第 34 卷 第 4 期 计 算 机 工 程 2008 年 2 月
Vol.34 No.4 Computer Engineering February 2008
·软件技术与数据库· 文章编号：1000—3428(2008)04—0084—03 文献标识码：A 中图分类号：TP311

基于 XML 的代理关键字处理方法

葛 欣[1]，丁恩杰[2]
(1. 中国矿业大学计算机学院，徐州 221008； 2. 中国矿业大学信息与电气工程学院，徐州 221008)

摘 要：代理关键字被普遍应用于数据仓库的构建。在 XML Schema 映射为数据仓库结构的情况下，该文提出一种引入代理关键字的处理方法，利用 Java 工具包，将 XML Schema 转换成 XOM 树结构，在转换成关系模式时加入代理关键字，实现了 XML 文档与数据仓库间的直接转换。实验结果证明了该算法的有效性。

关键词：代理关键字；数据仓库；数据转换

Surrogate Keys Processing Method Based on XML

GE Xin[1], DING En-jie[2]
(1. School of Computer, China University of Mining and Technology, Xuzhou 221008;
2. School of Information & Electronic Engineering, China University of Mining and Technology, Xuzhou 221008)

【Abstract】 With the universal application of surrogate keys in data warehouse creation, this paper presents a method to deal with surrogate keys when mapping XML Schema to data warehouse model. It maps XML Schema to XOM tree by using Java toolkit, and adds surrogate keys during transforming the tree to relation schema, which realizes the creation of data warehouse through XML. Experimental results show that the method is effective.

【Key words】 surrogate keys; data warehouse; data transformation

可扩展标记语言(XML)是自ACSCII文本文件出现以来可移植性高、最灵活的文档格式[1]。通过使用标签，它可以将数据和数据间的关系表示在一个文本文件中。这些标签可以是系统本身提供的，也可以由用户根据需要来设定。“与平台无关的传输格式”这一特点使其迅速发展成为数据存储、数据交换等方面的标准。然而，XML毕竟不是数据库，当进行复杂的数据分析与处理时，还需要借助成熟的数据库管理技术。针对这种情况，研究人员对XML数据文档与关系数据库之间的转换做了大量的研究，提出了很多转换算法。目前，万维网联盟(World Wide Web Consortium, W3C)组织推出的DTD, XML-Schema等技术已经满足大部分的实际需求，可以很好地完成XML数据与关系数据间的转换。但是，数据仓库不同于一般的操作型数据库，它是围绕一个或多个主题，通过对多种操作型数据源中的数据抽取、清洗、转换后加载形成的。为了正确地表示历史数据，数据仓库中维度表和事实表之间的每个连接都应该采用没有明确含义的整型代理关键字来建立[2]。因此，在进行XML数据文档与数据仓库间的转换时，如何处理代理关键字是必须要解决的问题。

1 代理关键字

1.1 代理关键字的概念

代理关键字，有时又称为虚义关键字、指定关键字，就是在填充维度时按照需要，顺序分配的多个整数值[2]。在实际设计中，它可以是一个以字段属性为自动编号的标识符列，用来识别表中的每条记录。例如，为第 1 条记录分配的代理关键字的值为 1，第 2 条分配的值为 2 等。代理关键字仅仅用于维度表到事实表的连接，它本身没有特别的含义。

1.2 代理关键字的作用

(1)对操作型数据的变化进行缓冲。代理关键字能够保证数据仓库不会受到自然关键字编码更新、删除、再生与重用等操作型规则的影响。在许多机构中，历史操作型编码(例如，非活动性账号编码或者过时的产品编码)会在废弃了一段时间之后重新使用。这些编码在操作型数据库中往往是被设定为主键的属性列，操作型数据库中保存的数据的存活期是较短的，只要避开这个时间周期，即使重新使用已经废弃的操作型编码也不会出现关键字重复的错误。但是数据仓库就不同了，它要把数据保留多年。如果关键字仅仅依赖于使用操作型编码，那就容易遇到数据采集或者在合并情况下关键字重叠的问题。因此，当事务标识编号在跨地域范围内不是唯一的或者需要进行重用时，代理关键字就是必要的。此外，当构建数据仓库需要对来自多个操作型源系统的数据进行合并时，如果它们之间缺乏一致的源关键字，也只有通过使用代理关键字将它们合并到维度表中。

(2)支持处理维度表属性的改变。在数据仓库中维度被看作是与时间无关的[2]。但事实上，由于实际需求的变化，维度属性不可能永远固定不变，因此这种变化相当缓慢。代理关键字技术支持对属性值在操作领域发生变化时的处理。例如，当涌水量参数所属的部门发生改变时，如果通过插入一行新记录来反映部门属性值的变化，如表 1 所示。

表 1 参数基本信息

参数编号(自然关键字)	参数名称	部门
SC001190	涌水量	运通组
SC001190	涌水量	安质科

基金项目：国家自然科学基金资助项目(70533050)
作者简介：葛 欣(1980－)，女，博士研究生，主研方向：数据仓库，数据挖掘技术及应用；丁恩杰，教授、博士生导师
收稿日期：2007-04-28 **E-mail：**gexin@cumt.edu.cn

涌水量就有了两个记录行，这时如果将参数编号作为主键就出现了关键字重复的错误使用代理关键字就能够避免这种问题的出现，如表 2 所示。通过为相同的参数编码(即实体的自然属性编码)分配两个不同的代理关键字。每个代理关键字可以唯一标识一个实体在某个时间跨度内的属性概况。

表 2　参数基本信息

代理关键字	参数编号(自然关键字)	参数名称	部门
12345	SC001190	涌水量	运通组
25984	SC001190	涌水量	安质科

这样，利用代理关键字也保证了事实表不被修改。

(3)当维度表中的主键包含了所有属性列时，引入代理关键字来标识每条记录，有利于简化维度表和事实表之间的连接。

(4)使用代理关键字还可以获得性能上的优势。虽然代理关键字只占据一个整数的空间，但是能最大限度地满足维度行以后可能需要的序号或者最大编号。

2　XML Schema 与数据仓库模型的转换

Shanmugasundaram J提出了将XML文档的DTD (Document Type Definition)映射为关系模式的结构映射方法[3]；Dongwon L等提出了保持语义约束的CPI算法[4]。与DTD模式相比，Schema模式具有强大的复杂数据类型定义和数据结构描述功能。因此，对基于Schema的XML与关系模式映射算法的研究得到了更多的关注。文献[5-9]提出了基于固定映射方式的XML Schema到关系数据库的转换算法。针对其局限性，贝尔实验室的Bohannon P等提出了基于最小存储代价的转换方法[10]。

笔者利用 Java 中提供的工具包，将 XML 文件转化成 XOM 树状结构，分析树结构中每个结点的性质，引入代理关键字，将树结构转化成平面的关系模式 *R*，进而转化成标准的 SQL 语句，实现了 XML Schema 到数据仓库结构的映射。转换流程如图 1 所示。

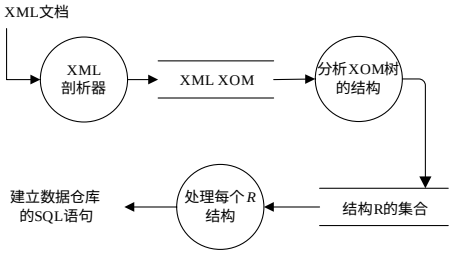

图 1　转换处理的流程

2.1　解析成 XOM 树结构

XOM与DOM类似，也是一种面向对象的XML应用编程接口(XML API)，可以利用它在内存中建立 XML 文档的完整映像，进而转换成某种面向对象的解释。由于它的目标是用最小的相关对象的方法集来捕捉和实施XML的精确信息集[11]，因此它提供了更加丰富、正交性强的API，可以在编程语言中调用这些API来遍历、筛选和转换XML文档中的对象。

对于一个 XML Schema 文档，利用 Java 中提供的 XML 剖析器能够生成对应的 XOM 树，其结构类似于图 2。每个结点都是派生自 Node 类的对象，其中 Text, Comment, ProcessingInstrction 节点仅仅是对父节点性质的说明，不能有子节点。而 Document, Element 节点是对 XML Schema 文档中对象的抽象，可以带有子节点。每个类型的节点可以在各层上出现多次，但一个 XML 文档只能有一个 Document 对象。

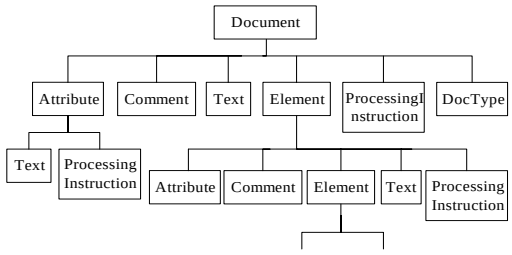

图 2　XML Schema 转换成 XOM 树的结构

2.2　XOM 到关系模式的转换

在 XOM 树结构中，每层对应一个关系，层中没有孩子的结点直接对应为关系中的属性，有孩子的结点则表示了两个关系间的联系。由于 Comment 和 ProcessingInstruction 仅仅是对 XML 文档的注释和名称空间的说明，在转换时被忽略。DocType 指示了当前 XML 文档的 DTD 描述，在进行 Schema 转换时也被忽略。

定义　关系模式 *R* 是一个包含了 4 个元素的结构 *R*(*U, K, FK, I*)，其中，*R* 表示关系名；*U* 表示 *R* 关系所包含的非码属性集合；*K* 表示 *R* 关系中的主码；*FK* 表示 *R* 关系的外码；*I* 表示作用在属性集 *U* 上的一组约束。为了简化结构体，这里的 *U, K, FK* 记录了属性的名称和类型。

算法　将 XOM 树结构转换成关系模式 *R* 的集合。

输入：XOM 树的根节点 root

输出：关系模式 *R* 的集合

广度优先搜索 XOM 树，对每个节点 node：

(1)查找是否存在一个以 node 父节点的名称命名的结构 *R*。

(2)若没有，则创建之；若存在，则对其操作。

(3)在 *K* 中添加“父结点名_id”属性列为代理关键字，类型设为“自动增加”。

(4)判断 node 类型：

如果是 Attribute 则将该属性名及其类型添加到 *U* 中；

判断类型是否为 ID，如果是则在 *I* 中添加“该属性不能为空”的 SQL 约束语句；

如果是 Element 则判断其是否有孩子节点；

如果没有，则将其属性名和类型添加到 *U* 中；

如果有，则在 *FK* 中添加“结点名_id”属性列为外码，类型设为整型，同时将该结点入队。

(5)只要队不为空，节点出队，转到(1)执行。

对于上述算法生成的每个结构 *R*，将其中的每个分量处理成字符串后作为参数，传给 Java 存储过程，就可以建立相应的数据仓库模型。

2.3　实验

为测试上述算法，笔者采用的开发环境是 jdk1.5.0+jre1.5.0+Eclipse+Myeclipse+ Tomcat5.0。输入要解析的 schema 文件，以.xsd 结尾。例如：

```
<schema>
<element name="e1" type="ct1" />
<complexType name="ct1">
<sequence>
    <element name="e11" type="string" />
  <element name="e12" type=" string" minOccurs="0" maxOccurs="unbounded" />
    </sequence>
```

```
    <attribute name="a11" type=" string" />
    </complexType>
……
</schema>
```

输出解析生成的数据仓库文件，以.sql 结尾，见图 3。

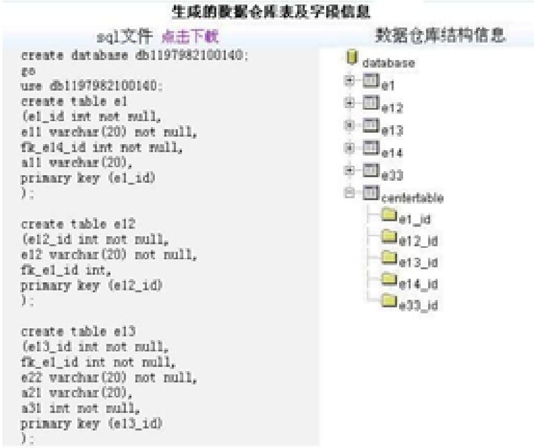

图 3　转换后的 SQL 文件界面图

其中主要的转换程序 Database.java 结构如图 4 所示。

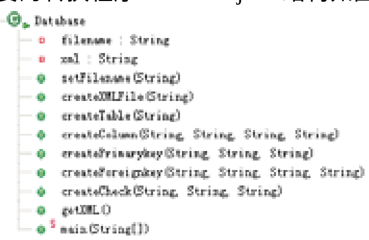

图 4　Database.java 程序结构界面图

通过该系统对不同 XML 文件的测试，证明了算法的有效性。

2.4　讨论

这样构建的数据仓库会使得维度表没有达到关系规范化中的第三范式(3NF)[3]，将这种结构倒退的维度表称之为退化维度表，但是从数据仓库的使用和高性能的角度来说，这种退化维度表反而会比规范化维度表更能满足这两个基本设计要求。

一般来说，通过规范化维度表节省下来的磁盘空间都少于整个设计结构所需磁盘空间总量的 1%[2]。规范化处理增加了维度表的数量，使跨表查询的连接操作变得复杂，这样不仅降低了查询性能，也增加了优化器的开销。另外，规范化维度表制约了跨属性的浏览操作，并禁止使用对位图的索引。因此，可以牺牲这点维度表空间来换取高性能与易使用方面的优点。

3　结束语

本文通过对象映射的方式，利用 XOM 树结构实现了由 XML Schema 到数据仓库模型的转换。算法侧重于数据仓库结构的建立和对于属性级上的限制，主要依赖于 XOM 树本身提供的信息提取功能来实现。该算法借助于现有的技术，避免了直接映射的繁琐。随着各大厂商不断推出对 XML 相关技术的支持，这种基于对象映射的方法会有良好的发展前景。

参考文献

[1] Harold E R. XML 技术手册[M]. 北京: 中国电力出版社, 2001.
[2] Kimball R, Ross M. 数据仓库工具箱[M]. 北京: 电子工业出版社, 2003.
[3] 萨师煊, 王　珊. 数据库系统概论[M]. 北京: 高等教育出版社, 2006.
[4] Dongwon L. CPI: Constraints-preserving In-lining Algorithm for Mapping XML DTD to Relational Schema[J]. Data & Knowledge Engineering, 2001, 7(6): 3-25.
[5] Florescu D, Kossmann D. A Performance Evaluation of Alternative Mapping Schemes for Storing XML in a Relational Database[C]//Proc. of the VLDB. Inria, France: [s. n.], 1999.
[6] Klettke M, Meyer H. XML and Object-relational Database Systems Enhancing Structural Mappings Based on Statistics[C]//Proc. of WebDB. Dallas: [s. n.], 2000: 63-68.
[7] Schmidt A, Kersten M, Windhouwer M. Efficient Relational Storage and Retrieval of XML Documents[C]//Proc. of WebDB. Dallas: [s. n.], 2000: 47-52.
[8] Madhavan J, Bernstein P A, Rahm E. Generic Schema Matching with Cupid[C]//Proc. of the 27th International Conference on Very Large Databases. Roma: Morgan Kaufmann Publishers, 2001: 49-58.
[9] Varlamis I, Vazirgiannis M. Bridging XML-schema and Relational Databases a System for Generating and Manipulating Relational Databases Using Valid XML Documents[C]//Proceedings of the ACM Symposium on Document Engineering. Atlanta, Georgia, USA: [s. n.], 2001: 105-110.
[10] Shanmugasundaram J, Tufte K, Zhang C, et al. Relational Databases for Querying XML Documents: Limitations and Opportunities[C]//Proc. of VLDB. Edinbergh, Scotland: [s. n.], 1999.
[11] Bohannon P, Freire J, Roy P, et al. From XML Schema to Relations: a Cost-based Approach to XML Storage[C]//Proceedings of the 18th International Conference on Data Engineering. San Jose, CA: IEEE Computer Society Press, 2002: 64-75.

~~~~~~

(上接第 83 页)

参考文献

[1] SAMSUNG. S3C2410X 32-Bit RISC Microprocessor User's Manual[Z]. 2003.
[2] 詹荣开. 嵌入式 Bootloader 技术内幕[EB/OL]. (2003-12-22). http://www.900.ibm.com/developerWorks/cn/linux/l-btloader/index.html.
[3] Garin L. ARM Bootloader 的实现——C 和 ASM 混合编程[EB/OL]. [2007-02-16]. http://embedded.homeunix.org.
[4] 陈　赜. ARM9 嵌入式技术及 Linux 高级实践教程[M]. 北京: 北京航空航天大学出版社, 2006.
~~~~~~