(T10)比較LinqToSql的AsEnumerable、AsQueryable  
CourseGUID: 5ba9a6fe-7475-4b0c-8b99-bbcf7f5e2e1c  
=======================================================================  
(T10)比較LinqToSql的AsEnumerable、AsQueryable  
=======================================================================  
0. Summary

-----------

1. Web Form Application - Linq Query

1.1. TSQL

1.2. Set up SQL Authentication

1.3. Create New Project : Sample

-----------

2. Linq to SQL

2.1. Add Connection

2.2. DataClasses1.dbml

2.3. Program.cs

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

0. Summary

1.

Deferred/Lazy Operators  V.S.  Immediate/Greedy Operators

Based on the behavior of query execution, Linq can be classified into 2 categories.

1.1. Deferred/Lazy Operators use deferred execution.

E.g.  select, where, Take, Skip ...

1.2. Immediate/Greedy Operators use immediate execution.

E.g.  count, average, min, max, ToList ...

1.3.

ToList, ToArray, ToDictionary, ToLookup, Cast, OfType, AsEnumerable, AsQueryable

are Linq Conversion Operators.

2.

2.1.

Queryable.AsQueryable<TElement>

(IEnumerable<TElement>)

Reference:

[https://msdn.microsoft.com/en-us/library/bb507003(v=vs.110).aspx](https://msdn.microsoft.com/en-us/library/bb507003%28v=vs.110%29.aspx)

<https://stackoverflow.com/questions/17366907/what-is-the-purpose-of-asqueryable>

Converts a generic IEnumerable<T> to a generic IQueryable<T>.

The main use of AsQueryable operator is unit testing to mock a queryable in-memory data source

3.

3.1.

Enumerable.AsEnumerable<TSource>

(this IEnumerable<TSource> source)

Reference:

[https://msdn.microsoft.com/en-us/library/bb335435(v=vs.110).aspx](https://msdn.microsoft.com/en-us/library/bb335435%28v=vs.110%29.aspx)

Returns the input typed as IEnumerable<T>.

3.2.

AsEnumerable operator split the Linq query into 2 parts.

In another words, AsEnumerable() move query processing to the client side.

3.2.1.

Linq to SQL part

The Linq query before AsEnumerable() is Linq to SQL part which reads data from SQL Server database to application.

3.2.2.

Linq to Objects part

The Linq query after AsEnumerable() is Linq to Objects part which process to the local client side machine.

4.

IQueryable<T> V.S.  IEnumerable<T> in Entity Framework

In the future, you might learn Entity Framework,

then you might see the following.

4.1.

//var xxxDbContext = new XxxDbContext();

// **IQueryable<Gamer>** gamers =xxxDbContext.Gamers;

//var gamer = Gamers.Where(g=>g.Level);

**IQueryable<T>** is **Deferred/Lazy** Operation type which use deferred execution.

It means it will generate the following TSQL.

Select ... From .... Where...

4.2.

//var xxxDbContext = new XxxDbContext();

//**IEnumerable<Gamer>** gamers = IQueryable.Gamers;

//var gamer = Gamers.Where(g=>g.Level);

**IEnumerable<T>** is **Immediate/Greedy** Operation type which use immediate execution.

It means it will generate the following TSQL.

Select ... From ....

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

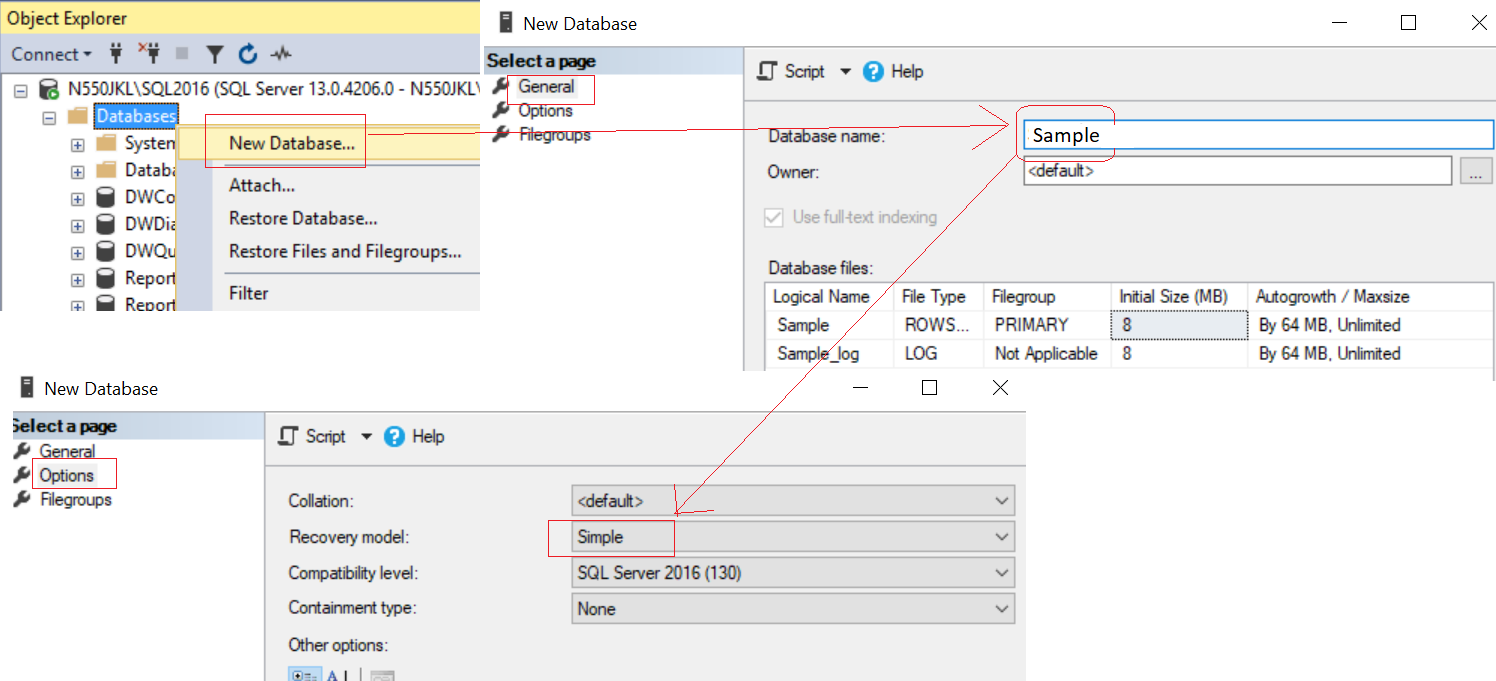
1. Web Form Application - Linq Query

1.1. TSQL

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

Database Name : Sample

Options --> Recovery Model : Simple



--Create an Sample DataBase and Run the following TSQL

Create Table Gamer

(

     Id int primary key IDENTITY(1,1),

     Name nvarchar(100),

     Gender nvarchar(50),

        Score int

)

GO

Insert into Gamer values ('Name1', 'Male', 3500)

Insert into Gamer values ('Name2', 'Female', 4000)

Insert into Gamer values ('Name3', 'Male', 5000)

Insert into Gamer values ('Name4', 'Female', 7000)

Insert into Gamer values ('Name5', 'Female', 3000)

Insert into Gamer values ('Name6', 'Male', 4500)

Insert into Gamer values ('Name7', 'Male', 4000)

Insert into Gamer values ('Name8', 'Male', 5500)

Insert into Gamer values ('Name9', 'Female', 6500)

Insert into Gamer values ('Name10', 'Female', 3500)

GO

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 New Project : Sample

File --> New --> Project... -->

Visual C# -->  **Console App** **(.Net Framework)** -->

Name: **Sample**

Graphical user interface, application, email

Description automatically generated

Graphical user interface, application

Description automatically generated

Graphical user interface, text, application, Excel

Description automatically generated

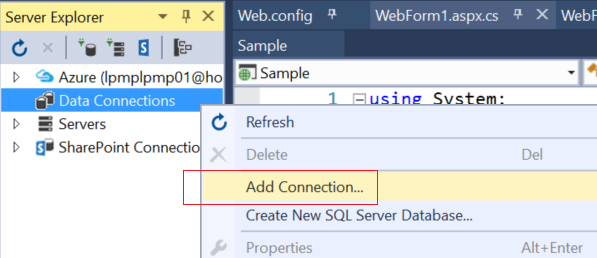
2. Linq to SQL

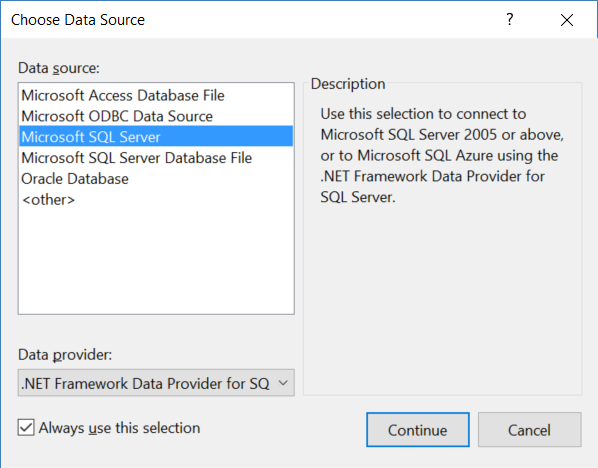
2.1. Add Connection

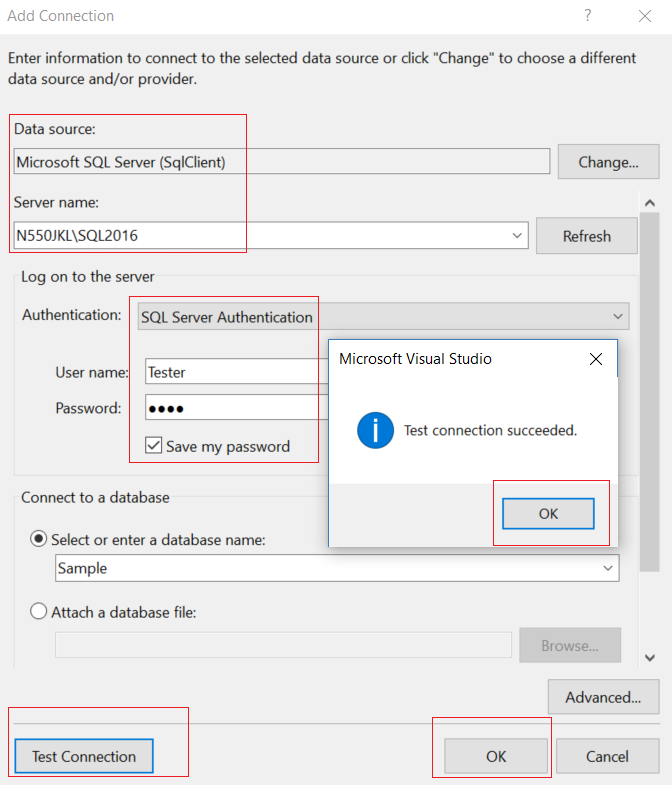
Server Explorer --> Data Connections --> Right click --> Add Connection...

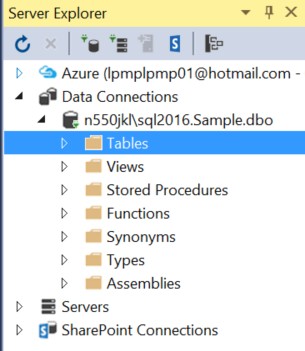
--> Microsoft SQL server -->

Enter your server and database details ....









2.2. DataClasses1.dbml

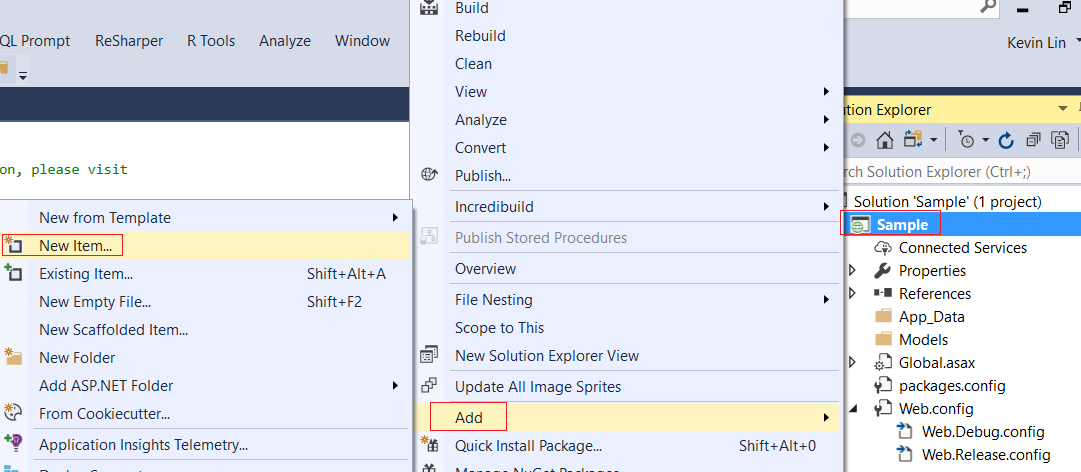
ProjectName --> Right Click --> Add --> New Item...

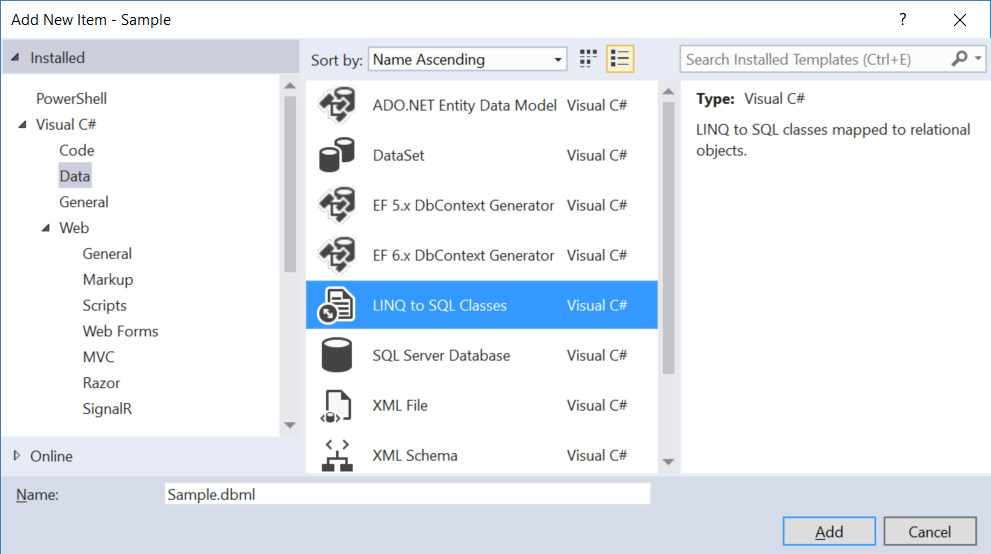
--> Linq to SQL classes -->

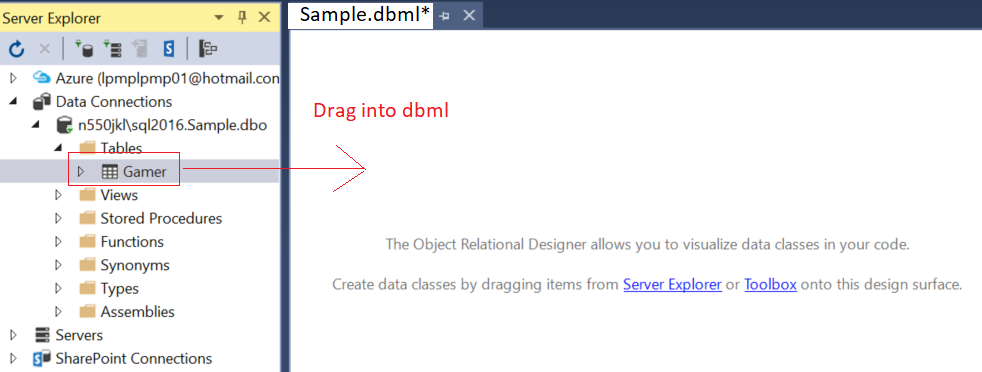
Name : **Sample.dbml**

-->

Drag Table from Server Explorer into DBML

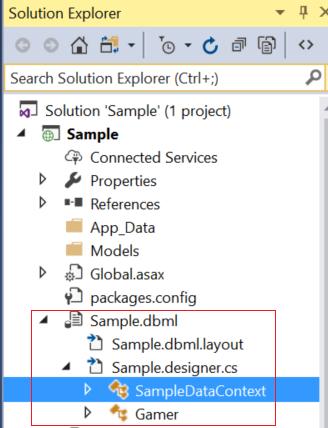






Graphical user interface, application

Description automatically generated



2.3. Program.cs

using System;

using System.Collections.Generic;

using System.Linq;

namespace Sample

{

    class Program

    {

        static void Main(string[] args)

        {

            // 1. ==========================================

            //Top5MaleByScore()

            Console.WriteLine("1. Top5MaleByScore() ===================== ");

            Top5MaleByScore();

            // 2. ==========================================

            //Top5MaleByScore2()

            Console.WriteLine("2. Top5MaleByScore2() ===================== ");

            Top5MaleByScore2();

            // 3. ==========================================

            //Top5MaleByScore3()

            Console.WriteLine("3. Top5MaleByScore3() ===================== ");

            Top5MaleByScore3();

            Console.ReadLine();

        }

        // 1. ==========================================

        //Top5MaleByScore()

        private static void Top5MaleByScore()

        {

            SampleDataContext dataContext = new SampleDataContext();

            //Get Top 5 Male by Score

            IQueryable<Gamer> top5MaleByScore =

                dataContext.Gamers

                .Where(g => g.Gender == "Male")

                .OrderByDescending(g => g.Score)

                .Take(5);

            foreach (Gamer gamer in top5MaleByScore)

            {

                Console.WriteLine(gamer);

            }

        }

        //1.1.

        //Id==8,Name==Name8,Gender==Male,Score==5500

        //Id==3,Name==Name3,Gender==Male,Score==5000

        //Id==6,Name==Name6,Gender==Male,Score==4500

        //Id==7,Name==Name7,Gender==Male,Score==4000

        //Id==1,Name==Name1,Gender==Male,Score==3500

        //1.2.

        //Notice that the following SQL Query is executed against the database.

        //exec sp\_executesql N'SELECT TOP (5) [t0].[Id], [t0].[Name], [t0].[Gender], [t0].[Score]

        //FROM[dbo].[Gamer] AS[t0]

        //WHERE[t0].[Gender] = @p0

        //ORDER BY[t0].[Score] DESC',N'@p0 nvarchar(4000)',@p0=N'Male'

        // 2. ==========================================

        //Top5MaleByScore2()

        private static void Top5MaleByScore2()

        {

            SampleDataContext dataContext = new SampleDataContext();

            //Get Top 5 Male by Score

            IEnumerable<Gamer> top5MaleByScore =

                dataContext.Gamers.AsEnumerable()

                .Where(g => g.Gender == "Male")

                .OrderByDescending(g => g.Score)

                .Take(5);

            foreach (Gamer gamer in top5MaleByScore)

            {

                Console.WriteLine(gamer);

            }

        }

        //1.1.

        //Id==8,Name==Name8,Gender==Male,Score==5500

        //Id==3,Name==Name3,Gender==Male,Score==5000

        //Id==6,Name==Name6,Gender==Male,Score==4500

        //Id==7,Name==Name7,Gender==Male,Score==4000

        //Id==1,Name==Name1,Gender==Male,Score==3500

        //1.2.

        //Notice that the following SQL Query is executed against the database.

        //SELECT [t0].[Id], [t0].[Name], [t0].[Gender], [t0].[Score]

        //FROM[dbo].[Gamer] AS[t0]

       // 3. ==========================================

        //Top5MaleByScore3()

        private static void Top5MaleByScore3()

        {

            SampleDataContext dataContext = new SampleDataContext();

            //Get Top 5 Male by Score

            IEnumerable<Gamer> top5MaleByScore =

                dataContext.Gamers

                .Where(g => g.Gender == "Male")

                .AsEnumerable()

                .OrderByDescending(g => g.Score)

                .Take(5);

            foreach (Gamer gamer in top5MaleByScore)

            {

                Console.WriteLine(gamer);

            }

        }

        //3.1.

        //Id==8,Name==Name8,Gender==Male,Score==5500

        //Id==3,Name==Name3,Gender==Male,Score==5000

        //Id==6,Name==Name6,Gender==Male,Score==4500

        //Id==7,Name==Name7,Gender==Male,Score==4000

        //Id==1,Name==Name1,Gender==Male,Score==3500

        //3.2.

        //Notice that the following SQL Query is executed against the database.

        //exec sp\_executesql N'SELECT [t0].[Id], [t0].[Name], [t0].[Gender], [t0].[Score]

        //FROM[dbo].[Gamer]

        //AS[t0]

        //WHERE[t0].[Gender] = @p0',N'@p0 nvarchar(4000)',@p0=N'Male'

    }

    public partial class Gamer

    {

        public override string ToString()

        {

            return $"Id=={Id},Name=={Name},Gender=={Gender},Score=={Score}";

        }

    }

}

/\*

1.

Deferred/Lazy Operators  V.S.  Immediate/Greedy Operators

Based on the behavior of query execution, Linq can be classified into 2 categories.

1.1. Deferred/Lazy Operators use deferred execution.

E.g.  select, where, Take, Skip ...

1.2. Immediate/Greedy Operators use immediate execution.

E.g.  count, average, min, max, ToList ...

1.3.

ToList, ToArray, ToDictionary, ToLookup, Cast, OfType, AsEnumerable, AsQueryable

are Linq Conversion Operators.

2.

Queryable.AsQueryable<TElement>

(IEnumerable<TElement>)

Reference:

[https://msdn.microsoft.com/en-us/library/bb507003(v=vs.110).aspx](https://msdn.microsoft.com/en-us/library/bb507003%28v=vs.110%29.aspx)

<https://stackoverflow.com/questions/17366907/what-is-the-purpose-of-asqueryable>

Converts a generic IEnumerable<T> to a generic IQueryable<T>.

The main use of AsQueryable operator is unit testing to mock a queryable in-memory data source

3.

3.1.

Enumerable.AsEnumerable<TSource>

(this IEnumerable<TSource> source)

Reference:

[https://msdn.microsoft.com/en-us/library/bb335435(v=vs.110).aspx](https://msdn.microsoft.com/en-us/library/bb335435%28v=vs.110%29.aspx)

Returns the input typed as IEnumerable<T>.

3.2.

AsEnumerable operator split the Linq query into 2 parts.

In another words, AsEnumerable() move query processing to the client side.

3.2.1.

Linq to SQL part

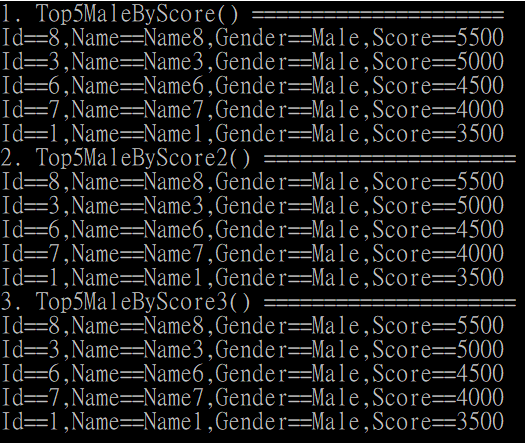
The Linq query before AsEnumerable() is Linq to SQL part which reads data from SQL Server database to application.

3.2.2.

Linq to Objects part

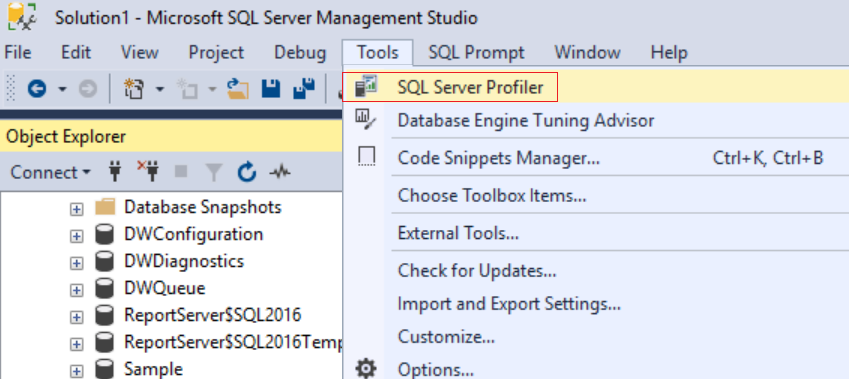
The Linq query after AsEnumerable() is Linq to Objects part which process to the local client side machine.

\*/



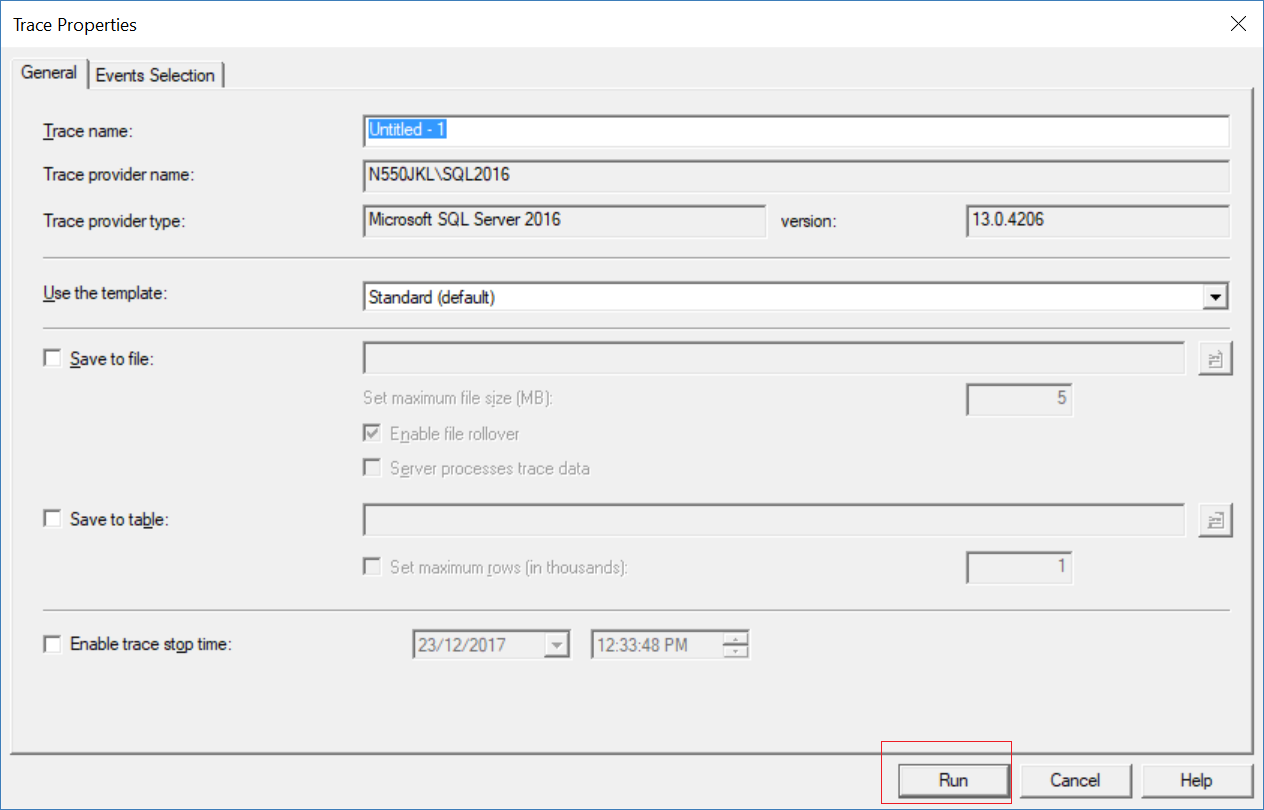
2.4. SQL Profiler

Tools --> SQL Server Profiler



Graphical user interface, text, application, email

Description automatically generated



Now, go back to VS2017, and run again

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

// 1. ==========================================

//Top5MaleByScore()

private static void Top5MaleByScore()

{

    SampleDataContext dataContext = new SampleDataContext();

    //Get Top 5 Male by Score

    IQueryable<Gamer> top5MaleByScore =

        dataContext.Gamers

        .Where(g => g.Gender == "Male")

        .OrderByDescending(g => g.Score)

        .Take(5);

    foreach (Gamer gamer in top5MaleByScore)

    {

        Console.WriteLine(gamer);

    }

}

//1.1.

//Id==8,Name==Name8,Gender==Male,Score==5500

//Id==3,Name==Name3,Gender==Male,Score==5000

//Id==6,Name==Name6,Gender==Male,Score==4500

//Id==7,Name==Name7,Gender==Male,Score==4000

//Id==1,Name==Name1,Gender==Male,Score==3500

//1.2.

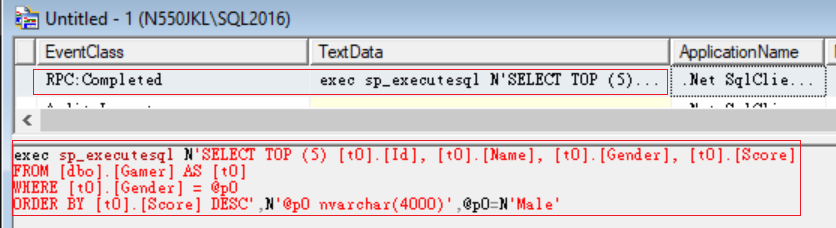
//Notice that the following SQL Query is executed against the database.

//exec sp\_executesql N'SELECT TOP (5) [t0].[Id], [t0].[Name], [t0].[Gender], [t0].[Score]

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

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

//ORDER BY[t0].[Score] DESC',N'@p0 nvarchar(4000)',@p0=N'Male'



// 2. ==========================================

//Top5MaleByScore2()

private static void Top5MaleByScore2()

{

    SampleDataContext dataContext = new SampleDataContext();

    //Get Top 5 Male by Score

    IEnumerable<Gamer> top5MaleByScore =

        dataContext.Gamers.AsEnumerable()

        .Where(g => g.Gender == "Male")

        .OrderByDescending(g => g.Score)

        .Take(5);

    foreach (Gamer gamer in top5MaleByScore)

    {

        Console.WriteLine(gamer);

    }

}

//1.1.

//Id==8,Name==Name8,Gender==Male,Score==5500

//Id==3,Name==Name3,Gender==Male,Score==5000

//Id==6,Name==Name6,Gender==Male,Score==4500

//Id==7,Name==Name7,Gender==Male,Score==4000

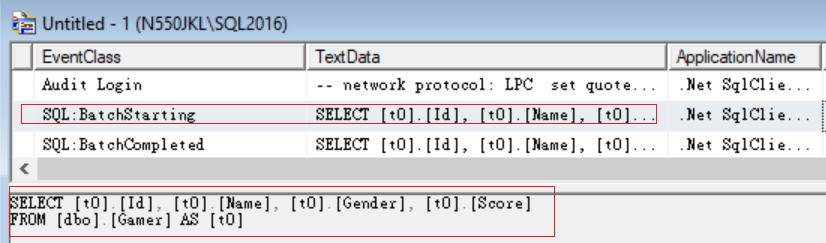
//Id==1,Name==Name1,Gender==Male,Score==3500

//1.2.

//Notice that the following SQL Query is executed against the database.

//SELECT [t0].[Id], [t0].[Name], [t0].[Gender], [t0].[Score]

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



// 3. ==========================================

//Top5MaleByScore3()

private static void Top5MaleByScore3()

{

    SampleDataContext dataContext = new SampleDataContext();

    //Get Top 5 Male by Score

    IEnumerable<Gamer> top5MaleByScore =

        dataContext.Gamers

        .Where(g => g.Gender == "Male")

        .AsEnumerable()

        .OrderByDescending(g => g.Score)

        .Take(5);

    foreach (Gamer gamer in top5MaleByScore)

    {

        Console.WriteLine(gamer);

    }

}

//3.1.

//Id==8,Name==Name8,Gender==Male,Score==5500

//Id==3,Name==Name3,Gender==Male,Score==5000

//Id==6,Name==Name6,Gender==Male,Score==4500

//Id==7,Name==Name7,Gender==Male,Score==4000

//Id==1,Name==Name1,Gender==Male,Score==3500

//3.2.

//Notice that the following SQL Query is executed against the database.

//exec sp\_executesql N'SELECT [t0].[Id], [t0].[Name], [t0].[Gender], [t0].[Score]

//FROM[dbo].[Gamer]

//AS[t0]

//WHERE[t0].[Gender] = @p0',N'@p0 nvarchar(4000)',@p0=N'Male'

