Hibernate学习笔记

目录

[Hibernate学习笔记 1](#_Toc484628371)

[Hibernate课程安排 2](#_Toc484628372)

[Day1 3](#_Toc484628373)

[内容总览 3](#_Toc484628374)

[1.1、Web内容回顾 3](#_Toc484628375)

[Javaee三层框架 3](#_Toc484628376)

[MVC思想 4](#_Toc484628377)

[1.2、Hibernate概述 4](#_Toc484628378)

[1.3、Hibernate入门案例 5](#_Toc484628379)

[搭建Hibernate环境 5](#_Toc484628380)

[1.4、Hibernate配置文件详解 7](#_Toc484628381)

[Hibernate映射配置文件 7](#_Toc484628382)

[Hibernate核心配置文件 8](#_Toc484628383)

[1.5、Hibernate中api的使用 8](#_Toc484628384)

[Configuration-类 8](#_Toc484628385)

[SessionFactory-接口（重点） 8](#_Toc484628386)

[Session-接口（重点） 9](#_Toc484628387)

[Transaction-接口 9](#_Toc484628388)

[1.6、解决配置文件没有提示问题 10](#_Toc484628389)

[Day2 10](#_Toc484628390)

[2.1、实体类编写规则 10](#_Toc484628391)

[2.2、hibernate主键生成策略 10](#_Toc484628392)

[2.3、实体类操作 12](#_Toc484628393)

[crud操作（增删改查） 12](#_Toc484628394)

[实体类对象状态 15](#_Toc484628395)

[2.4、hibernate的一级缓存 17](#_Toc484628396)

[什么是缓存？ 17](#_Toc484628397)

[Hibernate缓存 17](#_Toc484628398)

[验证一级缓存存在 17](#_Toc484628399)

[Hibernate一级缓存执行流程 18](#_Toc484628400)

[Hibernate一级缓存特性 19](#_Toc484628401)

[2.5、hibernate的事务操作 20](#_Toc484628402)

[事务相关概念 20](#_Toc484628403)

[hibernate事务代码规范写法（重点） 21](#_Toc484628404)

[Hibernate绑定session 23](#_Toc484628405)

[2.6、hibernate其他API（查询） 23](#_Toc484628406)

[Query对象 23](#_Toc484628407)

[Criteria对象 24](#_Toc484628408)

[SQLQuery对象 25](#_Toc484628409)

[Day3 26](#_Toc484628410)

[3.1列表功能实现 27](#_Toc484628411)

[3.2表与表之间关系回顾 27](#_Toc484628412)

[1、一对多（客户和联系人） 27](#_Toc484628413)

[2、多对多（用户和角色） 27](#_Toc484628414)

[3、一对一 28](#_Toc484628415)

[3.3 Hibernate一对多操作（重点） 28](#_Toc484628416)

[1、一对多映射配置 28](#_Toc484628417)

[2、一对多级联保存 30](#_Toc484628418)

[3、一对多级联删除 31](#_Toc484628419)

[4、inverse属性 33](#_Toc484628420)

[3.4 Hibernate多对多操作 33](#_Toc484628421)

[1、多对多映射配置 33](#_Toc484628422)

[2、多对多级联保存（重点） 33](#_Toc484628423)

[3、多对多级联删除 33](#_Toc484628424)

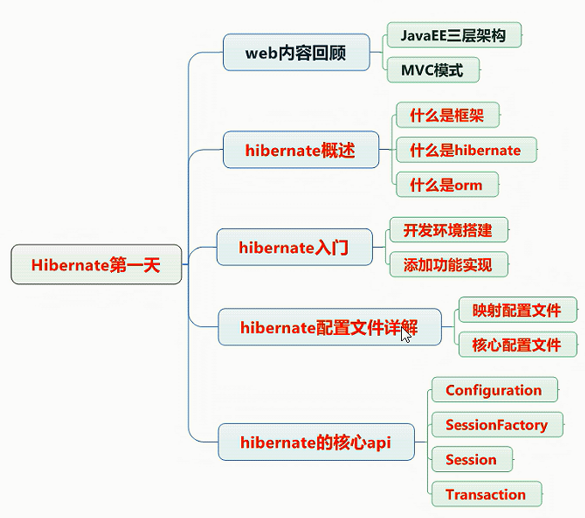
[4、维护第三张表 33](#_Toc484628425)

# Hibernate课程安排

* 第一天：hibernate入门和基本操作
* 第二天：hibernate概念和api使用
* 第三天：hibernate配置一对多和多对多
* 第四天：hibernate查询操作

# Day1

## 内容总览



## 1.1、Web内容回顾

### Javaee三层框架

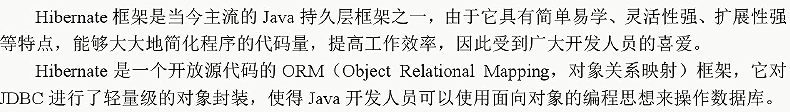
1. Web层：struts2框架
2. Service层：spring框架
3. Dao层：Hibernate框架
   1. 对数据库进行crud操作

### MVC思想

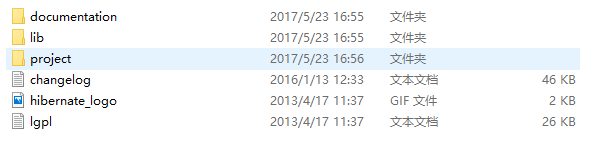
分成思想

1. M：模型
2. V：视图
3. C：控制器

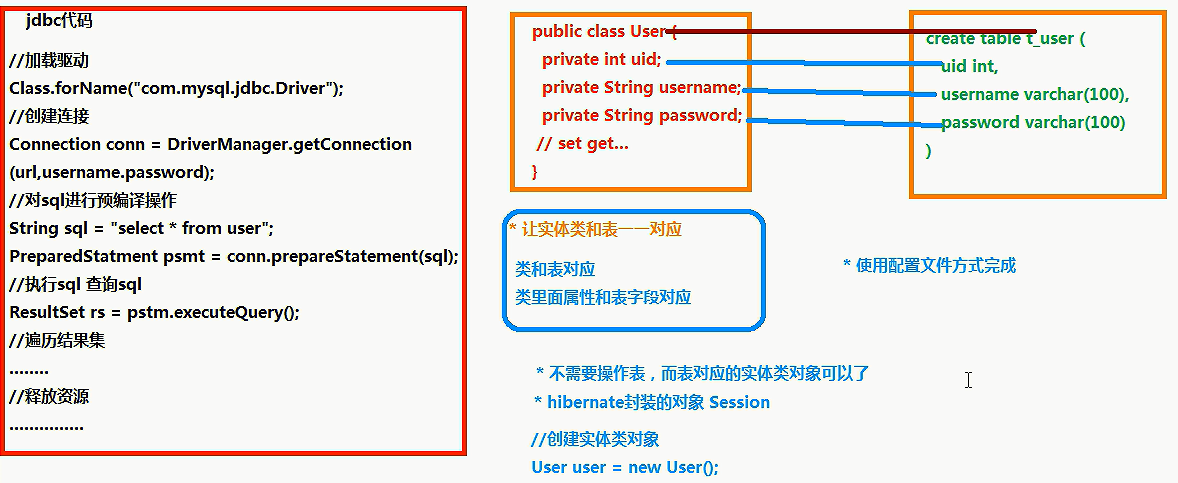
## 1.2、Hibernate概述



1. **什么是框架**
   1. 写程序，使用框架之后，帮我们实现一部分功能，使用框架好吃，少写一部分代码实现功能
2. **什么是Hibernate框架（重点）**
   1. Hibernate框框应用在javaee三层结构中dao层框架
   2. 在dao层里面对数据库crud操作，使用Hibernate实现crud操作，Hibernate底层代码就是JDBC，而Hibernate对JDBC进行封装，使用Hibernate的好处，不需要写复杂JDBC代码了，不需要写SQL语句实现。
3. **Hibernate开源的轻量级的框架**
4. **Hibernate版本**
   1. Hibernate3.x
   2. Hibernate4.x
   3. Hibernate5.x



1. **什么是ORM思想**
   1. Hibernate使用orm思想对数据库进行crud操作
   2. Web阶段，javabean，更正确的叫法是实体类（Javabean）
   3. ORM： Object Relational Mapping 对象关系映射
   4. 文字描述：
      1. 让实体类和数据表进行一一对应关系，让实体类首先和数据库边进行对应，让实体类属性和表里面字段对应
      2. 不需要直接操作数据库表，而操作实体类对象



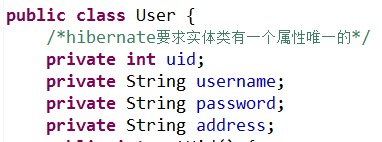


## 1.3、Hibernate入门案例

### 搭建Hibernate环境

第一步、导入Hibernate的jar包

第二步、创建实体类



使用hibernate时候，不需要自己手动创建表，hibernate帮创建表

第三步、配置实体类和数据库表一一对应关系

使用配置文件实现映射关系

1. 创建xml格式的配置文件

映射配置文件名称和位置没有固定要求

建议：在实体类所在包里面创建，实体类名称bm.xml

1. 配置是xml格式的文件，在配置文件中要引入约束

约束有：DTD、scheme，在hibernate中xml使用DTD约束

映射关系代码如下：



第四步：创建hibernate的核心配置文件

1. 核心配置文件格式xml，但是核心配置文件名称和位置是固定

* 位置：必须在scr目录下
* 名称：必须hibernate.cfg.xml

1. 引入DTD约束
2. Hibernate操作过程中，只会加载核心配置文件，其他配置文件不会加载

第一部分：配置数据库信息

第二部分：配置hibernate信息

第三部分：把映射文件放到配置文件里面来

**实现添加操作**

第一步：加载hibernate核心配置文件

第二步：创建SessionFactory对象

第三步：使用SessionFactory创建session对象（ 此Session非网页阶段的session）

第四步：开启事务

第五步：写具体逻辑crud操作

第六步：提交事务

第七步：关闭资源

看到的效果添加一个表

## 1.4、Hibernate配置文件详解

### Hibernate映射配置文件

1. 映射配置文件的名称和位置是不固定的
2. 映射配置文件中，标签中的name属性都是与实体相关的内容
   1. Class标签name属性值：实体类全路径
   2. Id标签name属性值是实体变量的名称
3. Id标签和property标签，column属性可以省略
   1. olumn属性可以省略，其默认值与name一致
4. 在property标签里面有个type属性。设置生成表字段的类型

（hibernate中有自动对应实体类中的类型，不需要写type属性）

### Hibernate核心配置文件

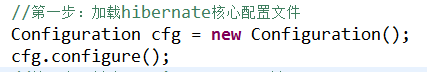
1. 配置要求位置



1. 配置三部分要求
   1. 数据库配置部分（必须）
   2. Hibernate配置部分（可选）
   3. 映射文件引用（必须）
2. 核心配置文件名称和位置是固定的
   1. 位置：src下
   2. 名称hibernate.cfg.xml

## 1.5、Hibernate中api的使用

### Configuration-类



1. 到src下找到名称为：hibernate.cfg.xml配置文件，创建对象，把配置文件放到对象里面（加载核心配置文件）
2. 其他加载方法：

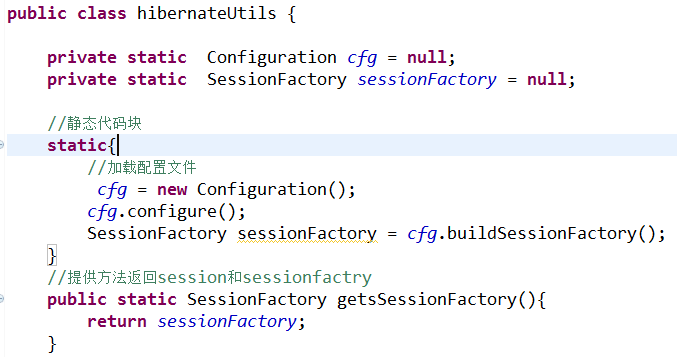
指定路径



### SessionFactory-接口（重点）

1. 使用configuration对象创建sessionFactory对象
   1. 创建sessionFactory过程中做事情
      1. 根据核心配置文件中，有数据库配置，有映射文件部分，到数据库里面根据映射关系把表创建。前提是要写
2. 创建sessionFactory过程中，这个过程特别耗资源
   1. 在hibernate操作中，建议一个项目一般创建一个sessionFactory对象
3. 具体实现
   1. 写工具类，写静态代码块实现

-静态代码块在类加载时候执行，执行一次



### Session-接口（重点）

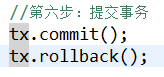
1. Session类似于JDBC中connection
2. 调用session里面不同的方法实现一个crud操作
   1. 添加：sava方法
   2. 修改：uqdata方法
   3. 删除：delete方法
   4. 根据id查询get方法
   5. ·············------还有很多方法
3. Session对象单线程对象，。
   1. 只能一个用户用，不可共用

### Transaction-接口

1. 事务对象



1. 事务提交和回滚方法



回滚就是出现错返回。

1. 事务概念
   1. 事务四大特性

原子性、一致性、隔离性、持久性

## 1.6、解决配置文件没有提示问题

1. 可以上网
2. 把约束文件引入到myeclipse中

# Day2

## 2.1、实体类编写规则

1. 实体类里面属性私有的
2. 私有属性值要有set和get方法
3. 要求实体类中有属性有唯一一个值（一般为id）
4. 实体类属性建议不适用基本数据类型，使用基本数据类型对应的包装类
   1. 八个基本数据类型对应的包装类

Int ----integer

Char-----character

其他都是对应首字母大写：比如double ---- Double

* 1. 比如表示学生分数，假如：int score；

比如学生得0分，int score = 0；

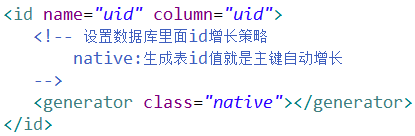
如果表示缺考，，int score =0，不能准确表示学生是否参加考试。

-解决：使用包装类可以了，integer score = 0，表示得0分

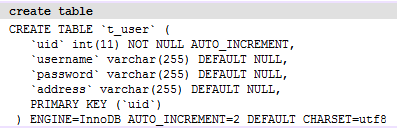
· integer score = null，表示缺考

## 2.2、hibernate主键生成策略

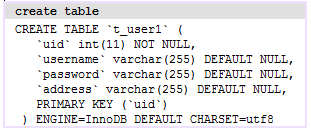
1. Hibernate要求实体类里面有一个属性作为唯一一个值，对应表主键，主键可以不同生成策略
2. Hibernate主键生成策略有很多值，不需要记但只要记两个常用的：

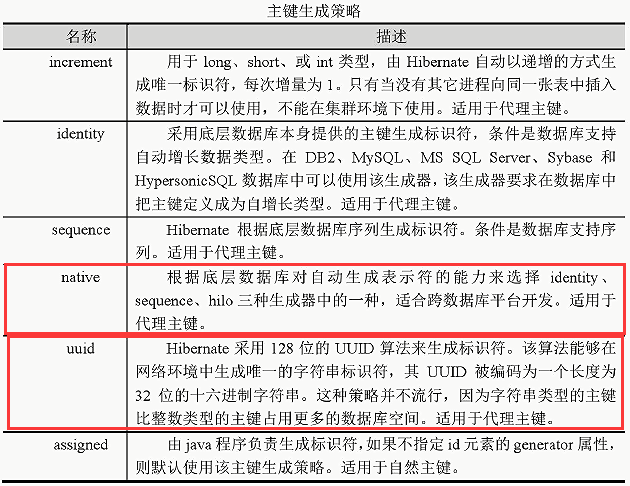


1. 在class属性里面有很多值
   1. Native;根据使用的数据库帮选择哪个值



* 1. UUID：之前web阶段写代码生成UUID值，hibernate自动生成UUID值





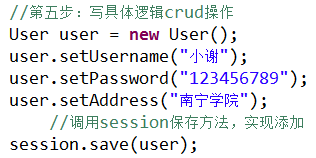
1. 演示UUID的使用
   1. 使用UUID生成策略，实体类ID属性类型，必须字符串类型

## 2.3、实体类操作

### crud操作（增删改查）

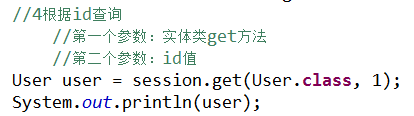
都是调用session的方法来实现对数据库进行crud操作

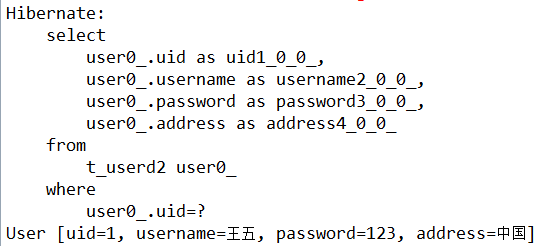
#### 添加操作



#### 根据id查询





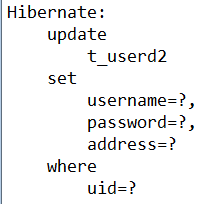


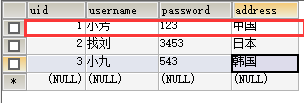
#### 修改操作

根据id查询，返回对象

再使用session的updata方法进行修改





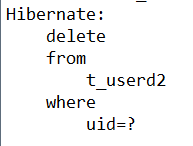


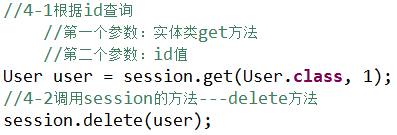
使用SaveOrUpdata是将整列的都修改，只要某个属性没有赋值就为NULL

#### 删除操作

方法一：

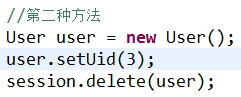
找到对应id后删除





方法二：

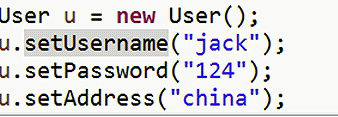
直接设置id值后删除



### 实体类对象状态

1. 实体类状态有三种
   1. **瞬时态（添加）**

对象里面没有设置id值，对象与我们session没有关联





* 1. **持久态（修改）**

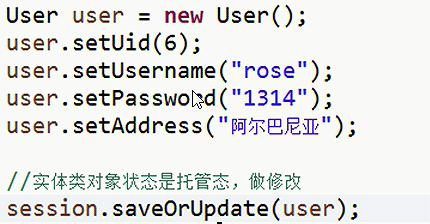
对象里面有id值，对象与session有关联





* 1. **托管态/游离态(修改)**

对象里面有id值，对象与session没有关联





## 2.4、hibernate的一级缓存

### 什么是缓存？

1. 数据存到数据库里面，数据库本来就是文件系统，使用流方式进行操作文件效率不高。
   1. 把数据存到系统的内存里面，不需要使用流的方式直接读取内存的数据
   2. 把数据放到内存中可以提高读取效果

### Hibernate缓存

1. Hibernate框架中提供很多优化方式，hibernate的缓存就是一个优化方式
2. Hibernate缓存特点

第一类：hibernate的一级缓存

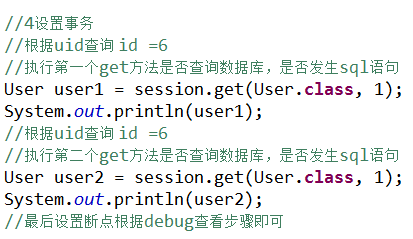
* + 1. Hibernate的一致缓存默认打开（默认可以直接使用了）
    2. Hibernate一级缓存有使用范围，是session范围，从创建到关闭
    3. Hibernate的一致缓存中，存储数据必须是持久态数据

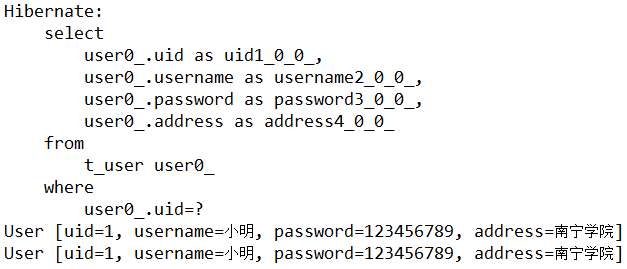
第二类：hibernate的二级缓存

1. 目前已经不使用了，代替技术redis
2. 二级缓存是默认不打开的，需要配置
3. 二级缓存使用范围，是sessionFactory

### 验证一级缓存存在

1. 验证方式
   1. 首先根据uid=1查询，返回对象
   2. 其次再根据uid = 1查询，返回对象





结论如下：

第一次输出的时候出现sql底层语句

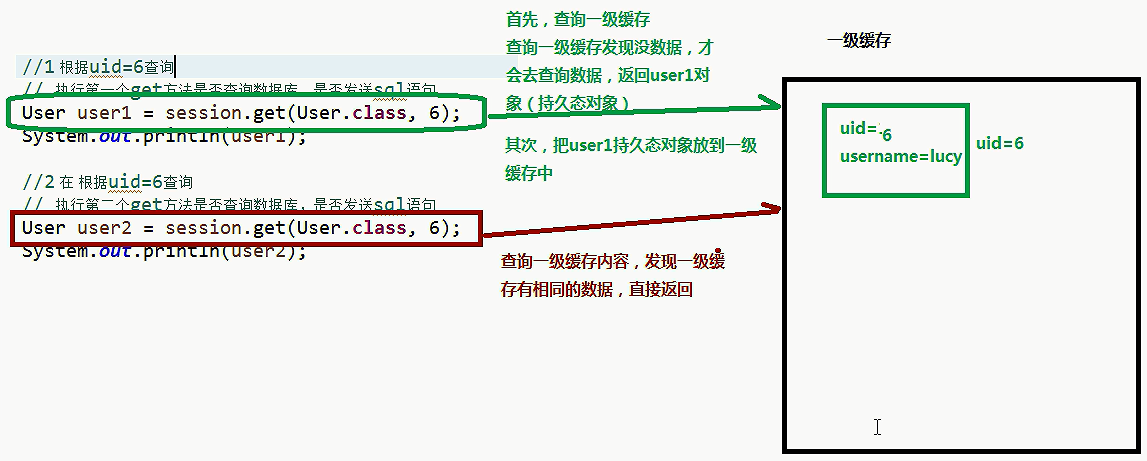
第二次输出的时候直接输出对应的值，因为值在缓存中无需使用SQL语句进行查询获取值

### Hibernate一级缓存执行流程

首先，查询一级缓存，没有对应的数据才去查询数据库

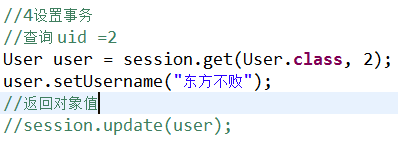
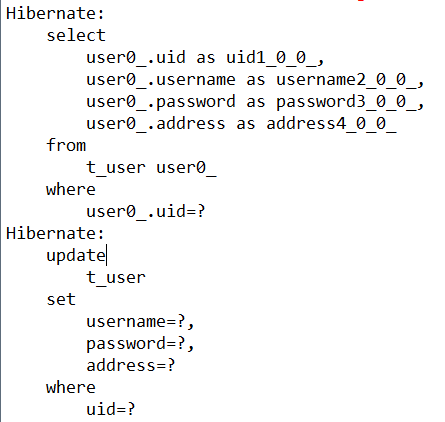
其次，把user1持久态对象放到一级缓存中（不是把对象存放到缓存中而是把数据存放到缓存中）

然后，查询一级缓存，发现有对应的数据，直接返回对应的数据

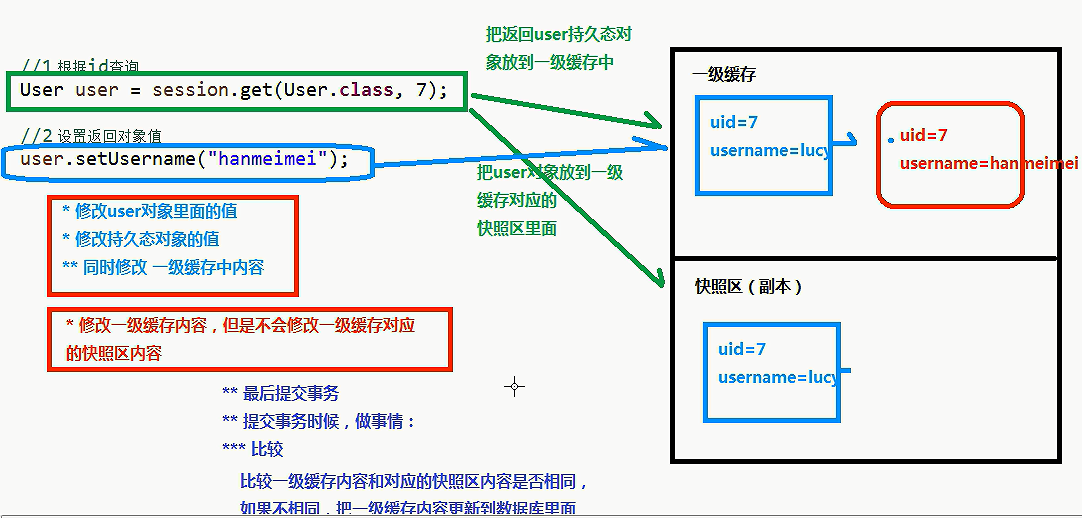


### Hibernate一级缓存特性

1. 持久态自动更新数据库

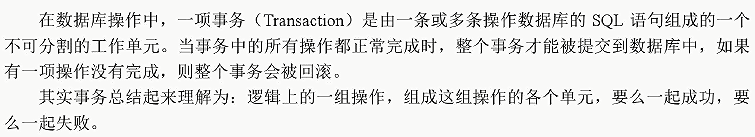
1. 执行过程（了解）
   1. 把首次获取的对象数据放到一级缓存区和一级缓存快照区（副本）
   2. 当修改数据提交后只修改一级缓存区数据，不会去修改快照区。
   3. 当一级缓存区中的数据有改动时，系统会比较一级缓存区与快照区，不一致时，提交事务时会更新到数据库



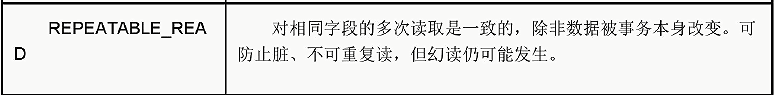
## 2.5、hibernate的事务操作

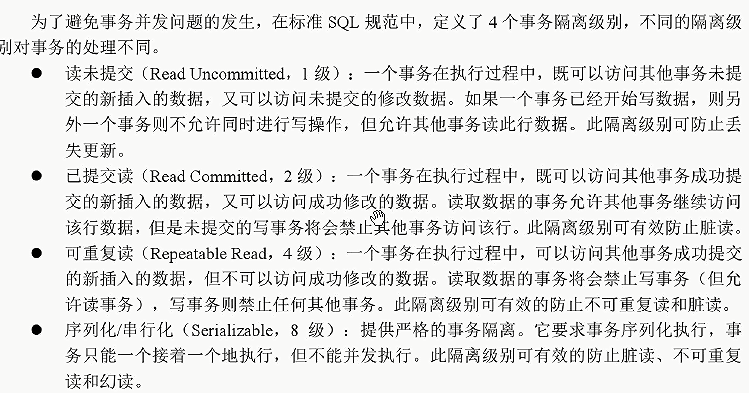
### 事务相关概念

1. 什么是事务

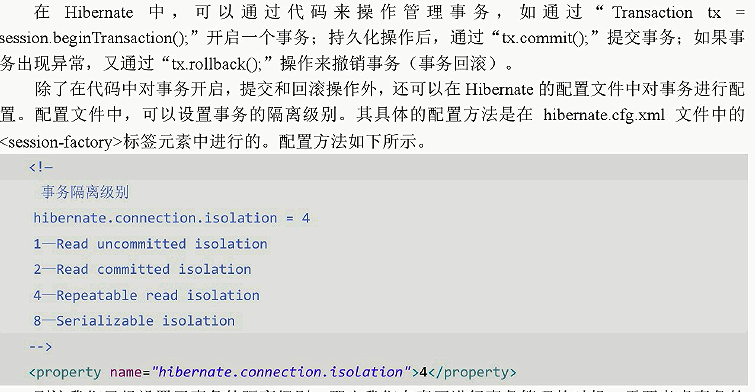


1. 事务特性
2. 不考虑隔离性产问题
   1. 脏读
   2. 不可重复读
   3. 虚度
3. 设置事务隔离级别
   1. Mysql默认隔离级别：repeattable read



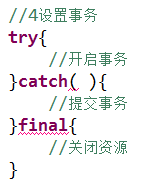


Hibernate中如何设置隔离级别



### hibernate事务代码规范写法（重点）

1. 代码结构

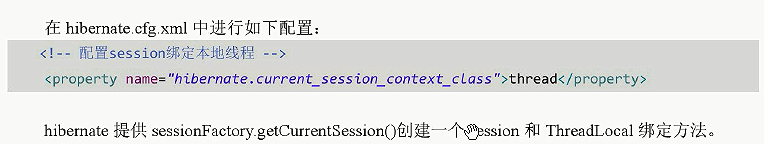


事务代码规范模板

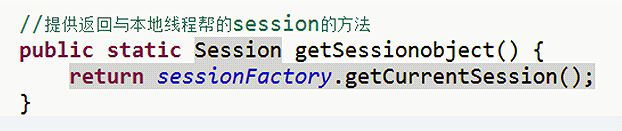


### Hibernate绑定session

1. Session类似于JDBC的cennection，之前web阶段学过threadLocal
2. 帮实现与本地线程绑定session
3. 获取域本地线程session
   1. 在Hibernate核心配置文件中配置



* 1. 调用sessionFactory里面的方法可得到



1. 获取与本地线程绑定session时候，关闭session报错，不需要手动关闭

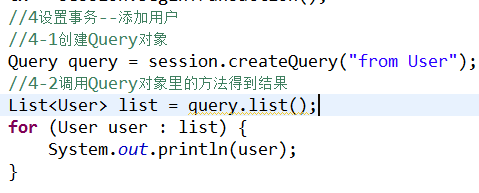
当线程关闭时session也跟着关闭



## 2.6、hibernate其他API（查询）

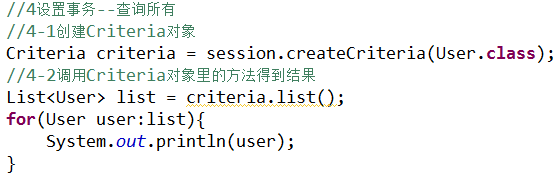
### Query对象

1. 使用query对象，不需要写sql语句，但要写HQL语句
   1. hql：hibernate query language 提供查询语言，这个hql语句与sql语句很相似
   2. hql和sql的区别：
      1. 使用普通sql操作是表和字段
      2. 使用hql操作是实体类和属性
2. 查询所有hql语句：
   1. From 实体类名称
3. Query对象的使用
   1. 创建Query对象
   2. 调用Query对象的方法得到结果

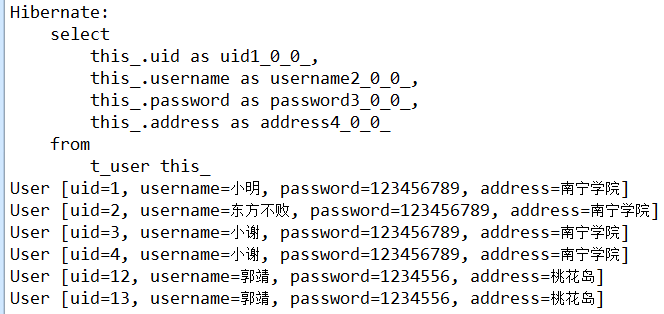


### Criteria对象

1. 使用这个对象查询操作，但是使用这个对象时候，不需要使用查询语句，直接调用其方法
2. 实现过程
   1. 创建Criteria对象
   2. 调用对象里的方法得到结果



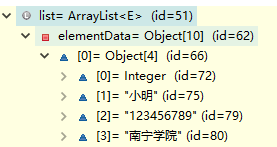
结果：

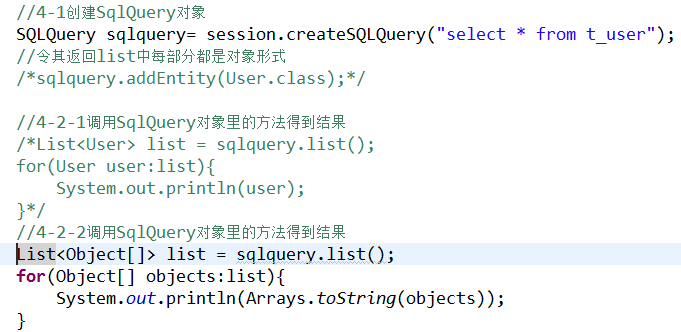


### SQLQuery对象

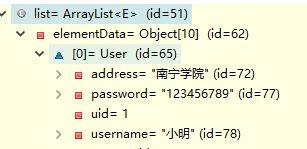
1. 使用hibernate时候，调用底层SQL语句实现
2. 实现过程：
   1. 创建SQLQuery对象
   2. 调用SQLQuery对象里面的方法

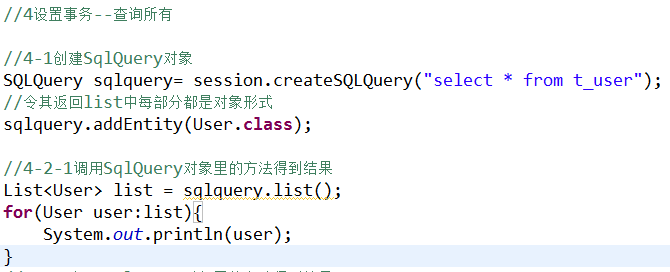
返回list集合每个部分都是数组





返回list集合每部分都是对象（如何实现）





# Day3



## 3.1列表功能实现

## 3.2表与表之间关系回顾

### 1、一对多（客户和联系人）

1. 分类和商品的关系，一个分类里面有多个商品，一个商品智能属于一个分类
2. 客户和联系人
   1. 客户：与公司有业务往来，、百度、新浪、360
   2. 联系人：公司里面的员工，百度里面可以很多个员工，，联系人工
   3. 客户四一联系人是多
   4. 一个客户里面可以有多个联系人，一个联系人只能属于一个客户
3. 一对多表，通过外键建立关系



### 2、多对多（用户和角色）

1. 订单和商品关系，一个订单里有多个商品，一个商品属于多个订单
2. 用户和角色多不多关系
   1. 用户：小王、王五、赵六
   2. 角色：总经理、秘书、司机、保安
   3. 一个用户里面可有多个角色，一个角色里面可以有多个用户
3. 多对多建表，创建第三张表进行记录



### 3、一对一

1. 在中国，一个男人只有一个妻子（一夫一妻制）

## 3.3 Hibernate一对多操作（重点）

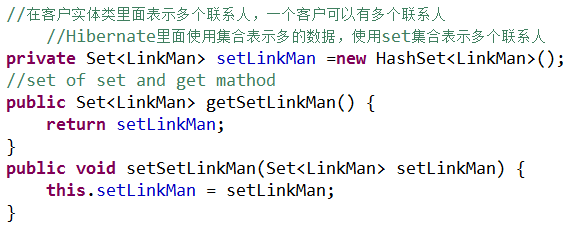
### 1、一对多映射配置

第一步：创建实体类，（以客户和联系人为例） 两个实体类

第二步：让两个实体类之间互相表示

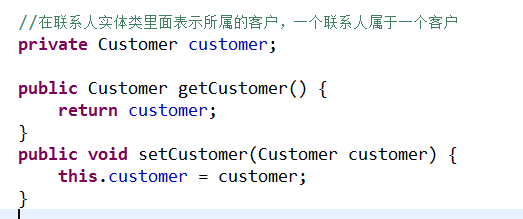
* 1. 在客户实体里面表示多个联系人

一个客户里面有多个联系人



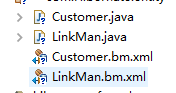
* 1. 在联系人里面要表示所属客户

一个联系人只能属于一个客户

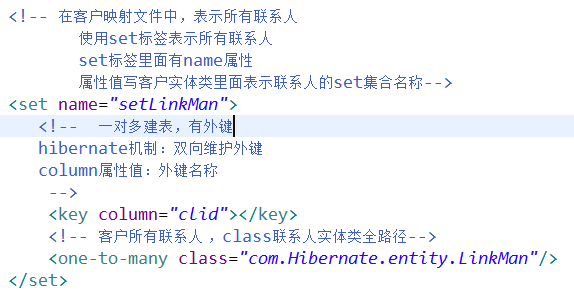


第三步：写两个实体基本映射配置文件

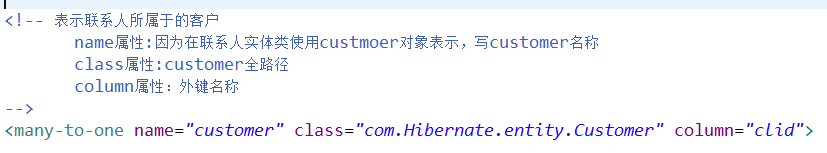
1. 一般一个实体类对应一个映射文件
2. 把映射最基本配置完成



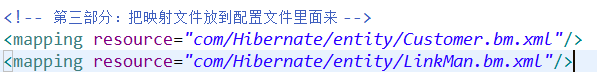
1. 在映射文件中，配置一对多关系
   1. 在客户映射文件中，表示所有联系人

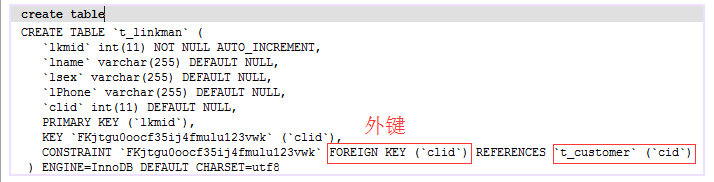


* 1. 在联系人映射文件中，表示所属客户



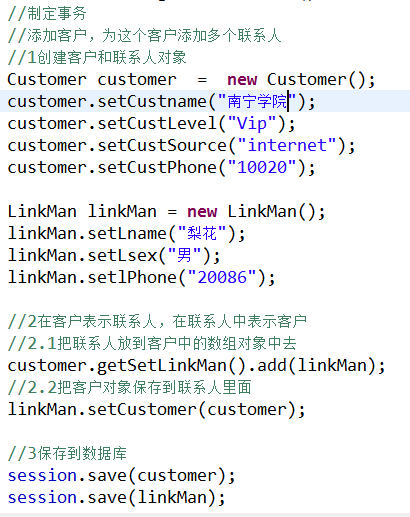
第四步：创建核心配置文件，把映射文件引入到核心配置文件中





### 2、一对多级联保存

1. 添加一个客户，为这个客户添加多个联系人
2. 操作步骤：（复杂的写法）



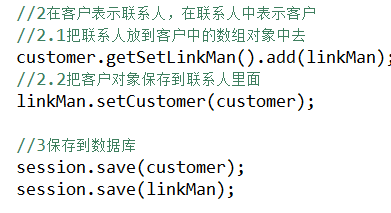




1. 操作步骤：（简单的写法）
   1. 第一步::在客户映射文件中进行配置



* 1. 第二步：创建客户和联系人对象，只需要把联系人放到客户里面就可以了

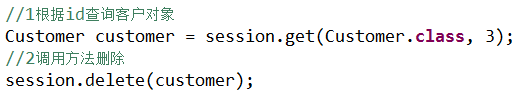


### 3、一对多级联删除

1. 删除某一个客户，这个客户里面的所有联系人也删除
2. 具体实现：
   1. 第一步：在客户映射文件set标签，进行配置

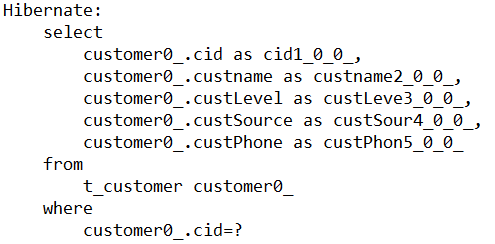


* 1. 第二步：在代码中直接删除客户，
     1. 根据ID查询对象，调用session里面的delete方法删除

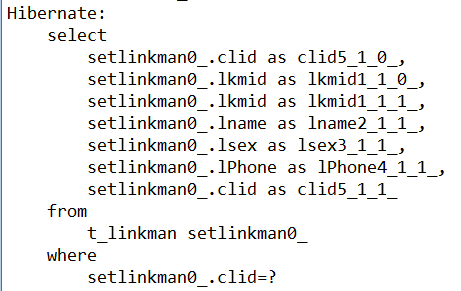


* + 1. 执行过程：

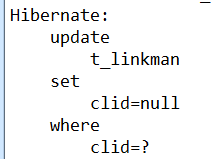
1、根据id查询客户



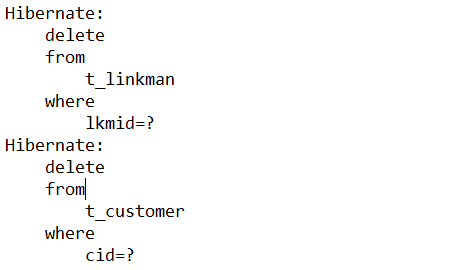
2、根据外键查询联系人



3、把联系人的外键设置成null



4、删除联系人和客户



### 4、一对多级联修改

修改某个联系人所属于的客户

### 5、inverse属性

## 3.4 Hibernate多对多操作

### 1、多对多映射配置

### 2、多对多级联保存（重点）

### 3、多对多级联删除

### 4、维护第三张表