一种基于关系数据库 XML 存储方案的数据库模型①

马竹娟[1]，汪宏喜[2]

[1](安徽农业大学 信息与计算机学院，合肥 230036)
[2](安徽农业大学 理学院，合肥 230036)

摘 要：存储方案是 XML 数据管理研究领域的核心问题之一。底层的存储表达在性能上对上层的查询和优化有着重要的影响，因此有效地建立 XML 文档的存储方案是首要问题。以关系数据库为基础，提出了一种基于边模型映射的 XML 存储方案的数据库模型。在该模型中，首先依据文档中的结点类型对 XML 文档树进行结构细化；其次采用联合表来存储结点的值，并在该表中采用结点序号和路径相结合的方式，仅记录每一个元素结点的路径；最后在这种新的存储模式的基础上，实现了针对该模型的数据库操作。实验表明，该模型在查询速度和存储空间方面较以往的工作都有明显的改善。

关键词：存储方案；边模型；文档树；联合表；数据库模型

Relational Database Model of XML Storage Schema Based on Relational Database

MA Zhu-Juan[1], WANG Hong-Xi[2]

[1](School of Information and Computer Science, Anhui Agricultural University, Hefei 230036, China)
[2](School of Natural Sciences, Anhui Agricultural University, Hefei 230036, China)

Abstract: Storage schema is a key research field in the community of XML data management. About the performance, expressions of the underlying storage have very important effects on the top queries and optimizations. Therefore, effectively building the storage schema of the XML is the primary problem. This paper proposed an edge-mapping XML document storage schema, which effectively maps the XML database onto relational database. In this schema: firstly, the XML document tree was structured according to the type of the document node. Secondly, a union-table was used to store the value of nodes, and we only recorded the path of the element nodes by the mean of the node sequence number combined with the path. Finally, based on this storage schema, we implemented the database operations on the new database model. Experiments showed that the model can speed up the query procedures and can spare more memory space.

Keywords: storage schema; edge model; document tree; union-table; database model

利用关系数据库对 XML 数据进行管理，就是将 XML 文档看作为有具体数据的对象树，关系映射理论经过中间件进行转换，从而映射到关系数据库中。

1 引言

在 XML 数据处理中，底层的存储表达在性能上对上层的查询和优化有着非常重要的影响，因此有效地建立 XML 文档的存储方案是首要问题。目前 XML 数据库的存储策略主要有以下三种：(1)利用文件系统的平面文件，这种方式的最大优点是容易实现，且不需要使用数据库系统和存储管理器，缺点是每次访问 XML 文档时都需要解析它，并且在查询处理期间，整个被解析的文件都必须驻留在内存里，一般不支持索引查询，也不容易修改文档；(2)面向对象数据库管理系统（OODBMS），其最明显方法就是把每一个 XML 元素存储成一个独立的对象，但由于 XML 元素通常

① 收稿时间:2010-08-13;收到修改稿时间:2010-09-11

很小，这种方法的空间开销通常很高[1]；(3)利用关系数据库系统（RDBMS），传统的关系数据库系统目前已趋以成熟，且得到广泛的应用，最近的研究成果[2,3]说明了如何将 XML 文档映射并存储到关系数据库系统中。

文献[2]描述了边模型的映射方法，将一个 XML 文档看成有序有向标签图，按照广度优先顺序赋予有向图中的每个结点唯一的 ID，将图中的每条边信息存储在边表(Edge)中。这种方法优点是易于实现，但缺点是，该方法着重于维护 XML 文档树的边信息，所以当进行路径表达式计算时会产生大量的连接操作。文献[3]提出一种基于路径索引的 XML 文档的关系存储模型 XRel。它是将 XML 文档依据其文档树的结构分解成结点，再根据结点的类型分别存储于相关的关系表中，并记录每一个结点的路径信息。这种数据库模式的优点在于它与 Xpath 标准紧密结合，有效地支持了基于 Xpath 的查询。缺点是由于区间编码是对文档进行两次遍历后产生，使得每次添加和删除结点时都需要相当复杂的操作。

本文结合边关联方法和基于路径的方法，提出一种优化的方案 XUni。其基本思想是首先将一个 XML 文档看成有向标签图，然后按照广度优先顺序赋予图中的每个结点唯一的 ID(即序号)，并按结点的类型有区别地将它们的存储在同一张关系表中，即联合表(Union-table)。同时对所有的元素结点采用结点序号与路径相结合的方式记录其路径信息，以避免不同的元素结点产生相同的路径信息。

2　基于关系数据库的XML存储方案

在 Xpath 数据模型中，格式良好(well-formed)的 XML 文档被抽象地映射成由 7 种不同类型结点构成的有序树模型，这 7 种类型的结点分别是名字空间结点、处理指令结点、注释结点、根结点、元素结点、属性结点、和文本结点[4]。由于元素、属性、文本这三种信息涵盖了 XML 文档的主要内容，所以本文以这三种结点为例进行讨论。本文以图 1 所示的 XML 文档片段为例。图 2 是对应于图 1 实例文档的有序树，其中不同的形状表示不同类型的结点。按照广度优先的顺序，有序树中的每个结点赋予唯一的序号，按序号升序存储到关系表中。

```
<pub>
    <book page="356">
        <title> Introduction to XML</title>
        <reference>
          <book page="490">
              <title>Database System Concepts</title>
          </book>
        </reference>
    </book>
    <book page="380">
        <title> A Query Language for XML</title>
    </book>
</pub>
```

图 1　一个 XML 文档实例

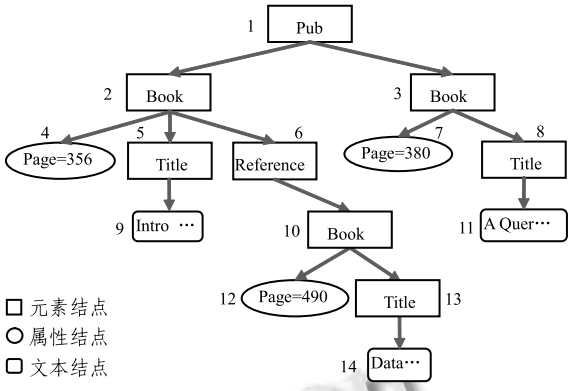


图 2　对应于图 1 的有序树

2.1 建立联合表

由于建立多张关系表往往会因为其索引而导致数据库空间增大[2]。因此本文采用联合表的形式，即将元素结点、属性结点和文本结点的信息存储在一张表中。联合表的数据模型形如:

Uni-Tab = Uni (Order, Source, Name, Lorder, Rorder, ValintPointer, ValstringPointer, PathPointer)

其中 Order 为主码。Order 是对应结点的广度优先遍历序号；Source 是该结点的父结点序号，取值为-1 时表示该项为空；Name 是该结点的名称，文本结点的名称均为 Null；Lorder（Rorder）表示该结点的左（右）兄弟结点的 Order，取值为-1 时表示该项为空；ValintPointer 是属性值指针，若该结点不是属性结点，则该项为 Null；ValstringPointer 是文本值指针，若该结点不是文本结点，则该项为 Null；PathPointer 为路

径指针，由于路径表达式在 Xpath 查询中被频繁地使用，因此为了提高查询以及更新的效率，每个元素结点都记录下从根结点到当前结点的路径信息，其余结点 Path 信息为 Null。同时为避免不同的结点出现相同的路径信息，如结点 2 和结点 3 的路径信息均为#/pub#/book。本文采用结点序号与路径相结合的方式，如结点 2 的 Path 信息记录为#/1-pub#/2-book。

由于联合表对整个 XML 文档进行存储，若要在其中记录所有文档中的信息，则该表在系统运行时占用的空间非常大，甚至在 XML 文档非常大的情况下，系统无法运行。因此在本文的联合表中，只记录结点的基本信息，比如结点序号、结点名称等。将占用空间比较大的属性值、文本值以及结点的路径信息分开存储，在联合表中只记录它们的存储位置(Valint Pointer, ValstringPointer, PathPointer)，因此在进行数据库操作时，根据具体指针存取相应的数据即可。数据可能在内存中，也可能存放在外部海量存储设备中。图 3 为联合表和各个存储具体值的位置之间的关系。

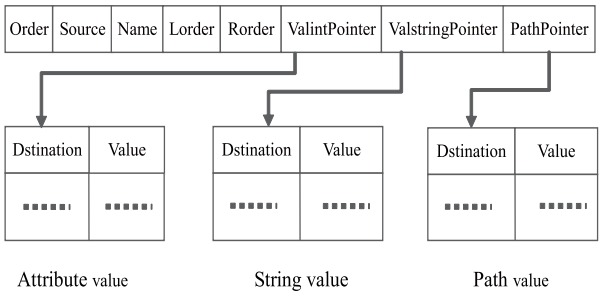


图 3 值的存储

2.2 节点的区分及各节点之间的联系

XML 文档在联合表的存储模式下，可以根据以下的五个性质，有效地实现节点的区分以及建立各节点的关系：性质 1：图中任意元组 n，若 n.source = -1，则 n 是根结点。性质 2：图中任意元组 n，若 n.path ≠null,，则 n 是元素结点；若 n.path=null 且 n.name≠null 则 n 是属性结点。性质 3：图中任意两个元组 n1、n2，若 n1.path＝prefix of n2.path，则 n1、n2 是元素结点，且 n1 是 n2 的祖先结点，n2 是 n1 的子孙结点。性质 4：图中任意两个元组 n1、n2，若 n1.source＝n2.source，则 n1 和 n2 是兄弟结点。性质 5：图中任意两个元组 n1、n2，若 n1.source＝n2.order，则 n2 是 n1 的父结点，n1 是 n2 的孩子结点。

3 新存储模式下的数据库操作

3.1 重构 XML 文档

在 XML 数据库管理中最重要的是查询操作，而在查询操作中最重要的是将在 RDBMS 中得到的查询结果有效准确地转换成 XML 文档，即 XML 文档的重构。在新的存储模式下，可以根据前面提到的五个性质，有效地实现对原 XML 文档的重构。

例如构建 page=356 对象的文档，它对应在关系上的查询为：通过 Valint 域找到该对象的 Order 为 4 和 Source 为 2，根据性质 3 查找以 2 为祖先结点的所有子孙元素结点，根据性质 2、性质 4 和性质 5 查找子元素结点对应的属性结点和文本结点。最后构建 XML 文档为：

```
<book page="356">
    <title> Introduction to XML</title>
    <reference>
        <book page="490">
            <title>Database System Concepts</title>
        </book>
    </reference>
</book>
```

3.2 XML 文档的更新

XML 文档的更新一般分为三种操作：插入一个新结点、删除一个结点和修改结点内容。删除结点和修改结点内容很容易实现，以下主要讨论的是插入操作。以元素结点为例，（设在在元素结点 m 上插入新元素结点 n）n.Order := maxorder + 1；n.Source := m.Order ；n.Lorder:= -1 ； n.Rorder:= -1 ； n.Valint=null ；n.Valstring=null；n.Path := m.Path &#/ n.Order-n.Name（&表示连接）；若结点 m 有孩子结点，则根据新结点 n 插入的位置，分为以下三种情况：n.Order := maxorder + 1 ； n.Source := m.Order ； n.Valint=null ；n.Valstring=null；n.Path := m.Path &#/ n.Order-n.Name（&表示连接）；（1）左右兄弟同时存在，设左兄弟结点为 n1，右兄弟结点为 n2，则：n.Source := n1.Source , n.Rorder := n2.Order, n2.Lorder := n.Order ,n1.Rorder := n.Order , n.Lorder :=n1.Order。（2）只有左兄弟存在，设左兄弟结点为 n1，则：n.Source := n1.Source , n1.Rorder := n.Order , n.Lorder :=n1.Order , n.Rorder := -1。（3）只有右兄弟存在，设右兄弟结点为 n1，则：n.Source := n1.Source ,

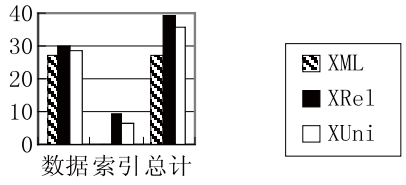
n1.Lorder := n.Order , n.Rorder :=n1.Order , n.Lorder := -1。若 n 为属性结点或文本结点，则把 n 的名称和值输入相关的关系表中，方法与上述类似。

4 实验分析

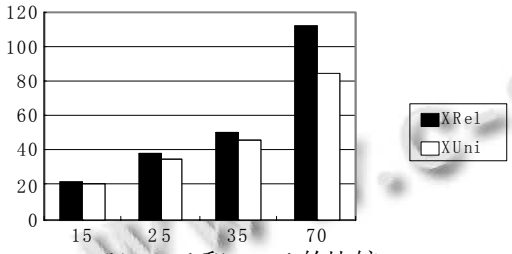
实验与 XRel 方法进行对比，主要考虑文档存储性能，采用 XMark[5]作为实验数据集。实验的硬件环境：CPU 为 P(R)4 2.8GHz，内存为 1G。软件环境为 Windows XP，开发平台为 C++，数据库服务器为 SQL Server 2000。

4.1 存储性能分析

图 4 显示了两种模型映射方法在存储空间上的比较。在图 4(a)中，将本方案与纯 XML 文档、XRel 模型进行比较。实验表明，XUni 较 XRel 在存储空间上有明显的提升。这主要是因为 XRel 方法是将不同类型的结点分别存储在对应的表中，多张表格产生大量的索引空间，而 XUni 采用的是一张表格，因此在总体上占用相对较小的空间。从图 4(b)可以看到，随着 XML 文档的增大，XUni 的效果也越为明显。



(a) 纯文本文件、XRel 和 XUni 的比较



(b) XRel 和 XUni 的比较

图 4　存储空间比较（单位：M）

4.2 查询响应速度分析

实验选用的查询从短路径、长路径、查找属性、匹配等六个方面进行考查：Q1：/pub/book；Q2：/pub/book/reference/book/title；Q3：//book/title；Q4：//@page；Q5：/pub/book[price>35.00]；Q6：/pub/book [@page=’356’]/title。从图 4 中可以清楚的看出，XUni 方法比 XRel 都有提高，尤其 Q6 最为明显，这主要是因为 Q6 在 XUni 节省了因索引和连接操作所占用了的时间。

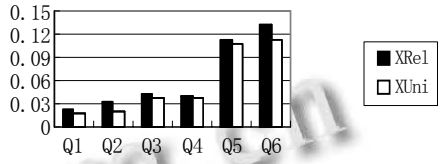


图 5　查询时间比较(单位：s)

5 结束语

本文提出了一种优化的 XML 文档存储模型。该模型将边关联的方法和基于路径的方法相结合，首先细化了文档树中的结点类型；其次采用联合表的形式存储结点值；最后通过结点序号和路径相结合的方式记录了元素结点的路径信息。针对这种存储模式，实现了一种针对 XML 数据进行处理的数据库模型。实验表明，本存储方案及其相应的存储模型在存储空间及查询速度方面较具有代表性的 XRel 方法有明显的优势。

参考文献

1 冯建华,钱乾,廖雨果,李国良,塔娜,周立柱.纯 XML 数据库研究综述.计算机应用研究,2006,(6):1−7.

2 Florescu D, Kossman D. Storing and querying XML data using an RDBMS. IEEE Data Engineering Bulletin, 1999, 22(3):27−34.

3 Yoshikawa M, Amagara T, Shimura T, Uemura S. XRel: A path-based approach to storage and retrieval of XML documents using relational databases. ACM Trans. on Internet Technology (TOID), 2001,1(1):110−141.

4 乔磊,李曦,龚育昌,周志鹏.一种支持 XML 的文件系统构建模型.小型微计算机系统,2008,29(8):1431−1436.

5 Schmidt A, Waas F, Kersten M, Carey M, Manolescu I, Busse R. XMark: a benchmark for XML data management. Proc. of the 28th International Conferences on Very Large Data Bases. Hong Kong, China, 2002: 974−985.