基于XML文档的关系数据库与面向对象数据库之间的信息交互

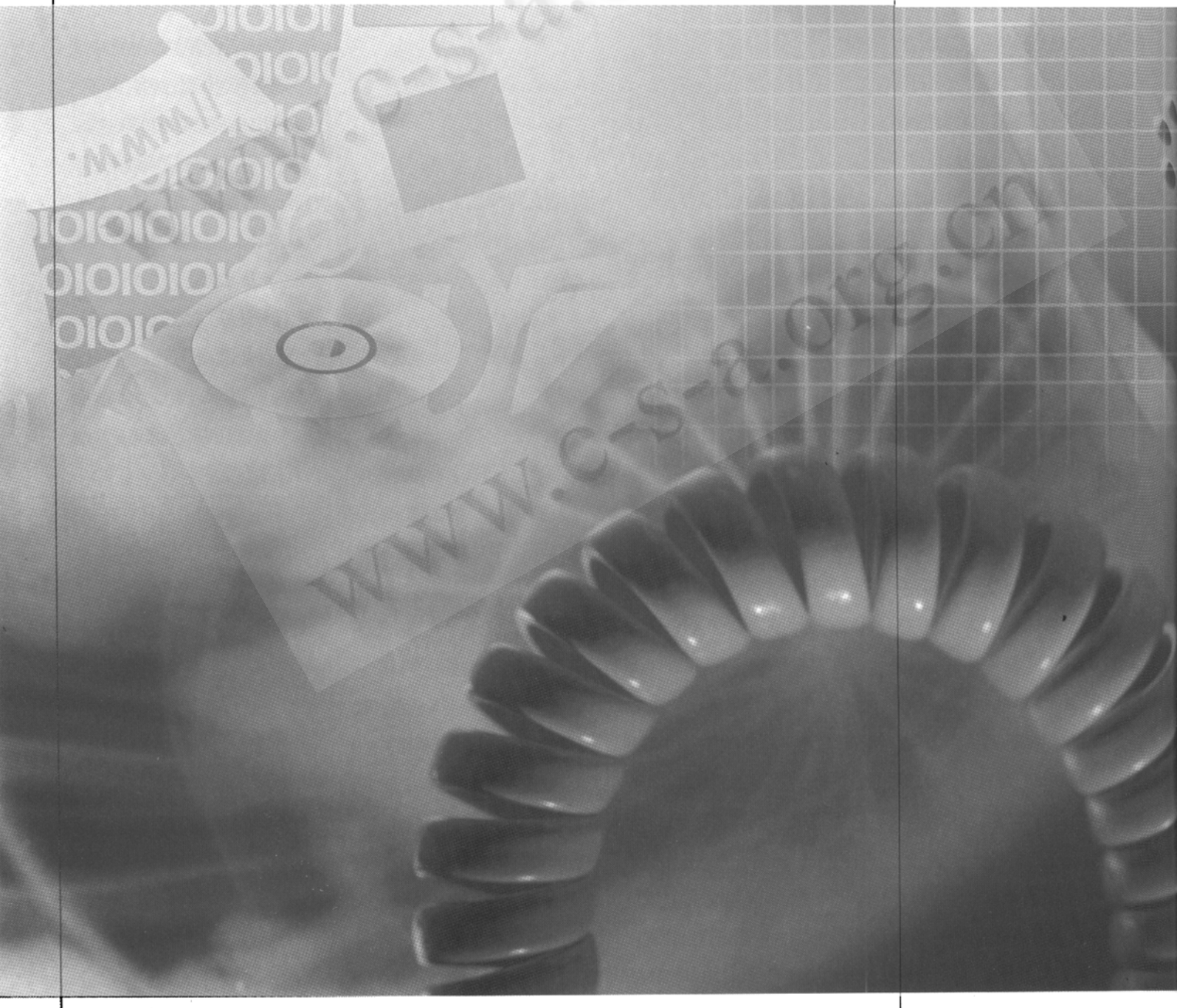
摘要：针对传统关系数据库与面向对象数据库之间信息交换所存在的不足，本文分析了XML文档和关系数据库模型/面向对象数据库模型的对应关系，提出了基于XML文档为中介的数据库间信息交互的算法。最后举例说明了如何利用XML文档来实现数据库间的信息交互。

关键词：XML 关系数据库 面向对象数据库 信息交互

高阳 谭力民 （长沙中南大学工商管理学院 410083）

1 引言

传统关系数据库建立在关系模型基础上，利用关系来表示现实世界中实体与实体之间的各种联系。因此关系数据库之间的信息交互可利用相应关系表结构的文本文件作为中间文件来完成，即用元组表示记录，记录的域之间用分隔符进行隔离。面向对象数据库是由类和对象组成的集合体，对象嵌套层次结构和类层次结构所形成的对象横向和纵向复



杂结构不仅导致类之间的层次关系，而且也使类内部具有嵌套层次结构，因此在关系数据库与面向对象数据库之间以及面向对象数据库相互之间不能再使用简单的文本文件来进行信息交互。从本质上讲，造成这种结果的原因是两者在结构上的区别，即前者使用关系表的二维结构来表示数据库，后者则采用树状结构来表示数据库。因此如何利用现有技术来实现关系数据库与面向对象数据库之间的信息交互就成了需要解决的问题。

2 XML

XML (eXtensible Markup Language)是全球信息网协会(W3C)推荐在Web上使用的数据表示和交换标准，它源于SGML (Standard Generalize Markup Language)这种老资格的通用标记语言，采用标记(<message>...</message>)来表示信息内容，它不仅提供关于数据本身的信息?而且侧重于提供对数据结构的描述，是标识和描述的集合，具有自描述性。

XML数据由嵌套和标记元素组成，标记包含对文档存储形式和逻辑结构的描述，这种嵌套标记元素结构使XML很适合描述Web上的半结构化数据；此外，这种标记代表数据的含义而不是显示数据结构，也使XML可用来描述内容而非表现形式；再加上XML文档一般是成型的合法结构文档，使一般的应用软件能解读它，并通过标记语言的意义对其进行特定的处理，使XML具有良好的可扩展性。

XML实际上是一种定义语言，使用者可自由定义标签，并通过元素之间的嵌套包含来体现层次关系。它主要有3个要素：Schema (模式)，XSL (eXtensible Stylesheet Language)，XLL (eXtensible Link Language)。其中，Schema规定XML文件的逻辑结构，定义XML文件中的元素、元素属性以及元素与元素之间的关系，它可帮助XML分析程序校正XML文件标记的合法性；XSL用于规定XML文档样式，它能在客户端使Web浏览器改变文档的表示方法，从而使其不需再与服务器进行通信交互；XLL主要用于进一步扩展目前Web上已有的简单链接[1]。由此可见，XML文档的结构树是由元素、子元素、属性、PCDATA以及元素与后三者的连接构成。

最重要的XML文档管理模型中是表模型和树模型，前者将XML文档看成单一的或一系列的表，类似于关系数据库中的关系模型；后者则利用XML的元素及属性来表达对象的内部结构和外部链接，类似于面向对象数据库的对象树结构[2]，因此XML可为关系数据库与面向对象数据库间的信息交互提供一种公共的格式。

3 基于XML实现关系数据库与面向对象数据库信息交互的算法

数据库之间信息交互的目的就是在目的数据库中再现源数据库中的信息。信息包括信息内容和信息结构两层含义，前者是构成信息的最小基本单位，如关系数据库中的字段、面向对象数据库中的属性等；后者是信息的构成形式，如关系数据库中的关系、面向对象数据库中的类等，信息的内容和结构相结合构成完整意义的信息。因此信息交互要求保证信息的内容和结构在数据库间完整转移，利用XML文档进行数据库间信息交互就是用XML作为信息交互的中介，完成信息的全面转移。

3.1 基于XML实现关系数据库与面向对象数据库信息交互算法的体系结构

在数据库将进行信息交互时，首先判断源数据库和目的数据库的数据模型的归类；再根据源数据库数据模型的类别与XML建立映射关系，转换为XML文档；最后再由XML文档映射到目的数据库中去，完成具体的信息交互。因此具体的算法思想如下：

{调用源数据库；
if (源数据库的数据模型是关系数据库)
{调用目的数据库；
if (目的数据库的数据模型是关系数据库)
使用关系表的文本文件进行信息交互；
else if (目的数据库的数据模型是面向对象数据库)
{调用模块1将源数据库的信息转换为XML文档文件；
调用模块2将XML文档文件映射到目的数据库
}
else 输出目的数据库数据模型有误；
}
else if (源数据库的数据模型是面向对象数据库)
{调用模块3将源数据库的信息转换为XML文档文件；
调用目的数据库；
if (目的数据库的数据模型是关系数据库)
调用模块4将XML文档文件映射到目的数据库；
else if (目的数据库的数据模型是面向对象数据库)
调用模块2将XML文档文件映射到目的数据库；}
else 输出目的数据库数据模型有误；
else 输出源数据库数据模型有误；
}

3.2 算法中的模块分析

在上述算法中，模块主要完成数据库与XML文档的交互，其中，模块1和3是将源数据库的信息转换为XML文档文件，模块2和4是将XML文档映射到目的数据库。由于模块对应的标的不同，其功能和实现方法也不相同。

模块1的作用是将关系数据库中的信息转化为XML文档，其实现步骤如下：

① 将信息中的每个关系表转化为XML文档中的元素；

② 将关系表中的元组转化为XML文档中的子元素，并将关系表

中各元组的属性赋给相应的子元素

③ 输入新的关系表，递归调用①、②，直至信息输入完毕；

④ 通过关系表的主键、外键将各元素、子元素串接成树状结构的XML文档，合并属性；

⑤ 删除关系数据库中因规范化而导致的冗余，优化XML文档。

模块2是将XML文档映射到面向对象数据库中，完成数据传递。具体实现步骤为：

① 按深度有限原则从树状XML文档的树叶往树根进行搜索；

② 每碰到除叶结点和根结点以外的内部结点，查看数据库中是否存在对应的抽象数据类型，若不存在，则创建，并保证该结点所有子结点是该抽象数据类型的域；

③ 重复②，直至根结点的直接子结点(次根结点)；

④ 创建基于次根结点所对应抽象数据类型的对象，该对象是根结点所对应的对象；

⑤ 将XML文档中各元素的属性值填入对象的相应域。

在此模块中，若元素只包含简单的PCDATA，则只需将其看成简单数据域；若元素本身结构较为复杂，则需将其对应为元素。

模块3表示将面向对象数据库的信息转化为XML文档文件，具体操作为：

① 将数据库中的对象对应生成XML文档中的元素；

② 对对象中包含的每个简单数据域用元素的属性来表示；

③ 对对象中每个复杂的子对象域递归调用①、②生成新元素，并将生成元素作为当前元素的子元素；

④ 直至对象中所有的域都完成为止。

模块4是将XML文档文件映射到关系数据库，生成相应的关系表，步骤如下：

① 对每个含有子元素或混合内容的元素生成一个表和一个主键列；

② 对每个含有混合内容的元素生成一个单独表，该元素保存的PCDATA数据，通过父表主键连接

③ 对每个单值属性和每个只出现一次的，且只含PCDATA类型数据在父表中生成一列，如果子元素或属性可选，则此列可以为NULL。

④ 对每个多值属性和每个多次出现的，只含PCDATA类型的子元素，生成一个单独表保存值，并通过父表的主键连接。

⑤ 对每个有子元素或含混合内容的子元素，通过父表的主键连接父表到子元素的表。

从理论上讲，面向对象数据库的结构更接近XML文档，但由于XML文档管理模式的多样性和传统对象/关系映射理论的发展，XML文档必将成为关系数据库与面向对象数据库信息交互的有效中介[3] [4] [5]。

4 应用举例

当前市面上占主导地位的数据库仍是关系数据库，但由于面向对象数据库的优势所在，它必将是未来发展趋势。在此用对象关系数据库的代表Oracle8i作为示例说明如何用XML实现数据库间的交互。Oracle8i允许创建对象数据类型，因此假设在数据库中存在以下创建的学生信息：

```
CREATE TYPE Address-TYPE AS Object
(street-name-or-route  varchar2 (40),
house-or-box-number  integer,
city-name  varchar2 (30),
state-code  varchar2 (10),
zip-code  integer)
CREATE TYPE Degree-TYPE AS Object
(department-name  varchar2 (30),
degree-name  varchar2 (30),
degree-field  varchar2 (20))
CREATE TYPE Student-TYPE AS Object
(student-id  number (20),
address  Address-TYPE,
degree-plan  Degree-TYPE)
CREATE TABLE STUDENT AS Student-
TYPE
(student-id  Not  NULL,
address  Not  NULL,
degree-plan  Not  NULL)
```

假设在表中存在一组信息，如果要将其传送到另一面向对象数据库中去，则需先将其转化为XML文档文件，表示如下：

```
<? xml version=".0"?>
<student>
<student-ID= "10253">
<ADDRESS street-name-or-route= "North Main Street" house-or-box-
number= "23"
city-name= "Philadelphia" state-code= "PA" zip-code= "45032">
</ADDRESS>
<DEGREE department-name= "management" degree-name= "master" degree-
field= "Computer nerwork">
</DEGREE>
</student>
```

然后将XML文档传送至面向对象数据库，按模块2的步骤依次在数据库中建立抽象数据类型ADDRESS-TYPE、DEGREE-TYPE、STUDENT-TYPE，再按抽象数据类型STUDENT-TYPE在数据库中创建对象，最后将XML文档中各元素的属性值填入对象的相应域。这样数据库间的信息交互就通过XML文档这个中间文件实现了。

参考文献

1 Goldfarb C F,Prescod P.The XML Handbook, Prentice Hall PTR 1998。

2 张伟都，周海东，钟共凌等，面向XML的数据管理系统，《计算机工程与应用》2001,20。

3 Bourret B.XML and Database,[EB] http://www.rpbourret.com/xml/XML and Database.htm。

4 Michael Blaha,William Premerlani.Using UML to Design Database Application [2].Rational Software Corporation,1999。

5 Scott W.Ambler.The Design of a Robust Persistence Layer for Relational Database [M].Ambysoft Inc.,October 1999。