Hibernate的关联映射之1-N关联映射

# 单向1-N关联

对于单向的1-N关联关系，只需要在1的一端增加Set类型的属性，该属性记录当前实体的关联实体。同样以员工-部门为例(Employee-->Department)。两个持久化类如下：

|  |
| --- |
| **public** **class** Employee {  **private** Integer id;  **private** String name;  //set,get,toString,无参构造器  **public** **class** Department {  **private** Integer id;  **private** String name;  **private** Set<Employee> employees;  //set,get,toString,无参构造器 |

## 基于无连接表的单向1-N关联

对于1-N的单向关联，需要在1的一端增加对应的集合映射元素，如<set.../>、<bag.../>。在集合元素中需要增加<key.../>子元素，该子元素用来映射外键。同时集合元素中需要使用<one-to-many.../>元素来映射1-N关联关系。

|  |
| --- |
| <hibernate-mapping package="com.hibernate.domain" >  <class name="Department" table="department">  <id name="id" column="departmentID">  <generator class="native" />  </id>    <property name="name" column="departmentName" />    <!-- 映射集合属合 -->  <set name="employees" inverse="true" >  <!-- 指定关联的外键列 -->  <key column="departmentID" />  <!-- 用以映射到关联类属性 -->  <one-to-many class="Employee"/>  </set>  </class>  </hibernate-mapping>  对于上面的映射文件，映射<set.../>元素时并没有指定cascade属性，在默认的情况下，对主表实体的持久化操作不会级联到从表实体。  <hibernate-mapping package="com.hibernate.domain">  <class name="Employee" table="employee">  <id name="id" column="employeeID">  <generator class="native" />  </id>  <property name="name" column="employeeName" />  </class>  </hibernate-mapping> |
| **public** **class** oneToManyTest {    **private** SessionFactory sessionFactory;  **private** Session session;  **private** Transaction transaction;  @Before  **public** **void** init() {  //创建配置对象  Configuration config = **new** Configuration().configure();  //创建会话工厂对象  sessionFactory = config.buildSessionFactory();  //创建会话对象  session = sessionFactory.openSession();  //开启事务  transaction = session.beginTransaction();  }  @After  **public** **void** destory() {  //提交事务  transaction.commit();  //关闭会话  session.close();  //关闭会话工厂  sessionFactory.close();  }    @Test  **public** **void** add(){  Department department = **new** Department();  department.setName("国防部");    //建立两个对象  Employee employee1 = **new** Employee();  employee1.setName("chentmt1");    Employee employee2 = **new** Employee();  employee2.setName("chentmt2");    Set<Employee> emps = **new** HashSet<Employee>();  emps.add(employee1);  emps.add(employee2);    //设置Department和Employee之间的关联关系  department.setEmployees(emps);    session.save(department); //....1  session.save(employee2);  session.save(employee1);    }  }  分析上面代码段：当程序运行到....1的时候，系统会持久化该对象：Department，而且这个对象已经关联了两个Employee对象。在这个时候hibernate需要完成两件事：  1. 执行insert语句想department表中插入一条记录  2. Hibernate试图执行update语句，将当前的department表记录关联的  employee表记录的外键departmentID修改为该department记录的主键的值。 |

## 基于有连接表的单向1-N关联

 对于有连接表的1-N关联映射，映射文件不再使用<one-to-many..../>元素映射关联实体，而是使用<many-to-many.../>元素，但是为了保证当前实体是1的一端，我们需要增加一个属性：unique="true"。

|  |
| --- |
| <hibernate-mapping package="com.hibernate.domain" >  <class name="Department" table="department">  <id name="id" column="departmentID">  <generator class="native" />  </id>  <property name="name" column="departmentName" />    <set name="employees" inverse="true" table="department\_employee">  <!-- 指定关联的外键列 -->  <key column="departmentID" />  <!-- 用以映射到关联类属性 -->  <many-to-many class="Employee" column="employeeID" unique="true"/>  </set>  </class>  </hibernate-mapping> |

# 双向1-N关联

对于1-N关联，Hibernate推荐使用双向关联，而且不要让1端控制关联关系，而使用N的一端控制关联关系。

双向的1-N关联，两端都需要增加对关联属性的访问，N的一端增加引用到关联实体的属性，1的一端增加集合属性，集合元素为关联实体。

两个持久化类和上面差不多，只需要在Employee中增加对Department的引用属性即可。

## 无连接表的双向1-N关联

对于无连接表的双向1-N关联。N的一端需要增加<many-to-one.../>元素来映射关联属性，而1的一端需要使用<set.../>或者<bag.../>元素来映射关联属性。同时<set.../>或者<bag../>元素需要增加<key.../>子元素映射外键列，并且使用<one-to-many.../>子元素映射关联属性。

在前面已经提到对于1-N关联映射，通常不提倡1的一端控制关联关系，而应该由N的一端来控制关联关系。此时我们可以再<set.../>元素中指定inverse="true"，用于指定1的一端不控制关联关系。

|  |
| --- |
| <hibernate-mapping package="com.hibernate.domain" >  <class name="Department" table="department">  <id name="id" column="departmentID">  <generator class="native" />  </id>  <property name="name" column="departmentName" />  <!-- 映射集合属合 -->  <set name="employees" inverse="true" >  <!-- 指定关联的外键列 -->  <key column="departmentID" />  <!-- 用以映射到关联类属性 -->  <one-to-many class="Employee"/>  </set>  </class>  </hibernate-mapping> |
| <hibernate-mapping package="com.hibernate.domain">  <class name="Employee" table="employee">  <id name="id" column="employeeID">  <generator class="native" />  </id>  <property name="name" column="employeeName" />  <!-- 用于映射N-1关联实体，指定关联实体类为 :Department,指定外键为：departmentID-->  <many-to-one name="department" column="departmentID" />  </class>  </hibernate-mapping> |

## 有连接表的双向1-N关联

有连接表的双向1-N关联。1的一端使用集合元素映射，然后在集合元素中增加<many-to-many../>子元素，该子元素映射到关联类。为保证该实体是1的一端，需要增加unique="true"属性。N的一端则使用<join.../>元素来强制使用连接表。

在N的一端使用<join../>元素强制使用连接表，因此将使用<key.../>子元素来映射连接表中外键列，且使用<many-to-one../>来映射连接表中关联实体的外键列；反过来，1的一端也将在<set.../>元素中使用<key.../>和<many-to-many.../>两个子元素，他们也映射到连接表的两列。为了保证得到正确的映射关系，应该满足如下关系：<join.../>里<key.../>子元素的column属性和<set.../>里<man-to-many.../>元素的column属性相同；<join.../>里<many-to-many.../>子元素的column属性和<set../>元素的<key../>子元素的column属性应该相等。

|  |
| --- |
| <hibernate-mapping package="com.hibernate.domain" >  <class name="Department" table="department">  <id name="id" column="departmentID">  <generator class="native" />  </id>  <property name="name" column="departmentName" />  <set name="employees" inverse="true" table="department\_employee">  <!-- 指定关联的外键列 -->  <key column="departmentID" />  <!-- 用以映射到关联类属性 -->  <many-to-many class="Employee" column="employeeID" unique="true"/>  </set>    </class>  </hibernate-mapping> |
| <hibernate-mapping package="com.hibernate.domain">  <class name="Employee" table="employee">  <id name="id" column="employeeID">  <generator class="native" />  </id>  <property name="name" column="employeeName" />    <!-- 使用join元素强制使用连接表 -->  <join table="department\_employee">  <!-- key映射外键列 -->  <key column="employeeID" />  <!-- 映射关联实体 -->  <many-to-one name="department" column="departmentID" />  </join>  </class>  </hibernate-mapping> |