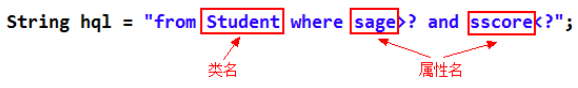
Hibernate支持原始SQL语句查询，支持QBC查询及Hibernate特有的HQL查询。

HQL，Hibernate Query Language，Hibernate查询语言，它与SQL非常相似。但，HQL是面向对象的查询语言，而SQL是面向二维表的。HQL查询语句中使用的是类名与属性名，而SQL语句使用的是表名与字段名。



注：不使用类的别名的情况下，也可使用表的字段名。但若使用“类别名.属性名”的形式，则不能使用字段名。



QBC，Query By Creteria，标准查询，一种比HQL更为面向对象的查询方法。

## 1、API

### 1）Query接口

Hibernate进行HQL查询的接口，支持动态参数绑定。使用Session对象的createQuery方法可获取Query对象。

Query query = session.createQuery(hql);

### 2)SQLQuery接口

Hibernate进行SQL原生查询的接口，支持动态参数绑定，是Query接口的子接口。使用Session对象的createSQLQuery()方法可获取SQLQuery对象。

SQLQuery sqlQuery = session.createSQLQuery(sql);

其查询出的结果对象默认为Object，当然，若结果为List，则其元素为Object。使用SQLQuery的addEntity(Xxx.class)方法，可以将其结果泛型设定为指定类型。

### 3）Criteria接口

Criteria，标准，准则，Hibernate进行Criteria查询的接口，与Query接口无关。使用Session对象的createCriteria()方法可获取Criteria对象。

Criteria criteria = session.createCriteria(Xxx.class);

## 2、分类查询

### 1）查询测试前的准备工作

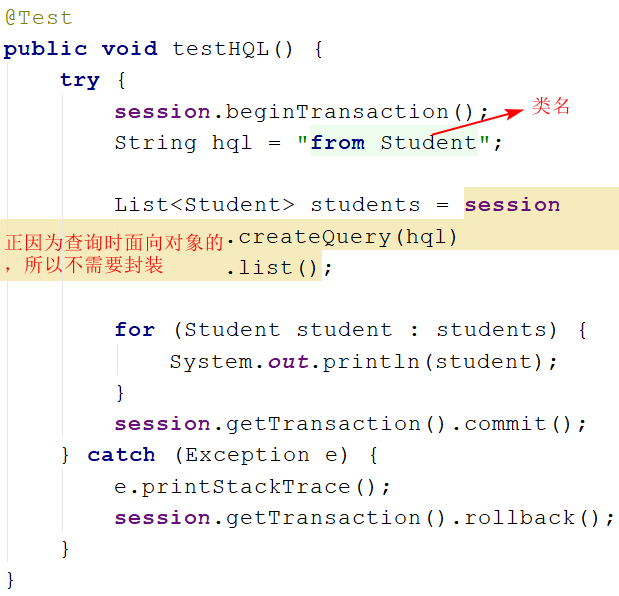
1. 定义查询的实体
2. 定义主配置文件
3. 定义HibernateUtils工具类
4. 定义测试类
5. 准备测试数据

### 2）查询所有（其查询结果为List集合）

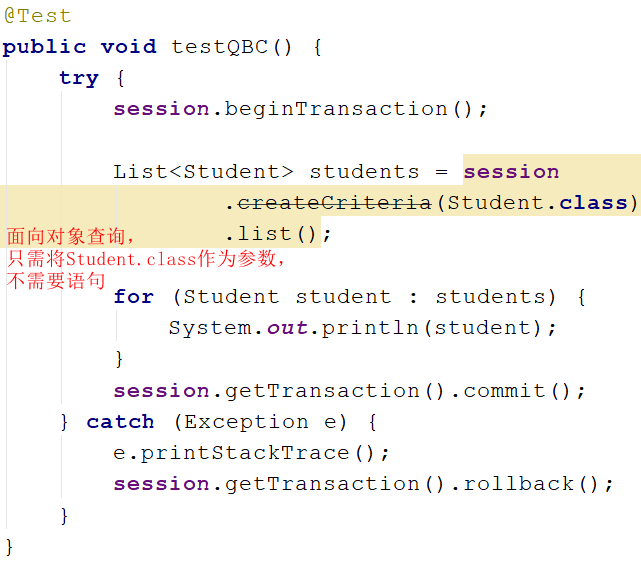
SQL查询：



HQL查询：

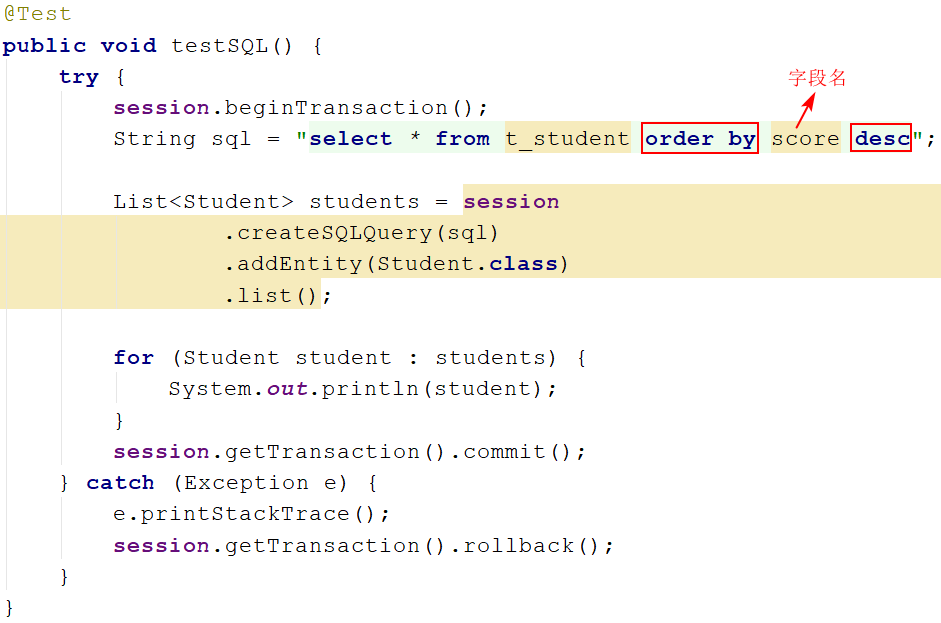


QBC查询：

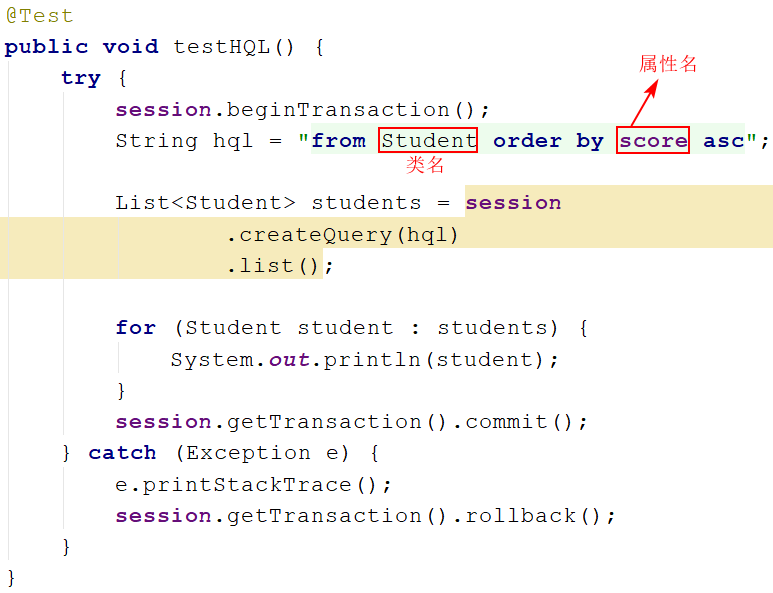


### 3）查询结果排序（order by，asc表示升序，desc表示降序）

SQL查询：

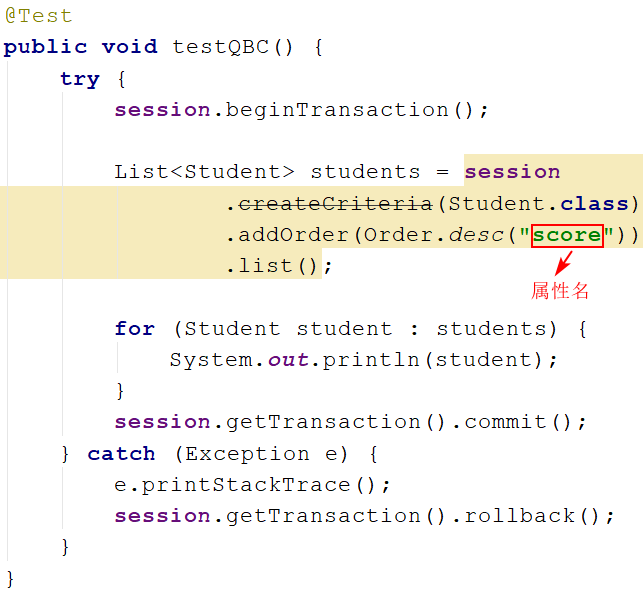


HQL查询：



QBC查询：

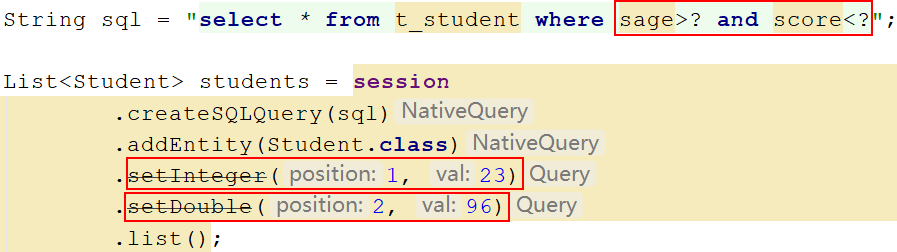
通过addOrder()方法可以向Criteria对象添加排序功能。Order类有两个静态方法desc()与asc()，分别用于降序和升序功能，他们的参数为排序属性名，注意非字段名。



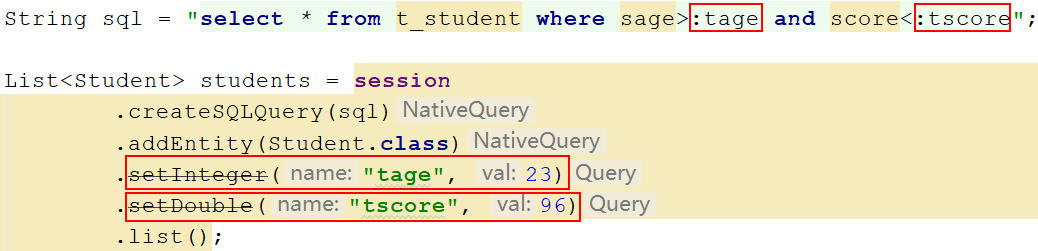
### 4）动态参数绑定查询（即参数由代码传递过来，而非固定在查询中）

SQL查询：

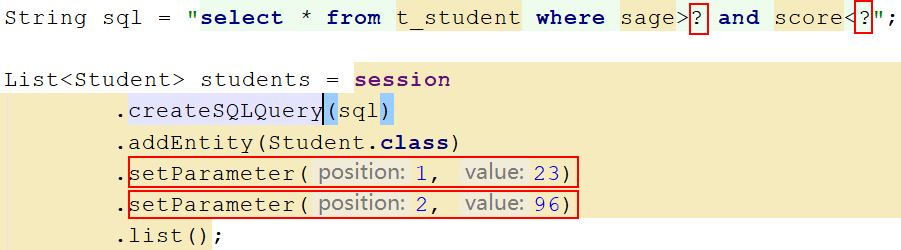
方式一：使用“?”占位符，setXxx(int position, Object value)绑定参数。position表示第几个占位符，从1开始计数。



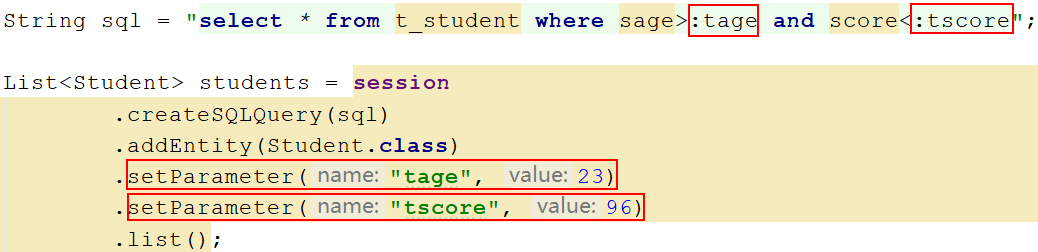
方式二：



方式三：使用“?”占位符占位，setParameter(int position, Object value)绑定参数。position表示从第几个占位符，从1开始计数。

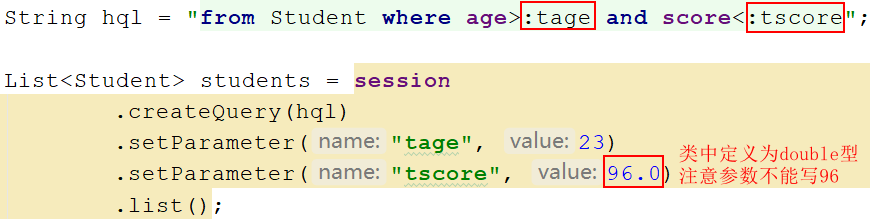


方式三：使用别名占位，setParamter(String name, Object value)绑定参数。别名的命名规则为，冒号后跟别名，别名可是任意名称。



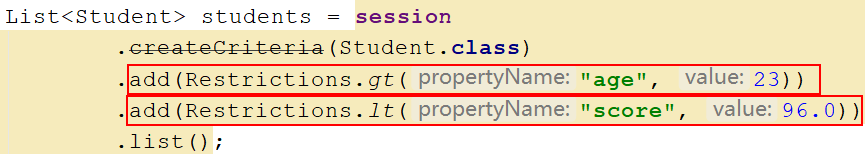
HQL查询：

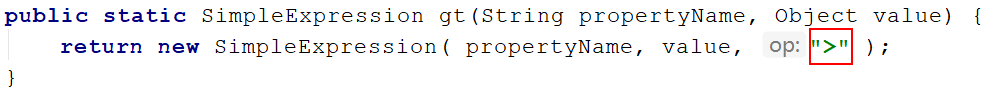
使用HQL查询时，以上四种方式均可成立，且用法相同。

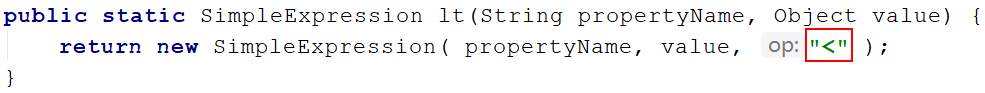


QBC查询：

需要向Criteria对象添加限制条件Restrictions，即添加查询条件。Restriction（限制，约束）。Restriction具有很多静态方法可用于表示关系比较。（注意double数据类型的写法）







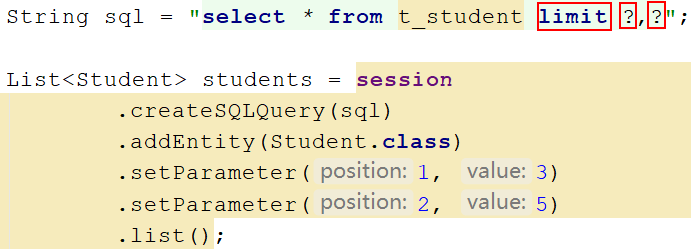
### 5）分页查询（limit startIndex,pageSize）

分页查询的SQL语句：

select \* from ... where ... limit startIndex, pageSize

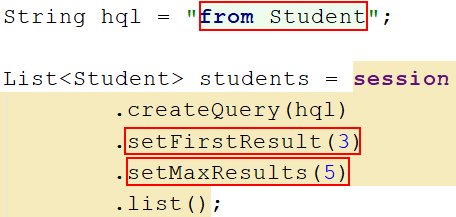
startIndex为开始索引，设置从总查询结果集的第几条记录作为本查询结果子集的开始，即本页的开始。总查询结果集从0开始计数。pageSize为该页所包含的记录数。

SQL查询：

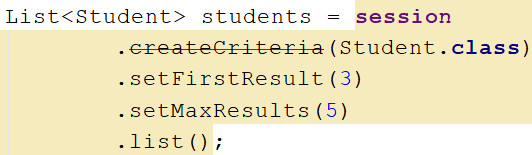


HQL查询：

Query接口中具有setFirstResult()方法，用于设置从总结果集的第几条记录作为本查询结果子集的开始，即本页的开始。总查询结果集从0开始计数。而setMaxResult()方法用于设置该页所包含的记录数。

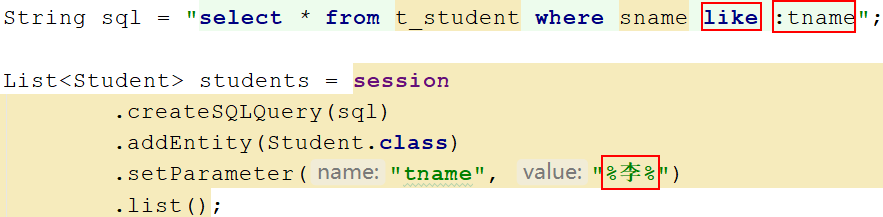


QBC查询：

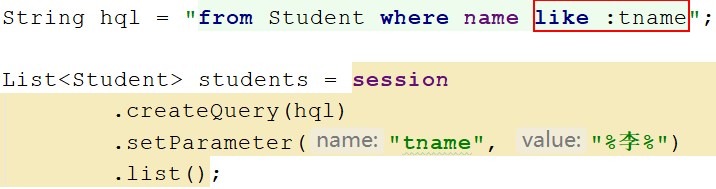


### 6）模糊查询（like %xxx%）

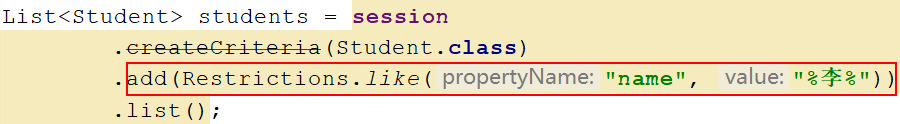
SQL查询：



HQL查询：



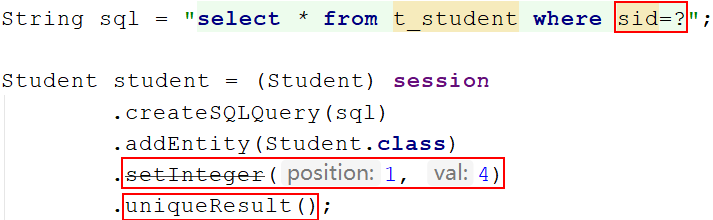
QBC查询：



### 7）唯一性查询（where）

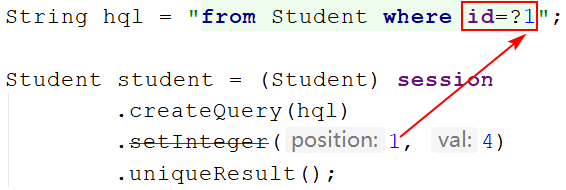
SQL查询：

SQLQuery接口的uniqueResult()方法表示其查询结果为一条数据，其返回值为Object。

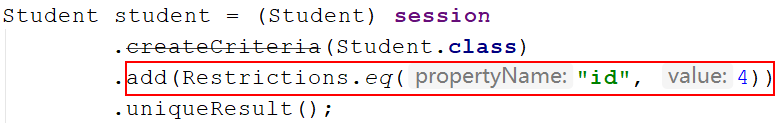


HQL查询：

注：传统样式查询参数（`？`）不再支持；使用JPA样式的序号参数（例如，`？1’）



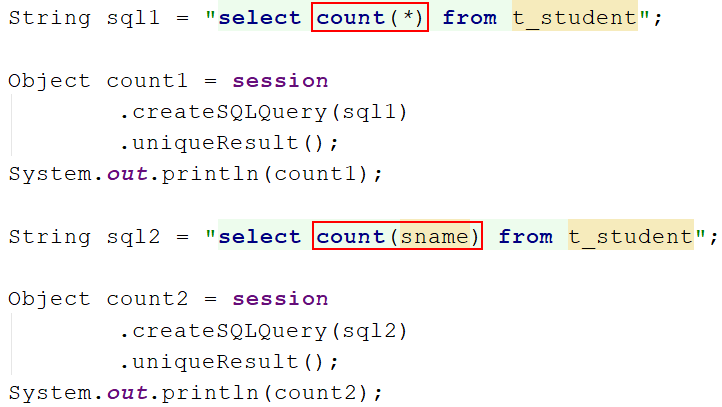
QBC查询：



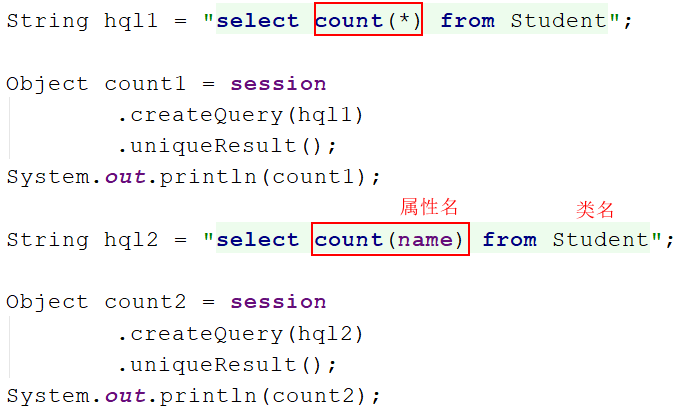
### 8）聚合函数查询（count(\*)、count(sname)）

SQL查询：

注意区分count(\*)与count(name)的不同。count(\*)与count(id)等价，均表示未空记录数。而count(name)则表示字段name的非空记录数。

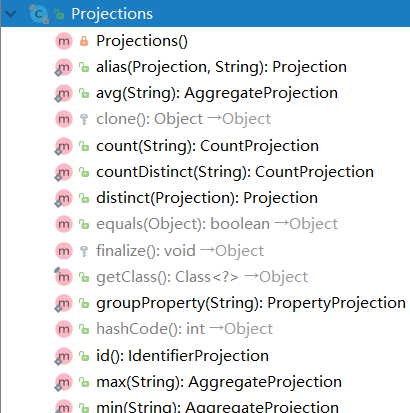


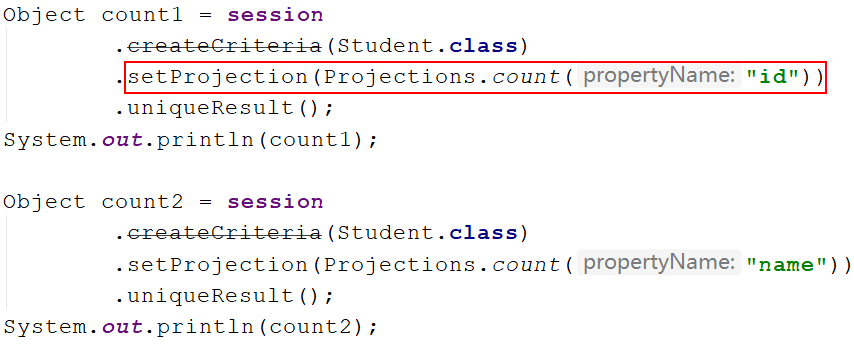
HQL查询：



QBC查询：

Projection，中文：投影。Projection中包含很多聚合函数的静态方法。



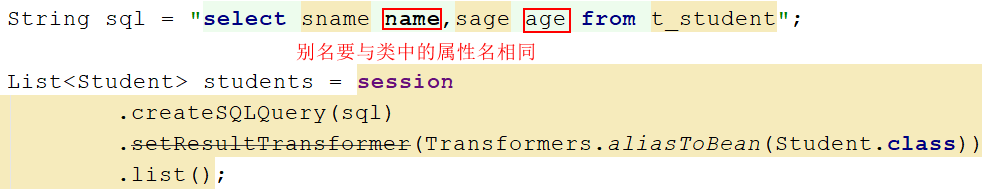


### 9）投影查询（查询部分字段）

SQL查询：

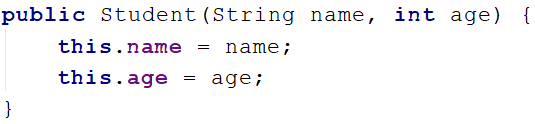
要查询的字段的别名要与实体类的属性名相同，这样才能将结果转化为相应类的对象。

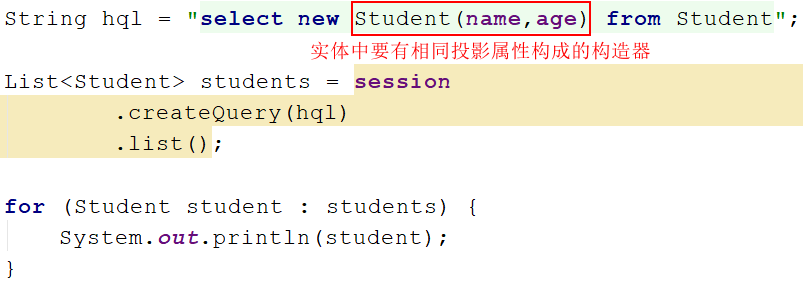
aliasToBean()方法首先会创建一个空的Student对象，然后会将别名与Student属性名对比，再用查询出的值初始化创建的Student对象。



HQL查询：

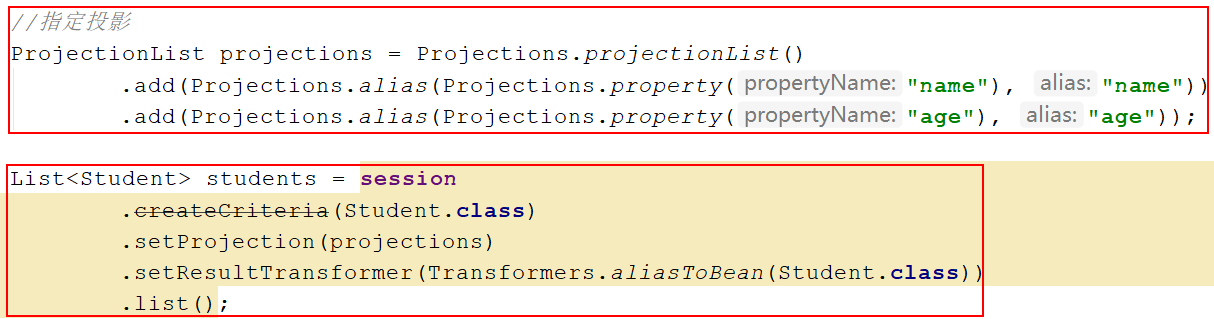
该查询要求实体类中具有相同投影作为参数的带参构造器。所以，首先要在Student类中添加相应的带参构造器。





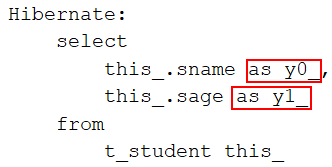
BQC查询：

如果直接将name与age属性指定为投影，则可以看到其输出的Student对象的属性值全部为空。

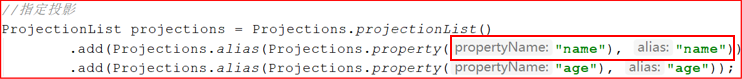


首先，setResultTransformer()方法的参数Transformers.aliasToBean()会根据别名自动创建指定类型对象。

查看产生的SQL语句可以发现，Hibernate为每个投影属性指定了一个别名，而这个别名与Student类的属性值无关。所以生成的对象属性值均为空。



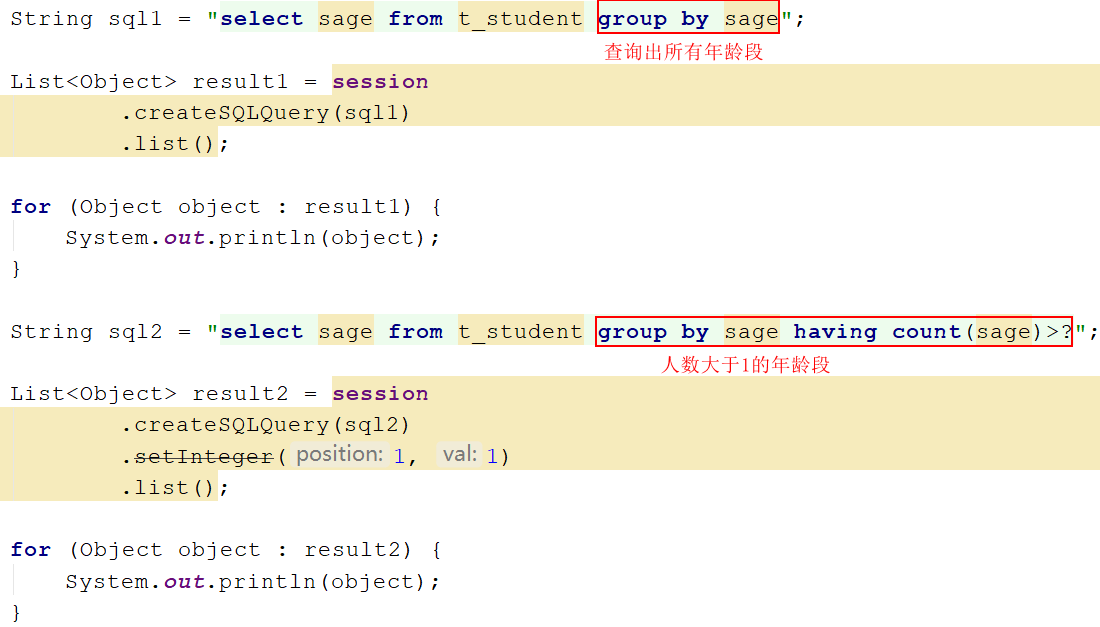
所以需要手工为每个投影指定别名，别名要与实体类的属性相同。



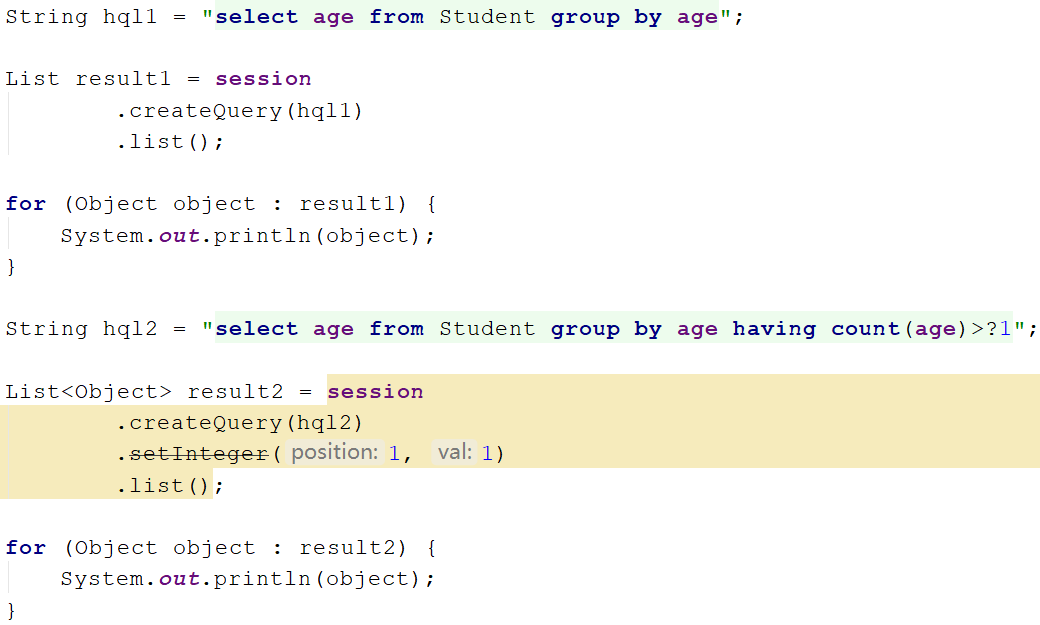
### 10）分组查询（group by xxx having count(xxx)）

SQL查询：

having，对分组结果进行筛选。下面代码实现的功能是，查询出所有年龄段，及人数多于1人的年龄段。



HQL查询：



QBC查询：

Hibernate不支持having操作。

