基于 Hibernate 实现对异构数据库的集成

Integration of Heterogeneous databases based on Hibernate

阳王东 （中南大学信息科学与工程学院 410083）
唐伟佳 （中科软件集团南方信息产业有限公司 413000）

摘要：利用面向对象的 hibernate 数据访问技术，不但可以利用动态的数据库配置来实现不同物理位置的数据的透明访问，本文通过分析数据集成中数据主题域的划分，建立一套与物理数据结构隔离的数据对象，在此基础上利用类的反射机制的实现算法定义了一套统一的数据访问接口，消除了数据集成中对物理数据结构的依赖。

关键词：数据集成 映射 数据访问 异构数据库

本文基于对业务数据进行主题域的划分，采用一种数据对象技术 hibernate 来实现上述的两层数据集成，提供一套只与数据主题域中的数据对象相关的访问接口，不但屏蔽了 hibernate 的实现细节，还屏蔽了数据库中的物理存储结构。并基于该技术介绍了一个城市交通管理数据集成的样例和技术实现方式。

1 Hibernate 介绍

Hibernate 是一个开放源代码（sourceforge）的 O/R Mapping（对象关系映射框架），它于 2001 年推出来最初的 alpha 版本以来，目前已经发展到 3.0（beta）版本。它是对 JDBC 进行了轻量级的对象封装，使 Java 程序员可以随心所欲地使用对象编程思维来操纵关系数据库。它提供一整套面向 Java 环境的对象/关系数据库映射、数据持久化、数据事务操作等开发工具包，还提供一种面向对象的查询语言 HQL，可以实现各种数据查询和数据存取操作[5]。

1.1 Hibernate 体系结构（见图 1）

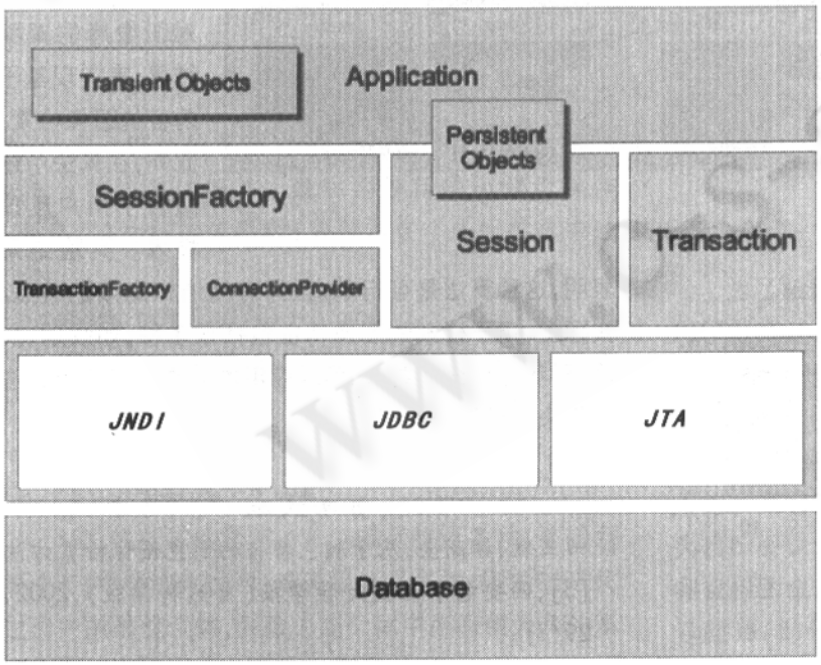

图 1

1.2 结构说明

（1）SessionFactory：它是 Session 的工厂，对编译过的映射文件的一个线程安全的，不可变的缓存快照，相当于 ConnectionProvider 的客户，可持有事务之间重用的数据缓存。

（2）会话 Session：代表应用程序和持久化层之间的一次对话，是单线程，生命期短促的对象。它封装了一个 JDBC 连接，也是 Transaction 的工厂，持有持久化对象的缓存。

(3) 持久化对象(Persistent Object)及其集合(Collection):生命期短促的单线程的对象,包含了持久化状态和商业功能。它们可能是普通的 JavaBeans,唯一特别的是他们现在从属于且仅从属于一个 Session。

(4) 临时对象(Transient Object)及其集合(Collection):目前没有从属于一个 Session 的持久化类的实例。它们可能是刚刚被程序实例化,还没有来得及被持久化,或者是被一个已经关闭的 Session 所实例化的。

(5) 事务,Transaction :(可选) 单线程,生命期短促的对象。应用程序用它来表示一批工作的原子操作,是底层的 JDBC、JTA 或者 CORBA 事务的抽象。一个 Session 可能跨越多个 Transaction 事务。

(6) ConnectionProvider :(可选)JDBC 连接的工厂和池,从底层的 Datasource 或者 DriverManager 抽象而来,对应用程序不可见。

(7) TransactionFactory :(可选)事务实例的工厂,对应用程序不可见。

1.3 Hibernate 特点

(1) 数据面向对象化。客户对数据的访问不再直接和数据库打交道,而是通过对 java 类的访问,从而使数据库对客户透明,并且可以动态设置与数据库的映射。

(2) 灵活的配置。可以通过多种配置方式进行配置。

(3) 查询方法多样化。自定义了一套查询的语言 HQL。

(4) 连接方式多样。用户可以自行提供的 JDBC 连接及驱动来建立数据库连接。

2 基于主题域的数据对象映射

2.1 定义

根据集成数据的业务领域及范围定义好数据集成的两个层面:数据主题域和数据对象。

定义 1:数据主题域 D 为一个相对完备的数据集合,他是由若干数据对象 Q 组成,也就是有,D = {Q1, Q2,……,Qm}。一个数据主题域在实践中可以是一个独立的数据源。

定义 2:设 Q 为数据主题域 D 中的一个数据对象,Q 有若干数据元素组成,也就是有,Q = { q1,q2,……qn},数据对象在应用中可以是数据库中的表和视图。

2.2 约定

在数据主题域上我们进行以下约定:

约定 1:设在数据主题域 D 上任意数据访问关系 Φ(q1,q2,……,qk),q1,q2,……,qk ∈ D,则? Qi1,Qi2,…,Qik 是 D 上的数据对象,有 Φ(Qi1,Qi2,…,Qik),也就是说在 D 上的数据访问关系相对于主题数据域 D 是封闭的。

约定 2:一个数据主题域 D 是共用同一个数据库连接。

2.3 映射规则

从需要集成的数据主题域到 hibernate 的持久化数据类的集合之间建立以下映射规则:

规则 1:对于一个数据主题域 D,则在 hibernate 中创建一个与之对应的会话配置文件,在会话配置文件中指明该数据主题域对应的数据源,例如:hibernate_D. cfg. xml。

规则 2:对于数据主题域 D 中每个数据对象 Q,则创建一个与之相对应的数据持久化类 O,并且为每一个数据持久化类配置一个与数据对象的映射文件,其中 Q 的每一个数据元素对应 O 的一个类属性。

2.4 城市交通管理数据集成的样例

例如在城市交通管理数据集成上,可以分为信号控制系统数据(SignalCtrl)、122/119/110 接处警系统数据(PoliceCall)、交通诱导系统数据(LED)、GPS 巡逻车系统数据(GPS)、车驾管系统数据(Vehicle)、视频监控系统(Video)等。而且这些数据分别属于不同的交通管理业务系统,分别存放在不同的业务数据库中,而这些数据库管理系统有所不相同,例如信号控制系统数据用的是 Oracle 数据库,GPS 系统用的是 MS SQLServer 数据库等。

这样把每一个业务系统划分为一个数据主题域,这样共有 6 各数据主题域,SignalCtrl, PoliceCall, LED, GPS, Vehicle, Video,然后为每个数据主题域创建一个会话配置文件:

- hibernate_SignalCtrl. cfg. xml
- hibernate_PoliceCall. cfg. xml
- hibernate_LED. cfg. xml
- hibernate_GPS. cfg. xml
- hibernate_Vehicle. cfg. xml

· hibernate_Video. cfg. xml

我们以信号控制系统数据(SignalCtrl)为例来配置它的会话配置文件和数据持久化类的映射文件,SignalCtrl 中共有 17 个数据对象,其中数据对象可以相对于与数据库中的表、视图。分别为:

CrossingConfig:监测线圈配置数据

LightConfig:信号等配置数据

Trafficflow:交通流量数据

……

(1) 下面为 SignalCtrl 的每一个数据对象创建一个数据持久化类:

```
public class CrossingConfig implements Serializable
{
    public Integer crossingId;//线圈编号
    public String crossingName;//线圈名称
……}
public class LightConfig implements Serializable { }
public class Trafficflow implements Serializable { }
……
```

(2) 然后为每一个数据持久化类创建一个与数据主题域 SignalCtrl 中相应的数据对象的映射文件(代码略)。

(3) 最后为数据主题域 SignalCtrl 设置会话配置文件 hibernate_SignalCtrl. cfg. xml(代码略)。

数据访问者要对数据对象进行数据访问时,只要访问这些数据持久化类即可,hibernate 就会根据指定的数据主题域的会话配置文件构建数据库会话对象,并把该数据主题域的所有数据持久化类实例化持久化对象,并建立与相应的数据库的连接。而持久化对象根据映射文件中描述的映射规则来实现对对应的数据库表进行操作。如果数据库结构发生改变,则可以动态修改这些映射文件。而这一点对于数据访问者是透明的。

3 统一的数据访问接口的定义

为了实现对数据的统一和标准的访问接口,要向数据访问者屏蔽 hibernate 的数据操作细节,本文定义了一个数据访问接口。

```
Public Interface DataOperate {
    Public void init(String domain);
    Public void beginTransaction();
    Public void endTransaction();
    Public void rollbackTransaction();
    Public String query(String oquery);
    Public Boolean insert(String insert);
    Public Boolean update(String update); Public
    Boolean delete(String delete);
}
```

接口方法说明:

· init:根据传入的数据主题域的名称,系统读入相应的会话配置文件,建立数据库连接,启动一个数据库会话对象,实例化该数据主题域中的所有数据持久化类。

· beginTransaction:在数据库会话中启动一个数据库事务,对于数据的修改、插入和删除需要启动数据库事务,而查询则不必要。

· endTransaction:提交数据库操作事务。

· rollbackTransaction:回滚数据库操作事务。

· query:查询操作,参数是 hibernate 定义的面向对象的查询语句,其实是用面向对象的查询语句来实现对集成的数据主题域的数据项进行查询。结果以 XML 文档返回。

· 插入、修改和删除方法的参数是描述的数据及数据操作的语句。

接口调用过程序列图如图 2 所示。

对于数据访问者只要根据集成的主题域数据生成相应的数据访问请求,再调用数据访问接口就行了,而数据访问组件则根据对数据访问请求,利用 Java 的反射机制来调用访问的数据对象,从而实现对数据库的访问。最后把数据结果封装为 XML 文档返回。实现了对于数据访问者而言,hibernate 也是透明的,他只需要关心定义数据集成的数据接口规范就可以了。

4 数据访问接口的实现

Hibernate 事先可以根据数据主题域的配置文件中配置的不同的数据主题域对应的数据库,建立相应的连接会话 session。

(1) 数据查询操作。假设要对数据域 Qi 和 Qj 中的相应数据项进行查询,f(Qi(qi1,qi2,…qin),Qj(qj1,qj2,…qjm)),则可以把数据查询转换为面向对象的查

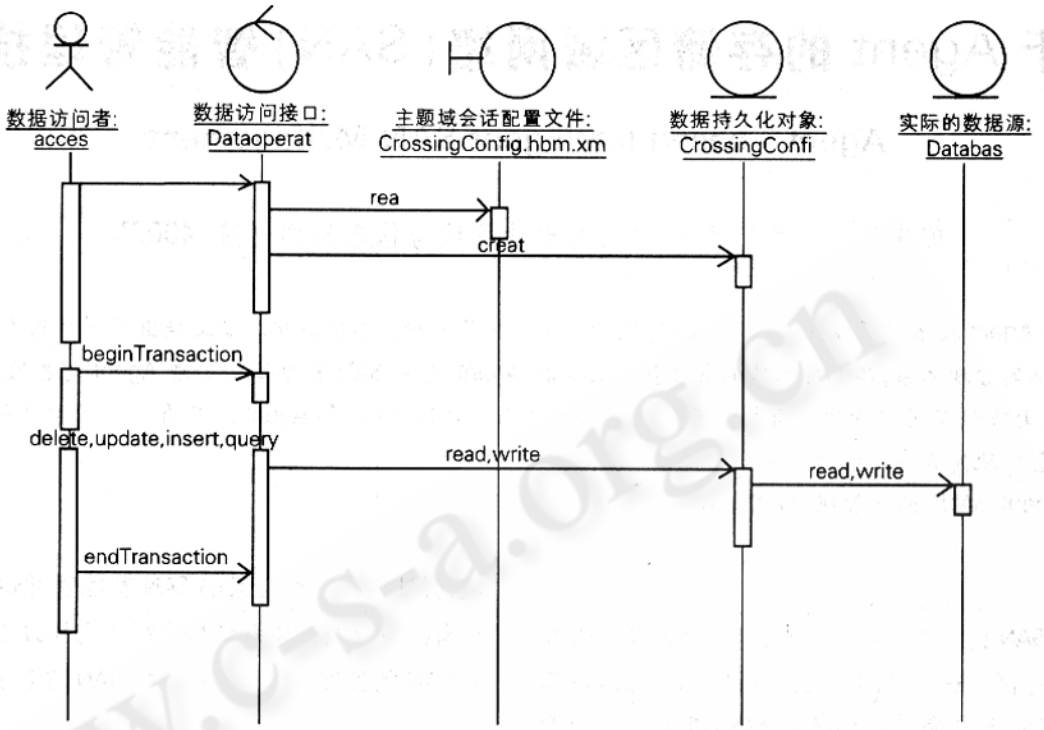

图 2

询语句：select a. qi1, a. qi2, …, a. qin, b. qj1, b. qj1, b. qj2, …b. qjm from Qi a, Qj b where …。然后通过 session 执行上述查询语句，并取得返回结果，返回结果是一个对象列表，然后把对象列表的值取出来封装为 XML 文档。

（2）数据插入操作。要在数据域 Qi 中插入一条数据记录，insert（Qi（qi1, qi2, … qin） value（v1, v2, …, vn），则根据 Java 类的反射机制来实现对该主题域对应的数据对象的操作，具体实现算法略。

（3）数据修改操作，数据修改先要利用数据对象从数据库查询出满足条件的数据，然后修改数据对象的相关属性，再保存该数据对象，数据修改方法与数据插入相类似，只不过数据插入是新创建一个数据对象，而数据修改则是从数据库查询出满足条件的数据而创建的，都是利用了 Java 的反射机制来实现对数据对象属性的赋值。

（4）数据删除操作，它与数据查询操作类似，传给 hibernate 一个删除语句，利用连接会话 session 执行删除语句。

5 总结

本文主要介绍了 hibernate 的技术架构，并介绍了对集成数据集合划分主题域，从而建立一种到 hibernate 的数据对象的映射机制，通过 hibernate 的数据对象屏蔽了在数据集成中，数据访问者对数据库物理结构的依赖。

参考文献

1 王海波、耿晖、姜吉发等，基于 XML 的数据交换的实现[J]，计算机应用，2001, 21(4)。
2 徐周、黄上腾，基于 XML 实现数据库间信息交换的方法[J]，计算机工程，2001, 27(4)。
3 陈殿波、隋树林，异构数据源数据共享的实现，计算机与信息技术。
4 http://hibernate. sourceforge. net/hibernate – mapping – 2. 0. dtd.
5 http://www. hibernate. org/hib_docs/reference/zh – cn/pdf/hibernate_reference. pdf.