# **[LINQ标准查询操作符（二）——Join、GroupJoin、GroupBy、Concat、](http://www.cnblogs.com/dotnetmvc/p/3680151.html)**

# **四、联接操作符**

联接是指将一个数据源对象与另一个数据源对象进行关联或者联合的操作。这两个数据源对象通过一个共同的值或者属性进行关联。

LINQ有两个联接操作符：Join和GroupJoin。

## **1. Join**

Join操作符类似于T-SQL中的inner join，它将两个数据源相联接，根据两个数据源中相等的值进行匹配。例如，可以将产品表与产品类别表相联接，得到产品名称和与其相对应的类别名称。以下的代码演示了这一点：

[IMG_256](http://www.cnblogs.com/dotnetmvc/p/javascript:void(0);" \o "复制代码)

//查询语法

var query =

(from p in db.Products

join c in db.Categories on p.CategoryID equals c.CategoryID

where p.CategoryID == 1

select new { p.ProductID, p.ProductName, c.CategoryID, c.CategoryName }).ToList();

生成的sql:

SELECT

[Extent1].[ProductID] AS [ProductID],

[Extent1].[ProductName] AS [ProductName],

[Extent2].[CategoryID] AS [CategoryID],

[Extent2].[CategoryName] AS [CategoryName]

FROM [dbo].[Products] AS [Extent1]

INNER JOIN [dbo].[Categories] AS [Extent2] ON [Extent1].[CategoryID] = [Extent2].[CategoryID]

WHERE (1 = [Extent1].[CategoryID]) AND ([Extent1].[CategoryID] IS NOT NULL)

//方法语法

var q =

db.Products

.Join

(

db.Categories,

p => p.CategoryID,

c => c.CategoryID,

(p, c) => new { p.ProductID, p.ProductName, c.CategoryID, c.CategoryName }

)

.Where(p => p.CategoryID == 1)

.ToList();

生成的sql:  
SELECT   
    [Extent1].[ProductID] AS [ProductID],   
    [Extent1].[ProductName] AS [ProductName],   
    [Extent2].[CategoryID] AS [CategoryID],   
    [Extent2].[CategoryName] AS [CategoryName]  
    FROM  [dbo].[Products] AS [Extent1]  
    INNER JOIN [dbo].[Categories] AS [Extent2] ON [Extent1].[CategoryID] = [Extent2].[CategoryID]  
    WHERE 1 = [Extent2].[CategoryID]

[IMG_257](http://www.cnblogs.com/dotnetmvc/p/javascript:void(0);" \o "复制代码)

以上代码为表述清晰加入了一个条件“where p.CategoryID == 1”，即仅返回产品类别ID为1的所有产品。

生成的sql语句略有不同。

## **2. GroupJoin**

GroupJoin操作符常应用于返回“主键对象-外键对象集合”形式的查询，例如“产品类别-此类别下的所有产品”。以下的代码演示了这一点：

[IMG_258](http://www.cnblogs.com/dotnetmvc/p/javascript:void(0);" \o "复制代码)

//查询语法

var query =

(from c in db.Categories

join p in db.Products on c.CategoryID equals p.CategoryID into r

select new

{

c.CategoryName,

Products = r

}).ToList();

//方法语法

var q =

db.Categories

.GroupJoin

(

db.Products,

c => c.CategoryID,

p => p.CategoryID,

(c, p) => new

{

c.CategoryName,

Products = p

}

)

.ToList();

生成的sql:

SELECT

[Project1].[CategoryID] AS [CategoryID],

[Project1].[CategoryName] AS [CategoryName],

[Project1].[C1] AS [C1],

[Project1].[ProductID] AS [ProductID],

[Project1].[ProductName] AS [ProductName],

[Project1].[SupplierID] AS [SupplierID],

[Project1].[CategoryID1] AS [CategoryID1],

[Project1].[QuantityPerUnit] AS [QuantityPerUnit],

[Project1].[UnitPrice] AS [UnitPrice],

[Project1].[UnitsInStock] AS [UnitsInStock],

[Project1].[UnitsOnOrder] AS [UnitsOnOrder],

[Project1].[ReorderLevel] AS [ReorderLevel],

[Project1].[Discontinued] AS [Discontinued]

FROM ( SELECT

[Extent1].[CategoryID] AS [CategoryID],

[Extent1].[CategoryName] AS [CategoryName],

[Extent2].[ProductID] AS [ProductID],

[Extent2].[ProductName] AS [ProductName],

[Extent2].[SupplierID] AS [SupplierID],

[Extent2].[CategoryID] AS [CategoryID1],

[Extent2].[QuantityPerUnit] AS [QuantityPerUnit],

[Extent2].[UnitPrice] AS [UnitPrice],

[Extent2].[UnitsInStock] AS [UnitsInStock],

[Extent2].[UnitsOnOrder] AS [UnitsOnOrder],

[Extent2].[ReorderLevel] AS [ReorderLevel],

[Extent2].[Discontinued] AS [Discontinued],

CASE WHEN ([Extent2].[ProductID] IS NULL) THEN CAST(NULL AS int) ELSE 1 END AS [C1]

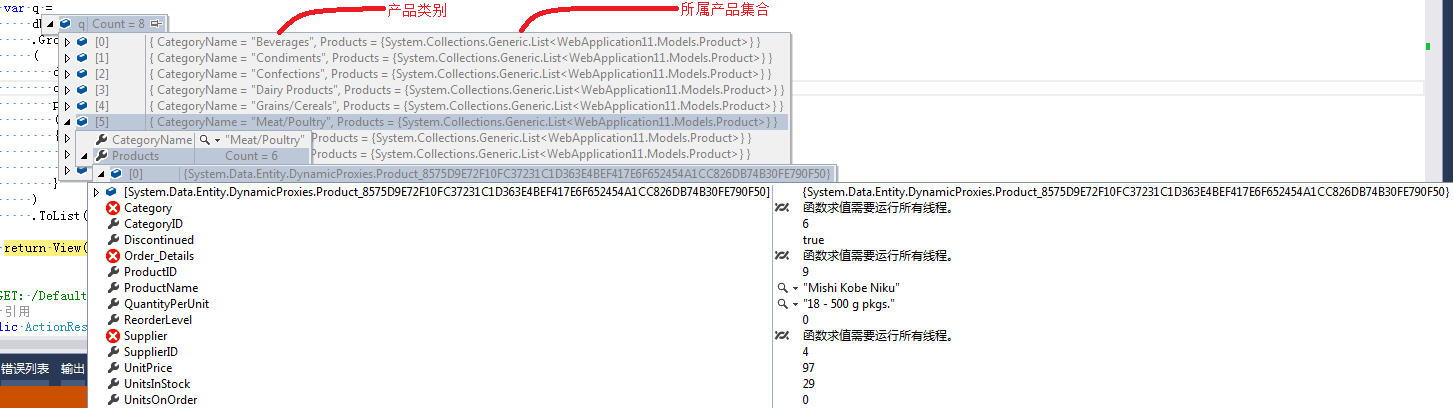
FROM [dbo].[Categories] AS [Extent1]

LEFT OUTER JOIN [dbo].[Products] AS [Extent2] ON [Extent1].[CategoryID] = [Extent2].[CategoryID]

) AS [Project1]

ORDER BY [Project1].[CategoryID] ASC, [Project1].[C1] ASC

[IMG_259](http://www.cnblogs.com/dotnetmvc/p/javascript:void(0);" \o "复制代码)

返回的结果为：  


# **五、分组操作符**

分组是根据一个特定的值将序列中的元素进行分组。LINQ只包含一个分组操作符：**GroupBy**。

下面的示例中使用了产品表，以CategoryID作为分组关键值，按照产品类别对产品进行了分组。

[IMG_261](http://www.cnblogs.com/dotnetmvc/p/javascript:void(0);" \o "复制代码)

//查询语法

var query =

(from p in db.Products

group p by p.CategoryID).ToList();

//方法语法

var q =

db.Products

.GroupBy(p => p.CategoryID)

.ToList();

生成的sql:

SELECT

[Project2].[C1] AS [C1],

[Project2].[CategoryID] AS [CategoryID],

[Project2].[C2] AS [C2],

[Project2].[ProductID] AS [ProductID],

[Project2].[ProductName] AS [ProductName],

[Project2].[SupplierID] AS [SupplierID],

[Project2].[CategoryID1] AS [CategoryID1],

[Project2].[QuantityPerUnit] AS [QuantityPerUnit],

[Project2].[UnitPrice] AS [UnitPrice],

[Project2].[UnitsInStock] AS [UnitsInStock],

[Project2].[UnitsOnOrder] AS [UnitsOnOrder],

[Project2].[ReorderLevel] AS [ReorderLevel],

[Project2].[Discontinued] AS [Discontinued]

FROM ( SELECT

[Distinct1].[CategoryID] AS [CategoryID],

1 AS [C1],

[Extent2].[ProductID] AS [ProductID],

[Extent2].[ProductName] AS [ProductName],

[Extent2].[SupplierID] AS [SupplierID],

[Extent2].[CategoryID] AS [CategoryID1],

[Extent2].[QuantityPerUnit] AS [QuantityPerUnit],

[Extent2].[UnitPrice] AS [UnitPrice],

[Extent2].[UnitsInStock] AS [UnitsInStock],

[Extent2].[UnitsOnOrder] AS [UnitsOnOrder],

[Extent2].[ReorderLevel] AS [ReorderLevel],

[Extent2].[Discontinued] AS [Discontinued],

CASE WHEN ([Extent2].[ProductID] IS NULL) THEN CAST(NULL AS int) ELSE 1 END AS [C2]

FROM (SELECT DISTINCT

[Extent1].[CategoryID] AS [CategoryID]

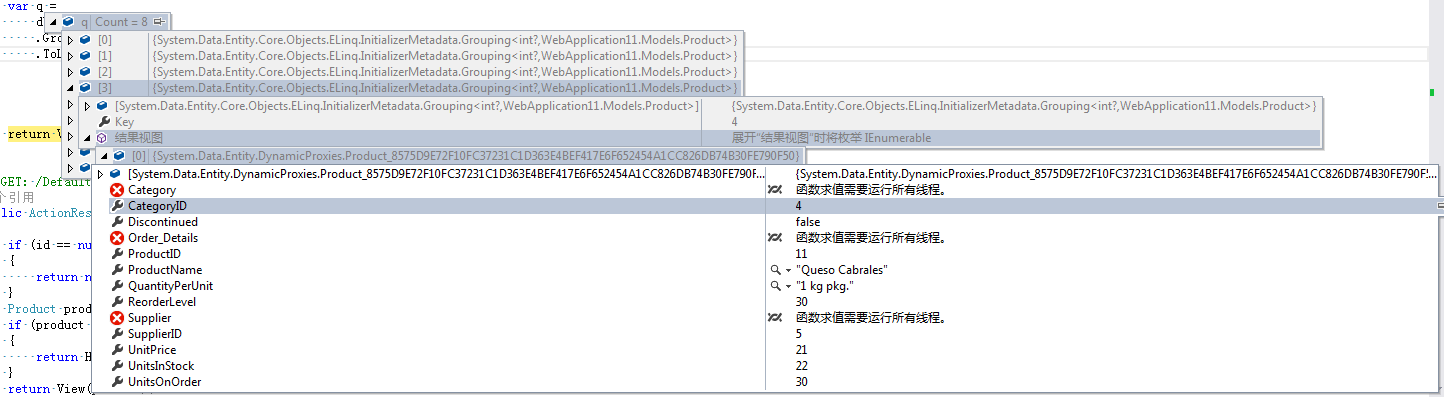
FROM [dbo].[Products] AS [Extent1] ) AS [Distinct1]

LEFT OUTER JOIN [dbo].[Products] AS [Extent2] ON ([Distinct1].[CategoryID] = [Extent2].[CategoryID]) OR (([Distinct1].[CategoryID] IS NULL) AND ([Extent2].[CategoryID] IS NULL))

) AS [Project2]

ORDER BY [Project2].[CategoryID] ASC, [Project2].[C2] ASC

[IMG_262](http://www.cnblogs.com/dotnetmvc/p/javascript:void(0);" \o "复制代码)



执行GroupBy得到的序列中包含的元素类型为IGrouping<TKey, T>，其Key属性代表了分组时使用的关键值，遍历IGrouping<TKey, T>元素可以读取到每一个T类型。在此示例中，对应的元素类型为IGrouping<int, Products>，其Key属性即为类别ID，遍历它可以读取到每一个产品对象。

# **六、串联操作符**

串联是一个将两个集合联接在一起的过程。在LINQ中，这个过程通过**Concat**操作符来实现。

在下面的示例中，将会把类别名称串联在产品名称之后：

[IMG_264](http://www.cnblogs.com/dotnetmvc/p/javascript:void(0);" \o "复制代码)

//方法语法

var q =

db.Products

.Select(p => p.ProductName)

.Concat

(

db.Categories.Select(c => c.CategoryName)

)

.ToList();

生成的sql:

SELECT

[UnionAll1].[ProductName] AS [C1]

FROM (SELECT

[Extent1].[ProductName] AS [ProductName]

FROM [dbo].[Products] AS [Extent1]

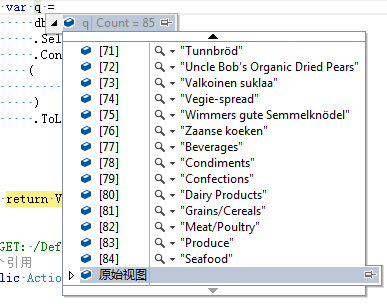
UNION ALL

SELECT

[Extent2].[CategoryName] AS [CategoryName]

FROM [dbo].[Categories] AS [Extent2]) AS [UnionAll1]

[IMG_265](http://www.cnblogs.com/dotnetmvc/p/javascript:void(0);" \o "复制代码)

返回结果77+8=85  


http://www.cnblogs.com/dotnetmvc/p/3680151.html