# [LINQ标准查询操作符（一）——select、SelectMany、Where、OrderBy、OrderByDescending、ThenBy、ThenByDescending和Reverse](http://www.cnblogs.com/dotnetmvc/p/3679779.html)

### 一、投影操作符

#### 1. Select

Select操作符对单个序列或集合中的值进行投影。下面的示例中使用select从序列中返回Employee表的所有列：

[IMG_256](http://www.cnblogs.com/dotnetmvc/p/javascript:void(0);)

//查询语法

var query =

from e in db.Employees

where e.FirstName.StartsWith("M")

select e;  
生成的sql:  
SELECT   
    [Extent1].[EmployeeID] AS [EmployeeID],   
    [Extent1].[LastName] AS [LastName],   
    [Extent1].[FirstName] AS [FirstName],   
    [Extent1].[Title] AS [Title],   
    [Extent1].[TitleOfCourtesy] AS [TitleOfCourtesy],   
    [Extent1].[BirthDate] AS [BirthDate],   
    [Extent1].[HireDate] AS [HireDate],   
    [Extent1].[Address] AS [Address],   
    [Extent1].[City] AS [City],   
    [Extent1].[Region] AS [Region],   
    [Extent1].[PostalCode] AS [PostalCode],   
    [Extent1].[Country] AS [Country],   
    [Extent1].[HomePhone] AS [HomePhone],   
    [Extent1].[Extension] AS [Extension],   
    [Extent1].[Photo] AS [Photo],   
    [Extent1].[Notes] AS [Notes],   
    [Extent1].[ReportsTo] AS [ReportsTo],   
    [Extent1].[PhotoPath] AS [PhotoPath]  
    FROM [dbo].[Employees] AS [Extent1]  
    WHERE [Extent1].[FirstName] LIKE N'M%'

[IMG_257](http://www.cnblogs.com/dotnetmvc/p/javascript:void(0);)

[IMG_258](http://www.cnblogs.com/dotnetmvc/p/javascript:void(0);)

//方法语法

var q =

db.Employees

.Where(e => e.FirstName.StartsWith("M"))

.Select(e => e);  
  
生成的sql:  
SELECT   
    [Extent1].[EmployeeID] AS [EmployeeID],   
    [Extent1].[LastName] AS [LastName],   
    [Extent1].[FirstName] AS [FirstName],   
    [Extent1].[Title] AS [Title],   
    [Extent1].[TitleOfCourtesy] AS [TitleOfCourtesy],   
    [Extent1].[BirthDate] AS [BirthDate],   
    [Extent1].[HireDate] AS [HireDate],   
    [Extent1].[Address] AS [Address],   
    [Extent1].[City] AS [City],   
    [Extent1].[Region] AS [Region],   
    [Extent1].[PostalCode] AS [PostalCode],   
    [Extent1].[Country] AS [Country],   
    [Extent1].[HomePhone] AS [HomePhone],   
    [Extent1].[Extension] AS [Extension],   
    [Extent1].[Photo] AS [Photo],   
    [Extent1].[Notes] AS [Notes],   
    [Extent1].[ReportsTo] AS [ReportsTo],   
    [Extent1].[PhotoPath] AS [PhotoPath]  
    FROM [dbo].[Employees] AS [Extent1]  
    WHERE [Extent1].[FirstName] LIKE N'M%'

[IMG_259](http://www.cnblogs.com/dotnetmvc/p/javascript:void(0);)

[IMG_260](http://www.cnblogs.com/dotnetmvc/p/javascript:void(0);)

可以简写为：

var qq =

db.Employees

.Where(e => e.FirstName.StartsWith("M"))

.ToList();

[IMG_261](http://www.cnblogs.com/dotnetmvc/p/javascript:void(0);)

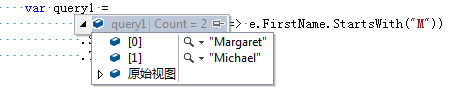
=========================================================

当然，你也可以返回单个列，例如：

[IMG_262](http://www.cnblogs.com/dotnetmvc/p/javascript:void(0);)

            var query =  
                     from e in db.Employees  
                     where e.FirstName.StartsWith("M")  
                     select e.FirstName;  
            var query1 =  
                    db.Employees.Where(e => e.FirstName.StartsWith("M"))  
                    .Select(e => e.FirstName);  
  
生成的sql:  
SELECT   
    [Extent1].[FirstName] AS [FirstName]  
    FROM [dbo].[Employees] AS [Extent1]  
    WHERE [Extent1].[FirstName] LIKE N'M%'

[IMG_263](http://www.cnblogs.com/dotnetmvc/p/javascript:void(0);)

 如果像上面这样，只查询单列，则返回类似于无名称的数组，并不能用于生成Json。  


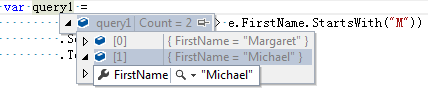
如果想返回单列匿名类，应

var query1 =

db.Employees.Where(e => e.FirstName.StartsWith("M"))

.Select(e => new { e.FirstName })

.ToList();

返回结果为：  


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

你也可以返回序列中的某几列，例如：

[IMG_266](http://www.cnblogs.com/dotnetmvc/p/javascript:void(0);)

var query =

(from e in db.Employees

where e.FirstName.StartsWith("M")

select new

{

e.FirstName,

e.LastName,

e.Title

}).ToList();

var query1 =

db.Employees.Where(e => e.FirstName.StartsWith("M"))

.Select(e => new

{

e.FirstName,

e.LastName,

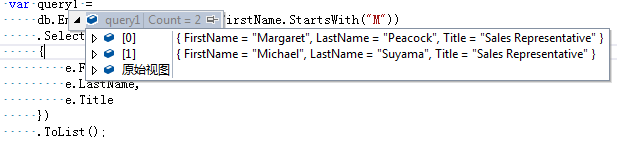
e.Title

})

.ToList();  
生成的sql:  
SELECT   
    1 AS [C1],   
    [Extent1].[FirstName] AS [FirstName],   
    [Extent1].[LastName] AS [LastName],   
    [Extent1].[Title] AS [Title]  
    FROM [dbo].[Employees] AS [Extent1]  
    WHERE [Extent1].[FirstName] LIKE N'M%'

[IMG_267](http://www.cnblogs.com/dotnetmvc/p/javascript:void(0);)

查询多列，返回的是一个匿名类



## 2. SelectMany

SelectMany操作符提供了将多个from子句组合起来的功能，它将每个对象的结果合并成单个序列。下面是一个示例：

[IMG_269](http://www.cnblogs.com/dotnetmvc/p/javascript:void(0);)

var query =

(from e in db.Employees

from o in e.Orders

select o).ToList();

//方法语法

var q =

db.Employees

.SelectMany(e => e.Orders)

.ToList();

生成的sql:

SELECT

[Extent1].[OrderID] AS [OrderID],

[Extent1].[CustomerID] AS [CustomerID],

[Extent1].[EmployeeID] AS [EmployeeID],

[Extent1].[OrderDate] AS [OrderDate],

[Extent1].[RequiredDate] AS [RequiredDate],

[Extent1].[ShippedDate] AS [ShippedDate],

[Extent1].[ShipVia] AS [ShipVia],

[Extent1].[Freight] AS [Freight],

[Extent1].[ShipName] AS [ShipName],

[Extent1].[ShipAddress] AS [ShipAddress],

[Extent1].[ShipCity] AS [ShipCity],

[Extent1].[ShipRegion] AS [ShipRegion],

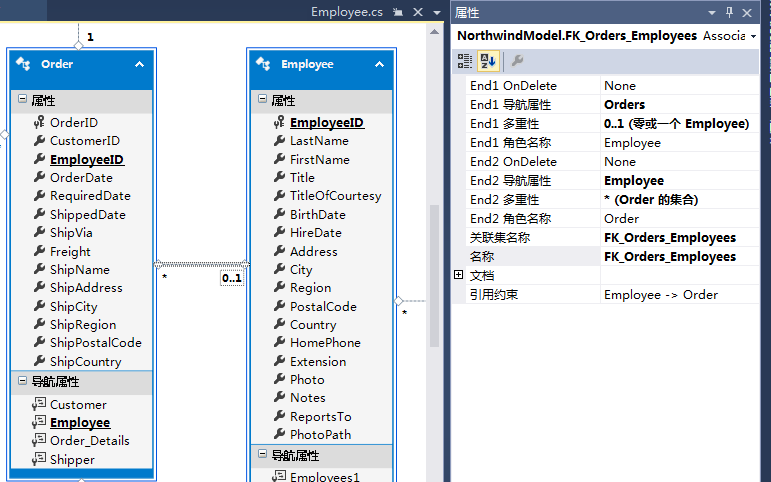
[Extent1].[ShipPostalCode] AS [ShipPostalCode],

[Extent1].[ShipCountry] AS [ShipCountry]

FROM [dbo].[Orders] AS [Extent1]

WHERE [Extent1].[EmployeeID] IS NOT NULL

[IMG_270](http://www.cnblogs.com/dotnetmvc/p/javascript:void(0);)



场景：Employee与Order为0或1对多的关系，也就是Order的EmployeeID可以为null

从生成的sql语句也可以看到，只查询出了Order的EmployeeID不为null的所有Order记录。

# 二、限制操作符

Where是限制操作符，它将过滤标准应用在序列上，按照提供的逻辑对序列中的数据进行过滤。

Where操作符不启动查询的执行。当开始对序列进行遍历时查询才开始执行，此时过滤条件将被应用到查询中。Where操作符的使用方法已经在第一节中出现过，这里不再冗述。

# 三、排序操作符

排序操作符，包括OrderBy、OrderByDescending、ThenBy、ThenByDescending和Reverse，提供了升序或者降序排序。

## 1. OrderBy

OrderBy操作符将序列中的元素按照升序排列。下面的示例演示了这一点：

[IMG_272](http://www.cnblogs.com/dotnetmvc/p/javascript:void(0);)

//查询语法

var query =

(from e in db.Employees

orderby e.FirstName

select e).ToList();

//方法语法

var q =

db.Employees

.OrderBy(e => e.FirstName)

.ToList();

生成的sql:

SELECT

[Extent1].[EmployeeID] AS [EmployeeID],

[Extent1].[LastName] AS [LastName],

[Extent1].[FirstName] AS [FirstName],

[Extent1].[Title] AS [Title],

[Extent1].[TitleOfCourtesy] AS [TitleOfCourtesy],

[Extent1].[BirthDate] AS [BirthDate],

[Extent1].[HireDate] AS [HireDate],

[Extent1].[Address] AS [Address],

[Extent1].[City] AS [City],

[Extent1].[Region] AS [Region],

[Extent1].[PostalCode] AS [PostalCode],

[Extent1].[Country] AS [Country],

[Extent1].[HomePhone] AS [HomePhone],

[Extent1].[Extension] AS [Extension],

[Extent1].[Photo] AS [Photo],

[Extent1].[Notes] AS [Notes],

[Extent1].[ReportsTo] AS [ReportsTo],

[Extent1].[PhotoPath] AS [PhotoPath]

FROM [dbo].[Employees] AS [Extent1]

ORDER BY [Extent1].[FirstName] ASC

[IMG_273](http://www.cnblogs.com/dotnetmvc/p/javascript:void(0);)

因为未使用Select，所以返回的为动态包装类。  
这里可以使用OrderBy的重载方法OrderBy(Func<T,TKey>,IComparer<Tkey>)来指定序列的排序方式。

## 2. OrderByDescending

OrderByDescending操作符将序列中的元素按照降序排列。用法与OrderBy相同，这里不再演示。

## 3. ThenBy

ThenBy操作符实现按照次关键字对序列进行升序排列。此操作符的查询语法与方法语法略有不同，以下代码演示了这一点：

[IMG_274](http://www.cnblogs.com/dotnetmvc/p/javascript:void(0);)

//查询语法

var query =

(from e in db.Employees

orderby e.FirstName, e.LastName

select e).ToList();

//方法语法

var q =

db.Employees

.OrderBy(e => e.FirstName)

.ThenBy(e => e.LastName)

.ToList();

生成的sql:

SELECT

[Extent1].[EmployeeID] AS [EmployeeID],

[Extent1].[LastName] AS [LastName],

[Extent1].[FirstName] AS [FirstName],

[Extent1].[Title] AS [Title],

[Extent1].[TitleOfCourtesy] AS [TitleOfCourtesy],

[Extent1].[BirthDate] AS [BirthDate],

[Extent1].[HireDate] AS [HireDate],

[Extent1].[Address] AS [Address],

[Extent1].[City] AS [City],

[Extent1].[Region] AS [Region],

[Extent1].[PostalCode] AS [PostalCode],

[Extent1].[Country] AS [Country],

[Extent1].[HomePhone] AS [HomePhone],

[Extent1].[Extension] AS [Extension],

[Extent1].[Photo] AS [Photo],

[Extent1].[Notes] AS [Notes],

[Extent1].[ReportsTo] AS [ReportsTo],

[Extent1].[PhotoPath] AS [PhotoPath]

FROM [dbo].[Employees] AS [Extent1]

ORDER BY [Extent1].[FirstName] ASC, [Extent1].[LastName] ASC

[IMG_275](http://www.cnblogs.com/dotnetmvc/p/javascript:void(0);)

## 4. ThenByDescending

ThenByDescending操作符实现按照次关键字对序列进行降序排列。此操作符的查询语法与方法语法略有不同，以下代码演示了这一点：

[IMG_276](http://www.cnblogs.com/dotnetmvc/p/javascript:void(0);)

//查询语法

var query =

from e in db.Employees

orderby e.FirstName,e.LastName descending

select e;

//方法语法

var q =

db.Employees

.OrderBy(e => e.FirstName)

.ThenByDescending(e => e.LastName)

.Select(e => e);

foreach (var item in query)

{

Console.WriteLine(item.FirstName);

}

生成的sql:  
  
SELECT   
    [Extent1].[EmployeeID] AS [EmployeeID],   
    [Extent1].[LastName] AS [LastName],   
    [Extent1].[FirstName] AS [FirstName],   
    [Extent1].[Title] AS [Title],   
    [Extent1].[TitleOfCourtesy] AS [TitleOfCourtesy],   
    [Extent1].[BirthDate] AS [BirthDate],   
    [Extent1].[HireDate] AS [HireDate],   
    [Extent1].[Address] AS [Address],   
    [Extent1].[City] AS [City],   
    [Extent1].[Region] AS [Region],   
    [Extent1].[PostalCode] AS [PostalCode],   
    [Extent1].[Country] AS [Country],   
    [Extent1].[HomePhone] AS [HomePhone],   
    [Extent1].[Extension] AS [Extension],   
    [Extent1].[Photo] AS [Photo],   
    [Extent1].[Notes] AS [Notes],   
    [Extent1].[ReportsTo] AS [ReportsTo],   
    [Extent1].[PhotoPath] AS [PhotoPath]  
    FROM [dbo].[Employees] AS [Extent1]  
    ORDER BY [Extent1].[FirstName] ASC, [Extent1].[LastName] DESC

[IMG_277](http://www.cnblogs.com/dotnetmvc/p/javascript:void(0);)

## 5. Reverse

Reverse将会把序列中的元素按照从后到前的循序反转。需要注意的是，Reverse方法的返回值是void，以下代码演示了这一点

[IMG_278](http://www.cnblogs.com/dotnetmvc/p/javascript:void(0);)

//方法语法

var q =

db.Employees

.Select(e => e.FirstName)

.ToList();

q.Reverse();

[IMG_279](http://www.cnblogs.com/dotnetmvc/p/javascript:void(0);)

http://www.cnblogs.com/dotnetmvc/p/3679779.html