(T11)討論 LinqToObject 的 IGroupingKeyValue、GroupBy
CourseGUID: 5ba9a6fe-7475-4b0c-8b99-bbcf7f5e2e1c
==
(T11)討論 LinqToObject 的 IGroupingKeyValue、GroupBy
==

==

0. Summary

GroupBy organize a flat sequence of items and
return a sequence of IGrouping<K,V> based on specific keys.

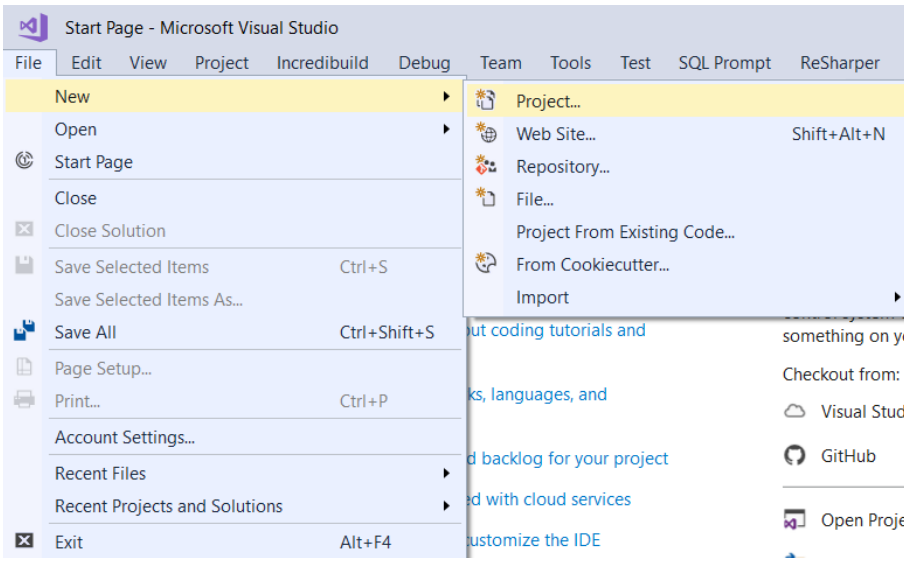
1. New Project

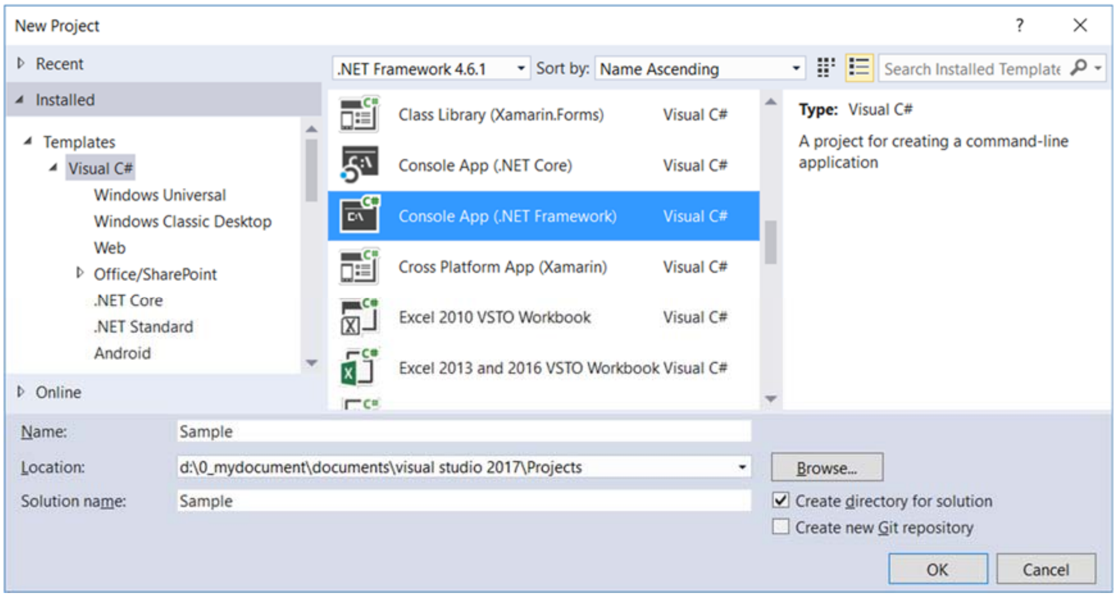
1.1. Create New Project : Sample

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

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

Name: **Sample**




New Project
Recent
Installed
Templates
Visual C#
Windows Universal
Windows Classic Desktop
Web
Office/SharePoint
.NET Core
.NET Standard
Android
Online
.NET Framework 4.6.1
Sort by: Name Ascending
Search Installed Template
Class Library (Xamarin.Forms) Visual C#
Console App (.NET Core) Visual C#
Console App (.NET Framework) Visual C#
Cross Platform App (Xamarin) Visual C#
Excel 2010 VSTO Workbook Visual C#
Excel 2013 and 2016 VSTO Workbook Visual C#
Browse...
OK
Cancel

Type: Visual C#

A project for creating a command-line application

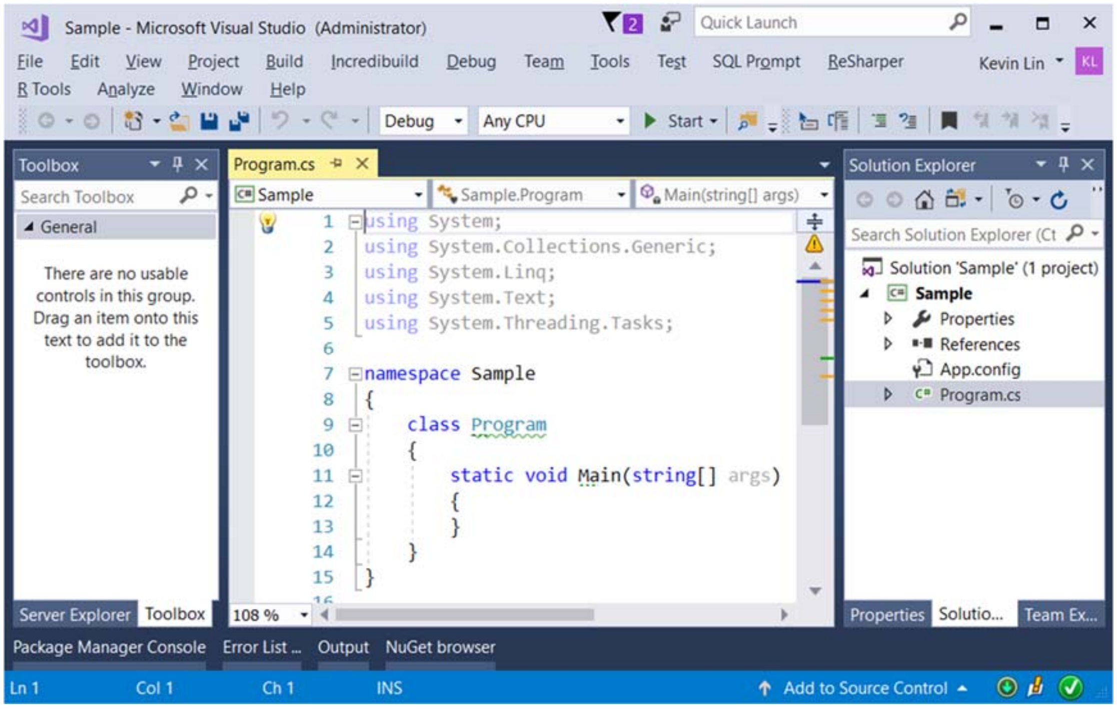
Name: Sample

Location: d:\0_mydocument\documents\visual studio 2017\Projects

Solution name: Sample

[✓] Create directory for solution

[ ] Create new Git repository


Sample - Microsoft Visual Studio (Administrator)
Quick Launch
File Edit View Project Build Incredibuild Debug Team Tools Test SQL Prompt ReSharper
Kevin Lin
R Tools Analyze Window Help
Debug
Any CPU
Start
Toolbox
Search Toolbox
General
There are no usable controls in this group. Drag an item onto this text to add it to the toolbox.
Program.cs
Sample
Sample.Program
Main(string[] args)
Solution Explorer
Search Solution Explorer (Ct
Solution 'Sample' (1 project)
Sample
Properties
References
App.config
Program.cs
Server Explorer Toolbox
108 %
Properties Solutio... Team Ex...
Package Manager Console Error List ... Output NuGet browser
Ln 1 Col 1 Ch 1 INS
Add to Source Control

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Sample
{
    class Program
    {
        static void Main(string[] args)
        {
        }
    }
}
```

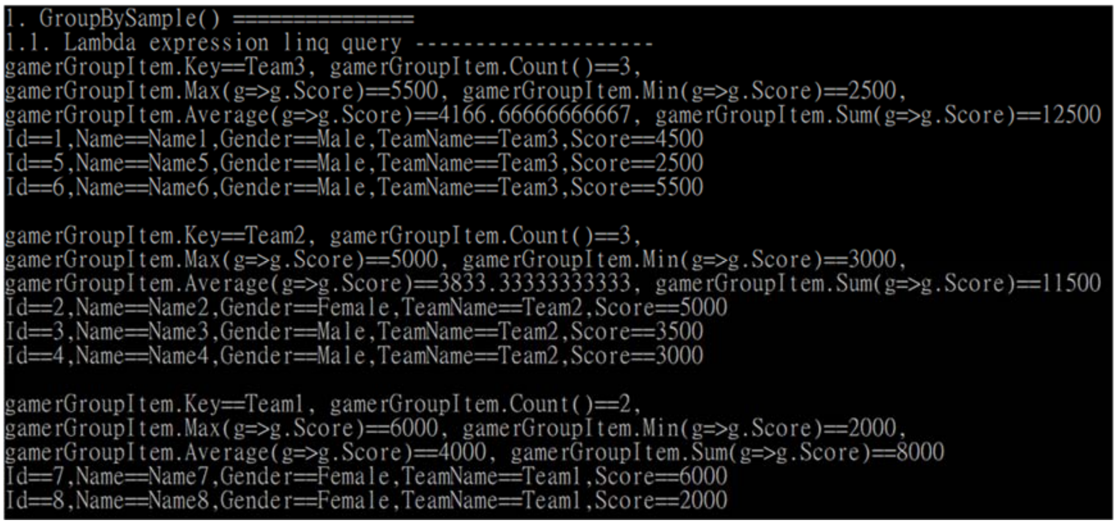
==

2. Sample : Program.cs

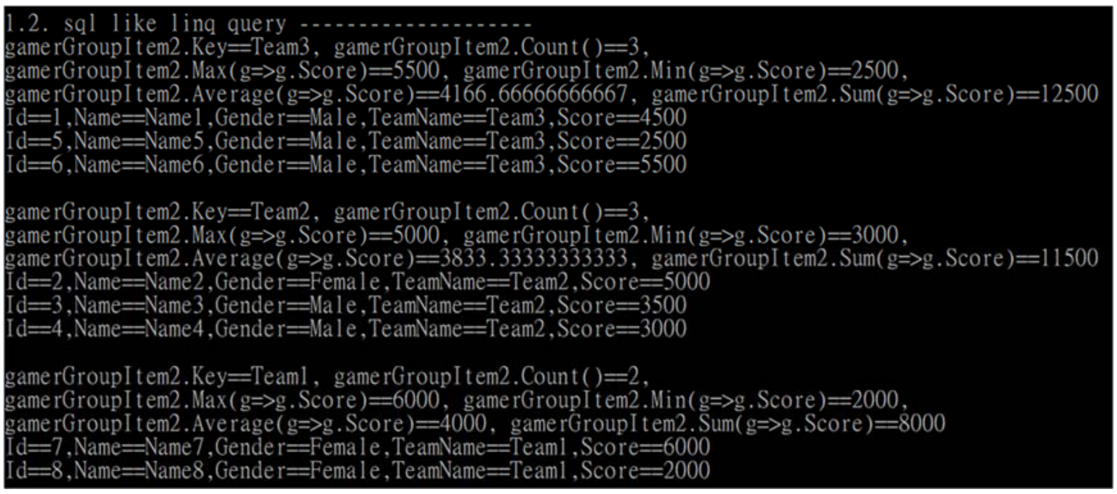
```
using System;
using System.Collections.Generic;
using System.Linq;
using OnLineGame;
namespace Sample
{
    class Program
     {
        static void Main(string[] args)
         {
            // 1. ========================================
            //GroupBySample
            Console.WriteLine("1. GroupBySample() =============== ");
            GroupBySample();
            // 2. =========================================
            //GroupByIntoSample
            Console.WriteLine("2. GroupByIntoSample() =============== ");
            GroupByIntoSample();
            // 3. =========================================
            //GroupByIntoMultipleKeysSample
            Console.WriteLine("3. GroupByIntoMultipleKeysSample() =============== ");
            GroupByIntoMultipleKeysSample();
            Console.ReadLine();
        }


        // 1. =========================================
        //GroupBySample
        static void GroupBySample()
        {
            List<Gamer> gamersList = GamerHelper.GetSampleGamers();
            //1.1. Lambda expression linq query --------------------
            Console.WriteLine("1.1. Lambda expression linq query -------------------- ");
            IEnumerable<IGrouping<string, Gamer>> gamerGroupEnumerable =
                gamersList.GroupBy(g => g.TeamName);
            foreach (IGrouping<string, Gamer> gamerGroupItem in gamerGroupEnumerable)
            {
                Console.WriteLine($"gamerGroupItem.Key=={gamerGroupItem.Key}, 
gamerGroupItem.Count()=={gamerGroupItem.Count()}, \r\ngamerGroupItem.Max(g=>g.Score)=={gamerGroupItem.Max(
g => g.Score)}, gamerGroupItem.Min(g=>g.Score)=={gamerGroupItem.Min(g =>
g.Score)}, \r\ngamerGroupItem.Average(g=>g.Score)=={gamerGroupItem.Average(g => g.Score)}, 
gamerGroupItem.Sum(g=>g.Score)=={gamerGroupItem.Sum(g => g.Score)}");
                foreach (Gamer gamer in gamerGroupItem)
                {
                    Console.WriteLine(gamer);
                }
                Console.WriteLine();
            }
            // 1.1. Lambda expression linq query --------------------
            // gamerGroupItem.Key==Team3, gamerGroupItem.Count()==3,
            // gamerGroupItem.Max(g=>g.Score)==5500, gamerGroupItem.Min(g=>g.Score)==2500,
            // gamerGroupItem.Average(g=>g.Score)==4166.66666666667, gamerGroupItem.Sum(g=>g.Score)==12500
            // Id==1,Name==Name1,Gender==Male,TeamName==Team3,Score==4500
            // Id==5,Name==Name5,Gender==Male,TeamName==Team3,Score==2500
            // Id==6,Name==Name6,Gender==Male,TeamName==Team3,Score==5500
            // gamerGroupItem.Key==Team2, gamerGroupItem.Count()==3,
            // gamerGroupItem.Max(g=>g.Score)==5000, gamerGroupItem.Min(g=>g.Score)==3000,
            // gamerGroupItem.Average(g=>g.Score)==3833.33333333333, gamerGroupItem.Sum(g=>g.Score)==11500
            // Id==2,Name==Name2,Gender==Female,TeamName==Team2,Score==5000
            // Id==3,Name==Name3,Gender==Male,TeamName==Team2,Score==3500
            // Id==4,Name==Name4,Gender==Male,TeamName==Team2,Score==3000
            // gamerGroupItem.Key==Team1, gamerGroupItem.Count()==2,
            // gamerGroupItem.Max(g=>g.Score)==6000, gamerGroupItem.Min(g=>g.Score)==2000,
            // gamerGroupItem.Average(g=>g.Score)==4000, gamerGroupItem.Sum(g=>g.Score)==8000
            // Id==7,Name==Name7,Gender==Female,TeamName==Team1,Score==6000
            // Id==8,Name==Name8,Gender==Female,TeamName==Team1,Score==2000
            //1.2. sql like linq query --------------------
            Console.WriteLine("1.2. sql like linq query -------------------- ");
            IEnumerable<IGrouping<string, Gamer>> gamerGroupEnumerable2 =
                from gamer in GamerHelper.GetSampleGamers()
                group gamer by gamer.TeamName;
            foreach (IGrouping<string, Gamer> gamerGroupItem2 in gamerGroupEnumerable2)
            {
                Console.WriteLine($"gamerGroupItem2.Key=={gamerGroupItem2.Key},
gamerGroupItem2.Count()=={gamerGroupItem2.Count()}, \r\ngamerGroupItem2.Max(g=>g.Score)=={gamerGroupItem2.
Max(g => g.Score)}, gamerGroupItem2.Min(g=>g.Score)=={gamerGroupItem2.Min(g =>
g.Score)}, \r\ngamerGroupItem2.Average(g=>g.Score)=={gamerGroupItem2.Average(g => g.Score)},
gamerGroupItem2.Sum(g=>g.Score)=={gamerGroupItem2.Sum(g => g.Score)}");
                foreach (Gamer gamer2 in gamerGroupItem2)
                {
                    Console.WriteLine(gamer2);
                }
                Console.WriteLine();
            }
        }
        // 1.2. sql like linq query --------------------
        // gamerGroupItem2.Key==Team3, gamerGroupItem2.Count()==3,
        // gamerGroupItem2.Max(g=>g.Score)==5500, gamerGroupItem2.Min(g=>g.Score)==2500,
        // gamerGroupItem2.Average(g=>g.Score)==4166.66666666667, gamerGroupItem2.Sum(g=>g.Score)==12500
        // Id==1,Name==Name1,Gender==Male,TeamName==Team3,Score==4500
        // Id==5,Name==Name5,Gender==Male,TeamName==Team3,Score==2500
        // Id==6,Name==Name6,Gender==Male,TeamName==Team3,Score==5500
        // gamerGroupItem2.Key==Team2, gamerGroupItem2.Count()==3,
        // gamerGroupItem2.Max(g=>g.Score)==5000, gamerGroupItem2.Min(g=>g.Score)==3000,
        // gamerGroupItem2.Average(g=>g.Score)==3833.33333333333, gamerGroupItem2.Sum(g=>g.Score)==11500
        // Id==2,Name==Name2,Gender==Female,TeamName==Team2,Score==5000
        // Id==3,Name==Name3,Gender==Male,TeamName==Team2,Score==3500
        // Id==4,Name==Name4,Gender==Male,TeamName==Team2,Score==3000
        // gamerGroupItem2.Key==Team1, gamerGroupItem2.Count()==2,
        // gamerGroupItem2.Max(g=>g.Score)==6000, gamerGroupItem2.Min(g=>g.Score)==2000,
        // gamerGroupItem2.Average(g=>g.Score)==4000, gamerGroupItem2.Sum(g=>g.Score)==8000
        // Id==7,Name==Name7,Gender==Female,TeamName==Team1,Score==6000
        // Id==8,Name==Name8,Gender==Female,TeamName==Team1,Score==2000



        // 2. ========================================
        //GroupByIntoSample
        static void GroupByIntoSample()
        {
            List<Gamer> gamersList = GamerHelper.GetSampleGamers();
            //2.1. sql like linq query --------------------
            Console.WriteLine("2.1. sql like linq query -------------------- ");
            var gamerGroups =
                from gamer in gamersList
                group gamer by gamer.TeamName into gGroup
                orderby gGroup.Key
                select new
                {
                    Key = gGroup.Key,
                    Gamers = gGroup.OrderBy(x => x.Name)
        };
      foreach (var gamerGroupsItem in gamerGroups)
      {
        Console.WriteLine($"gamerGroupsItem.Key=={gamerGroupsItem.Key},
gamerGroupsItem.Gamers.Count()=={gamerGroupsItem.Gamers.Count()}, \r\ngamerGroupsItem.Gamers.Max(g=>g.Scor
e)=={gamerGroupsItem.Gamers.Max(g => g.Score)},
gamerGroupsItem.Gamers.Min(g=>g.Score)=={gamerGroupsItem.Gamers.Min(g =>
g.Score)}, \r\ngamerGroupsItem.Gamers.Average(g=>g.Score)=={gamerGroupsItem.Gamers.Average(g => g.Score)},
gamerGroupsItem.Gamers.Sum(g=>g.Score)=={gamerGroupsItem.Gamers.Sum(g => g.Score)}");
        foreach (var gamer in gamerGroupsItem.Gamers)
        {
          Console.WriteLine(gamer);
        }
        Console.WriteLine(); Console.WriteLine();
      }
      // 2.1. sql like linq query --------------------
      // gamerGroupsItem.Key==Team1, gamerGroupsItem.Gamers.Count()==2,
      // gamerGroupsItem.Gamers.Max(g=>g.Score)==6000, gamerGroupsItem.Gamers.Min(g=>g.Score)==2000,
      // gamerGroupsItem.Gamers.Average(g=>g.Score)==4000,
gamerGroupsItem.Gamers.Sum(g=>g.Score)==8000
      // Id==7,Name==Name7,Gender==Female,TeamName==Team1,Score==6000
      // Id==8,Name==Name8,Gender==Female,TeamName==Team1,Score==2000
      // gamerGroupsItem.Key==Team2, gamerGroupsItem.Gamers.Count()==3,
      // gamerGroupsItem.Gamers.Max(g=>g.Score)==5000, gamerGroupsItem.Gamers.Min(g=>g.Score)==3000,
      // gamerGroupsItem.Gamers.Average(g=>g.Score)==3833.33333333333,
gamerGroupsItem.Gamers.Sum(g=>g.Score)==11500
      // Id==2,Name==Name2,Gender==Female,TeamName==Team2,Score==5000
      // Id==3,Name==Name3,Gender==Male,TeamName==Team2,Score==3500
      // Id==4,Name==Name4,Gender==Male,TeamName==Team2,Score==3000
      // gamerGroupsItem.Key==Team3, gamerGroupsItem.Gamers.Count()==3,
      // gamerGroupsItem.Gamers.Max(g=>g.Score)==5500, gamerGroupsItem.Gamers.Min(g=>g.Score)==2500,
      // gamerGroupsItem.Gamers.Average(g=>g.Score)==4166.66666666667,
gamerGroupsItem.Gamers.Sum(g=>g.Score)==12500
      // Id==1,Name==Name1,Gender==Male,TeamName==Team3,Score==4500
      // Id==5,Name==Name5,Gender==Male,TeamName==Team3,Score==2500
      // Id==6,Name==Name6,Gender==Male,TeamName==Team3,Score==5500
      //2.2. Lambda expression linq query --------------------
      Console.WriteLine("2.2. Lambda expression linq query -------------------- ");
      var gamerGroups2 =
        gamersList.GroupBy(g => g.TeamName).OrderBy(group => group.Key).Select(group => new
        {
          Key = group.Key,
          Gamers = group.OrderBy(x => x.Name)
        });
      foreach (var gamerGroupsItem2 in gamerGroups2)
      {
        Console.WriteLine($"gamerGroupsItem2.Key=={gamerGroupsItem2.Key},
gamerGroupsItem2.Gamers.Count()=={gamerGroupsItem2.Gamers.Count()}, \r\ngamerGroupsItem2.Gamers.Max(g=>g.S
core)=={gamerGroupsItem2.Gamers.Max(g => g.Score)},
gamerGroupsItem2.Gamers.Min(g=>g.Score)=={gamerGroupsItem2.Gamers.Min(g =>
g.Score)}, \r\ngamerGroupsItem2.Gamers.Average(g=>g.Score)=={gamerGroupsItem2.Gamers.Average(g =>
g.Score)}, gamerGroupsItem2.Gamers.Sum(g=>g.Score)=={gamerGroupsItem2.Gamers.Sum(g => g.Score)}");
        foreach (var gamer2 in gamerGroupsItem2.Gamers)
        {
          Console.WriteLine(gamer2);
        }
        Console.WriteLine(); Console.WriteLine();
```

```
            }
        }
        // 2.2. Lambda expression linq query --------------------
        // gamerGroupsItem2.Key==Team1, gamerGroupsItem2.Gamers.Count()==2,
        // gamerGroupsItem2.Gamers.Max(g=>g.Score)==6000, gamerGroupsItem2.Gamers.Min(g=>g.Score)==2000,
        // gamerGroupsItem2.Gamers.Average(g=>g.Score)==4000,
gamerGroupsItem2.Gamers.Sum(g=>g.Score)==8000
        // Id==7,Name==Name7,Gender==Female,TeamName==Team1,Score==6000
        // Id==8,Name==Name8,Gender==Female,TeamName==Team1,Score==2000
        // gamerGroupsItem2.Key==Team2, gamerGroupsItem2.Gamers.Count()==3,
        // gamerGroupsItem2.Gamers.Max(g=>g.Score)==5000, gamerGroupsItem2.Gamers.Min(g=>g.Score)==3000,
        // gamerGroupsItem2.Gamers.Average(g=>g.Score)==3833.33333333333,
gamerGroupsItem2.Gamers.Sum(g=>g.Score)==11500
        // Id==2,Name==Name2,Gender==Female,TeamName==Team2,Score==5000
        // Id==3,Name==Name3,Gender==Male,TeamName==Team2,Score==3500
        // Id==4,Name==Name4,Gender==Male,TeamName==Team2,Score==3000
        // gamerGroupsItem2.Key==Team3, gamerGroupsItem2.Gamers.Count()==3,
        // gamerGroupsItem2.Gamers.Max(g=>g.Score)==5500, gamerGroupsItem2.Gamers.Min(g=>g.Score)==2500,
        // gamerGroupsItem2.Gamers.Average(g=>g.Score)==4166.66666666667,
gamerGroupsItem2.Gamers.Sum(g=>g.Score)==12500
        // Id==1,Name==Name1,Gender==Male,TeamName==Team3,Score==4500
        // Id==5,Name==Name5,Gender==Male,TeamName==Team3,Score==2500
        // Id==6,Name==Name6,Gender==Male,TeamName==Team3,Score==5500


       // 3. =========================================
        //GroupByIntoMultipleKeysSample
        static void GroupByIntoMultipleKeysSample()
        {
            List<Gamer> gamersList = GamerHelper.GetSampleGamers();
            //3.1. Lambda expression linq query -----------
            Console.WriteLine("3.1. Lambda expression linq query ----------- ");
            var gamerGroups =
                gamersList
                .GroupBy(gamer => new { gamer.TeamName, gamer.Gender })
                .OrderBy(gamer => gamer.Key.TeamName).ThenBy(gamer => gamer.Key.Gender)
                .Select(group => new
                {
                    TeamName = group.Key.TeamName,
                    Gender = group.Key.Gender,
                    Gamers = group.OrderBy(g => g.Name)
                });
            foreach (var gamerGroup in gamerGroups)
            {
                Console.WriteLine($"gamerGroup.TeamName=={gamerGroup.TeamName},
gamerGroup.Gender=={gamerGroup.Gender},
gamerGroup.Gamers.Count()=={gamerGroup.Gamers.Count()},\r\ngamerGroup.Gamers.Min(g=>g.Score)=={gamerGroup
.Gamers.Min(g=>g.Score)}, gamerGroup.Gamers.Max(g=>g.Score)=={gamerGroup.Gamers.Max(g =>
g.Score)}, \r\ngamerGroup.Gamers.Average(g=>g.Score)=={gamerGroup.Gamers.Average(g => g.Score)},
gamerGroup.Gamers.Sum(g=>g.Score)=={gamerGroup.Gamers.Sum(g => g.Score)}");
                foreach (Gamer gamer in gamerGroup.Gamers)
                {
                    Console.WriteLine(gamer);
                }
                Console.WriteLine();
            }
            // 3.1. Lambda expression linq query -----------
            // gamerGroup.TeamName==Team1, gamerGroup.Gender==Female, gamerGroup.Gamers.Count()==2,
            // gamerGroup.Gamers.Min(g=>g.Score)==2000, gamerGroup.Gamers.Max(g=>g.Score)==6000,
            // gamerGroup.Gamers.Average(g=>g.Score)==4000, gamerGroup.Gamers.Sum(g=>g.Score)==8000
            // Id==7,Name==Name7,Gender==Female,TeamName==Team1,Score==6000
            // Id==8,Name==Name8,Gender==Female,TeamName==Team1,Score==2000
            // gamerGroup.TeamName==Team2, gamerGroup.Gender==Female, gamerGroup.Gamers.Count()==1,
            // gamerGroup.Gamers.Min(g=>g.Score)==5000, gamerGroup.Gamers.Max(g=>g.Score)==5000,
            // gamerGroup.Gamers.Average(g=>g.Score)==5000, gamerGroup.Gamers.Sum(g=>g.Score)==5000
            // Id==2,Name==Name2,Gender==Female,TeamName==Team2,Score==5000
            // gamerGroup.TeamName==Team2, gamerGroup.Gender==Male, gamerGroup.Gamers.Count()==2,
            // gamerGroup.Gamers.Min(g=>g.Score)==3000, gamerGroup.Gamers.Max(g=>g.Score)==3500,
            // gamerGroup.Gamers.Average(g=>g.Score)==3250, gamerGroup.Gamers.Sum(g=>g.Score)==6500
            // Id==3,Name==Name3,Gender==Male,TeamName==Team2,Score==3500
            // Id==4,Name==Name4,Gender==Male,TeamName==Team2,Score==3000
            // gamerGroup.TeamName==Team3, gamerGroup.Gender==Male, gamerGroup.Gamers.Count()==3,
            // gamerGroup.Gamers.Min(g=>g.Score)==2500, gamerGroup.Gamers.Max(g=>g.Score)==5500,
            // gamerGroup.Gamers.Average(g=>g.Score)==4166.66666666667,
gamerGroup.Gamers.Sum(g=>g.Score)==12500
            // Id==1,Name==Name1,Gender==Male,TeamName==Team3,Score==4500
            // Id==5,Name==Name5,Gender==Male,TeamName==Team3,Score==2500
            // Id==6,Name==Name6,Gender==Male,TeamName==Team3,Score==5500
            //3.2. SQL like linq query -----------
            Console.WriteLine("3.2. SQL like linq query ----------- ");
            var gamerGroups2 =
                 from gamer in gamersList
                 group gamer by new
                 {
                     gamer.TeamName,
                     gamer.Gender
                 } into gamerGroup
                 orderby gamerGroup.Key.TeamName ascending,
                                gamerGroup.Key.Gender ascending
                 select new
                 {
                     TeamName = gamerGroup.Key.TeamName,
                     Gender = gamerGroup.Key.Gender,
                     Gamers = gamerGroup.OrderBy(g => g.Name)
                 };
            foreach (var gamerGroup in gamerGroups2)
            {
                Console.WriteLine($"gamerGroup.TeamName=={gamerGroup.TeamName}, 
gamerGroup.Gender=={gamerGroup.Gender}, 
gamerGroup.Gamers.Count()=={gamerGroup.Gamers.Count()},\r\ngamerGroup.Gamers.Min(g=>g.Score)=={gamerGroup
.Gamers.Min(g => g.Score)}, gamerGroup.Gamers.Max(g=>g.Score)=={gamerGroup.Gamers.Max(g => 
g.Score)}, \r\ngamerGroup.Gamers.Average(g=>g.Score)=={gamerGroup.Gamers.Average(g => g.Score)}, 
gamerGroup.Gamers.Sum(g=>g.Score)=={gamerGroup.Gamers.Sum(g => g.Score)}");
                foreach (Gamer gamer in gamerGroup.Gamers)
                {
                    Console.WriteLine(gamer);
                }
                Console.WriteLine();
            }
        }
        // 3.2. SQL like linq query -----------
        // gamerGroup.TeamName==Team1, gamerGroup.Gender==Female, gamerGroup.Gamers.Count()==2,
        // gamerGroup.Gamers.Min(g=>g.Score)==2000, gamerGroup.Gamers.Max(g=>g.Score)==6000,
        // gamerGroup.Gamers.Average(g=>g.Score)==4000, gamerGroup.Gamers.Sum(g=>g.Score)==8000
        // Id==7,Name==Name7,Gender==Female,TeamName==Team1,Score==6000
        // Id==8,Name==Name8,Gender==Female,TeamName==Team1,Score==2000
        // gamerGroup.TeamName==Team2, gamerGroup.Gender==Female, gamerGroup.Gamers.Count()==1,
        // gamerGroup.Gamers.Min(g=>g.Score)==5000, gamerGroup.Gamers.Max(g=>g.Score)==5000,
        // gamerGroup.Gamers.Average(g=>g.Score)==5000, gamerGroup.Gamers.Sum(g=>g.Score)==5000
        // Id==2,Name==Name2,Gender==Female,TeamName==Team2,Score==5000
        // gamerGroup.TeamName==Team2, gamerGroup.Gender==Male, gamerGroup.Gamers.Count()==2,
        // gamerGroup.Gamers.Min(g=>g.Score)==3000, gamerGroup.Gamers.Max(g=>g.Score)==3500,
        // gamerGroup.Gamers.Average(g=>g.Score)==3250, gamerGroup.Gamers.Sum(g=>g.Score)==6500
        // Id==3,Name==Name3,Gender==Male,TeamName==Team2,Score==3500
        // Id==4,Name==Name4,Gender==Male,TeamName==Team2,Score==3000
        // gamerGroup.TeamName==Team3, gamerGroup.Gender==Male, gamerGroup.Gamers.Count()==3,
        // gamerGroup.Gamers.Min(g=>g.Score)==2500, gamerGroup.Gamers.Max(g=>g.Score)==5500,
        // gamerGroup.Gamers.Average(g=>g.Score)==4166.66666666667,
gamerGroup.Gamers.Sum(g=>g.Score)==12500
        // Id==1,Name==Name1,Gender==Male,TeamName==Team3,Score==4500
        // Id==5,Name==Name5,Gender==Male,TeamName==Team3,Score==2500
        // Id==6,Name==Name6,Gender==Male,TeamName==Team3,Score==5500
    }
}


namespace OnLineGame
{
    public class Gamer
    {
        public int Id { get; set; }
        public string Name { get; set; }
        public string Gender { get; set; }
        public string TeamName { get; set; }
        public int Score { get; set; }
        public override string ToString()
        {
            return $"Id=={Id},Name=={Name},Gender=={Gender},TeamName=={TeamName},Score=={Score}";
        }
    }
    public class GamerHelper
    {
        // Create a List<Gamer> which contains numberOfGamers Gamers.
        public static List<Gamer> GetSampleGamers()
        {
            return new List<Gamer>
            {
                new Gamer{Id=1,Name = "Name1",Gender = "Male",TeamName = "Team3",Score = 4500},
                new Gamer{Id=2,Name = "Name2",Gender = "Female",TeamName = "Team2",Score = 5000},
                new Gamer{Id=3,Name = "Name3",Gender = "Male",TeamName = "Team2",Score = 3500},
                new Gamer{Id=4,Name = "Name4",Gender = "Male",TeamName = "Team2",Score = 3000},
                new Gamer{Id=5,Name = "Name5",Gender = "Male",TeamName = "Team3",Score = 2500},
                new Gamer{Id=6,Name = "Name6",Gender = "Male",TeamName = "Team3",Score = 5500},
                new Gamer{Id=7,Name = "Name7",Gender = "Female",TeamName = "Team1",Score = 6000},
                new Gamer{Id=8,Name = "Name8",Gender = "Female",TeamName = "Team1",Score = 2000},
            };
        }
    }
}
/*
```

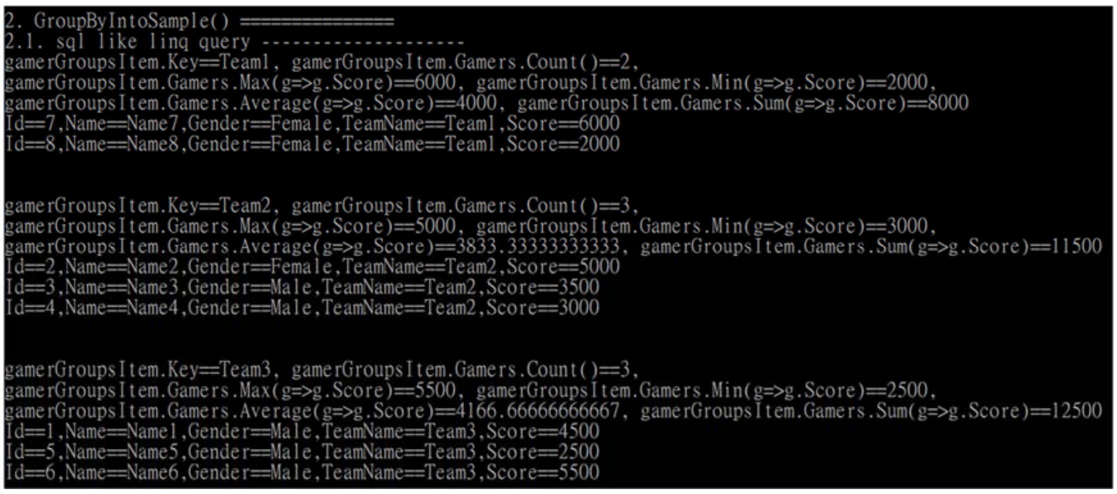
```
GroupBy organize a flat sequence of items and
return a sequence of IGrouping<K,V> based on specific keys.
*/
```



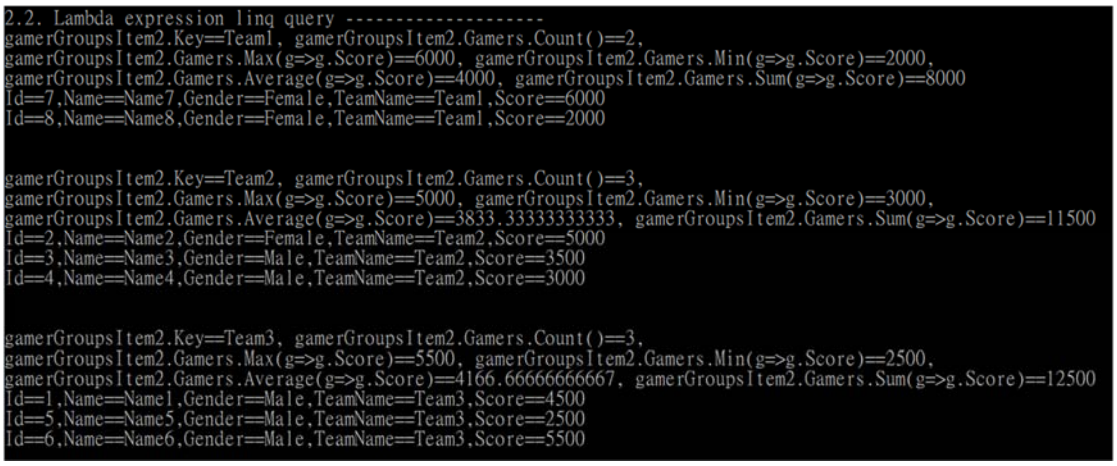
```
1. GroupBySample() ===============
1.1. Lambda expression linq query --------------------
gamerGroupItem.Key==Team3, gamerGroupItem.Count()==3,
gamerGroupItem.Max(g=>g.Score)==5500, gamerGroupItem.Min(g=>g.Score)==2500,
gamerGroupItem.Average(g=>g.Score)==4166.66666666667, gamerGroupItem.Sum(g=>g.Score)==12500
Id==1,Name==Name1,Gender==Male,TeamName==Team3,Score==4500
Id==5,Name==Name5,Gender==Male,TeamName==Team3,Score==2500
Id==6,Name==Name6,Gender==Male,TeamName==Team3,Score==5500

gamerGroupItem.Key==Team2, gamerGroupItem.Count()==3,
gamerGroupItem.Max(g=>g.Score)==5000, gamerGroupItem.Min(g=>g.Score)==3000,
gamerGroupItem.Average(g=>g.Score)==3833.33333333333, gamerGroupItem.Sum(g=>g.Score)==11500
Id==2,Name==Name2,Gender==Female,TeamName==Team2,Score==5000
Id==3,Name==Name3,Gender==Male,TeamName==Team2,Score==3500
Id==4,Name==Name4,Gender==Male,TeamName==Team2,Score==3000

gamerGroupItem.Key==Team1, gamerGroupItem.Count()==2,
gamerGroupItem.Max(g=>g.Score)==6000, gamerGroupItem.Min(g=>g.Score)==2000,
gamerGroupItem.Average(g=>g.Score)==4000, gamerGroupItem.Sum(g=>g.Score)==8000
Id==7,Name==Name7,Gender==Female,TeamName==Team1,Score==6000
Id==8,Name==Name8,Gender==Female,TeamName==Team1,Score==2000
```



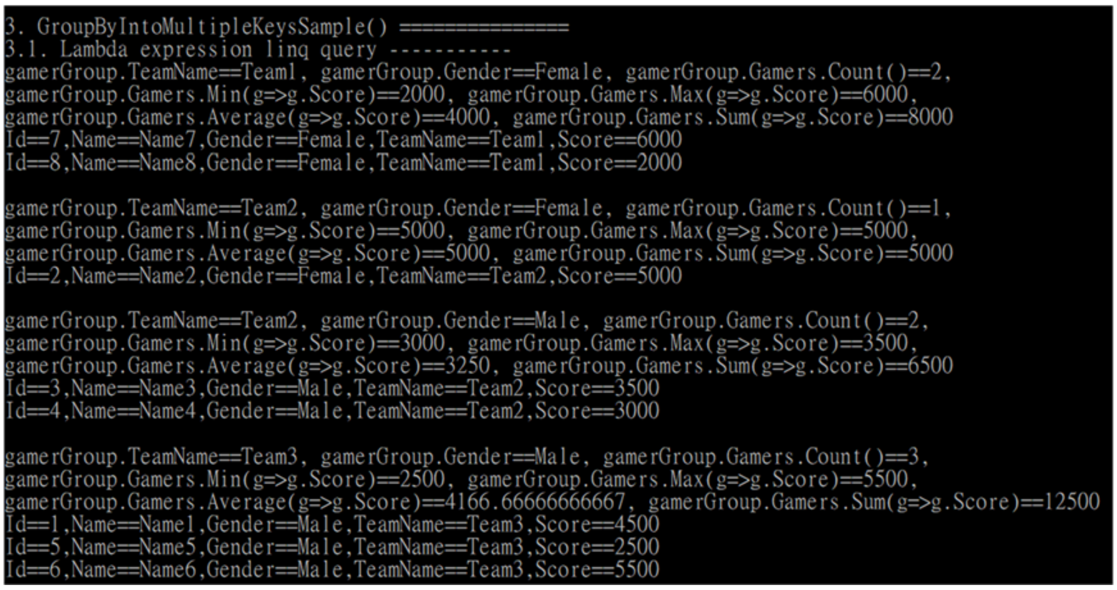
```
1.2. sql like linq query --------------------
gamerGroupItem2.Key==Team3, gamerGroupItem2.Count()==3,
gamerGroupItem2.Max(g=>g.Score)==5500, gamerGroupItem2.Min(g=>g.Score)==2500,
gamerGroupItem2.Average(g=>g.Score)==4166.66666666667, gamerGroupItem2.Sum(g=>g.Score)==12500
Id==1,Name==Name1,Gender==Male,TeamName==Team3,Score==4500
Id==5,Name==Name5,Gender==Male,TeamName==Team3,Score==2500
Id==6,Name==Name6,Gender==Male,TeamName==Team3,Score==5500

gamerGroupItem2.Key==Team2, gamerGroupItem2.Count()==3,
gamerGroupItem2.Max(g=>g.Score)==5000, gamerGroupItem2.Min(g=>g.Score)==3000,
gamerGroupItem2.Average(g=>g.Score)==3833.33333333333, gamerGroupItem2.Sum(g=>g.Score)==11500
Id==2,Name==Name2,Gender==Female,TeamName==Team2,Score==5000
Id==3,Name==Name3,Gender==Male,TeamName==Team2,Score==3500
Id==4,Name==Name4,Gender==Male,TeamName==Team2,Score==3000

gamerGroupItem2.Key==Team1, gamerGroupItem2.Count()==2,
gamerGroupItem2.Max(g=>g.Score)==6000, gamerGroupItem2.Min(g=>g.Score)==2000,
gamerGroupItem2.Average(g=>g.Score)==4000, gamerGroupItem2.Sum(g=>g.Score)==8000
Id==7,Name==Name7,Gender==Female,TeamName==Team1,Score==6000
Id==8,Name==Name8,Gender==Female,TeamName==Team1,Score==2000
```



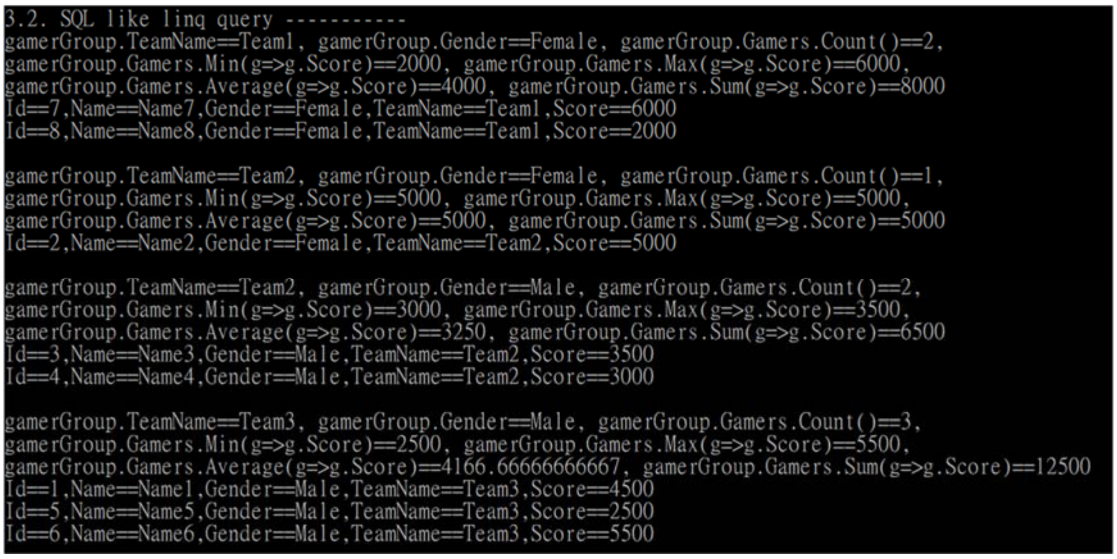
```
2. GroupByIntoSample() ==============
2.1. sql like linq query --------------------
gamerGroupsItem.Key==Team1, gamerGroupsItem.Gamers.Count()==2,
gamerGroupsItem.Gamers.Max(g=>g.Score)==6000, gamerGroupsItem.Gamers.Min(g=>g.Score)==2000,
gamerGroupsItem.Gamers.Average(g=>g.Score)==4000, gamerGroupsItem.Gamers.Sum(g=>g.Score)==8000
Id==7,Name==Name7,Gender==Female,TeamName==Team1,Score==6000
Id==8,Name==Name8,Gender==Female,TeamName==Team1,Score==2000


gamerGroupsItem.Key==Team2, gamerGroupsItem.Gamers.Count()==3,
gamerGroupsItem.Gamers.Max(g=>g.Score)==5000, gamerGroupsItem.Gamers.Min(g=>g.Score)==3000,
gamerGroupsItem.Gamers.Average(g=>g.Score)==3833.33333333333, gamerGroupsItem.Gamers.Sum(g=>g.Score)==11500
Id==2,Name==Name2,Gender==Female,TeamName==Team2,Score==5000
Id==3,Name==Name3,Gender==Male,TeamName==Team2,Score==3500
Id==4,Name==Name4,Gender==Male,TeamName==Team2,Score==3000


gamerGroupsItem.Key==Team3, gamerGroupsItem.Gamers.Count()==3,
gamerGroupsItem.Gamers.Max(g=>g.Score)==5500, gamerGroupsItem.Gamers.Min(g=>g.Score)==2500,
gamerGroupsItem.Gamers.Average(g=>g.Score)==4166.66666666667, gamerGroupsItem.Gamers.Sum(g=>g.Score)==12500
Id==1,Name==Name1,Gender==Male,TeamName==Team3,Score==4500
Id==5,Name==Name5,Gender==Male,TeamName==Team3,Score==2500
Id==6,Name==Name6,Gender==Male,TeamName==Team3,Score==5500
```



```
2.2. Lambda expression linq query -------------------
gamerGroupsItem2.Key==Team1, gamerGroupsItem2.Gamers.Count()==2,
gamerGroupsItem2.Gamers.Max(g=>g.Score)==6000, gamerGroupsItem2.Gamers.Min(g=>g.Score)==2000,
gamerGroupsItem2.Gamers.Average(g=>g.Score)==4000, gamerGroupsItem2.Gamers.Sum(g=>g.Score)==8000
Id==7,Name==Name7,Gender==Female,TeamName==Team1,Score==6000
Id==8,Name==Name8,Gender==Female,TeamName==Team1,Score==2000


gamerGroupsItem2.Key==Team2, gamerGroupsItem2.Gamers.Count()==3,
gamerGroupsItem2.Gamers.Max(g=>g.Score)==5000, gamerGroupsItem2.Gamers.Min(g=>g.Score)==3000,
gamerGroupsItem2.Gamers.Average(g=>g.Score)==3833.33333333333, gamerGroupsItem2.Gamers.Sum(g=>g.Score)==11500
Id==2,Name==Name2,Gender==Female,TeamName==Team2,Score==5000
Id==3,Name==Name3,Gender==Male,TeamName==Team2,Score==3500
Id==4,Name==Name4,Gender==Male,TeamName==Team2,Score==3000


gamerGroupsItem2.Key==Team3, gamerGroupsItem2.Gamers.Count()==3,
gamerGroupsItem2.Gamers.Max(g=>g.Score)==5500, gamerGroupsItem2.Gamers.Min(g=>g.Score)==2500,
gamerGroupsItem2.Gamers.Average(g=>g.Score)==4166.66666666667, gamerGroupsItem2.Gamers.Sum(g=>g.Score)==12500
Id==1,Name==Name1,Gender==Male,TeamName==Team3,Score==4500
Id==5,Name==Name5,Gender==Male,TeamName==Team3,Score==2500
Id==6,Name==Name6,Gender==Male,TeamName==Team3,Score==5500
```



```
3. GroupByIntoMultipleKeysSample() ==============
3.1. Lambda expression linq query -----------
gamerGroup.TeamName==Team1, gamerGroup.Gender==Female, gamerGroup.Gamers.Count()==2,
gamerGroup.Gamers.Min(g=>g.Score)==2000, gamerGroup.Gamers.Max(g=>g.Score)==6000,
gamerGroup.Gamers.Average(g=>g.Score)==4000, gamerGroup.Gamers.Sum(g=>g.Score)==8000
Id==7,Name==Name7,Gender==Female,TeamName==Team1,Score==6000
Id==8,Name==Name8,Gender==Female,TeamName==Team1,Score==2000

gamerGroup.TeamName==Team2, gamerGroup.Gender==Female, gamerGroup.Gamers.Count()==1,
gamerGroup.Gamers.Min(g=>g.Score)==5000, gamerGroup.Gamers.Max(g=>g.Score)==5000,
gamerGroup.Gamers.Average(g=>g.Score)==5000, gamerGroup.Gamers.Sum(g=>g.Score)==5000
Id==2,Name==Name2,Gender==Female,TeamName==Team2,Score==5000

gamerGroup.TeamName==Team2, gamerGroup.Gender==Male, gamerGroup.Gamers.Count()==2,
gamerGroup.Gamers.Min(g=>g.Score)==3000, gamerGroup.Gamers.Max(g=>g.Score)==3500,
gamerGroup.Gamers.Average(g=>g.Score)==3250, gamerGroup.Gamers.Sum(g=>g.Score)==6500
Id==3,Name==Name3,Gender==Male,TeamName==Team2,Score==3500
Id==4,Name==Name4,Gender==Male,TeamName==Team2,Score==3000

gamerGroup.TeamName==Team3, gamerGroup.Gender==Male, gamerGroup.Gamers.Count()==3,
gamerGroup.Gamers.Min(g=>g.Score)==2500, gamerGroup.Gamers.Max(g=>g.Score)==5500,
gamerGroup.Gamers.Average(g=>g.Score)==4166.66666666667, gamerGroup.Gamers.Sum(g=>g.Score)==12500
Id==1,Name==Name1,Gender==Male,TeamName==Team3,Score==4500
Id==5,Name==Name5,Gender==Male,TeamName==Team3,Score==2500
Id==6,Name==Name6,Gender==Male,TeamName==Team3,Score==5500
```



```
3.2. SQL like linq query -----------
gamerGroup.TeamName==Team1, gamerGroup.Gender==Female, gamerGroup.Gamers.Count()==2,
gamerGroup.Gamers.Min(g=>g.Score)==2000, gamerGroup.Gamers.Max(g=>g.Score)==6000,
gamerGroup.Gamers.Average(g=>g.Score)==4000, gamerGroup.Gamers.Sum(g=>g.Score)==8000
Id==7,Name==Name7,Gender==Female,TeamName==Team1,Score==6000
Id==8,Name==Name8,Gender==Female,TeamName==Team1,Score==2000

gamerGroup.TeamName==Team2, gamerGroup.Gender==Female, gamerGroup.Gamers.Count()==1,
gamerGroup.Gamers.Min(g=>g.Score)==5000, gamerGroup.Gamers.Max(g=>g.Score)==5000,
gamerGroup.Gamers.Average(g=>g.Score)==5000, gamerGroup.Gamers.Sum(g=>g.Score)==5000
Id==2,Name==Name2,Gender==Female,TeamName==Team2,Score==5000

gamerGroup.TeamName==Team2, gamerGroup.Gender==Male, gamerGroup.Gamers.Count()==2,
gamerGroup.Gamers.Min(g=>g.Score)==3000, gamerGroup.Gamers.Max(g=>g.Score)==3500,
gamerGroup.Gamers.Average(g=>g.Score)==3250, gamerGroup.Gamers.Sum(g=>g.Score)==6500
Id==3,Name==Name3,Gender==Male,TeamName==Team2,Score==3500
Id==4,Name==Name4,Gender==Male,TeamName==Team2,Score==3000

gamerGroup.TeamName==Team3, gamerGroup.Gender==Male, gamerGroup.Gamers.Count()==3,
gamerGroup.Gamers.Min(g=>g.Score)==2500, gamerGroup.Gamers.Max(g=>g.Score)==5500,
gamerGroup.Gamers.Average(g=>g.Score)==4166.66666666667, gamerGroup.Gamers.Sum(g=>g.Score)==12500
Id==1,Name==Name1,Gender==Male,TeamName==Team3,Score==4500
Id==5,Name==Name5,Gender==Male,TeamName==Team3,Score==2500
Id==6,Name==Name6,Gender==Male,TeamName==Team3,Score==5500
```