关系数据库中存储 XML 文档的技术*

薛俊义[1],刘琴[2],许卓明[2],刘传汉[3]

(1.盐城工学院 计算机工程系,江苏 盐城　224003
2.河海大学 计算机及信息工程学院,江苏 南京　210098
3.上海交通大学 计算机科学系,上海　200030)

摘　要:XML 及其相关技术日益渗透至计算机科学的各个层面,为了实现 XML 的潜能,XML 的有效存储是一个必须首要解决的技术环节。对该技术进行了较为详尽的阐述,并介绍了 Oracle 为存储 XML 文档所进行的功能扩展。

关键词:XML;数据库;数据存储;模式映射;Oracle

中图分类号:TP311.132.3　　**文献标识码**:A　　**文章编号**:1671－5322(2002)04－0041－05

可扩展标识语言 XML(the eXtensible Markup Language)已发展成为 Web 上数据表示与交换的标准格式。为实现 XML 在数据管理、交换与共享中的潜能,XML 的有效存储是一个关键的技术环节。源于 XML 通用的数据(结构化、半结构化与元结构化)表示能力、XML 文档内容的两重性(传统类型的数据或文本内容)及其不同的应用目的与领域,使得 XML 的存储问题至今难以找到一种“最佳的”通用解决方案。这正是导致这方面的争论尚无定论的原因所在。大量的研究与(原型)产品提出了各种 XML 存储技术。按数据存储系统的不同,大体可分为如下 3 种方案:(1)在文件系统中的文本文件方式存储 XML;(2)在数据库系统(关系或对象数据库)中存储 XML;(3)开发 XML 专用库(XML Native Repository)的所谓‘自然’(native)方式存储 XML。

不管 XML 的存储方案孰优孰劣,以及最终的结局如何,我们(至少在现阶段)有充分的理由需要研究 XML 在关系(包括对象—关系)数据库中如何存储的问题:(1)大量的商务业务数据已经并将继续在关系数据库中存储与管理,而基于关系数据库的事务处理应用需要集成与交换 XML 格式的数据;(2)RDBMS 已提供了有效、完善及用户熟悉的功能服务设施(如:数据存储与索引、查询优化、事务管理、安全与恢复等),因此,需求与技术两种驱动导致近来对 XML 关系存储方法的大量研究及其原型开发[1~2];主要的数据库厂商 Oracle、IBM、Microsoft 也纷纷进行 XML 扩充,推出 XML－Enabled 产品支持。

那么如何在这两种不同结构的数据之间进行模式映射(Schema Mapping)呢?这已经成为解决这一数据交换技术的关键。

1　XML 文档及其模式

目前,XML[3]正渐渐取代 HTML,成为网络上标准的文本标记语言,它是 SGML(Standard Generalized Markup Language,标准通用标记语言,是第一个标准化的信息结构化技术)的一个优化子集,继承了 SGML 的可扩展性,结构化和有效性。一个 XML 文档是由元素组成的,元素又可包含子元素,也可具有属性,如图 1 所示。从数据的角度来说,XML 与 HTML 相比主要有两大优势:通过语义标记来说明数据的语义(如图 1 中,标记 < phone > 说明的是数据的意义即电话号码,而非 HTML 中那样用标记来表示数据的外观);文档内容和显示相分离,使得数据具有独立性,而文档的显示则由 XSL(eXtensible Style Language)来完成。

我们知道 XML 具有可扩展性,即 XML 文档

* 收稿日期:2002－06－02

作者简介:薛俊义(1973-),男,江苏东台人,盐城工学院计算机系助教。

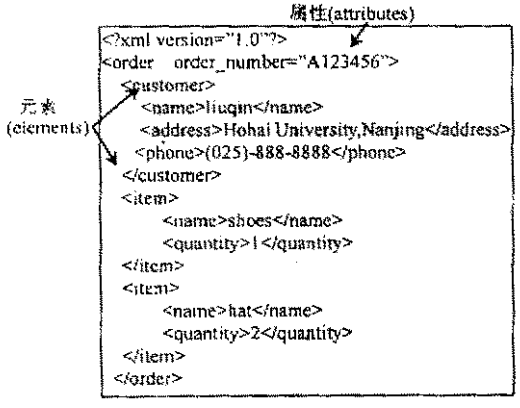

图 1 XML 文档

Fig.1 XML document

中的标记可以由开发者自己定义,那么用来定义这些标记、XML 文档结构以及语法的语言就是 XML 文档模式语言。至今已经提出多种 XML 文档模式语言,如 DTD、XML Schema、XDR、SOX、Schematron、DSD 等。其中 DTD(document type definition)是最早也是最成熟的 XML 文档模式语言(如图 2 就是图 1 的 DTD),它定义了一套规则去控制 XML 文档的结构(即决定存在哪些元素和属性,它们在何地出现,出现多少次,元素如何嵌套,元素和属性如何结合等等问题)。而 XML Schema[4] 弥补了 DTD 的不足并将会逐渐取代它:首先,它本身就是 XML 文档,可直接用 XML 解析器来解析;其次,提供了丰富的数据类型,并可以定义新的数据类型;其三,XML Schema 支持继承性(即利用一个现存的 XML Schema 来建立新的 XML Schema);另外,XML Schema 还支持命名空间。

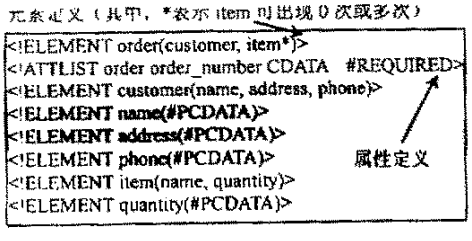

图 2 DTD 文档

Fig.2 DTD document

2 XML 存储于传统数据库中的一般技术

基于关系数据库的 XML 文档存储查询过程一般有 4 个步骤[5]:(a)建立 XML 文档和关系表的映射;(b)解析 XML 文档,将其数据载入关系表;(c)将针对 XML 文档的半结构查询语言转换为针对关系表的 SQL 查询语言,执行 SQL 查询语言;(d)将 SQL 查询语言执行的关系结果转换为目标 XML 文档。下面我们重点讨论第一步。

2.1 存储于关系数据库的一般技术

目前,存储于关系数据库大致有四类技术[6]:通用的映射技术;用户提供映射方法,目前商品化 RDBMS 大多采用这种方法,我们将在第 4 小节以 Oracle 为例进行介绍;从文档模式推导出关系模式;分析 XML 数据和查询。

2.1.1 通用的映射技术

映射[1]把 XML 文档模型化成一个有序的带标记的有向图(如图 3 为图 1 的有向图)。在图中,每一 XML 元素用带有标识符 id 的结点表示,元素与子元素的关系用子元素的名字命名的有向边来表示,每一结点的出边都是有序的,叶子结点表示属性值或元素值。

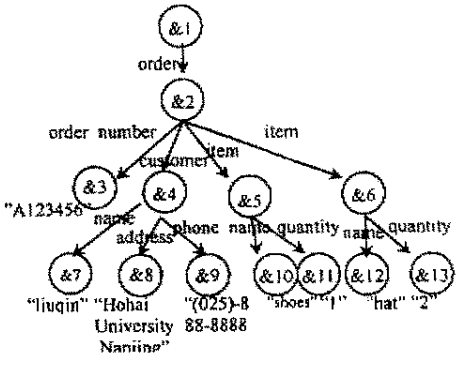

图 3 XML 文档的有向图

Fig.3 The digraph of XML document

该技术共有 6 种映射方法,即 3 种存储边的方法和 2 种存储叶子方法的组合。

2.1.1.1 映射边

(1) Edge Approach:存储图中所有的边在一张表中,该表称为边表。表的结构:

Edge(source, ordinal, name, flag, target)

其中 source—边的起点对象的 ID,ordinal—边的顺序号,name—边名,flag—标记边是否指向叶结点,target—边的目标对象的 ID。表的主键是(source, ordinal)。

(2) Attribute Approach:存储所有边名相同的边到一张表中。表结构:Aname(source, ordinal, flag, target)(各参数意义及主键同前)

(3) Universal Table:用一张统一的表存储所有的边,相当于外连接所有的 Attribute 表。表的结构:

Universal(source, $ordinal_{n1}$, $flag_{n1}$, $target_{n1}$, $ordinal_{n2}$, $flag_{n2}$, $target_{n2}$, . .) 其中,n1,n2…为边名。

2.1.1.2　映射值

(1) Separate Value Tables:为每一数据类型建立一张单独的表,称之为值表。表结构:

Vtype(id, value)

其中 type—数据类型,id—由映射自动产生的叶结点的标记。

(2) Inling:值和边存储在一张表中,在 Edge Approach 情况下,相当于外连接边表和所有的值表(其余情况类似)。由于需要为每一不同的数据类型建立一列,从而表中存在许多空值。

从映射生成的数据库的大小、重构 XML 文档所需的时间、执行不同的 XML 查询所需的时间等方面对这 6 种映射方法进行比较分析,发现 Attribute + Inling 是最优的映射。

该技术有利于存储那些没有文档模式的 XML 文档,不需要用户参与映射过程,该方法也比较简单,对于任何 XML 文档都映射生成一个固定的关系模式,不过生成的关系模式不是数量众多,包含大量的空值,就是在查询时需要执行大量的连接操作。这项技术一般只作为进一步研究的基础。

2.1.2　从文档模式推导出关系模式

映射首先对 DTD 进行简化[2],然后将 DTD 表示为 DTD 图(如图 4 为图 2 的 DTD 图,其中节点代表元素、属性和操作符,边表示父子关系,且每一元素在图中仅出现一次,属性和操作符出现次数与 DTD 中相同),然后依据 DTD 图并利用如下两种技术建立关系模式。

(1) The Shared Inlining 技术为 DTD 图中的节点入度大于 1 或等于 0 的元素节点建立关系(如图 4 的 name, order 元素);入度等于 1 的元素节点内联到其它元素关系中(如 address 内联到 customer);"*"的儿子节点元素单独建立关系(如 item 元素);相互递归的元素节点,选择其中之一单独建立关系。

(2) The Hybrid Inlining 技术它在 The Shared Inlining 的基础上进行了改进,将 DTD 图中节点入度大于 1、但不递归、不经过"*"到达的节点元素内联到其它元素中(如 name 元素)。

根据 The Hybrid Inlining 技术可见所有的关系都有由映射自动生成的 id 属性,非根元素相应的关系表都有 parentid 属性,parentid 是外键,指向父元素,且存在 parentcode 列指明元素在兄弟元素中的顺序。

在把基于 XML 的查询转化成 SQL 查询时,以上两种技术显示出了不同的性能,Shared 相对减少了生成的 SQL 数目但增加了每个 SQL 查询的连接运算,Hybrid 相对减少了每个 SQL 查询的连接运算但增加了 SQL 查询。具体的结果要根据 DTD 类型和查询类型来具体分析。

从文档模式推导出关系模式最大的不利在于它无法处理没有文档模式的 XML 文档,而且现实中的文档模式是非常复杂的,该技术只能在对文档模式做了一些简化工作之后,才能推导出关系模式。同时由于文档模式信息的存在,这类方法能用关系数据库处理大多数 XML 查询,而且能取得较好的查询效率。

2.1.3　分析 XML 数据和查询

XML 是半结构化数据的一种数据形式,所以可利用处理半结构化数据的技术去存储和管理 XML。

在 STORED 中,当 XML 文档给定时,依据 XML 数据自动生成 STORED 映射和关系模式,半结构化数据的查询和更新也能自动的重写成关系模式的查询和更新。事先知道的一些查询可以被应用在产生 STORED 映射和关系模式的过程中,由此可见此技术是在分析 XML 数据和查询的基础上产生的映射。

为了能产生一个"好"的关系模式("好"的定义取决于不同的具体的查询或更新的应用场合中),STORED 利用了一种启发式的算法—数据挖掘算法,自动产生 STORED 映射和关系模式。这种映射是无损映射,部分不满足映射的数据会被存储在另外的一个溢出图中,也就是说 STORED 映射把半结构化的数据存储到一些混合模式(关系模式和溢出模式)的实例中。

映射中用到的数据模型与 2.1.1 节中使用的模型相同,可依据不同的"类型"分离对象,然后为不同类型的对象建立关系模式。例如:依据图 1 和图 3,我们可分离出如下的 3 个关系模式 order, customer, item,若除了这个 customer 元素外,还有一 customer 元素,它的 address 元素中还包含了子元素,那么依据"类型"不同可分离出两张表 customer1,customer2。

当然这种选择不是唯一的,也不一定是最优的。另外,这些关系模式还不能满足更新 XML 时的需求。例如,我们无法在 order.customer 下再添加一个 phone 元素,此时我们需存储这些数据在

溢出图中,溢出图也可转化成关系表存储在关系数据库中。上述的情况可生成溢出关系 G1(source, phone)。

STORED 技术虽然可以不需要 XML 文档模式,但是文档信息的存在(且 XML 文档遵守模式)可有利于最小化溢出图,提高系统的效率。

该技术利用了关系和半结构化的混合技术去处理 XML 文档,它不需要文档信息,能根据不同的应用场合产生合适的关系模式,但是这类技术往往比较复杂,具体实现时需要借助其它相关领域的知识来完成。

2.2 小结

检验一个模式映射优劣的一个重要的标准是基于这些映射的 SQL 的查询效率如何[7~8]。在存储 XML 文档时,可以发现有 3 种理想的组建关系模式的方法,有利于提高查询效率:(1)依照现实世界的规律组织元素(如可以把客户的姓名和地址存储在一起);(2)把相同的元素组织在一起(如存储所有的客户元素在一起);(3)按照 XML 文档的顺序组织元素。

The Shared Inlining 和 The Hybrid Inlining 利用了第 1、2 两种方法;The Edge Approach(以 name 作为簇集健)和 The Attribute Approach 利用了第二种方法,丧失了第一种方法的优势,这样当查询涉及元素间相互关系时(如:查找住在 Hohai University 的客户的信息),edge 和 attribute 需多次运用 SQL 的连接操作,此时就不如 Shared 和 Hybrid 查询效率高,此外 Attribute Approach 由于在表中未保留子元素的标记,从而在重构和查询时需要 DTD 来决定哪张表里包含了所需的元素;STORED 能依据不同的查询,产生不同的关系模式,同时考虑到了 XML 的更新问题,能生成无损的映射,这些都是其它的映射不能比及的,但是由于 STORED 语言的表达能力有限,它不能处理带有递归结构的 XML 文档,而其它两类技术都能完成。

总的说来,XML 文档模式的信息对取得好的查询性能是很重要的,而另一方面,有些应用场合需要去处理没有文档模式的 XML 文档。这些映射的一个共同的弱点是在进行 XML 文档重构时需要花费大量的时间,都需要额外的努力去把基于 XML 的查询转换成基于关系数据库的查询,且无法处理复杂的递归查询,另外依据不同的映射方法、不同的 XML 查询语言,需用不同的转换技术。由于关系模式和 XML 之间的异构性[9],XML 存储到关系数据库时会造成语义(如文档的名称、实体的定义、命名空间的定义、元素的顺序等等)的丢失。

3 Oracle8i/9i 中存储 XML 文档的技术简介

在 Oracle8i 中存储 XML 文档有几种可选方案[11]:大对象(LOB)存储,整个 XML 文档以字符大对象(CLOB)存储,这种方法一般用来存储有着不规则结构的 XML;BFILE 存储,XML 被存储在 DBMS 以外的地方(如文件系统),但有关的元数据可以存储在数据库服务器的对象 – 关系表中,并且可以利用这些数据来快速索引和查询 XML 文档;对象 – 关系存储(即用数据库中的表来存储 XML 文档),一般用来存储规则结构的 XML 文档。Oracle 还支持混合存储且允许在表定义期间用户自定义映射的粒度。实现这些功能的部件是 XSU(XML – SQL Utility),它被打包在 Oracle8i(8.1.7 或更高版本)和 Oracle9i 中。

XSU 提供了缺省的 XML 到 SQL 的映射,可映射到关系表(不推荐)或对象 – 关系表中,映射规则:简单元素(只包含字符串的元素,如 address 元素)映射到数量型的列中,包括子元素的元素映射成对象类型(如元素 customer 映射成对象类型 customerType),重复出现的元素映射到集合类型,(如 item 映射成表类型 itemlistType)。但在映射的过程中,忽略了元素的属性(除顶层纯量型的属性,如 order-number 外)和顺序,而且映射的过程中要求元素名和列名相同。这样依照映射规则可以生成如下的对象—关系表 order:

```
CREATE TYPE orderType as OBJECT
(@order-number varchar(7),
customer customerType,
item itemlistType);
CREATE TABLE order as TABLE OF orderType;
```

除了缺省的映射外,用户也可从 3 个方面进行修改(即由用户提供映射方法):修改目标模式(对象 – 关系模式),可以在原来的目标模式上建立视图;利用 XSLT(XML Stylesheet Language Transformation)转换 XML 文档,例如我们可以利用 XSLT 转化 XML 文档的属性到元素,这样属性的数据就不会丢失了;修改 XSU 的 XML-to-SQL 映射,包括元素名到列名可以大小写不敏感匹配,可以告诉 XSU 在 XML 文档中相应于数据库中一行的那个元素的名字(缺省为 ROW,图 1 中的文档

为 order)。

XSU 为用户提供了 3 种访问方式:命令行方式,XSU Java API,XSU PL/SQL API。可采用如下的 XSU Java API 的方式将图 1 中的 XML 文档存到数据库中。

```
String xmlDoc =" …the actual xml document… ";
Connection conn =
DriverManager.getConnection(," scott "," tiger ");
OracleXMLSave Sav = new
OracleXMLSave(conn," order ");
Sav.insertXML(xmlDoc);
```

Oracle9i 在 8i 的基础上,在存储方面增加了一个重要的特性:支持 XMLType(一种新的系统定义的对象类型),允许把整个或部分 XML 文档存于数据库表中的被定义为 XMLType 类型的列,系统并相应地提供了创建、查询、索引、更新 XML-Type 列等机制。

虽然 Oracle 数据库已经能支持 XML,但我们可以看到这种支持是很有限的,体现在对象 – 关系模式需用户建立,它只是提供了相同名字的元素到列的数据复制,只能处理一些简单结构的 XML 文档。

4　结束语

本文总结分析了几种典型的映射技术,这些映射技术各有优劣,这给用户在选择合适的映射技术时带来了不便,从而引发了一个标准化的问题——如何制定统一的标准来衡量扩展 XML 的数据库的性能,如何规范化映射方法。有关这方面的工作已经在一些科研院所[1] [10]展开了,规范化的问题也给数据库带来了新的生机和挑战。

参考文献:

[1] Florescu D, Kossmann D. Storing and Querying XML Data using an RBMS[J]. IEEE Data Engineering Bulletin, 1999, 22(3): 27 – 34.

[2] Shanmugasundaram J, Tufte K, He G. Relational Databases for Querying XML Documents: Limitations and Opportunities[Z]. Proceeding of the 25th VLDB conference, Snowmas, 1999: 302 – 314.

[3] Ceri S, Fraternali P, Paraboschi S. XML: Current Developments and Future Challenges for the Database Community[Z]. Proceeding of the 6th international conference on EDBT, London, 2000.

[4] Roy J, Ramanujan A. XML Schema Language: Taking XML to the Next Level[J]. IEEE IT Pro, 2001.3(4): 37 – 40.

[5] Shanmugasundaram J, Krishnamurthy R, Tatarinov I. A Gerneral Technique for Querying XML Documents using a Relational Database System[Z]. proceeding of SIGMOD international conference, Chicago, 2001.

[6] Chaudhuri S, Shim K. Storage and Retrieval of XML Data Using Relational Databases[Z]. VLDB '01 Conference, Chicago, 2001.

[7] Schmidt A, Was F, Kersten M etc. Why And How To Benchmark XML Databases[Z]. proceeding of SIGMOD international conference, Chicago, 2001.

[8] Tian F, Dabid J, Chen J. The Design and Performance Evaluation of Alternative XML Storage Strategies[Z]. In the proceeding of SIGMOD international conference, Chicago, 2002.

[9] Kappel G, Kapsammer E, Retschitzegger W. XML and Relational Database Systems-A Comparison of Concepts[Z]. 2001 International Conference on Internet Computing, NewYork, 2001.

[10] Eisenberg A, Melton J. SQL/XML and the SQLX Informal Group of Companies[Z]. proceeding of SIGMOD international conference, Chicago, 2001.

A Survey of Techniques for Storing XML Documents in Relational Databases

XUE Jun-yi[1], LIU Qin', XU Zhuo-ming[2], LIU Chuan-han[3]

(1. Department of Computer Engineering of Yancheng Institute of Technology, Jiangsu Yancheng 224003, China; 2. College of Computer & Information Engineering, Hohai University, Jiangsu Nanjing 210098, China; 3. Department of Computer Science, SJTU, Shanghai 200030, China)

Abstract: XML and related technology has obviously affected almost every field of computer science. To realize XML potential, its effective storage is firstly faced technology. In this paper, we will detail these issues and introduce some XML-storage extensions of Oracle.

Keywords: XML; Database; Data Storage; Schema Mapping; Oracle