## Linq查询

语言集成查询 (LINQ) 是一系列直接将查询功能集成到 C# 语言的技术统称。LINQ可以查询SQL Server 数据库、XML 文档、ADO.NET 数据集以及支持 [IEnumerable](https://docs.microsoft.com/zh-cn/dotnet/api/system.collections.ienumerable) 或泛型 [IEnumerable<T>](https://docs.microsoft.com/zh-cn/dotnet/api/system.collections.generic.ienumerable-1) 接口的任何对象集合。

### Linq查询操作的三个部分

|  |  |
| --- | --- |
| （1）获取数据源。  （2）创建查询。  （3）执行查询。 |  |

例子

|  |
| --- |
| class Porgram  {  static void Main()  {  // 查询3步骤  // 1. 准备数据源  int[] numbers = new int[7] { 0, 1, 2, 3, 4, 5, 6 };  // 2. 创建查询字符串  // numQuery 变量是IEnumerable<int>类型  var numQuery =  from num in numbers  where (num % 2) == 0  select num;  // 3.执行查询  foreach (int num in numQuery)  {  Console.Write("{0,1} ", num);  }  }  } |

### LINQ 中的查询语法和方法语法

|  |
| --- |
| class Porgram  {  static void Main()  {  int[] numbers = { 5, 10, 8, 3, 6, 12};  //查询语法:  IEnumerable<int> numQuery1 =  from num in numbers  where num % 2 == 0  orderby num  select num;  //方法语法:  IEnumerable<int> numQuery2 = numbers.Where(num => num % 2 == 0).OrderBy(n => n);  foreach (int i in numQuery1)  {  Console.Write(i + " ");  }  Console.WriteLine(System.Environment.NewLine);  foreach (int i in numQuery2)  {  Console.Write(i + " ");  }    // Keep the console open in debug mode.  Console.WriteLine();  Console.WriteLine("按任意键退出");  Console.ReadKey();  }  }  /\*  Output:  6 8 10 12  6 8 10 12  \*/ |

### lambda表达式

Lambda 表达式是一种可用于创建 委托 或 表达式目录树 类型的 匿名函数 。 通过使用 lambda 表达式，可以写入可作为参数传递或作为函数调用值返回的本地函数。 Lambda 表达式对于编写 LINQ 查询表达式特别有用。

语法

|  |
| --- |
| (参数列表) => {表达式/语句组} |

给定参数返回表达式值或执行语句组

## 数据查询

### 使用查询查找实体

DbSet 和 IDbSet 可实现 IQueryable，因此可用作针对数据库编写 LINQ 查询的起点。

出现以下情况时，会针对数据库执行查询：

* 查询由 foreach (C#)语句枚举。
* 查询由集合操作（如 ToArray、ToDictionary 或 ToList）枚举。
* 在查询最外部指定了 LINQ 运算符，例如 First 或 Any。
* 调用了以下方法：DbSet 上的 Load 扩展方法、DbEntityEntry.Reload 和 Database.ExecuteSqlCommand。

从数据库返回结果时，上下文中不存在的对象会附加到上下文。 如果某个对象已存在于上下文中，则不返回现有对象。如果查询未从数据库返回任何行，则结果将是空集合，而不是 NULL。

例子 查找包含给定名称name的所有博客

|  |
| --- |
| public List<Blog> Query(string name)  {  using (var db = new BloggingContext())  {  // 查询所有包含字符串name的博客  var blogs = from b in db.Blogs  where b.Name.Contains(name)  select b;  return blogs.ToList();  }  } |

例子 查找等于给定名称的博客

|  |
| --- |
| public Blog QueryABlog(string name)  {  using (var db = new BloggingContext())  {  // 查询所有包含字符串name的博客  var blogs = from b in db.Blogs  where b.Name == name  select b;  return blogs.FirstOrDefault();  }  } |

### 使用主键查找实体

DbSet 上的 Find 方法使用主键值来尝试查找由上下文跟踪的实体。 如果在上下文中未找到实体，则会向数据库发送查询以在其中查找实体。 如果未在上下文中或数据库中找到实体，则返回 NULL。

Find 与使用查询有两个重要区别：

只有未在上下文中找到具有给定键的实体时，才会往返数据库。

Find 会返回处于“已添加”状态的实体。 即 Find 会返回已添加到上下文但尚未保存到数据库的实体。

例子

|  |  |
| --- | --- |
| |  | | --- | | public Blog Query(int id)  {  using (var db = new BloggingContext())  {  return db.Blogs.Find(id);  }  } | |

扩展阅读

<https://docs.microsoft.com/zh-cn/dotnet/csharp/tutorials/working-with-linq>

<https://docs.microsoft.com/zh-cn/dotnet/csharp/programming-guide/concepts/linq/>