　<hibernate-mapping>标签是NHibernate映射文件的根节点。

　　<hibernate-mapping                                     
　      schema="schemaName"                         数据库schema名称。

         default-cascade="none|save-update"      默认的级联风格，(可选 - 默认为 none):。

         auto-import="true|false"   　　　　　      指定我们在使用查询语句的时候是否可以使用非全限定名。

         assembly="AssemblyName"

         namespace="Namespace"　　　　　        指定映射文件中的类的应用程序集名称和其所在的名称空间名，用来生成类的非全限定名。

　　/>

**2.2 class**

　　<class>标签是定义一个持久化类的。

　　<class

name="ClassName" 持久化类的类名，这里可以是类的全名。

table="tableName" 对应的数据库表名。

discriminator-value="discriminator\_value" 辨别值，一个用于区分不同的子类的值，在多态行为时使用（在后面继承映射的课程中会讲到）。

mutable="true|false" 表明该类的实例可变。

schema="owner" 覆盖在根元素中指定的schema名字。

proxy="ProxyInterface" 指定一个接口，在延迟装载时作为代理使用。你可以在这里使用该类自己的名字。

dynamic-update="true|false" 指定用于UPDATE 的SQL将会在运行时动态生成，并且只更新那些改变过的字段。

dynamic-insert="true|false" 指定用于 INSERT的 SQL 将会在运行时动态生成，并且只包含那些非空值字段。

select-before-update="true|false" 指定NHibernate除非确定对象的确被修改了，UPDATE操作。

polymorphism="implicit|explicit" 界定是隐式还是显式的使用查询多态。

where="arbitrary sql where condition" 指定一个附加的SQL WHERE 条件，在抓取这个类的对象时会一直增加这个条件。

persister="PersisterClass" 指定一个定制的 IClassPersister。

batch-size="N" 指定一个用于根据标识符抓取实例时使用的"batch size"（批次抓取数量），默认值为1。

optimistic-lock="none|version|dirty|all" 乐观锁定，决定乐观锁定的策略。

lazy="true|false" 是否启用延迟加载。

abstract="true|false" 是否为抽象类。

　　/>

**2.3 id**

　　<id>标签定义了该属性到数据库表主键字段的映射。

　　<id

　　  name="PropertyName"                                   标识属性的名字。  
        type="typename"                                          NHibernate类型的名字  
        column="column\_name"                                主键字段的名字。  
        unsaved-value="any|none|null|id\_value"         一个特定的标识属性值，用来标志该实例是刚刚创建的，尚未保存。  
        access="field|property|nosetter|ClassName"    NHibernate用来访问属性值的策略。  
        <generator class="generatorClass"/>  
　　</id>

generator：主键生成策略。NHibernate提供了以下几种生成策略：

|  |  |  |
| --- | --- | --- |
| **名称** | **说明** | **是否常用** |
| **increment** | 用于为int类型生成 唯一标识。只有在没有其他进程往同一张表中插入数据时才能使用 | N |
| **identity** | 对DB2,MySQL, MS SQL Server, Sybase和HypersonicSQL的内置标识字段提供支持。数据库返回的主键值 返回的标识符是int类型的。 | N |
| **sequence** | 在DB2,PostgreSQL, Oracle, SAP DB, McKoi中使用序列（sequence)， 而在Interbase中使用生成器(generator)。返回的标识符是int类型的。 | Y |
| **seqhilo** | 使用一个高/低位算法来高效的生成int类型的标识符，给定一个数据库序列（sequence)的名字。 | N |
| **uuid.hex** | 用一个System.Guid的ToString()方法法生成字符串类型的标识符， 字符串的长度由format参数定义。 | N |
| **uuid.string** | 用一个新的System.Guid实例的byte[]转化为字符串作为标示符。 | N |
| **guid** | 使用新的System.Guid实例作为标示符。 |  |
| **guid.comb** | 使用Jimmy Nilsson的算法生成一个新的System.Guid标示符。 |  |
| **native** | 根据底层数据库的能力选择identity, sequence 或者hilo中的一个。 | Y |
| **assigned** | 让应用程序在 Save()之前为对象分配一个标示符。 | Y |
| **foreign** | 使用另外一个相关联的对象的标识符。通常和<one-to-one>联合起来使用。 | Y |

**2.4 composite-id**

　　<composite-id>为联合主键。

　　<composite-id                                                          
        name="PropertyName"                                                              
        class="ClassName"                                               联合主键类的类名。  
        unsaved-value="any|none"                                     
        access="field|property|nosetter|ClassName">  
  
        <key-property name="PropertyName" type="typename" column="column\_name"/>             联合主键的属性  
        <key-many-to-one name="PropertyName class="ClassName" column="column\_name"/>      联合主键多对一属性  
        ......  
　　</composite-id>

　<many-to-one

name="PropertyName" 属性名

column="column\_name" 数据库字段名

class="ClassName" 关联的类的名字

cascade="all|none|save-update|delete" 级联关系

fetch="join|select" 抓取

update="true|false"

insert="true|false"

property-ref="PropertyNameFromAssociatedClass" 指定关联类的一个属性，这个属性将会和本外键相对应

access="field|property|nosetter|ClassName"

unique="true|false" 是否生成一个唯一约束

optimistic-lock="true|false"

not-found="ignore|exception" 指定外键引用的数据不存在时如何处理

　　/>

<one-to-one>默认的cascade为all

Session 对象的3中状态：

1. transient
2. persistent
3. detached

在任何时候都要使用事务，即使是在读取、查询数据的时候，为什么呢？因为你不清楚数据库什么时候操作失败，如何恢复原来数据。

通过组件可以改善我们的对象模型

Component

NHibernate中映射关系是NHibernate中的亮点

NH集合

Bag：对象集合，每个元素可以重复。例如{1,2,2,6,0,0}，在.Net中相当于IList或者IList<T>实现。

Set：对象集合，每个元素必须唯一。例如{1,2,5,6}，在.Net中相当于ISet或者ISet<T>实现，Iesi.Collections.dll程序集提供ISet集合。

List：整数索引对象集合，每个元素可以重复。例如{{1,"YJingLee"},{2,"CnBlogs"},{3,"LiYongJing"}}，在.Net中相当于ArraryList或者List<T>实现。

Map：键值对集合。例如{{"YJingLee",5},{"CnBlogs",7},{"LiYongJing",6}}，在.Net中相当于HashTable或者IDictionary<Tkey,TValue>实现。

## 关联:

主键关联

唯一外键关联

　“一对多”双向关联映射指的是：在“一”的这端（“Class”类这端）包含“多”的属性（“Students”）；在“多”的这端包含“一”的属性（“Class”属性）。这样两个类构成的循环引用就是双向关联映射。

“inverse”属性的意思是反转，就是将操作权交给关联关系的另一点

设置cascade属性为all，用于从一端保存数据。