DOI:10.13374/j.issn1001-053x.1999.04.021

第21卷 第4期 北京科技大学学报 Vol.21 No.4
1999年 8月 Journal of University of Science and Technology Beijing Aug. 1999

JAVA 语言实现对象关系数据库的方法

刘庆文 杨 扬

北京科技大学 信息工程学院,北京 100083

摘 要 JAVA 语言的产生以及 JAVA 虚拟机的跨平台性为开发新的对象关系数据库提供了方便,提出了一种按照 Atkinson 正交持久化理论持久化对象,按照 Association Algebra 中关联算子计算关联集来访问持久对象集的对象关系数据库的原理与方法.

关键词 对象关系数据库; JAVA; PERSISTENCE

分类号 TP309

目前关系数据库的研究已基本趋近成熟,而对象关系数据库的研究成为重点.典型的对象关系数据库系统有两类:一类是由关系数据库系统增加对类的支持和对象机制而构成的,如 Informix 中 datablade 模块,这种系统保持了与关系数据库系统的兼容性,但却是阻抗失配的,即两种语言的类型系统不一致.另一类是在面向对象语言中增加对象持久功能而构成的,如 O2,ObjectStore[1],这种系统是阻抗一致的,但却缺少 SQL 这种集合范型的查询语言设施,另外,由于 ObjectStore 附着在 C++上,由于编译系统的支持,持久对象不能使用持久化模式中的方法.此外,Glasgow 大学 Atkinson 教授一直倡导正交持久化语言.正交持久化认为语言中所有的对象潜在都有同样的生命期.正交持久化设计的三原则是:(1)正交持久.不论什么类型的对象都可以持久化,也就是说不仅对象被持久化了,而且对象的类型也被持久化了;(2)持久传递.对象的生命期是由它们的可达性来决定的,如果一个对象在程序执行之后仍然存在,要么它被标识为一个持久根,要么它可以被一些已持久的对象引用;(3)持久独立.要求程序代码不管是对长期数据(产生于稳定存储并且生命期超过程序的执行)或者是暂态数据(在一个常规程序执行过程中产生并且只存在于程序的常规执行过程中).传统的程序设计语言如果没有专用编译程序就不能实现对象正交持久化,特别是不能支持方法.JAVA 为对象持久化提供了支持,因为只要支持 JAVA 虚拟机,就可以作到不仅使对象持久也可以使类型持久,同时实现正交三原则.对于对象关系数据库,只有对象持久化仅仅是实现了对象的无缝连接也就是阻抗匹配,仍然不能进行应用中需要的集合操作,因此还需要有对象数据库理论的集合语言.目前,Association Algebra 是有影响的一种对象关系理论[2].如果要在 JAVA 语言中直接支持集合范型语言则需要对 JAVA 语言的语法进行扩充.我们认为可以在 JAVA 语言的基础上实现对象的正交持久化,同时支持 Association Algebra 的客户机服务器的对象关系系统.

1 JAVA 对象关系数据库的原理

JAVA 对象关系数据库是基于 Atkinson 的正交持久化和 Su 的关联代数理论.一个典型的正交持久化程序[1]如下:

```
public class SaveSpag{
    public static void main(String[] args){
    Spaghtti sp1=new Spaghetti(27);
    Spaghtti sp2=new Spaghetti(5);
    sp1.add("something1");
    sp2.add("something2");
     try{
     PjavaStore pjs = PjavaStore.getStore();
    Pjs.newPRoot(r1,sp1);
catch(PJSexception e){...}}}
```

newPRoot 方法建立一个 String 和 sp1 之间的约束,并把它记录为一个持久根.在函数执行

1998-09-12 收稿 刘庆文 男,27岁,硕士

的最后，与 main 隐式相关的事物将要提交，随后新标识的持久对象和所有从持久对象可达的对象都被提升到持久状态. 用以下方法重用持久对象.

```
public class SpagShow{
public static void main (String[] args) {//start traction
try{ //catch store exception
PjavaStore pjs=PjavaStore.getStore();//obstain persistent store
Spaghetti sp=(Spaghetti)pjs.getPRoot(r1);//get persistent object
Sp.display();
}catch(PJSException e){...}}}//handle exception
```

PjavaStore 对象用来得到一个持久根，getPRoot 方法得到持久对象. Atkinson 的正交持久化是基于命名，我们认为对象持久应该是基于对象标识 oid 的. 因此，我们修改 JavaStore 的方法使之是支持 oid.JavaStore 在加入持久对象之前，首先寻找此对象的持久根，但不是以命名寻找而是以此对象的类为模板寻找，如果找到根则将此对象加入到此根的对象族之中，否则则建立以本对象的类为模板的根同时加入对象. 这样建立了数据库的模式. 在进行重用时，首先将模式类装入 Java 堆之中，然后生成对象.

当 JavaStore 中有很多同一类的对象时，则要通过 Association Algebra 的代数运算求得结果类对象集. Association Algebra 共有 9 种算子有丰富的语义表达能力，分别对对象的关联模式集进行运算以产生新的关联模式集. 我们通过 Association Algebra 对 SQL 进行修改扩充为 AQL (Association Query Language)，其中把 SQL 中的 WHERE 子句中的关系改为关联集.几种语句为：

```
SELECT CLASS.ATTRIBUTE
FROM ASSOCIATION_SET
WHERE CONDITION
PDATA CLASS.ATTRIBUTE WITH SOMETHING
FROM ASSOCIATION_SET
WHERE CONDITION
INSERT INTO ASSOCIATION_SET
WITH CLASS.ATTRIBUTE="... "
例如对类
class A{};class B extends A{};
class C { ... B obj; ...}可以有如下查询：
select A.attrib1, B.attrib2
from C*B
where some-condition
```

其中 attrib1 是类 A 的属性,attrib2 为类 B 的属性,*关联算子.

JAVA 语言不能直接执行 AQL 语句，因此我们通过服务器来执行 AQL 程序,并将结果返回到客户端.例如:

```
AQL.Execute("select A.attrib1,B.attrib2 from C*B where some-condition");
```

并返回 set, list, bag 等 Collection 类对象. AQL 还有 commitbegin 保留提交前现场，commitend 开始提交，abort 恢复提交前现场，AQL 支持 commit 的嵌套执行. 客户端与服务器是通过 JAVA RMI 进行联接的.

2 JAVA 对象关系数据库的实现方法

JAVA 对象关系数据库的结构如图 1 .

JAVA 对象关系数据库由 CLIENT 和 SERVER 两部分组成,SERVER 由交互 AQL,AQL 解释器,持久引擎和 JAVA 虚拟机组成. 交互 AQL 处理交互 AQL 的解释，AQL 解释器解释 AQL 语句、持久引擎管理对象的虚拟内存与库、JAVA 虚拟机管理类的装入、对象实例的生成、AQL 语句中引用的类的方法的解释. CLIENT 由应用程序、持久 API 和 JAVA 虚拟机组成. 应用程序通过持久 API 持久化对象.

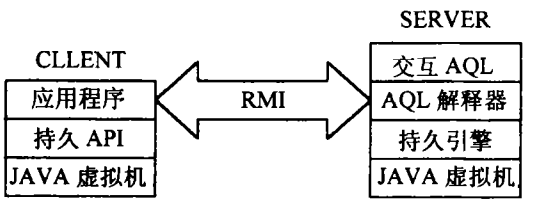

图 1 JAVA 对象关系数据库的结构

SERVER 不停的监听 AQL 语句解释的请求，在收到 AQL 语句时为此语句的建立解释线程然后继续监听. 而 AQL 解释线程启动，首先对 AQL 语句进行语法分析，然后对关联集求值，并返回结果. 在 server 端需要虚拟内存的支持. 1 个 Javastore 由 3 个表组成：类名表，类表(存放类的字节码)，对象表 (存放类的持久对象). 在 Javastore 被打开之后 3 个表都被读入到虚存之中，并生成虚拟地址与 javastore 中地址的对照表，如图 2 所示.

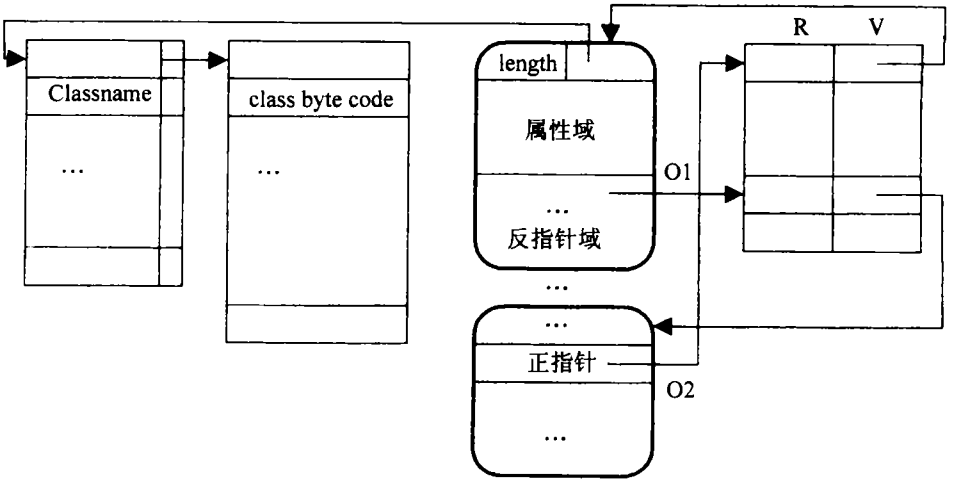

图 2 对象虚拟空间管理图

classname 表存放类名，class byte code 表存放类的字节码 O1 和 O2 是对象,RV 表存放实地址与虚拟地址的对照.

在 Javastore 之中对象中的引用是按实地址(Javastore 库中的地址)存放的，同时为了保正引用完整性在对象的最后还有引用者对象表 (反指针表). 为了能在虚存之中引用正确的对象，建立实地址和虚地址转换表 RV 表. 访问类是通过虚拟地址进行的，当需要访问一个引用对象时，通过 RV 表得到虚地址来访问对象.例如在图 2 中当 O2 引用 O1 时，对象中的引用地址是 O1 的实地址，通过 RV 表找到 O1 的虚拟地址就可以访问对象 O1 了.

当需要插入一个对象时，先查找类名表和类字节码表看是否此类已存在，若不存在则先将类名与字节码插入，然后在虚拟空间中分配空间，插入对象并填写 RV 表，同时建立反指针保证引用完整性.此时 RV 表中 R 为空，则访问引用对象时按对象中的地址访问. 若此类已存在，则已存在则直接插对象，并填写 RV 表和反指针域.当要删除一个对象时，通过 RV 表找到对象，若反指针域都为空，则删除对象并删初类名与字节码表项，否则为保证引用完整性仅删除反指针.

对于 AQL 的解释最关键的是关联集的求值，求值后的关联模式用 VV 表表示，图 3 显示了求值过程.

图 3 表示了关联集 Teacher*Student*Course 的求值过程. 首先由 Teacher 类的 RV 表与 Student 类的 RV 表进行关联算子 (*) 运算生成 VV 表 1，其中左边是与 Student 类关联的 Teacher 类

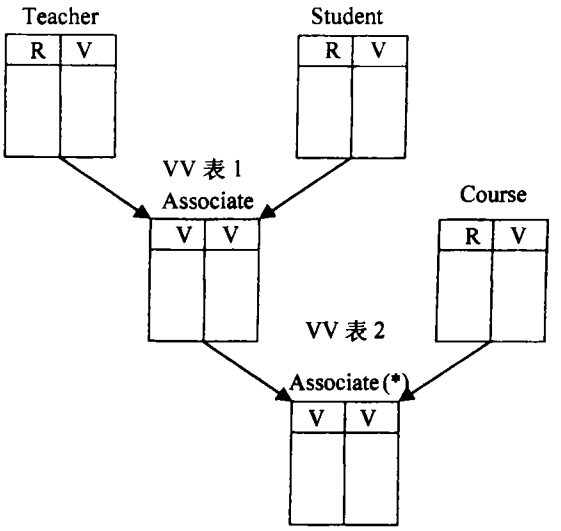

图 3 关联集 Teacher*Student*Course 求值

对象 OID 表，右边是与 Teacher 类关联的 Student 类对象，然后 VV 表 1 与 Course 表进行关联算子(*)运算生成 VV 表 2. 关联集的结果就是通过 VV 表的遍历来查询.

当开始一个事务时，类名表，类字节表，对象表，RV 表被保存，事务结束时类名表，类字节表，对象表被写入 Javastore 同时删除保留的现场.如果事务中间流产则恢复现场. Javastore 支持事务嵌套.

JAVA 对象关系数据库的实现在 client 端需要 JAVA 反射的支持.通过 JAVA 反射功能来得到对象的类型，属性与值，当对对象持久化时，对对象引用的对象进行遍历以使可达的都持久化，然后调用持久 API 写入 Javastore 之中.持久 API 主要有 openStore 打开仓库，createStore 创建仓库， addObject 加入对象， exectuteAQL 执行 AQL 语句，closeStore 关闭仓库，deleteStore 删除

仓库几种 API..

图 4 表示了对象的可达关系.图中 Obj1 引用 Obj2,Obj2 引用 Obj4,Obj1 引用 Obj3,Obj5 引用 Obj3. 持久化 Obj1 时要遍历 Obj1 的引用链并把 Obj2, Obj3, Obj4 都持久化(同时有反指针来保正完整性). 在 CLIENT 端有表<ref,virtual>来防止对象的重复加入. 当同时持久化 Obj1, Obj5 时,必须对 Obj3 检查,若已加入到 Javastore 之中则不必重复加入.

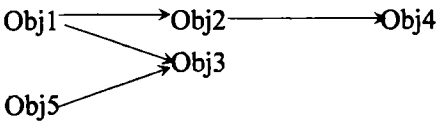

图 4 对象可达关系图

AQL 的结果返回到集合类型 set, list, bag 等类型之中. 应用程序用这些类中的方法访问集合中的个体元素.集合中的元素并不在 CLIENT 端,Javastore 使用懒惰方法访问持久对象,即只有当访问到该元素时才从 SERVER 中装入此对象.集合中的对象在装入时,先装入类的字节码,然后生成新对象,并用 JAVA 的反射功能用持久对象中的属性设置它的属性并加入到它的集合类中,同时凡是此对象可达的对象都被装入,以此来完成持久对象的访问.

由于 Javastore 是一个外挂的系统,不是对虚拟机进行重新开发,因此只能持久对象中的 public 属性,对于 private 和 protected 属性无法持久化.

4 结论

在对象关系数据库的研究之中,在没有 JAVA 虚拟机之前,对象的方法不能持久化也不能被访问. JAVA 产生之后,借助于正交持久化的理论可以产生带方法的模式,按照关联代数的方法可以用集合化的方法访问持久对象和它的方法,加上 JAVA 的跨平台性可以很自然的分布到网络中,形成网络对象关系数据库,这将优于以往的系统.

参 考 文 献

1 Bindou R.R. Object-oriented Databases Technology, Applications and Products.NY: McGraw Hill Database Experts' Series,1994

2 Stanley Y, Su W. Association Algebra: A Mathematical Foundation for Object-oriented Database. IEEE Transaction on Knowledge and Data Engineering,1993,5(5):93

An Implementation Method of Object-relational Database Using JAVA

Liu Qingwen, Yang Yang

ABSTRACT JAVA language and the platform independence of JAVA virtual machine provide convience for new generation of object relation database. A principle and methdology of object relation database which persistences JAVA objects are presened according to Atkinson's orthogonal persistence theory and accesses persistent object set by assiciation set by calculating association operators.

KEY WORDS object relation database; JAVA; persistence