**[Hibernate --调用存储过程](http://cwfmaker.iteye.com/blog/453781)（一）**

Java代码 [复制代码](http://cwfmaker.iteye.com/blog/453781) [[收藏代码http://cwfmaker.iteye.com/images/spinner.gif](javascript:void())](javascript:void())

1. CREATE   TABLE  `proctab` (
2. `id`  int ( 11 )  NOT   NULL  auto\_increment,
3. `Name`  varchar ( 20 ),
4. `age`  int ( 11 ),
5. PRIMARY   KEY   (`id`)
6. )

CREATE TABLE `proctab` (

`id` int ( 11 ) NOT NULL auto\_increment,

`Name` varchar ( 20 ),

`age` int ( 11 ),

PRIMARY KEY (`id`)

)

简单的存储过程

Java代码 [复制代码](http://cwfmaker.iteye.com/blog/453781) [[收藏代码http://cwfmaker.iteye.com/images/spinner.gif](javascript:void())](javascript:void())

1. create   PROCEDURE  proc()
2. begin
3. select   \*   from  proctab;
4. end ;

create PROCEDURE proc()

begin

select \* from proctab;

end ;

一种方法是通过hibernate传统的xml映射方式去调用

Java代码 [复制代码](http://cwfmaker.iteye.com/blog/453781) [[收藏代码http://cwfmaker.iteye.com/images/spinner.gif](javascript:void())](javascript:void())

1. <class name="com.test.User" table="proctab">
2. <id name="id" column="id">
3. <generator class="native"/>
4. </id>
5. <property name="name" column="name" type="string" />
6. <property name="age" column="age" type="integer" />
7. </class>
8. <sql-query name="getUser" callable="true">
9. <return alias="user" class="com.test.User">
10. <return-property name="id" column="id" />
11. <return-property name="name" column="name" />
12. <return-property name="age" column="age" />
13. </return>
14. {call proc()}
15. </sql-query>

<class name="com.test.User" table="proctab">

<id name="id" column="id">

<generator class="native"/>

</id>

<property name="name" column="name" type="string" />

<property name="age" column="age" type="integer" />

</class>

<sql-query name="getUser" callable="true">

<return alias="user" class="com.test.User">

<return-property name="id" column="id" />

<return-property name="name" column="name" />

<return-property name="age" column="age" />

</return>

{call proc()}

</sql-query>

调用方法

Java代码 [复制代码](http://cwfmaker.iteye.com/blog/453781) [[收藏代码http://cwfmaker.iteye.com/images/spinner.gif](javascript:void())](javascript:void())

1. Session ss= HibernateSessionFactory.getSession()
2. List li=ss.getNamedQuery("getUser").list();
3. ss.close();

Session ss= HibernateSessionFactory.getSession()

List li=ss.getNamedQuery("getUser").list();

ss.close();

及其类似jdbc的方法

Java代码 [复制代码](http://cwfmaker.iteye.com/blog/453781) [[收藏代码http://cwfmaker.iteye.com/images/spinner.gif](javascript:void())](javascript:void())

1. Session session =HibernateSessionFactory.getSession();
2. Connection conn = session.connection();
3. ResultSet rs =null;
4. CallableStatement call = conn.prepareCall("{Call proc()}");
5. rs = call.executeQuery();
6. rs.close();
7. session.close();

Session session =HibernateSessionFactory.getSession();

Connection conn = session.connection();

ResultSet rs =null;

CallableStatement call = conn.prepareCall("{Call proc()}");

rs = call.executeQuery();

rs.close();

session.close();

这种方法基本上就是jdbc，不过很简单，总是感觉怪怪的~！   
  
还有就是通过强大的createSQLQuery来实现   
o(∩\_∩)o...哈哈  个人比较喜欢这种方法

Java代码 [复制代码](http://cwfmaker.iteye.com/blog/453781) [[收藏代码http://cwfmaker.iteye.com/images/spinner.gif](javascript:void())](javascript:void())

1. Session session =HibernateSessionFactory.getSession();
2. SQLQuery query = session.createSQLQuery("{Call proc()}");
3. List list =query.list();
4. session.close();

Session session =HibernateSessionFactory.getSession();

SQLQuery query = session.createSQLQuery("{Call proc()}");

List list =query.list();

session.close();

如果没有返回值 直接用execute的方法就可以了   
(\*^\_\_^\*)  忘了一个重要的问题就是存储过程的传参问题

Java代码 [复制代码](http://cwfmaker.iteye.com/blog/453781) [[收藏代码http://cwfmaker.iteye.com/images/spinner.gif](javascript:void())](javascript:void())

1. CallableStatement call = conn.prepareCall("{Call proc(?)}");
2. call.setString(1, 参数);
3. rs = call.executeQuery();

CallableStatement call = conn.prepareCall("{Call proc(?)}");

call.setString(1, 参数);

rs = call.executeQuery();

Java代码

1. SQLQuery query = session.createSQLQuery("{Call proc(?)}");
2. query.setString(0, 参数);
3. List list =query.list();

**注意：以上使用hibernate的session.createSQLQuery(“{Call proc(?)}”);这种创建SQLQuery 调用存储过程的方式，例子中呈现的只是存储过程是执行查询操作的功能。可能有针对增、删、改等操作的，估计有query.execute()。若没有，就没法使用，其实说是JDBC的方式，但是session、connection、callablestatement的获取同样是通过hibernate的方式获得的。**

[**Hibernate --调用存储过程**](http://cwfmaker.iteye.com/blog/453781)**（二）**

DB2中简单存储过程　selectAllUsers   
  
CREATE  procedure selectAllUsers    
DYNAMIC RESULT SETS 1   
BEGIN    
      DECLARE temp\_cursor1 CURSOR  WITH RETURN TO CLIENT  FOR    
      SELECT \* FROM  test;      
      OPEN temp\_cursor1;   
END;   
  
映射文件中关于存储过程内容如下   
  
<hibernate-mapping package="com.unmi.vo">   
    <class name="Test" table="TEST">   
　　　　　............   
    </class>   
  
    <sql-query callable="true" name="selectAllUsers">   
<return alias="aa" class="Test">   
    <return-property name="oborqt" column="OBORQT"/>   
    <return-property name="moorqt" column="MOORQT"/>   
    <return-property name="roschn" column="ROSCHN"/>   
    <return-property name="plandate" column="PLANDATE"/>   
</return>   
{ ? = call selectAllUsers() }   
    </sql-query>   
</hibernate-mapping>   
  
**{ ? = call selectAllUsers() } 也可以写成{ call selectAllUsers() }，**   
如果有参数就写成   
{ ? = call selectAllUsers(?,?,?) }   
代码中对query设置相应位置上的值就OK   
  
Java调用关键代码如下   
         
  Session session = HibernateUtil.currentSession();   
  Query query = session.getNamedQuery("selectAllUsers");   
  List list = query.list();   
  System.out.println(list);   
  
**要求你的存储过程必须能返回记录集，否则要出错   
如果你的存储过程是完成非查询任务就应该在配置文件用以下三个标签   
  <sql-insert callable="true">{call createPerson (?, ?)}</sql-insert>   
  <sql-delete callable="true">{? = call deletePerson (?)}</sql-delete>   
  <sql-update callable="true">{? = call updatePerson (?, ?)}</sql-update>   
  
有一点不好的地方就是调用存储过程获取记录集时，不能对Query使用   
setFirstResult(int)和setMaxResults(int)方法来分页   
  
  
  
要求你的存储过程必须能返回记录集，否则要出错**  
  
上面这句话是关键。  
==========================================================================  
hibernate调用mysql5.0存储过程  
关键代码：  
数据库存储过程  
Create procedure SP\_get\_UserInfo()   
as  
begin  
    select \*  
    from user  
   order by Id  
end  
  
hibernate相对应的配置文件User.hbm.xml  
<?xml version="1.0" encoding="utf-8"?>  
<!DOCTYPE hibernate-mapping PUBLIC "-//Hibernate/Hibernate Mapping DTD 3.0//EN"  
"http://hibernate.sourceforge.net/hibernate-mapping-3.0.dtd">  
<!--   
    Mapping file autogenerated by MyEclipse Persistence Tools  
-->  
<hibernate-mapping>  
    <class name="com.test.hibernate.db.User" table="user"  
        catalog="user">  
        <id name="id" type="java.lang.Long">  
            <column name="ID" />  
            <generator class="assigned" />  
        </id>  
        <property name="name" type="java.lang.String">  
            <column name="name" length="500" />  
        </property>  
        <property name="sex" type="java.lang.String">  
            <column name="sex" length="10" />  
        </property>  
        <!--   
            <property name="born" type="java.util.Date">  
            <column name="born" length="0" />  
            </property>  
        -->  
        <property name="born" type="java.lang.String">  
            <column name="born" length="0" />  
        </property>  
        <property name="address" type="java.lang.String">  
            <column name="address" />  
        </property>  
    </class>  
      
    <!--调用存储过程就在这里配 -->  
    <sql-query name="getUser" callable="true">  
        <return alias="user" class="com.test.hibernate.db.User">  
            <return-property name="id" column="ID" />  
            <return-property name="name" column="name" />  
            <return-property name="sex" column="sex" />  
            <return-property name="born" column="born" />  
            <return-property name="address" column="address"/>  
        </return>  
        { call SP\_get\_UserInfo() }  
    </sql-query>  
  
</hibernate-mapping>  
  
调用存储过程的Java代码：  
          
        Session querySession = HibernateSessionFactory.getSession();  
        querySession.clear();  
        List lst = uerySession.getNamedQuery("getUser").list();  
        for(int i=0;i<lst.size();i++)  
        {  
            usr=(User)lst.get(i);  
            System.out  
            .println(usr.getId() + " " + usr.getName() + " "  
                    + usr.getSex() + " " + usr.getBorn() + " "  
                    + usr.getAddress());  
        }  
       HibernateSessionFactory.closeSession();  
  
一个简单的存储过程调用就这样完成了！