基于 Hibernate 的 XML 数据存储方法①

Hibernate's XML-Based Data Storage Method

肖辉辉[1] 段艳明[1] 兰小机[2] (1.河池学院 计算机与信息科学系 广西 宜州 546300;
2.江西理工大学 建筑与测绘学院 江西 赣州 341000)

摘 要: 随着 Web 技术及其应用的快速发展，XML 已经成为万维网上信息表示和数据交换的一个重要标准。当前，学术界对于 XML 数据管理的研究，其内容广泛，如 XML 数据存储、XML 数据查询、XML 索引、Native XML 数据库、XML 流处理、XML 数据发布等。聚焦于其中的 XML 数据存储，即基于 Hibernate 的 DOM4J 技术把 XML 数据存储到关系数据库中的应用研究。

关键词: XML 关系数据库 PO POJO Hibernate

1 引言

近十年来，随着 Web 技术及其应用的快速发展，在互联网上，各种信息大量涌现，使网络逐渐成为汇集信息的海洋。这些信息形式各异，从其数据载体的结构特征角度上分析，这些信息基本上分为三类：结构化数据信息、半结构化数据信息和无结构化数据信息。其中半结构化数据具有不规则、多变的结构特征，在当今的互联网上最为广泛地存在。可扩展标记语言 XML 是半结构化数据常用的一种表现形式。随着 Web 技术的广泛应用，半结构化数据信息也日益膨胀，XML 已经成为万维网上信息表示和数据交换的一个重要标准。但随着 XML 数据的日益膨胀和 XML 应用的不断发展，给我们带来一个巨大的难题：即如何有效地管理 XML 数据？针对这个问题，当前，学术界对于 XML 数据管理的研究，其内容广泛，如 XML 数据存储、XML 数据查询[1]、XML 索引、XML 流处理、Native XML 数据库[2,3]、XML 数据发布等。

目前在存储 XML 数据的数据库技术上，形成两大阵营[4]：一种阵营主张在原有的传统关系数据库基础上，通过扩展 XML 支持模块或中间件，来完成 XML 数据库之间的格式转换和传输；另一种阵营主张利用 NXD 技术来解决 XML 文档的存储管理[5]。事实上，NXD 非常具有发展潜力，更适合 XML 文档的存储，而关系数据库技术则非常成熟。因此，在实际应用中如何寻求更切合实际的最佳实施策略对于 XML 文档的有效存储非常重要。本文试图通过在 XED(关系数据库)和 Hibernate，来实现 XML 数据的有效存储。

2 Hibernate技术

Hibernate 是对象/关系映射(ORM, Object Relational Mapping)的解决方案，简单的说，就是将 Java 对象与对象关系映射至关系数据库中的数据表与数据表之间的关系，Hibernate 提供了这个过程中自动对应转化的方案。

Hibernate 是 Java 应用和关系数据库之间的桥梁，它负责 Java 对象和关系数据库之间的映射。Hibernate 内部封装了通过 JDBC 访问数据库的操作，想上层应用提供了面向对象的数据访问 API。Hibernate 是一个开放源码的对象关系映射框架，它对 JDBC 进行了非常轻量级的对象封装，使得 Java 程序员随心所欲的使用面向对象编程思维来操作数据库。

2.1 Hibernate 结构体系及其核心组件[6]

Hibernate 可以作为模型层/数据访问层。它通过配置文件(hibernate.Properties 或 hibernate.cfg.xml)和映射文件(.hbm.xml)把 Java 对象或持久化对象(Persistent Object，PO)映射到数据库中的数据表，

① 基金项目:国家自然科学基金(40761017)
收稿时间:2009-02-13

然后通过操作 PO,对数据库表中的数据进行增、删、改、查等操作。Hibernate 的体系结构如图 1 所示。

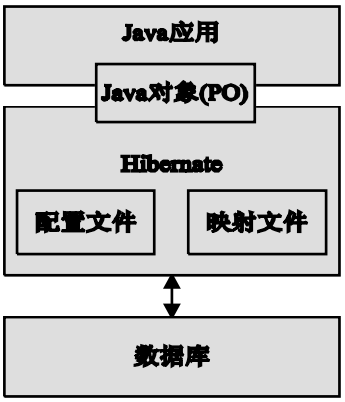

图 1　Hibernate 的体系结构图

除配置文件(hibernate.properties 或 hibernate.cfg.xml)、映射文件(.hbm.xml)和持久化类(PO)外，hibernate 的核心组件还包括以下几部分：Configuration 类：用来读取 Hibernate 配置文件，并生成 SessionFactory 对象。SessionFactory 接口：产生 Session 实例的工厂。Session 接口：用来操作 PO(持久化对象 Persistent Object)。它有 get()，load()，save()，update()和 delete()等方法用来对 PO 进行加载、保存、更新及删除等操作。它是 Hibernate 的核心接口。Query 接口：用来对 PO 进行查询操作。它可以从 Session 的 createQuery()方法生成。Transaction 接口：用来管理 Hibernate 事务。它的主要方法有 commit()和 rollback()，可以从 Session的beginTransaction()方法生成。Hibernate 配置文件主要用来配置数据库连接参数，例如数据库的驱动程序，URL，用户名和密码等。映射文件(.hbm.xml)用来把 PO 与数据库中的数据表、PO 之间的关系与数据表之间的关系，以及 PO 的属性与表字段一一映射起来，它是 Hibernate 的核心文件。持久化对象(PO)可以是普通的 JavaBeans，唯一特殊的是它们与(仅一个)Session 相关联。

2.2 Hibernate 的运行过程

Hibernate 的运行过程如下：应用程序先调用 Configuration 类，该类读取 hibernate 配置文件及映像文件中的信息，并用这些信息生成一个 SessionFactory 对象，然后从 SessionFactory 对象生成一个 Session 对象，并用 Session 对象生成 Transaction 对象；可通过 Session 对象的 get()，load()，save()，update()，delete()和 saveOrUpdate()等方法对 PO 进行加载、保存、更新及删除等操作；在查询的情况下，可通过 Session 对象生成一个 Query 对象，然后利用 Query 对象执行查询操作；如果没有异常，Transaction 对象将提交这些操作结果到数据库中。Hibernate 的运行过程如图 2 所示。

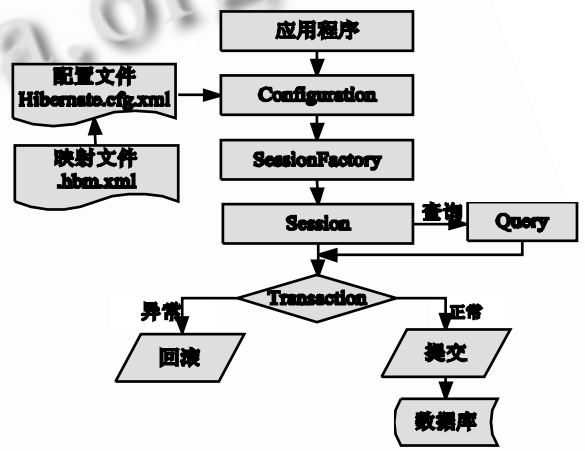

图 2　Hibernate 各组件之间的关系

3　基于Hibernate的XML数据存储过程

Hibernate 允许将 XML 数据映射到实体模型中，这样就可以使用一般的会话方法访问 XML 数据。这个功能主要来在底层关系数据库中存储和读取 XML 数据。Hibernate 使用 Dom4J 作为进行 XML 访问的 API，因为 Hibernate 内部已经使用 Dom4J 读取配置和映射文件。

XML 数据在关系数据库中存储的关键是要将 XML 模式映射为关系模式，因此，在将 XML 数据存入关系数据库之前应首先建立它们的映射关系。建立好映射文件后，根据该文档生成 XML 文档与数据库之间的关联关系，然后使用 Dom4J 实体模式的会话将数据导入到Hibernate,从而实现在底层关系数据库中导入 XML 数据。XML 数据存储到关系数据库中的过程如下：

① 创建 Hibernate 配置文件

Hibernate 通过配置文件来获得连接数据库和决定映射所需的所有信息，本论文使用配置文件 Hibernate.cfg.xml 来实现连接数据库和决定映射所需的所有信息。配置文件 Hibernate.cfg.xml 的核心代码如下:

```
<!—XML 文档实例的 XML 声明-->
<?xml version='1.0' encoding='UTF-8'?>
<!--XML 文档类型定义-->
<!DOCTYPE hibernate-configuration PUBLIC"-//Hibernate/Hibernate Configuration DTD 3.0//EN" "http://hibernate. source-forge.net/hibernate-configuration-3.0.dtd">
<!-- Generated by MyEclipse Hibernate Tools. -->
<hibernate-configuration>
<!--各属性的配置. -->
<session-factory>
<!—访问数据库的用户名 -->
<property name="connection.username">sa</property>
<!--数据库驱动程序的地址 -->
<property name="connection.url">
jdbc:microsoft:sqlserver://localhost:1433
</property>
<!--所使用的数据库的类型 -->
<property name="dialect">
org.hibernate.dialect.SQLServerDialect
</property>
<!--数据库驱动程序的名称 -->
<property name="myeclipse.connection.profile">
SQL Server2000
</property>
<!—访问数据库的密码 -->
<property
name="connection.password">whoamireal
</property>
<!--数据库驱动类 -->
<property name="connection.driver_class">
com.microsoft.jdbc.sqlserver.SQLServerDriver
</property>
<!--HBM 映射文件. -->
<mapping
resource="demo/Product.hbm.xml" />
</session-factory>
</hibernate-configuration>
```

② 编写 POJO

在理想环境中，获得任何 Java 对象并将它持久化到数据库中都应该是很轻松的。不需要为此编写特殊的代码，也不会有性能损失，而且结果是完全可移植的。Hibernate 已经非常接近这个目标了，至少与其他替代方式相比已经很方便了，但还是需要创建配置文件，而且要考虑微妙的性能问题。但无论如何，Hibernate 实现了它的基本目标—使我们能够在数据库中存储 POJO(Plain Old Java Object,普通 Java 对象)。图 3 说明了 Hibernate 如何在客户机代码和数据库之间起到桥梁的作用。由于篇幅所限制，我们把编写 POJO 的代码在此省略。

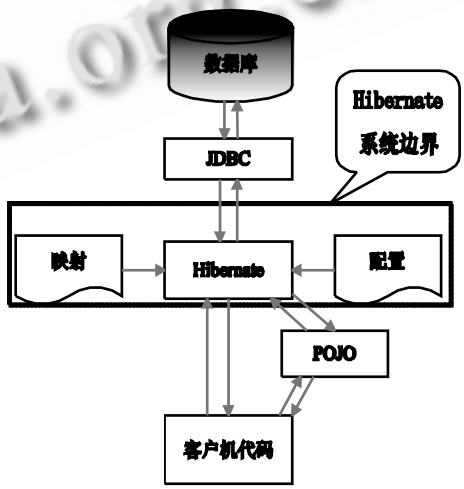

图 3 Hibernate 在 Java 应用程序中的角色

③ 创建映射文件

有了 POJO，就需要将它们映射到数据库，方法是

直接表示每个 POJO 的字段，或间接地表示为相关联的表的列值。创建映射文件的核心代码如下：

```
<!—XML 文档实例的 XML 声明-->
<?xml version="1.0" encoding="utf-8"?>
<!--XML 文档类型定义-->
<!DOCTYPE hibernate-mapping PUBLIC "
-//Hibernate/Hibernate Mapping DTD 3.0//EN"
"http://hibernate.sourceforge.net/hibernate-mapping-3.0.dtd">
<!--Mapping file autogenerated by MyEclipse Persistence Tools
    -->
<hibernate-mapping>
<!--指定 Product 类与数据库中 product 表的映射. -->
<class name="demo.Product" table="product" node="product" schema="dbo" catalog="hibernate_test">
    <!--name 表示 Product 类中的属性名字. -->
    <!--node 表示 XML 文档中的节点名字. -->
    <!--column 表示表中的字段名字. -->
        <id name="id" node="@prod_id" column="id"></id>
        <property name="sku" node="@sku" column="sku"/>
<property name="description" node="description"
column="description"/>
<property name="listPrice" node="list_price" column="list_price" />
<property name="basePrice" node="base_price" column="base_price"/>
<property name="orderPrice" node="order_price"column="order_price">
        <property name="b" node="b" column="b"/>
    </class>
</hibernate-mapping>
```

4　XML数据存储到关系数据库中的实现

Hibernate 支持采用 dom4j 作为操作 XML 树的 API。你可以写一些查询从数据库中检索 dom4j 树，随后可对这棵树做的任何修改都将自动同步回写到数据库。你甚至可以用 dom4j 解析一篇 XML 文档，然后使用 Hibernate 的任一基本操作将它写入数据库：persist()，saveOrUpdate，merge()，delete()，replicate()(合并操作 merge()目前还不支持)。通过创建 Hibernate 配置文件、编写 POJO、创建映射文件之后，我们可以编程来实现 XML 数据存储到底层 XED(关系数据库)中，实现的核心代码如下：

```
public class ImportXML
{   ……
 //构造一个 SAXReader 对象
 SAXReader Reader=new SAXReader();
 //解析一个 XML 文件并返回一个文档对象
  Document document=Reader.read("d:\\product.xml");
 //构造一个 List 对象
     List users = document.selectNodes("//product ");
  //打开一个传统的会话
  Session session =sessionFactory. OpenSession();
 //获得一个 Transaction 对象（启动事务）
  Transaction transaction = session.beginTransaction();
 //获得一个 dom4j 实体模式的会话对象
Session dom4jSession= session.getSession(EntityMode.DOM4J);
 //构造一个 Iterator 对象
 Iterator iter = users.iterator();
 //判断还有没有对象可以迭代
 while (iter.hasNext())
```

(下转第 163 页)

(上接第 192 页)

```
{
        //返回迭代的下一个对象
        Object next = iter.next();
       //利用 Hibernate 的 XML 持久性机制——saveOrUpdate 方法把 XML 数据存储到底层 XED(关系数据库)中
            dom4jSession.saveOrUpdate("demo.Product ", next );
    }
    ......
    }
```

5 结语

本文是基于 Hibernate 的 XML 数据存储方法的研究，由于 Hibernate 对数据库中数据操作的高效性，通过 Hibernate 来存储 XML 数据，使 XML 数据存储到 XED 数据库中不失一种好的存储方法的探索。对 XML 数据管理的研究是当前学术界的热点，本论文的研究只是涉及这个领域的一小部分，还有好多领域没有研究。如 Native XML 数据库、XML 数据查询路径、XML 索引等方面有待研究。

参考文献

1 王国仁,等.XML 数据管理技术.北京:电子工业出版社, 2007.

2 冯建华,钱乾,等.纯 XML 数据库研究综述.计算机应用研究, 2006,(6):1－7.

3 刘刚,喻成.Native XML 数据库的研究与应用.微机发展, 2005,15(8):65－67.

4 万常选.XML 数据库技术.北京:清华大学出版社, 2005.

5 兰小机,肖辉辉,段艳明.基于扩展 NXD 的 GML 空间数据库的数据查询系统的研究.大地测量与地球动力学, 2008,28(1):86－91.

6 Minter D,et al. Hibernate 基础教程.北京:人民邮电出版社, 2008.