hibernate的native sql查询 - xiaoluo501395377 - 博客园(1)

在我们的hibernate中，除了我们常用的HQL查询以外，还非常好的支持了原生的SQL查询，那么我们既然使用了hibernate，为什么不都采用hibernate推荐的HQL查询语句呢？这是因为HQL查询语句虽然方便我们查询，但是基于HQL的查询会将查询出来的对象保存到hibernate的缓存当中，如果在我们的一个大型项目中(数据量超过了百万级)，这个时候如果使用hibernate的HQL查询的话，会一次将我们查询的对象查询出来后放到缓存中，这个时候会影响我们的效率，所以当在大型项目中使用hibernate时我们的最佳实践就是--使用原生的SQL查询语句，而不使用HQL语句，因为通过SQL查询的话，是不会经过hibernate的缓存的。接下来我们就来看看hibernate的原生SQL查询

1.标量查询

在hibernate中，我们如果使用原生SQL查询的话，是通过SQLQuery接口进行的，我们首先来看看我们最基本的查询：

```
session.createSQLQuery("select * from t_student s").list()session.createSQLQuery("select ID,NAME,SEX from t_student s").list()
```

这就创建了最简单的两条SQL查询语句，此时返回的数据库表的字段值是保存在一个Object[]数组中的，数组中的每个元素就是查询出来的t_student表中的每个字段值，Hibernate会通过ResultSetMetadata来判断每个字段值的存放位置以及类型，接下来我们看下标量查询：

```
List stus = (List)session.createSQLQuery("select * from t_student s")
.addScalar("ID")                                                    .addScalar("NAME")
.setFirstResult(0).setMaxResults(20)                                                .list();
```

这个查询语句指定了查询的字符串以及返回的字段和类型

这条查询语句仍然会返回一个Object[]类型的数组，但是此时就不会通过ResultSetMetadata，而是我们明确指定的ID,NAME,SEX，此时虽然查询语句中有 * 将所有的字段查询出来，但是这个时候仅仅只会返回我们指定的三个字段值，其类型还是由ResultSetMetada来指定的。

2.实体查询

①返回一个实体对象

上述的标量查询返回的裸数据，保存到了Object[]数组当中，我们如果要一个实体对象，将其保存到实体对象时就可以使用 addEntity()方法来实现：

复制代码

```
@Test    publicvoid testSql1()    {        Session session = null;        try        {            session = HibernateUtil.openSession();            /*             * 原生的SQL语句查询，会将t_student表的所有字段查出来，存放的一个Object[]数组当中             * 如果希望转换成实体对象，只需要调用 addEntity(Student.class)即可，这时，会首先匹配
* Student对象里面的属性是否全部查询出来，如果没有，则报错(如果这个类是实体类)             * 注意：如果要调用addEntity方法，这个类必须是实体类，即加了@Entity注解或者在XML中配置了实体类             * 映射关系             */            List stus = (List)session.createSQLQuery("select * from t_student s")
.addEntity(Student.class)                                                .setFirstResult(0).setMaxResults(20)
.list();            for(Student stu : stus)            {                System.out.println(stu.getName());            }
}        catch (Exception e)        {            e.printStackTrace();        }        finally        {
HibernateUtil.close(session);        }    }
```

复制代码

这时我们就指定了将查询出来的对象存进Student这个实体类中，**注意：如果使用了实体对象，那么在查询时要将实体对象中有的属性全部在SQL语句中查询出来，否则就会报错**。

②返回多个实体对象

有的时候我们可能不只查询出一个实体对象，在使用连接查询时候，我们可能需要将几个表的数据都查询出来，并存放到对应的实体对象中去，这个时候我们应该怎么写呢?先看下如下一种写法：

复制代码

```
List stus = (List)session.createSQLQuery("select stu.*, cla.*, spe.*"                    + " from t_student stu left join t_classroom cla on stu.rid=cla.id"                    + " left join t_special spe on spe.id=cla.sid where stu.name like ?")                    .addEntity("stu", Student.class)
.addEntity("cla", Classroom.class)                                                .addEntity("spe", Special.class)
```

```
.setFirstResult(0).setMaxResults(20)                                    .setParameter(0, "%张%")
.list();
```

复制代码

我们这里通过addEntity的一个重载方法给每个别名指定了一个关联的实体类，这个时候就会出现字段名冲突的问题，我们的本意是查询出三个实体对象，分别取得其name属性，但是name属性在这三张表中的字段都是name，这个时候查询出来的三个实体对象中的属性值都是一样的。我们如果要解决这个方法，只需要用一个 {} 花括号占位符将每个表的所有属性值括起来即可。如下：

复制代码

```
@Test    publicvoid testSql3()    {        Session session = null;        try        {            session = HibernateUtil.openSession();            /**             * 当使用连接查询查询多个对象时，可以通过addEntity("alias", XXX.class)方法来根据             * 数据库表的别名来引入多个实体类，这时如果需要将查询出来的所有的对象分别存入实体类中，
* 只需要在查询出来的对象上添加 {} 号即可，此时就会自动帮我们分类             */            List stus = (List)session.createSQLQuery("select {stu.*}, {cla.*}, {spe.*}"                    + " from t_student stu left join t_classroom cla on stu.rid=cla.id"                    + " left join t_special spe on spe.id=cla.sid where stu.name like ?")                    .addEntity("stu", Student.class)
.addEntity("cla", Classroom.class)                                    .addEntity("spe", Special.class)
.setFirstResult(0).setMaxResults(20)                                    .setParameter(0, "%张%")
.list();            for(Object[] obj : stus)            {                Student stu = (Student)obj[0];
Classroom cla = (Classroom)obj[1];                Special spe = (Special)obj[2];
System.out.println(stu.getName() + ", " + cla.getName() + ", " + spe.getName());            }        }        catch (Exception e)        {            e.printStackTrace();        }        finally        {
HibernateUtil.close(session);        }    }
```

复制代码

③返回不受hibernate管理的实体对象

我们有时候的需求是这样的，将每个表其中的一些字段查询出来，然后存放到一个javabean对象中，但是我们的这个bean对象又不需要存放到数据库，不需要设置成实体对象，这个时候我们往往会创建一个 DTO 的数据传输对象来存放我们要储存的属性，例如定义了一个 StudentDTO 对象：

复制代码

```
publicclass StudentDTO{    privateint sid;    // 学生id    private String sname;    // 学生姓名    private String sex;
  // 学生性别    private String cname;    // 班级名    private String spename;    // 专业名        public StudentDTO(){}
public StudentDTO(int sid, String sname, String sex, String cname,            String spename)    {        super();
this.sid = sid;        this.sname = sname;        this.sex = sex;        this.cname = cname;        this.spename = spename;    }    ..............}
```

复制代码

这个时候我们来看看我们的查询语句：

复制代码

```
@Test    publicvoid testSql4()    {        Session session = null;        try        {            /**             * 对于非Entity实体对象的类，我们如果要保持数据，可以通过定义一个DTO对象             * 然后调用 setResultTransformer(Transformers.aliasToBean(StudentDTO.class))方法             * 来返回一个不受管的Bean对象
*/            session = HibernateUtil.openSession();            List stus = (List)session.createSQLQuery("select "
+ "stu.id as sid, stu.name as sname, stu.sex as sex, cla.name as cname, spe.name as spename"                    + " from t_student stu left join t_classroom cla on stu.rid=cla.id"                    + " left join t_special spe on spe.id=cla.sid where stu.name like ?")
.setResultTransformer(Transformers.aliasToBean(StudentDTO.class))
.setFirstResult(0).setMaxResults(20)                                    .setParameter(0, "%张%")
.list();            for(StudentDTO std : stus)            {                System.out.println(std.getSname() + ", " + std.getCname() + ", " + std.getSpename());            }        }        catch (Exception e)        {
e.printStackTrace();        }        finally        {            HibernateUtil.close(session);        }    }
```

复制代码

我们要将查询出来的这些不同的表的字段存放到一个不是实体对象当中，就可以调用 .setResultTransformer(Transformers.aliasToBean(StudentDTO.class)) 方法来创建一个非受管的 bean 对象，这个时候就hibernate就会将属性值存放到这个 bean 对象当中，**注意：查询出来的表的字段值存放到bean对象中，是通过调用 bean对象的 setter方法，如果该属性在bean对象中没有setter方法，则会报错**。

本篇随笔主要分析了一下如何来通过原生的SQL语句查询我们需要的信息，我们一定要记住，当数据量非常大的时候，强烈建议使用原生的SQL去查询数据，而不要使用HQL来查询，这样其实使用hibernate来说效率其实也不差，但是我们的增、删、改的操作则完全可以交给hibernate来完成。