Hibernate

@于龙

目录

[1 面向应用的持久层设计 2](#_Toc422925042)

[1.1 解耦合设计的目标 2](#_Toc422925043)

[1.2 DAO模式 2](#_Toc422925044)

[1.3 数据库连接池 2](#_Toc422925045)

[2 Hibernate概述 2](#_Toc422925046)

[3 Hibernate基础 2](#_Toc422925047)

[3.1 基础语义 2](#_Toc422925048)

[3.2 基础配置 3](#_Toc422925049)

[3.3 实体映射 3](#_Toc422925050)

# 面向应用的持久层设计

## 解耦合设计的目标

应用层解耦合：应用逻辑与数据逻辑相分离

资源层解耦合：逻辑结构与物理结构相分离

## DAO模式

1. 数据存储逻辑分离；
2. 数据访问底层实现分离；
3. 资源管理和调度分离；
4. 数据抽象；

## 数据库连接池

Apache Jakarta Commons DBCP

Proxool

# Hibernate概述

POJO Plain Ordinary Java Object 无格式普通Java对象

# Hibernate基础

## 基础语义

SessionFactory中保存了对应数据库配置的所有映射关系，同时也负责维护当前的二级数据缓存和Statement Pool。

SessionFactory创建过程复杂，代价高昂，因此要考虑其复用策略。

SessionFactory线程安全，一般一个数据库共享一个SessionFactory。

Session相当于JDBC 的Connection，是非线程安全的。

## 基础配置

数据库连接池：默认（仅适用于开发）、C3P0、dhcp、Proxool

## 实体映射

类/表映射

id映射：推荐每个表定义一个与业务逻辑无关的id

属性/字段映射

## 实体映射策略

实体粒度设计

面向设计的粒度细分：组合

面向性能的粒度细分：继承

实体层次设计

3种表模式

## 数据关联

一对一关联

主键关联

唯一外键关联

一对多关联

双向一对多关联

# Hibernate高级特性

## Hibernate持久化实现

实体对象生命周期

实体对象生命周期的三种状态：

1. Transient：与数据库中的记录无关
2. Persistent：实体处于Hibernate所管理的状态
3. Detached：对于P状态的实体，session关闭后就处于Detached状态

T和D的区别在于D能借助某个session实例再成为P状态

T与数据库记录无关联，D与数据库记录存在关联

## 数据缓存

对于ORM，缓存是持久层性能提升的关键

事务级缓存；Session

应用级缓存；SessionFactory

分布式缓存；

Hibernate数据缓存：

内部缓存Session

二级缓存SessionFactory