# 代码添加工作

第一步：加载hibernate核心配置文件（找到hibernate.cfg.xml）

Configuration cfg = **new** Configuration();

cfg.configure();

第二步：创建SessionFactory，开启会话

SessionFactory sessionFactory = cfg.buildSessionFactory();

Session session = sessionFactory.openSession();

第三步：开启事务

Transaction tx = session.beginTransaction();

第四四：CRUD操作

第五步：提交事务，关闭资源

tx.commit();

session.close();

sessionFactory.close();

# 配置文件详解

## Hibernate映射配置文件

1. 映射配置文件中，标签name属性值写实体类相关内容
2. class标签中name属性值实体类全路径
3. id标签和property标签name属性实体类属性名称
4. id标签和property标签，column属性可以省略
5. 不写，hibernate会自动生成
6. property标签type属性，设置生成表字段的类型，自动对应类型

## Hibernate核心配置文件

|  |
| --- |
| 1. 数据库配置必须 |
| 2.hibernate可选（格式化sql） |
| 3.映射文件必须 |
| 1. 核心配置文件和名称固定   （1）：位置src下面  （2）：名称hibernate.cfg.xml |
| Hibernate核心api的使用 Configuration  1:代码  Configuration cfg = new Configuration();  cfg.configure();  到src下面找到名称为heibernate.cfg.xml配置文件，创建对象，把配置文件放到对象里（加载核心配置文件） |

SessionFactory

|  |
| --- |
| 1 使用configuration对象创建sesssionFactory对象   1. 创建sessionFactory过程中做事情     当配备上面图片的配置时根据数据库配置，到数据库里根据映射关系把表创建  2 创建sessionFactory过程中，过程好资源（抽取工具类，写到静态代码块中） |

Session

|  |
| --- |
| 1 session类似于jdbc中的connection  2 调用sesssion里面的不同的方法实现crud操作   1. 添加save方法 2. 修改update方法 3. 删除delete方法 4. 根据id查询get方法   3 session对象单线程对象  （1）session对象不能共用，只能自己使用 |

Transaction

|  |
| --- |
| 1 事务对象  2 事务提交与回滚  3 事务概念   1. 事务四特性：原子性、一致性、隔离性、持久性 |

## Hibernate主键生成策略

|  |
| --- |
| 1 hibernate要求实体类里面有一个属性作为唯一值，对应表主键，主键可以不同生成策略   1. increment自动以递增的方式生成唯一标识符，增量唯一 2. identity 要求数据库中支持自动增长数据类型 3. sequence利用数据库底层序列生成标识符，需要数据库支持序列 4. native根据底层数据库自动选择identity、sequence、hilo 5. uuid 生成椅子成都为32位的十六进制的字符串（主键必须为字符串类型） |

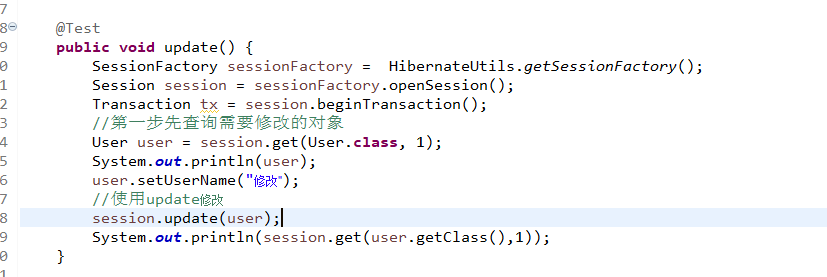
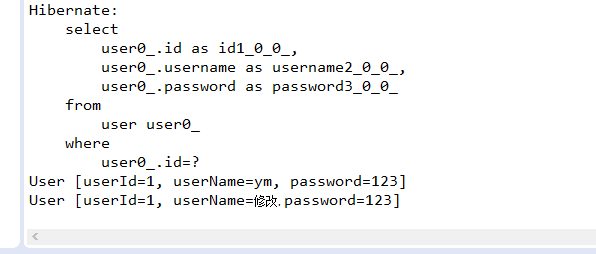
# 实体类CRUD操作

## 添加调用save

使用save时对save的对象是否设置id并不影响，无论是否设置id，都将在表中最后插入一条数据，该数据id使用已经设置好的主键增长策略。

## 根据id查询,调用session里面的get方法实现

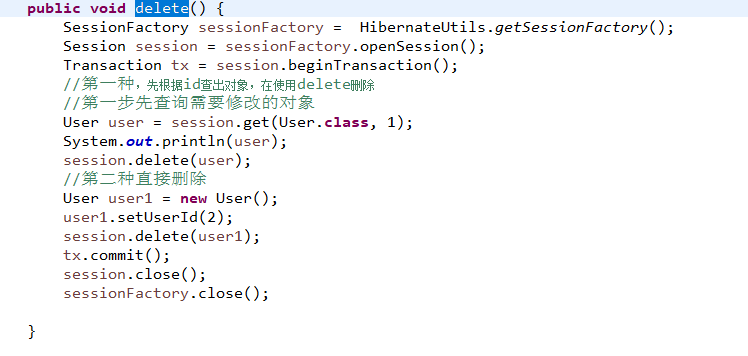
## 修改

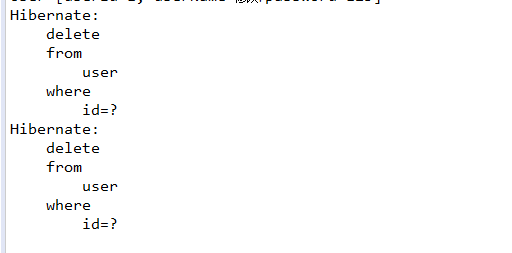
1. 根据id修改
2. 
3. 

update（）方法，具体实现步骤，先根据user对象中的主键查找，然后对其进行修改

使用update时，会对对象所有的值都进行修改，如果对象其他属性为kong,则在表中数据对应修改为null。

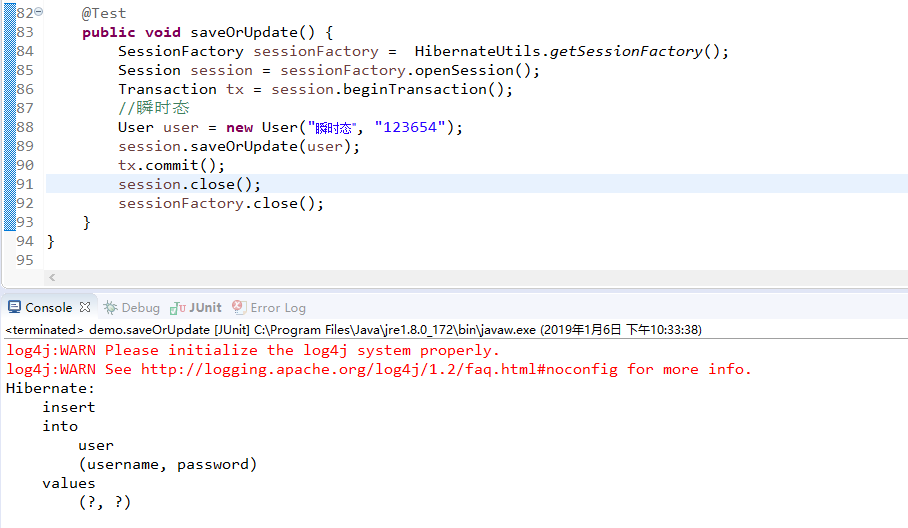
## 删除





## saveOrUpdate

该方法既能添加亦能修改，如果执行的对象时瞬时态，执行的为save,如果是托管态或者持久态则为update操作







# 实体类对象三种状态

## 瞬时态

对象中没有id值，与sesssion也没有关联

## 持久态

对象中有id值，与sesssion也有关联

## 托管态

对象中有id值，但与sesssion并没有关联

# Hibernate的一级缓存

## 什么是缓存

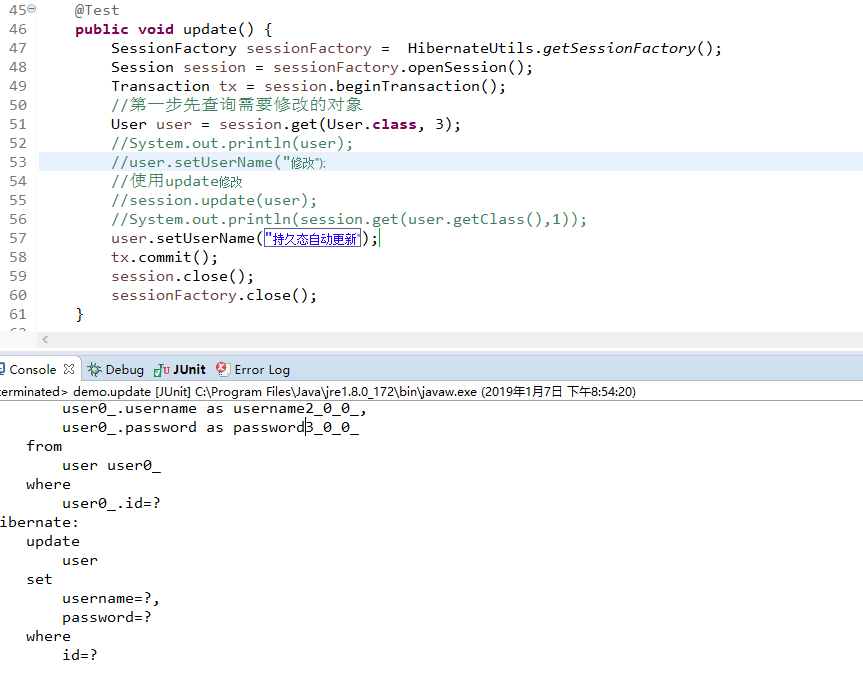
|  |
| --- |
| 1 数据存到数据库中，数据库中是文件系统，需要使用流来操作效率不高   1. 把数据存到中，不需要使用流方式，可以直接读取内存中的数据 2. 把数据放到内存中，提供读取效率 |

## Hibernate缓存

|  |
| --- |
| 1 hibernate框架中提供了很多的优化方式，hibernate的缓存就是一种优化方式  2 hibernate一级缓存：   1. 默认打开 2. 适用范围为session范围（从session创建开始到sesssion关闭） 3. 存储数据为持久态   3 二级缓存   1. 已经不适应，使用redis替代 2. 默认不打开 3. 使用范围为整个项目，即sesssionFactory |

## 缓存特性

持久态自动更新数据库



即是没有调用update方法，但是仍然执行了updat，对数据库中数据进行了更新

# Hibernate事务

## 事务隔离级别

1. mysql默认事务隔离级别 repetable read

## 事务代码规范写法

try{

开启事务

事务代码

提交事务

}catch（）{

捕捉异常，并回归

}finally{

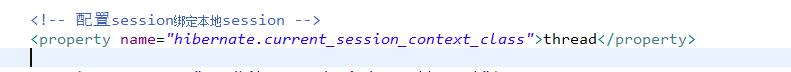
关闭操作

}



## 绑定session

|  |
| --- |
| 1 session类似于jdbc的connection 底层threadlocal   1. 帮实现于本地线程sesssion 2. 获取于本地线程session 3. 在hibernate核心配置文件 4. 使用sessionFactory.getCurrentSession获取绑定session 5. 当绑定线程后，不需要openSession,此时也不需要关闭否则会报错 |



# Hibernate中api的使用

## Query对象

|  |
| --- |
| 1 使用query对象，不需要写sql语句，但是需要写hql语句   1. sql操作的是表中字段，hql操作的是实体类和实体类的属性   2 查询所有hql语句   1. from实体类   3 query对象的使用 |
|  |

## Critica对象

|  |
| --- |
| 1 使用该对象不需要语句，直接调用已封装好的方法 |

## SqlQuery对象

|  |
| --- |
| 1 调用底层sql实现，但用的不多    或者用List<Object>接受 |

# Hibernate一对多操作

## 一对多映射配置

hibernate要求使用集合表示多的数据，使用set集合



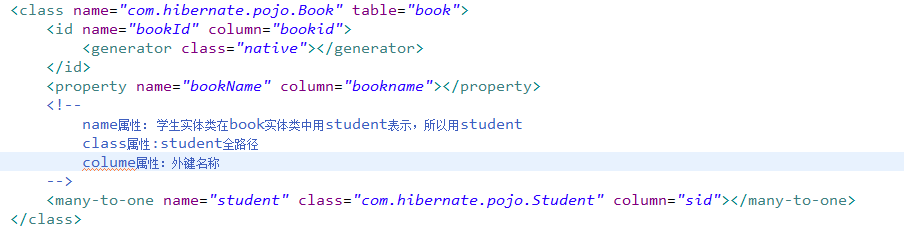
表示一的配置



### 配置一映射关系

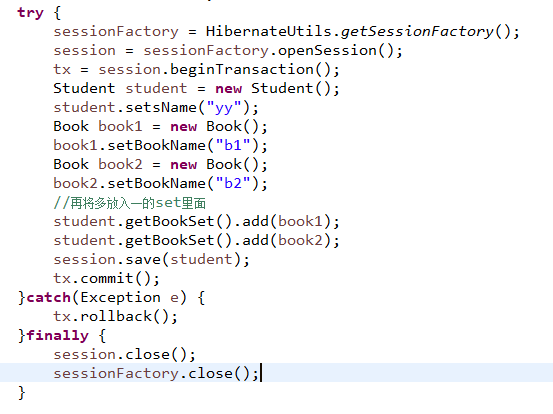


### 配置多的映射关系



## 一对多级联操作

### 添加

1. 添加一个学生，为这个学生添加多本书
2. 

### 删除

1. 删除一个学生，这个学生的所有书也删除



代码略

### 修改

一对多时让一方放弃关系维护，即在set标签上配置inverse，cascade属性失效，即Hibernate不会级联操作



# Hibernate多对多操作

## 多对多映射配置



## 多对对级联保存

与一对多相似 使用cascade

## 级联删除

set中配置cascade标签中的 delete属性

## 维护中间表

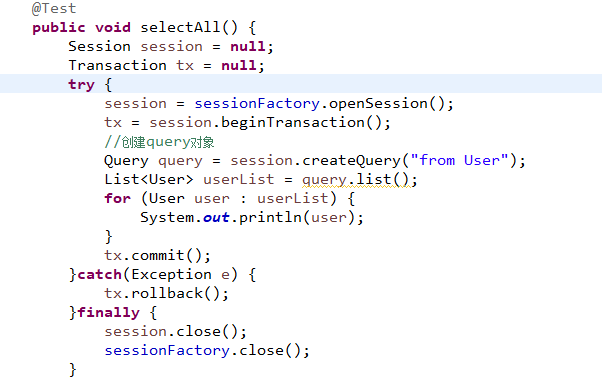
|  |
| --- |
| 1 让某个role拥有emp   1. 先根据id查出role 2. 将emp添加进set中 3. 保存   2 让某个role不拥有emp   1. 先根据id查出role 2. 从set中删除特定emp 3. 保存 |

# Hibernate查询

## Hql查询

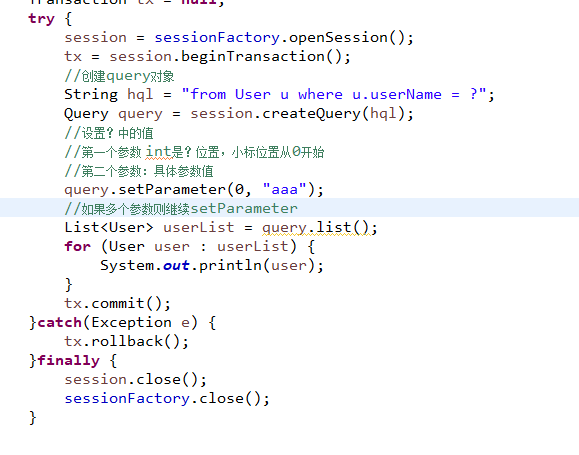
hql操作的是实体类和它的属性

### 查询所有



### 条件查询

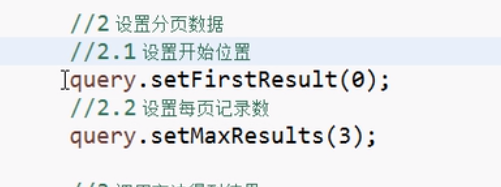
|  |
| --- |
| 1 from 实体类名称 where 实体类属性名称 = ？ and 实体类属性名称 = ？  2 from 实体类名称 where 实体类属性名称 = ？ and 实体类属性名称 like ？ |



### 排序查询

|  |
| --- |
| 1 from 实体类名称 order by 实体类属性名称 asc/dedsc |

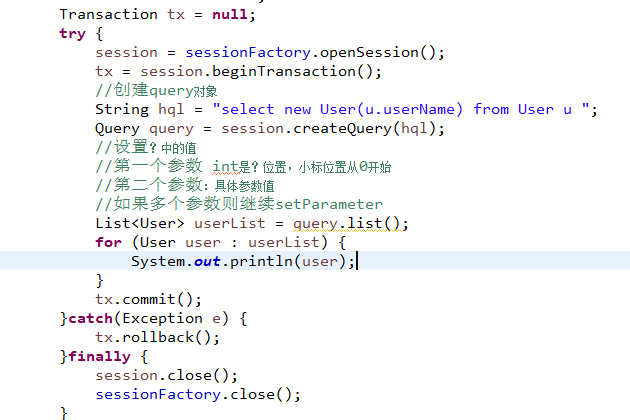
### 分页查询



调用该方法

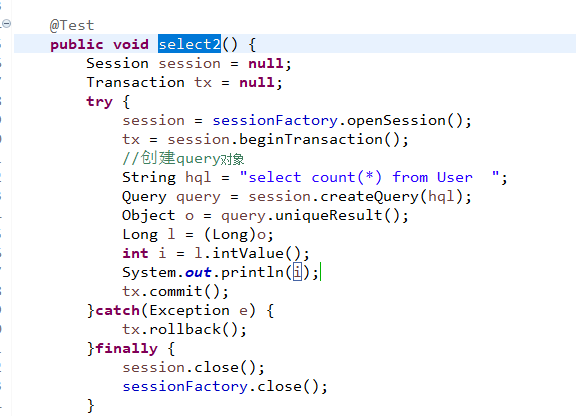
### 投影查询

|  |
| --- |
| 1 投影查询：查询部分的字段  2 hql写法 |



**注意：如果不添加new 实体类名称，那么查出来的list将会是list<Object>而不会封装成对象**

### 聚集函数的使用



**注意：先从Object转换成Long在转换成int类型**