1. 在使用Linq to Sql的时候，进行两个表的左连接的时候要注意defaultifempty的使用，
这个函数本来的意思即是：如果为空则使用默认值代替，默认值为 NULL ，当
然也可以使用defaultifempty的另一个重载指定默认。

2. Linq中使用Left Join

```
Create table Student(
    ID int identity(1,1) primary key,
    [Name] nvarchar(50) not null
)
Create Table Book(
    ID int identity(1,1) primary key,
    [Name] nvarchar(50)not null,
    StudentID int not null
)

insert into Student values('张三')
insert into Student values('李四')
insert into Student values('王五')
select * from student

--张三借的书
insert into Book values('红楼',1)
insert into Book values('大话红楼',1)
--李四借的书
insert into Book values('三国',2)
--王五没借书

--一本错误的记录
insert into Book values('错误时怎样练成的',111)

--左连接
select s.name,b.name from student as s
left join Book as b on s.id=b.studentid
--右连接
select s.name,b.name from student as s
right join Book as b on s.id=b.studentid
```

要用Linq实现左连接，写法如下

```
    DataClasses1DataContext db = new DataClasses1DataContext();
    var leftJoinSql = from student in db.Student
                      join book in db.Book on student.ID equals book.StudentID into temp
                      from tt in temp.DefaultIfEmpty()
                      select new
                      {
                           sname= student.Name,
                           bname = tt==null?"":tt.Name//这里主要第二个集合有可能为空。需要判断
                      };
```

用Linq实现右连接，写法如下

```
    DataClasses1DataContext db=new DataClasses1DataContext();
    var rightJoinSql = from book in db.Book
                       join stu in db.Student on book.StudentID equals stu.ID into joinTemp
                       from tmp in joinTemp.DefaultIfEmpty()
                       select new {
                            sname=tmp==null?"":tmp.Name,
                            bname=book.Name
                       };
```

3. Linq中GroupBy方法

```
public class StudentScore {
    public int ID { set; get; }
    public string Name { set; get; }
    public string Course { set; get; }
    public int Score { set; get; }
```

```
    public string Term { set; get; }
}
List<StudentScore> lst = new List<StudentScore>() {
    new StudentScore(){ID=1,Name="张三",Term="第一学期",Course="Math",Score=80},
    new StudentScore(){ID=1,Name="张三",Term="第一学期",Course="Chinese",Score=90},
    new StudentScore(){ID=1,Name="张三",Term="第一学期",Course="English",Score=70},
    new StudentScore(){ID=2,Name="李四",Term="第一学期",Course="Math",Score=60},
    new StudentScore(){ID=2,Name="李四",Term="第一学期",Course="Chinese",Score=70},
    new StudentScore(){ID=2,Name="李四",Term="第一学期",Course="English",Score=30},
    new StudentScore(){ID=3,Name="王五",Term="第一学期",Course="Math",Score=100},
    new StudentScore(){ID=3,Name="王五",Term="第一学期",Course="Chinese",Score=80},
    new StudentScore(){ID=3,Name="王五",Term="第一学期",Course="English",Score=80},
    new StudentScore(){ID=4,Name="赵六",Term="第一学期",Course="Math",Score=90},
    new StudentScore(){ID=4,Name="赵六",Term="第一学期",Course="Chinese",Score=80},
    new StudentScore(){ID=4,Name="赵六",Term="第一学期",Course="English",Score=70},
    new StudentScore(){ID=1,Name="张三",Term="第二学期",Course="Math",Score=100},
    new StudentScore(){ID=1,Name="张三",Term="第二学期",Course="Chinese",Score=80},
    new StudentScore(){ID=1,Name="张三",Term="第二学期",Course="English",Score=70},
    new StudentScore(){ID=2,Name="李四",Term="第二学期",Course="Math",Score=90},
    new StudentScore(){ID=2,Name="李四",Term="第二学期",Course="Chinese",Score=50},
    new StudentScore(){ID=2,Name="李四",Term="第二学期",Course="English",Score=80},
    new StudentScore(){ID=3,Name="王五",Term="第二学期",Course="Math",Score=90},
    new StudentScore(){ID=3,Name="王五",Term="第二学期",Course="Chinese",Score=70},
    new StudentScore(){ID=3,Name="王五",Term="第二学期",Course="English",Score=80},
    new StudentScore(){ID=4,Name="赵六",Term="第二学期",Course="Math",Score=70},
    new StudentScore(){ID=4,Name="赵六",Term="第二学期",Course="Chinese",Score=60},
    new StudentScore(){ID=4,Name="赵六",Term="第二学期",Course="English",Score=70},
};
//分组，根据姓名，统计Sum的分数，统计结果放在匿名对象中。
    var studentSumScore_1 = (   from l in lst
                                group l by l.Name into grouped
                                orderby grouped.Sum(m => m.Score)
                                select new { Name = grouped.Key, Scores = grouped.Sum(m => m.Score) }
                            ).ToList();
        foreach (var l in studentSumScore_1)
        {
            Console.WriteLine("{0}:总分{1}", l.Name, l.Scores);
        }


4.  distinct

4.1.    使用GroupBy：对需要Distinct的字段进行分组，取组内的第一条记录这样结果就是Distinct的数
据了。
        Console.WriteLine("Distinct1 By: A");
        var query1 = from e in User.GetData()
                     group e by new { e.A } into g
                     select g.FirstOrDefault();
        foreach (var u in query1)
            Console.WriteLine(u.ToString());

4.2     使用Distinct()扩展方法：需要实现IEqualityComparer接口。
        class UserCompare : IEqualityComparer<User>
        {
            public bool Equals(User x, User y)
            {
                return (x.A == y.A && x.B == y.B);
            }
            public int GetHashCode(User obj)
            {
                // return obj.GetHashCode();
                return obj.ToString().ToLower().GetHashCode();
            }
        }
        Console.WriteLine("Distinct2 By: A,B");
        var compare = new UserCompare();
        var query2 = User.GetData().Distinct(compare);
        foreach (var u in query2)
```

```
                Console.WriteLine(u.ToString());
```