

OrientX3.0: 一个支持更新的原生 XML 数据库系统

朱金清 张新 孟小峰

一、OrientX 系统简介

随着因特网应用的发展, XML 逐渐成为数据描述和数据交换的标准, 大量的 XML 文档出现在网络中, XML 数据的有效管理变得越来越迫切。XML 具有自描述性, 结构嵌套等特性, 将 XML 数据转换成其它数据模型的数据库管理, 将带来效率与复杂性问题。为此, 中国人民大学网络与移动数据管理实验室 (WAMDM) 孟小峰教授带领他的研究团队研究与开发了一个 Native XML 数据库管理系统 OrientX。OrientX 也是中国第一个自主研发的 Native XML 数据库系统。Native 意思原生的, 指的是处理 XML 数据是基于 XML 数据模型之上的。

根据 XML 数据库系统基于的数据模型可以将 XML 数据库管理系统分为两类, 一是 Native XML 数据库系统, 另一类是非 Native XML 数据库管理系统。前者是基于 XML 数据模型上, 即树模型。而后者是基于其它数据模型上, 如关系模型, 由于 XML 数据的结构复杂, 自描述性等特点, 转换成其它数据模型往往带来效率上的问题, 并带来更新维护及易操作性等问题。原生 XML 数据库管理系统在数据库的各个方面上进行了创新, 包括 XML 数据存储, 索引, 查询, 更新等。孟小峰教授带领他的研究团队从 2001 年起逐步对 Native XML 数据存储, 索引, 查询 (XQuery/XPath), 查询优化等实现技术进行了重点研究, 逐步发布了 OrientX 1.0, 1.5、2.0、2.5 版本。OrientX 的研发也一直围绕着如何高效处理 XML 数据, 随着研究不断深入, OrientX 的功能也在不断完善。2007 年 1 月 W3C 提出的 XQuery/Update 语法草案。OrientX 研究团队对 XML 数据的高效更新, XQuery/Update 的处理技术进行了深入研究, 于 2007 年 9 月发布了 OrientX 最新版本 3.0, 支持 W3C 于 07 年 1 月提出 XQuery/Update 语法草案, 为了有效支持 XML 的更新, 我们对原 Native XML 存储 OrientStore 进行了改进, 提出了一种更加有效的存储策略。与此同时, 针对系统由于多年的开发积累的系统结构不合理的特点, 我们对系统体系结构进行了改进, 使得系统更加稳定。

二、OrientX 系统特性

OrientX 系统作为一个 XML 数据库管理系统, 管理 XML 数据的基本单位是 XML 文档, 将 XML 文档组成一个个数据集进行管理 (相当于关系数据库中的数据库), OrientX 具有如下的系统特性:

1. 充分利用模式 (Schema) 信息

XML 模式是对 XML 数据的描述, 它虽然并不像关系数据库中的模式是用来严格限制数据的。但有效利用 XML 模式信息在 XML 存储, 索引, 查询处理与查询优化有很大的满足。正是出于这一点, OrientX 根据 XML 文档的模式信息提出了基于模式的存储方法, 索引方法, 并在查询处理与优化利用模式信息来提高查询效率。

2. 基于遵循 W3C 推荐的 XML 数据模型标准

XML 数据模型是处理 XML 数据的基础, OrientX 出发点就是基于 XML 数据模型来高效处理 XML 数据。OrientX 中 XML 数据的存储, 查询处理都是符合 W3C 提出 XML 数据模型标准的。

3. 压缩存储与多样化的数据存储方式

所谓压缩存储, OrientX 在将 XML 文档导入数据库中时, 会将 XML 文档中的节点名, 属性名等映射成一个唯一的整数 ID, 如图 1:

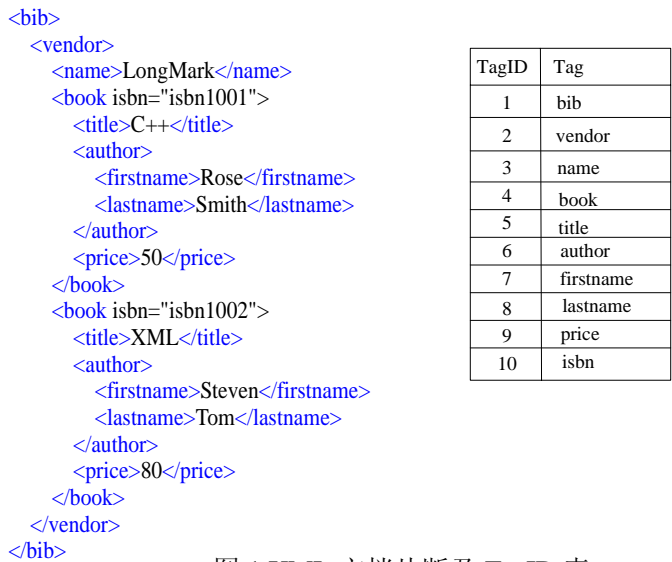


图 1 XML 文档片断及 TagID 表

如图 1， 左边是一个 XML 文档片断， 右边为相应的 TagID 表， 这样既可以压缩存储空间， 也可以提高查询速度(字符串匹配转化为整数比较)。

所谓数据组织和存储方式， 就是如何划分 XML 数据组成一个个的记录， 如何存储这些记录的关系等。 OrientX 系统支持四种存储策略： Element-based(EB)、 Subtree-based(SB)、 Clustering Subtree based (CSB)、 Clustering Element-based (CEB)。 用户可以根据查询数据的需求和文档的不同特点， 选择合适的存储策略。

4. 数据存取路径

数据存取路径， 是指如何从数据库中读取数据， 是顺序扫描、 还是通过索引随机读取等。 OrientX 系统提供最常见的类似于 DOM 接口， 可以根据当前 XML 数据结点导航遍历其父亲、 孩子和左右兄弟结点； 此外还支持三种索引： 基于元素的索引(Elementary Index)， 路径索引(Path Index)和值索引(Value Index)。

所谓基于元素的索引 (Elementary Index)， 是指对元素名称 (element name) 进行索引； 有了此索引， 就可以一下子读取到指定名称的所有元素结点在数据库中的地址 (Element node)。

所谓路径索引 (path index)， 就是对一个 XPath 路径进行索引； 通过该索引， 可以返回匹配指定路径的元素有序对。 例如对路径 A/D 建立索引， 通过此索引就可以返回一系列的(A,D)有序对， 它指示着符合路径 A/D 的嵌套关系 (父子或祖先后代关系) 的结点对在数据库中相应的地址。

所谓值索引 (Value Index)， 类似关系数据库中的索引， 它可以直接定位指定的 text 值或 attribute 值的元素结点。 例如， 假设在 book/@price 上建立值索引。

5. 支持 W3C 推荐的 XQuery/XPath/以及最新的 OrientX/Update 标准

如同关系数据库需要用 SQL 语言来查询数据一样， XML 数据库也需要有一种查询来从海量的 XML 数据检索出满足特定条件的数据片断。 在诸多的 XML 查询语言中， W3C 推荐的 XQuery 脱颖而出， 逐渐成为 XML 数据检索的标准。

OrientX 系统所支持的 XML 数据查询语言遵循 W3C 推荐的 XQuery 标准。 虽然 OrientX 系统能够解析所有的符合该规范的 XQuery 查询， 但是它并没有处理所有的 XQuery 情形， 例如 XQuery 中关于数据类型的表达式(cast as、 instance of、 type switch 等)， 用户自定义的函数， 部分内嵌的函数(如取系统时间， 日期等)， 这些情形的 XQuery， 当前版本的 OrientX 暂时没有支持。 OrientX 处理了 XQuery 的核心部分， 例如 FLWR Expr， Construct Expr， XPath Expr 等。

同样在如通过关系数据库中的 SQL 语言一样， XQuery 也包含更新的语句。 在 2007 年 1 月 W3C 推出了新的 XQuery/Update 草案， 因此 OrientX 也在 9 月份推出它的 3.0 版本， 其主要的特性之一就是支持 Update。 XQuery/Update 包含 insert、 delete、 rename、 replace 和 transform 五种类型的更新语句， 在 OrientX3.0 中主要支持前面四种更新操作。

三、OrientX 的发展历史

OrientX 从诞生之初，就是围绕着 XML 数据库中技术难点进行研究突破，包括 XML 数据的存储、索引、更新、查询、编码等一系列问题进行了重点研究，随着研究的不断深入，OrientX 在功能与体系结构上不断地完善。OrientX 从 2001 年的 1.0 版本，到现在的 3.0 版本已经有近 7 个年头，从基本的存储管理开始，到 XPath 查询处理，到 XQuery 查询处理，到索引管理，到 XML 代数，到现在的支持 XQuery/Update，接下来我们大概从全局浏览一下各个版本主要实现的功能：

1. OrientX1.0

OrientX1.0 版本主要实现存储管理、基本的导入导出操作、简单的元素级别 (Element-Based) 的和子树级别 (Subtree-Based) 的存储。对于一个符合 W3C 推荐的规范 Schema 的 XML 文档，我们都可以实现将其导入到系统内部，可以导出文档等。下面简单阐述一下 3 个主要功能。

a) 文件管理模块

文件管理模块式为了整个系统负责管理所有的涉及磁盘文件创建、打开、删除和读写的操作，实现文件读写，为上层存储模块提供独立于操作系统的接口。

b) 缓冲区模块

为了提高效率，减少 I/O 次数，OrientX 缓冲了最近访问过的数据块。OrientX 的缓冲区模块负责将物理页面读入到缓冲区、申请一个新的页面以及页面的释放等等。OrientX 采用经典的 LRU 页面置换算法。

c) 导入和导出

在对 XML 数据的管理之前，首先最简单的就是文档的导入和导出，对于一个符合 W3C 推荐的规范 Schema 的 XML 文档，我们都可以将其导入和导出。

2. OrientX1.5

OrientX1.5 主要实现了 XPath 查询引擎、XML 编码等。接下来简单介绍一下：

a) XPath 查询引擎

XPath 查询引擎负责将 XPath 查询转化为语法树，然后对语法树进行分析，生成 XPath 查询引擎所能识别的内部执行计划，接着根据一定的规则对这个计划进行优化，最后按优化后的执行计划执行将结构返回给用户。

b) XML 数据编码

为了保持 XML 中数据之间的关系，许多方法提出了实现方法。其中，编码来保持结点之间的关系有许多优势，比如节省存储空间、快速判断结点关系等。在 OrientX 中实现了区域编码方法——RegionCode。

3. OrientX2.0

OrientX2.0 与上一版本，有较大的飞跃，实现了 XQuery 的导航查询引擎和 XML 的更新。

a) XQuery 导航查询引擎

所谓基于导航的处理，就是根据 XQuery 查询的导航性语义，采用类似 DOM 接口的方式遍历 XML 文档，抽取出符合特定条件的数据片断。XQuery 导航查询引擎首先对 XQuery 查询语句进行词法和语法分析，然后生成相应的执行计划，接着对生成的计划进行优化，最后进行查询执行。

b) XML 更新

向用户提供更新语句，实现对 OrientX 数据库中存储的数据文档的增删改操作，包括增加、删除、修改元素、属性和值。增加的元素可以是同一文档中符合某些谓词的元素，也可以是其他文档中的元素。

4. OrientX2.5

OrientX2.0 中采用的是导航式处理；OrientX2.5 中实现 XQuery 的代数式处理。OrientX2.5 版本增加了 XQuery 代数处理模块，与已有 XQuery 查询处理模块在 OrientX 系统中具有相同的地位，提供相同的功能，只是实现策略不同。

XQuery 代数查询引擎

我们提出了一套基于语法的代数，命名为 XAlgebra。之所以称之为“基于语法的”，是因为 XAlgebra

里的操作符是与 XQuery 的语法单位相对应的，也就是说，XAlgebra 的每一个操作符对应于 XQuery 的有限确定个语法产生式，XQuery 的每一个语法产生式可以用 XAlgebra 的有限确定个操作符来表示。XAlgebra 中只设有 13 个操作符，但是可以表示绝大多数形式的 XQuery 查询。

5. OrientX 3.0

OrientX3.0 是最新版本，支持 W3C 推荐的 XQuery/Update 语法。详细特征在后续章节将逐一介绍。

四、OrientX3.0

OrientX 3.0 在系统原基础上，对系统的结构进行了调整，使得系统变得更可扩展，更加灵活。同时系统也增加了新的特性。OrientX 3.0 的主要特性有：1) 新的系统体系结构 2) 支持 W3C 新提出的 XQuery/Update 语法，3) 为了加有效地支持更新，实现了一种改的存储方法，4) 同时还提供了一系列的编程接口。现在主要按照下面几点进行展开分析：

1. OrientX 新的体系结构

OrientX3.0 将数据管理模块进行了重新封装修改，而且加入了一些新的接口——Update APIs。接下来简单介绍 OrientX3.0 的总体架构和各个组成模块的功能。如下图所示为 OrientX3.0 的架构图：

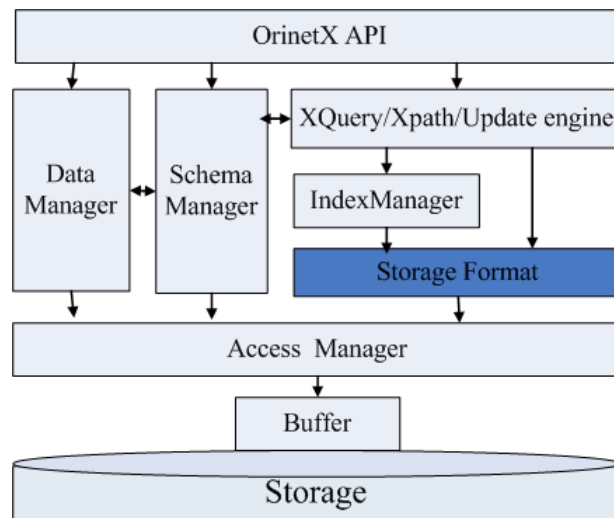


图 2. OrientX3.0 系统框架

从框架图 2 可以看出，OrientX3.0 主要包含以下几个模块：

a) 存储管理 (Storage Manager)

存储管理是 OrientX 系统的最底层的模块，它负责 XML 文档、Schema 文档、索引文件、以及系统临时文档的存储，其具体功能包括：划分记录块，确定物理文件结构和物理页内的记录顺序，缓冲系统最近访问的物理页，为上层的模块存取数据提供统一的接口。

b) 缓冲区管理 (Buffer Manager)

OrientX 系统为了减少 I/O 次数，缓冲了最近访问过的数据块，它为上层的模块提供的基本操作有：读一块物理页面到缓冲区、释放一块缓冲区、申请一块新的物理页面、释放一块物理页面。所有对数据库的操作都通过缓冲管理模块，所以申请和释放物理页面也是通过缓冲模块，再由缓冲模块调用底层的存储模块的方法，OrientX 采用近最少使用算法 (LRU)。

c) 存取管理 (Access Manager)

存取管理的功能主要是，对存储模块 (Storage Manager) 和缓冲区模块 (Buffer Manager) 进行包装，向上层数据管理模块提供统一的接口，使上层直接通过这个接口来存取数据。

d) Storage Format

Storage Format 是 OrientX3.0 新引进来的组件。根据存储记录的粒度，这些存储的方法可以分为基于元素结点 (EB)，基于子树的 (SB) 和基于文档的 (DB)。我们在研究过程中发现 XML 管理系统中模式对于设计良好的存储策略有着非常重要的意义。OrientX 系统根据模式信息设计了两种新的存储策略：基于元素结点的聚簇存储 (CEB)，基于子树的聚簇存储 (CSB)。OrientX 同样实现了上

子树的记录地址，子树也保存了父亲结点记录地址。如图 3 中的 R2 保存了 R5 与 R1 的地址，这种存储方法保证了结点间结构关系不丢失，但当一个记录的地址发生了变化时，可能需要修改多个记录，更新代价很高。

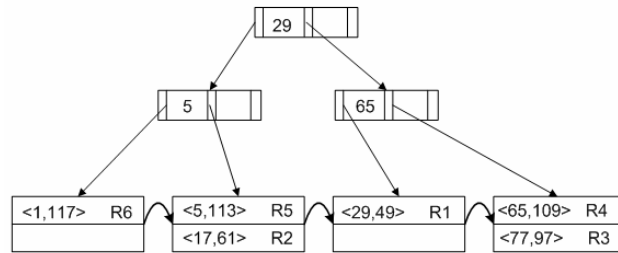


图 4 Code-Record Index

在 XML 存储时，每个结点还分配了一个编码，编码与结点的关系是一一对应的关系；在 OrientStore+ 中为结点编码与其存储地址建立映射关系 F 。若一个结点的编码为 c ，则 $F(c)$ 为其物理地址。易知 $F(c)$ 的值在系统中是唯一的。

这样，在记录中不再需要存储其子记录与父亲记录的地址，只需要保存相应结点的编码 c ，通过 $F(c)$ 得到相应的存储地址，而结点的编码通常是不改变的。

建立映射关系 F 的一种方法是建立结点编码(Code)到结点记录地址(Record-address)的索引，称之为 Code-Record index。图 4 是图 3 记录树的 B+ 树 Code-Record index。对于以子树为单位的 XML 存储，不需要建立每一个结点编码到记录地址的索引，只需要对每个记录的根结点建立索引即可。通过结点编码找其地址时，在 Code-Record Index 找到其最近的祖先结点编码对应的记录地址即可。如找编码 <33,37> 的地址，找 <29,49> 的记录地址即可。

在引入 Code-Record index 后，若一个记录的地址发生了改变，只需要对 Code-Record index 做修改，无需修改其它记录。建立索引只需要索引结点的编码，无需索引结点的记录地址。结点的记录地址发生了改变，不需要对索引进行修改，减小了索引的维护代价。

3. 基于代数的 Update 实现

XQuery/Update 语法草案是由 W3C 在 07 年 1 月提出来的，据我们最大的努力所知，目前只有 Galax 0.7.2, MonetDB/XQuery 支持 XQuery/Update 语法，XQuery/Update 语法主要有 5 类处理语句: insert, delete, replace, rename, transform。XQuery/Update 也都是基于 XQuery/XPath 数据模型之上。以下是两个 update 的语句示例：

例 1: insert node <year>2005</year>
after fn:doc("bib.xml")/books/book/publisher

在 publisher 结点后插入一个新的结点: <year>2005</year>。

例 2: rename node fn:doc("bib.xml")/books/book[1]/author[1]
as "principal-author"

将满足条件的 author 结点的结点名改为 "principal-author"。

XQuery/Update 语句可以与 XQuery 中的 FLWOR 语句任意嵌套组合，因此，为了有效地处理 XQuery/Update 语句，必须和已有查询处理引擎进行有效的结合。OrientX2.5 实现了 XML 代数查询处理，XML 代数处理最大的特点是，一次一集合方法，与关系代数一样，OrientX 中的代数也引入一系列的操作符。为了能与代数引擎有效结合，利用代数引擎查找出需要更新的数据，OrientX3.0 中 XQuery/Update 实现引入了四个操作符，分别为 Insert、Delete、Replace、Rename。这四类操作符分别对应相应的更新语句。新增加的四个操作符跟已存在的代数操作符一样，每一个操作输入是一个 pattern Tree，输出是满足条件 pattern Tree 实例，这样，新引入的四个操作符可以跟原来的操作符一样很好地实现流水线处理。

五. 总结

OrientX 系统是一个 Native XML 数据库管理系统，可供管理 XML 数据，具体包括导入模式 (schema) 文档创建数据库、删除数据库、导入存储 XML 文档、导出数据库或 XML

文档、XPath 和XQuery 查询, XML 数据更新等功能。

OrientX 系统为XML 数据管理相关研究工作提供了一个实验平台的。

OrientX 3.0 提供了Client-Server 的应用体系结构; 用户可以在客户端通过命令进行数据管理(建立删除数据库、导入导出文档、数据查询等操作)。

OrientX 系统提供一套API 接口, 支持XML 数据管理的应用开发。

关于OrientX的其它系统功能, 请登录我们的主页: <http://idke.ruc.edu.cn/OrientX>。

如果用户需要OrientX 系统相关的程序包、源代码、说明文档, 请向我们索取: OrientXRUC@gmail.com 或登录我们的主页: <http://idke.ruc.edu.cn/OrientX>。

参考文献

1. 孟小峰, 王宇等. OrientX: 一个 Native XML 数据库系统的实现策略. 第 20 届全国数据库学术会议, 2003, 10, 计算机科学, 卷 30(10), 111-115.
2. 罗道锋, 孟小峰等. OrientStore: Native XML 存储方法. 第 20 届全国数据库学术会议, 2003, 10, 计算机科学, 卷 30(10), 105-110.
3. 罗道峰, 孟小峰, 蒋喻. XML 数据扩展前序编码的更新方法. 第 20 届全国数据库学术会议, 2003, 10, 计算机科学, 卷 30(10).
4. 孟小峰, 罗道锋, 蒋喻, 王宇. OrientXA: 一种高校的 XQuery 查询代数, 2004, 软件学报, 卷 15(11), 1648-1660.
5. 张新, 孟小峰, 朱金清, 王伟, 黄静. OrientStore+: 一种支持高效更新的 Native XML 存储方法. 第 24 届中国数据库学术会议, 2007, 10, 计算机研究与发展, 卷 44(增刊), 368-373.
6. X. Meng, D. Luo and et al. OrientStore: A Schema Based Native XML Storage System. (Demo). In Proceedings of VLDB, 2003.
7. J. Wang, X. Meng, and S. Wang. Supex: A schema-guided path index for xml data. (Doctoral poster). In Proceedings of VLDB, 2002.
8. J. Yu, D. Luo, X. Meng, and H. Lu. Dynamically updating xml data: Numbering scheme revisited. In World Wide Web, 2005.
9. <http://www.w3.org/TR/xquery>