Java开源项目Hibernate深度探险 PDF转换可能丢失图片或格式，建议阅读原文 https://www.100test.com/kao_ti2020/144/2021_2022_Java_E5_BC_80_E6_BA_90_c104_144694.htm 一）Hibernate意义 在一个真正的OOAD中，我们的设计首先是做UML建摸，最终将一个系统涉及所有对象（这个东西不是东西那么简单）用类图来体现一个完整的设计，我们最后可能得到这几种类：控制业务逻辑的类，保存业务数据的类module（bean类），辅助类或者更多（具体问题具体分析，但是将业务所需数据归结为一个类module更适合分层）。到数据库低层实现的时候， 为了获取数据或者存储数据，你不得不为此加上一个操作数据库的控制逻辑，到此，你完美的设计估计会为此付出巨大的努力，因为你看到的业务数据层是一个复杂的模块，即使从面向对象观点来看，我们UML类图中的，业务数据层只是一个数据模块。Hibernate已经帮我们解决了业务数据层这个本来十分复杂的模块的底层实现，现在，我们只要在外层裹上我们的代表数据的类即可。 二）对象模型与关系数据库模型差异 在写出我初探Hibernate的感受之前，我觉得写下这一节还是很有必要的。带着问题研究远远比带着好奇研究要意义深远得多。 问题领域： 关系型数据库是存储数据的最好选择，但是随着OO技术日益发展，在persisitent层上关系型数据库的设计体系与OO体系格格不入，可以想象，当满脑子充斥着OOAD的你想到怎么隔离满天飞的SQL语句时，那是多么痛苦的表情。无论你的业务层设计多么完美，在真正储存数据或者加载数据时，你面对的无非是一大堆封装好的数据，这些数据在JDBC中已经完全失去对象（这里的对象称之为业

务对象或许更为确切）的意义，你整体的OOAD到此为止。为什么会造成这种情况呢？原因是对象模型与关系数据库模型根本设计体系之间的差别。 对象模型与关系数据库模型各自理论出发点是不同的：对象模型的理论体系可以简单归结为这两点： 1） 以对象看待世界。 2） 对象间关系（继承，关联，聚合，组合）维系着整体构成。 而关系数据库模型唯一出发点是有效储存数据，KEY是数据库的关键技术，关系在这里只是各个数据表的KEY之间的关联，这种关联我觉得应该称之为数据的关联，其表达的意义远远没有对象之间的关联那么深广。 那么，我现在最关心的问题是hibernate是怎么利用关系数据库的数据表KEY关联来表达对象之间的关系呢？ 在进入正式研究Hbernate之前，我们可以思索一下问题的似乎简单与似乎十分复杂的矛盾。 我们设计的代表数据层的所有类必须完美的体现在数据表之中。可以这样总结：class- à tableclass1(关系)---class2------〉table1---(关系)-----table2 问题的解决似乎很简单，特别是对于javabean构架，更是简单（看起来简单而已！！！）。 想象一个简单的javabean类：public class SimpleBean{ protected int id. protected String name. public int getId(){ return id. } public void setId(int id){ this.id=id. } public String getName(){ return name. } public void setName(String name){ this,name=name. }} 100Test 下载频道开通，各类考试题目直接下载。详细请访问 www.100test.com