**执行左外部联接（C# 编程指南）**

左外部联接是这样一个联接：在其中返回第一个集合的每个元素，而无论该元素在第二个集合中是否具有相关元素。可以使用 LINQ 执行左通过对分组联接的结果调用方法 [DefaultIfEmpty<TSource>](https://msdn.microsoft.com/zh-cn/library/bb360179.aspx) 外部联接连接。

下面的示例演示如何对分组联接的结果调用 [DefaultIfEmpty<TSource>](https://msdn.microsoft.com/zh-cn/library/bb360179.aspx) 方法来执行左外部联接。

若要生成两个集合的左外部联接，第一步是使用分组联接执行内部联接。（有关此过程的说明，请参见[如何：执行内部联接（C# 编程指南）](https://msdn.microsoft.com/zh-cn/library/bb397941.aspx)。）在此示例中，Person 对象的列表内部联接到 Pet 对象列表。Person 对象的匹配 Pet.Owner。

第二步是在结果集内包含第一个（左）集合的每个元素，即使该元素在右集合中没有匹配的元素也是如此。这是通过对分组联接中的每个匹配元素序列调用 [DefaultIfEmpty<TSource>](https://msdn.microsoft.com/zh-cn/library/bb360179.aspx) 来实现的。在此示例中，[DefaultIfEmpty<TSource>](https://msdn.microsoft.com/zh-cn/library/bb360179.aspx) 对匹配 Pet 对象每个序列。方法返回一个包含单个集合，因此，如果匹配 Pet 对象序列为任何 Person 对象为空，从而确保的默认值每 Person 对象在结果集合表示。

| **https://wizardforcel.gitbooks.io/msdn-csharp/content/img/note.jpg 注意** |
| --- |
| 引用类型的默认值为 **null**;因此，该示例检查空在访问每个 Pet 集合的每个元素之前引用。 |

class Person

{

public string FirstName { get; set; }

public string LastName { get; set; }

}

class Pet

{

public string Name { get; set; }

public Person Owner { get; set; }

}

public static void LeftOuterJoinExample()

{

Person magnus = new Person { FirstName = "Magnus", LastName = "Hedlund" };

Person terry = new Person { FirstName = "Terry", LastName = "Adams" };

Person charlotte = new Person { FirstName = "Charlotte", LastName = "Weiss" };

Person arlene = new Person { FirstName = "Arlene", LastName = "Huff" };

Pet barley = new Pet { Name = "Barley", Owner = terry };

Pet boots = new Pet { Name = "Boots", Owner = terry };

Pet whiskers = new Pet { Name = "Whiskers", Owner = charlotte };

Pet bluemoon = new Pet { Name = "Blue Moon", Owner = terry };

Pet daisy = new Pet { Name = "Daisy", Owner = magnus };

// Create two lists.

List<Person> people = new List<Person> { magnus, terry, charlotte, arlene };

List<Pet> pets = new List<Pet> { barley, boots, whiskers, bluemoon, daisy };

var query = from person in people

join pet in pets on person equals pet.Owner into gj

from subpet in gj.DefaultIfEmpty()

select new { person.FirstName, PetName = (subpet == null ? String.Empty : subpet.Name) };

foreach (var v in query)

{

Console.WriteLine("{0,-15}{1}", v.FirstName + ":", v.PetName);

}

}

// This code produces the following output:

//

// Magnus: Daisy

// Terry: Barley

// Terry: Boots

// Terry: Blue Moon

// Charlotte: Whiskers

// Arlene: