Session session=  this.getSession();  
  List list=session.createCriteria(MeterUser.class).add(Restrictions.eq("name", "11")).addOrder(Order.desc("userid")).list();

Restrictions的用法：

|  |  |
| --- | --- |
| 方法 | 说明 |
| Restrictions.eq | ＝ |
| Restrictions.allEq | 利用Map来进行多个等于的限制 |
| Restrictions.gt | ＞ |
| Restrictions.ge | ＞＝ |
| Restrictions.lt | ＜ |
| Restrictions.le | ＜＝ |
| Restrictions.between | BETWEEN |
| Restrictions.like | LIKE |
| Restrictions.in | in |
| Restrictions.and | and |
| Restrictions.or | or |
| Restrictions.sqlRestriction | 用SQL限定查询 |

有空再添加上，代码示例。

***QBC常用限定方法***

Restrictions.eq --> equal,等于.

Restrictions.allEq --> 参数为Map对象,使用key/value进行多个等于的比对,相当于多个Restrictions.eq的效果

Restrictions.gt --> great-than > 大于

Restrictions.ge --> great-equal >= 大于等于

Restrictions.lt --> less-than, < 小于

Restrictions.le --> less-equal <= 小于等于

Restrictions.between --> 对应SQL的between子句

Restrictions.like --> 对应SQL的LIKE子句

Restrictions.in --> 对应SQL的in子句

Restrictions.and --> and 关系

Restrictions.or --> or 关系

Restrictions.isNull --> 判断属性是否为空,为空则返回true

Restrictions.isNotNull --> 与isNull相反

Restrictions.sqlRestriction --> SQL限定的查询

Order.asc --> 根据传入的字段进行升序排序

Order.desc --> 根据传入的字段进行降序排序

MatchMode.EXACT --> 字符串精确匹配.相当于"like 'value'"

MatchMode.ANYWHERE --> 字符串在中间匹配.相当于"like '%value%'"

MatchMode.START --> 字符串在最前面的位置.相当于"like 'value%'"

MatchMode.END --> 字符串在最后面的位置.相当于"like '%value'"

例子  
查询年龄在20-30岁之间的所有学生对象  
List list = session.createCriteria(Student.class)  
      .add(Restrictions.between("age",new Integer(20),new Integer(30)).list();  
查询学生姓名在AAA,BBB,CCC之间的学生对象  
String[] names = {"AAA","BBB","CCC"};  
List list = session.createCriteria(Student.class)  
      .add(Restrictions.in("name",names)).list();  
查询年龄为空的学生对象  
List list = session.createCriteria(Student.class)  
      .add(Restrictions.isNull("age")).list();  
查询年龄等于20或者年龄为空的学生对象  
List list = session.createCriteria(Student.class)  
      .add(Restrictions.or(Restrictions.eq("age",new Integer(20)),  
                 Restrictions.isNull("age")).list();

--------------------------------------------------------------------  
**使用QBC实现动态查询**  
public List findStudents(String name,int age){

 Criteria criteria = session.createCriteria(Student.class);  
 if(name != null){  
  criteria.add(Restrictions.liek("name",name,MatchMode.ANYWHERE));  
 }  
 if(age != 0){  
  criteria.add(Restrictions.eq("age",new Integer(age)));  
 }  
 criteria.addOrder(Order.asc("name"));//根据名字升序排列  
 return criteria.list();  
}

-----------------------------------------------------------------------------------

今天用了写hibernate高级查询时用了Restrictions(当然Expression也是可以的)这个类.感觉不错.  
下面的代码写的不易读.其实核心就是一句  
Restrictions.or(Restrictions.like(),Restrictions.or(Restrictions.like,........))  
里面的or可以无限加的.还是比较好用  
  
Session session = getHibernateTemplate().getSessionFactory()  
                .openSession();  
        Criteria criteria = session.createCriteria(Film.class);  
        List<Film> list = criteria.add(  
            Restrictions.or(Restrictions.like("description", key,MatchMode.ANYWHERE),  
            Restrictions.or(Restrictions.like("name", key,MatchMode.ANYWHERE),  
                Restrictions.or(    Restrictions.like("direct", key,MatchMode.ANYWHERE),  
                Restrictions.or(Restrictions.like("mainplay",key,MatchMode.ANYWHERE),  
                        Restrictions.like("filearea", key,MatchMode.ANYWHERE)))))).list();  
  
        session.close();  
        return list;