Web 数据库技术简述*)

Web Database Techniques: A Survey

张志强　周立柱　冯建华

(清华大学计算机科学与技术系　北京 100084)

Abstract The popularity of the WWW has made people disseminate information very easy. The relevance of database concepts to the problems of managing and querying Web information has led to a significant body of recent research addressing these problems. In this paper, we discuss some Web database techniques in brief, and give some advises about future working

Keywords Web, XML, Database, Searching engine

1 引言

网络技术发展到今天，已渗透到社会生活的每一个角落，获得了巨大的成功。然而，电子商务、电子图书、远程教育等全新领域异军突起，随之而来的是 Web 文件的复杂化、多样化、智能化，而且要求同样的数据能根据不同用户的不同需求而以不同的效果、形式表达出来。但是在 Web 世界中，数据的表现形式是十分不规则的、多样的，很难利用传统的数据库技术来存储，管理所有的 Web 数据。而且将来的 Web 应该能够管理动态的内容，而不是静态的 HTML 网页，对于这种个性化的 Web 应用需要复杂的数据库服务。为此如何利用成熟的、已经发展得相当完善的数据库技术来对 Web 数据进行存储管理和有效的检索，已经成为当今网络和数据库领域共同关心的问题，甚至认为这是下一个世纪的重要的课题。

XML 技术的出现为解决上面的问题提供了可能性。XML 是互联网联合组织(W3C)创建的一组规范，以便于软件开发人员和内容创作者在网页上组织信息，其目的不仅在于满足不断增长的网络应用需求，同时还希望借此能够确保在通过网络进行交互合作时，具有良好的可靠性与互操作性。现在 XML 正在成为 INTERNET 上数据描述和交换的标准，并且在将来 XML 将代替 HTML 而成为 Web 上驻留数据的主要格式。

当前 Web 上的工作涉及的方面很多，本文只对查询语言、XML 数据的存储、XML 数据的查询处理和优化做一个简要的综述，有兴趣的读者还可以参见文[15]。

半结构化查询语言和数据模型已经获得了广泛的研究，如文[1,3,6,21]等，XML 数据也是半结构化数据，相应的关于半结构化数据的结果可以应用于 XML 数据。为了对 XML 数据进行有效的查询，相继提出了一些有影响的查询语言，如，LOREL[3,18]、XML-QL[12,13]、XML-GL[7]、XSL[8,31] 和 XQL[26,27] 等，在文[5]中对这几种语言做了一个详细的比较。

最近，出现了一些考察不同存储策略对查询处理的影响的项目，斯坦福大学[21]开发了一个专门存储和管理半结构化数据和 XML 数据的数据库系统 Lore 系统，它采用的查询处理步骤与传统数据库系统相类似，并采用了以开销模型为基础的优化方法[24]。另一种方法是将 XML 数据存储于关系数据库系统或面向对象数据库系统中[28,30,14]。在文[28]中描述了如何利用 DTD 将 XML 数据映射到一个关系数据库中，但是正如我们后面将要看到的，这种方法存在一些问题。文[14,30]考察了多种 XML 数据到关系数据库的映射策略，探索不同的 XML 数据的存储策略。

本文的组织如下：第 2 部分简要介绍两种查询语言；第 3 部分对 XML 数据的查询处理和优化进行了简单的描述；第 4 部分介绍了 XML 数据的存储；第 5 部分中讨论了今后可能的研究方向；最后在第 6 部分做了一个简要的总结。

2 查询 XML 数据

XML 现在正在成为 INTERNET 上进行数据交换的标准，因此 Web 上的 XML 数据将越来越多。现在的 Web 工具，如浏览器、搜索引擎等，都是采用面向 HTML 格式数据的操作，而对于 XML 数据我们需要

*)本文得到国家"863"项目(98031J015)和"973"重点基础研究发展项目(G1998030414)的支持。

一些数据库的操作，如数据提取、数据集成、数据转换、数据存储等。因此，XML 数据的查询语言应该能够表达基于内容的查询，允许应用从一个或多个 XML 数据源中准确提取所需要的信息。XML 数据与传统的关系数据或面向对象数据的根本区别在于，传统的数据都遵循一定的数据模式，而 XML 数据是自描述的，没有固定形式、由于 XML 数据的半结构化特点，SQL 和 OQL 都已经不再适用，因此需要建立新的查询语言。由于篇幅的限制，下面我们只简要地介绍一下 LOREL 和 XML-QL 这两种主要查询语言。前者可以对 XML 数据的结构进行存取，后者除了这一点之外，还可以对查询的结果进行构造。今后 XML 查询语言应具有更多的特征[20]。下面的查询例子都是以文[29]中的参考文献数据库为查询对象，如图 1 所示。

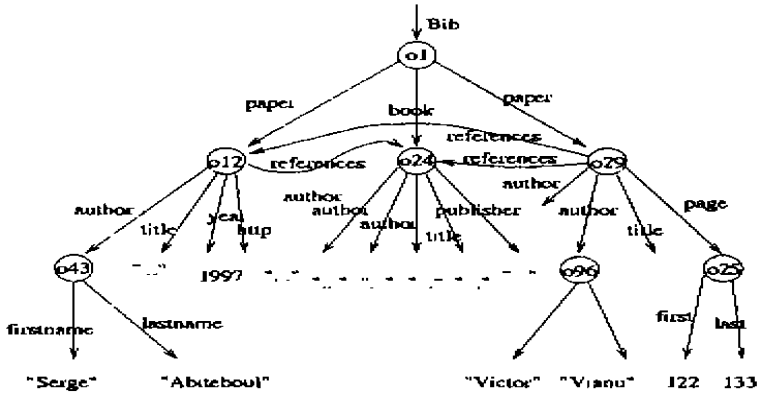

图 1　参考文献数据库

2.1　LOREL 查询语言

LOREL 是 Lore 系统的查询语言，它是通过对 OQL 扩展得到的，例如下面的查询返回 1995 年以后写的文章标题：

```
Select X.title
From Bib.paper X
Where X.year>1995
```

LOREL 具有以下几个特点：

(1)它具有一个强制类型转换功能，例如对于上面的查询，无论文章的年月是字符串形式，还是整数形式，都会将相应文章的标题返回。

(2)对于丢失的信息采用忽略的处理方式，由于实际的 XML 数据中经常出现数据丢失现象，如在有的文章中可能没有关于这篇文章的写作日期，LOREL 就简单地跳过这篇文章，接着查看下一篇文章，这样具有一定的容错能力，比较灵活。

(3)属性变量可以是单值，也可以是多值的，例如上面的查询中文章 X 的日期属性可能有多个值，如果这些值中有满足查询条件的值，则 X 就被选中。

(4)允许对不确定结构的数据进行查询，例如，给出所有那些被"Ullman"直接或间接引用的文章标题：

```
Select X.title
From Bib.paper X,Bib.paper Y
Where Y author.(lastname)? ="Ullman"
        Y.(reference)+=X
```

其中()?表示一个可选路径表达式，并且()+表示 Kleene 闭包。另外 From 语句并不是必要的，有时可以省略。

2.2　XML-QL 查询语言

XML-QL 的目标就是允许对 XML 数据进行查询、转换和集成。XML-QL 具有很多其它半结构化数据查询语言的特点，并且将它们扩展到一个有序的数据模型上。XML-QL 在 Where 语句中使用模式提取数据，例如，

```
Where<book>
    <publisher><name>Addison-Wesley</></>
    <title>$t</>
    <author>$a</>
    </>in www.a.b.c/bib.xml
Construct<result>
   <author>$a</>
   <title>$t</>
   </>
```

这个查询要求取回所有由 Addison-Wesley 公司出版的书的<author,title>对，这里</>是结束标签的简写(<publisher>或</name>等)，变量前加有符号 $。模式可以扩展到其他的 XML 组成部分，如 XML 的属性。

另外，XML-QL 可以用来对数据进行转换和集成，下面的例子表明通过参考文献数据源构造一个关于作者的数据库。这里采用了 Skolem 函数 PersonNodes()。

```
Where<paper><author.lastname?>$a</>
     <title>$t</>
     </Paper>in www.a.b.c/bib.xml
Construct<person id=PersonNodes($a)>
```

```
(name) $ a(/)
(papertitle) S t(/)
(/person)
```

3 XML 数据的查询处理和优化

目前关于 XML 数据的查询处理和优化的工作还都是针对专门的 XML 存储系统，对于底层采用数据库管理系统的情况，则是通过将以 XML 查询语言书写的查询转换为关系 SQL 语言或 OQL 语言查询，利用传统数据库系统的查询处理功能来完成用户的查询请求。

3.1 Lore 系统的查询处理过程

Lore 系统是存储 XML 数据的一个专门数据库系统，其查询处理和优化方式基本代表了今后其它 XML 数据专门系统的优化方式。因此，我们先来看一下 Lore 系统的查询处理步骤，这个过程与传统数据库中的查询处理过程很类似，大致有以下几步：

(a)对 Query 进行语法分析，并将其转换为传统的 OQL 的形式；

(b)逻辑查询计划生成器产生一个逻辑查询计划；

(c)这个逻辑查询计划可以产生很多的物理查询执行计划，这些物理查询计划由一些物理操作符构成，这些操作符可由查询执行引擎执行。查询优化器通过存储的统计信息和开销模型，从众多的物理查询执行计划中选出一个最优的；

(d)提交给查询计划执行引擎完成查询，并返回查询结果。

传统的对数据库的查询，是通过申明性查询语言，如 SQL 书写的一个表达式来描述的，当前对 SSD 和 XML 数据的查询处理不仅仅局限于此，还应与 IR(Information Retrieval)中的一些概念结合起来，如关键字搜索(Keywords)、相临搜索(Proximity)和相似搜索(Similarity)等。在 Lore 系统中提供了部分的关键字搜索和相临搜索等功能。

3.2 查询优化

目前关于查询优化的工作并不是很多，在实际当中应用得就更少了，总体上这些工作主要是从以下几个角度出发的，一种是以 Lore 系统为代表的按照传统数据库系统的优化模式进行，一种是通过关于数据的一些信息进行的语义优化，另一种主要是从理论角度出发的规则路径表达式的优化。下面我们简单地介绍一下，详细的细节参见相应的参考文献。

3.2.1 *Lore 系统的优化* 当前 Lore 系统的优化策略，实现了一些简单的启发式优化技术，如将选择操作放到查询树的下面等，Lore 系统的优化的主要特点是充分利用索引和采用开销模型。Lore 系统提供了多种索引技术以利于查询的有效执行，Lore 系统主要有如下5种索引，由于篇幅的原因我们不在这里详细介绍这几种索引，详见文[18,25]。

(1)Vindex：找出满足给定边标记(lable)和谓词的所有原子对象。

(2)Lindex：通过一个边的标记，找出所有给定对象的父对象。

(3)Bindex：通过给定的边标记，找出所有的通过这个标记连接的父-子对象对。

(4)Tindex：是使用倒排表来实现的，将给定的词 w 和标记 L 映射到一个输入标记为 L 的原子值表，且这些原子值中含有词 w，主要目的是为了支持各种 IR 型查询。

(5)Pindex：主要指 DataGuide，通过给定的一个路径 p，可以返回所有通过 p 能够达到对象的集合。DataGuide 是对所有在数据库中的路径，在某一个时间点的动态总结。

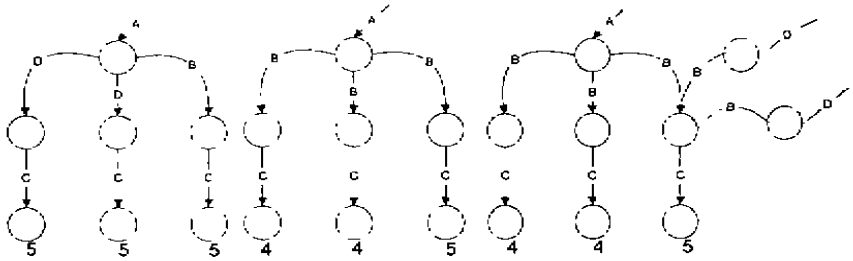

图2 三种不同情况的数据库及相应的优化策略

下面我们采用文[24]中的例子来简要地说明一下优化的动机和相应索引的应用，如图2所示，查询为“Select x From A.B x Where exists y in x.C:y=5”。对于这个查询在不同的数据库情况下，所采用的执行策略是不同的，例如在第一种情况下采用自顶向下的策略就是比较好的执行策略。而在第二种情况下、采用

自底向上的策略比较好，这里需要用到 Lindex 索引。对于第三种情况，显然采用自顶向下和自底向上的结合策略比较好，首先从上至下地执行，但不是执行到底，而是执行到中间并生成一个临时结果，然后通过 Vindex 找到满足条件的叶子结点、利用 Lindex 从底向上执行，直到前面到达的那些中间结点，最后将这两部分的临时结果进行连接得到最后的结果。

3.2.2 *以模式为基础的优化* 以模式为基础的查询优化(也称为语义优化)[25]，其基本思想就是通过已知的关于数据的一些信息来对查询进行优化，例如在图1所示的参考文献数据库中，对于查询条件为"Bib. * . last=130"的查询，如果没有关于数据模式的信息，查询处理器就得扫描整个的数据库来完成查询；如果我们知道数据相应的 DTD，则通过 DTD 我们知道只有 Bib. (paper|book) page 路径才能达到 last 属性，这样只需从根结点 Bib 开始扫描路径 Bib. (paper|book). page 就可以完成查询，这样就大大减少了需要检查的数据量，提高了对查询的响应速度。

3.2.3 *路径表达式的优化* 从优化路径表达式入手(理论的角度)，在文[4]中考虑了路径表达式的等价式，例如，考虑一个关于装配部件的数据库，已知有"part. (subpart)*. description" = "catalog. partdescription"成立，即可以通过 catalog 结点到达 description 结点，这样路径表达式为"part. (subpart)*. description. section. paragraph"的查询，就可以用路径表达式为"catalog. partdescription. section. paragraph"的查询来代替或改写。这使得查询处理器可以直接通过替换后的路径快速找到相关的数据，从而实现了对查询的优化。在文[21]中考虑了更复杂的查询，但是这方面的结果目前还只是停留在理论一级上，还没有被应用于实际当中。

4 XML 数据的存储

当前 XML 数据的存储方式主要有三种：存储于文件中、存储于数据库系统中、建立专门的存储系统。下面我们对这三种方式简单地描述一下，并总结了它们各自的优缺点。

4.1 文件存储(XML 文档)

这种方法就是将一个 XML 数据文档直接作为一个文件存储。这是当前普遍采用的方法，如下面图3所示的 XML 文档，它将作为操作系统的一个文件来存放。这种方法的优缺点如下：

优点：实现很简单；数据所占用的空间小；XML 数据以自然合理的方式聚集在一起。缺点：由于修改操作涉及到对文件的修改，所以当前很少能够支持修改操作；要求采用特殊的查询处理器来完成在 XML 数据文档中的查询；每次存取或浏览时都需要进行分析，且分析后的文件比原 XML 文档大得多，在处理的过程中需要驻留内存；建立索引很困难。

```
<book>
  <booktitle>Database System</booktitle>
  <author id="Holzner">
        <name>
            <firstname> Alberto </firstname>
            <lastname> Honlzner </lastname>
        </name>
        <address>
            <city> New York </city>
            <zip> 11111 </zip>
        </address>
  </author>
</book>
```

图3

```
<!ELEMENT  book(booktitle, author)>
<!ELEMENT  author(name,address)>
<!ATTLIST    author id ID #REQUIRED>
<!ELEMENT  name(firstname?,lastname)>
<!ELEMENT  firstname(#PCDATA)>
<!ELEMENT  lastname(#PCDATA)>
<!ELEMENT  address ANY>
```

图4

4.2 存储于数据库系统中

XML 数据存储在数据库系统中是最容易被人们想到的方法，目前最为常用的是采用关系数据库系统和面向对象数据库系统。

4.2.1 *采用关系数据库系统* 这种方法将 XML 数据存储于当前商品化的关系数据库系统中，利用已经成熟的技术来处理 XML 数据。但是需要将 XML 数据转换成关系数据。在文[28]中分析了采用传统的关系数据库引擎处理 XML 文档方法的优缺点，指出在 XML 数据上大部分半结构化的查询语义都能采用关系的方法处理，但是只在一些情况下是有效的。对 XML 文档的处理过程大致如下：首先，根据 DTD 生成一个关系模式；其次，将与该 DTD 一致的文档转化为相应的关系元组，并装入 RDBMS；然后将在 XML 文档上的半结构化查询转换为在关系数据上的 SQL 查询；最后将查询结果转化为 XML 数据。

```
book (bookID:integer,booktitle:string,authorID:integer)
author (firstname:string,lastname:string,authorID:integer)
address (city string,zip:string,addressID:integer)
```

图5

图4是图3中 XML 文档对应的 DTD，通过这个 DTD，可以将其转换成一个关系模式，例如一个可能的关系模式如图5所示。这种方法的优缺点如下：

优点：充分利用了现有的传统关系数据库的资源

和技术；当前的大数据库厂商已经或正在提供这种功能，如 Oracle，IBM DB2等；实现上工作量相对比较小。缺点：存储前后的转换工作很繁杂；为了清除半结构化数据与二维数据之间的差别，其转换工作使原有的半结构数据部分信息丢失；转换后，在很多情况下很简单的 XML 查询需要很多的 SQL 语句查询，即还很少的 SQL 语句查询但每个语句中含有大量的连接操作。

4.2.2 *采用面向对象数据库系统* 与关系数据库系统的情况相似，也需要进行 XML 数据到对象模式的转换。这里也可以通过 DTD 来导出对象模式，如对于上面的例子，一个可能的对象模式 ODMG 如下：

```
Class BOOK public type tuple
    (booktitle:string,author:List(AUTHOR))
Class AUTHOR public type tuple
    (firstname:string,lastname:string,address:AD-
    DRESS)
Class ADDRESS public type tuple
    (city:string,zip:string)
```

虽然面向对象数据库的复合类型和继承特性有助于解决 XML 数据的不规则性，但是为半结构化数据设计一个面向对象的模式是很困难的，性能也是一个关键的问题。XML 文档中的元素常常是很小的，将每一个元素都做一个对象来存储会使得系统存储空间的开销很大。

4.3 建立专门的存储系统

在为半结构化数据和 XML 数据建立的专门存储系统中，斯坦福大学研制的 Lore 系统是较为著名的一个，在这里我们简单地介绍一下 Lore 系统[21]。

Lore 系统是一个专门为存储半结构化数据和 XML 数据而设计的数据库系统，可以通过多种应用或者直接通过 Lore 的应用程序接口来存取 Lore 系统。系统大致分为两层：查询编译层和数据引擎层，

查询编译层由分析器、预处理器，查询计划生成器和查询优化器组成，分析器接受一个查询的文本描述，并将它转换成一个语法分析树，最后将这个分析树提交给预处理器。预处理器将 Lore 的查询转换成一个类-OOL 的查询。查询计划生成器将这个查询生成一个查询计划，并将这个计划提交给查询优化器。优化器需要对执行计划进行一些变换和确定一些索引的使用，最后将优化后的执行计划提交给数据引擎层处理。数据引擎层由 OEM 对象管理器，查询操作执行部分、外部数据管理器和其它的工具组成。查询操作执行部分主要完成上一层提交的查询执行计划。对象管理器用来完成 OEM 与底层文件结构的映射。它支持基本的元语，如取一个对象，比较两个对象等。

Lore 的存储管理策略很简单，首先将 XML 数据分解为一些基本的要素(element)、属性和文本字符串，采用直接深度优先聚集方法进行物理存储。Lore 将对象组织在物理磁盘页上，每个页上有很多的槽(slot)，每个槽中有一个单独的对象。Lore 支持跨越多个页的大对象，对多媒体类型是有用的。采用深度优先的方法将对象聚集在页上，这主要是由于系统采用深度优先的数据库扫描策略。当一个对象有多个父对象时，它与任一个父对象聚集在一起。

5 进一步的讨论

能够将 Web 作为一个巨大的数据库来查询是一个很有意思的探索目标，要实现这个目标我们必须解决很多的问题，例如，用户如何表达查询和搜索请求？系统或搜索引擎如何正确理解用户的查询请求并进行处理？系统如何高效存储和管理 XML 数据？等，因此对 Web 数据的查询/搜索和存储管理是关键。

在搜索方面最常用的方法就是采用搜索引擎，目前搜索引擎的工作主要集中在两个方面：一是搜索引擎对用户搜索请求的理解上，如自然语言理解；另一个是快速准确地找到相关信息。现在很多的搜索引擎都主要在这方面展开工作，如 Google 搜索引擎公司在分析超链方面的工作。但正如我们大家看到的，由于现在搜索引擎提供的搜索方式有限，基本都是关键字搜索，还不能支持更复杂一些的搜索请求，很难准确表达用户的搜索请求，用户不得不花费很大的力气对搜索引擎返回的结果进行再筛选。如何解决这个问题呢？XML 的出现为解决这个问题带来了可能性，因为 XML 的一个优点在于它允许对数据进行更精细的查询。在 XML 将最终取代 HTML 而成为 Web 上的主要数据驻留格式的趋势下，搜索引擎应该具有搜索 XML 数据的能力，而不是现在的只能搜索 HTML 数据。

目前的搜索引擎还都主要对那些静态网页进行搜索，对于动态生成的网页无能为力或能力极弱，这主要是因为现在动态网页一般都是利用数据库系统来作为底层基本数据的存储。当前的搜索引擎不能对这种数据库系统进行查询，这主要是因为现在的数据库系统处理不了这样的查询请求。所以对于 XML 数据库系统除了能实现传统数据库查询技术与 IR 查询技术的结合之外，还应该具有处理不固定模式查询的能力。同时，为了支持对用户请求的快速响应，要求系统必须采用高效的存储方法和处理策略。前面介绍的几种存储方案值得进一步的研究，研究的重点在于对各种存储策略在不同环境下性能的比较和分析，在专门的 XML 数据存储系统中特殊索引的设计和建立也是改善系统性能的重要手段，这里需要考虑 XML 数据中各要素的顺序，以及新的比较操作等因素。

结论 本文对查询语言、查询处理和优化以及存

储等方面进行了简单的分析和总结，并对今后的研究方向进行了讨论。目前 Web 上的工作除了上面提到的几个方面外，还有很多如 Web 数据的缓存[11,32]，XML 数据的索引技术[22,23]，XML 的视图技术[2,3,10,19]等，这里由于篇幅的原因不再详细介绍，有兴趣的读者可以查看相应的参考文献。

参 考 文 献

1 Abiteboul S. Query semi-structured data. In: Proc of the Intl Conf. on Database Theory (ICDT), Dephi, Greece, 1997

2 Abiteboul S, et al. Incremental Maintenance for Materialized Views over Semistructured Data. In: Proc. of VLDB' 98, August 1998

3 Abiteboul S, et al. The Lorel query language for semistructured data. International Journal on Digital Libraries, 1997, 1(1): 68～88

4 Abiteboul S, Vianu V. Regular path queries with constrains. In: Proc. of the ACM SIGACT-SIGMOD-SIGART Symposium on Priciples of Databse Systems (PODS), Tucson, Arizona, May 1997

5 Bonifati A, Ceri S. Comparative Analysis of Five XML Query Languages. ACM SIGMOD Record, 2000, 29(1): 68～79

6 Buneman P. Same to [4]: 117～121

7 Ceri S, et al. XML-GL: a :Graphical Language for Querying and Restructuring WWW Data. In: Proc. of 8th Int. World Wide Web Conference, WWW8, Toronto, Canada, May 1999. www8.org/1ullpaper.html

8 Clark J. Xsl transformations (xslt) specification. 1999. http://www.w3.org/TR/WD-xslt

9 Calvanese D, et al. Rewriting regular expressions and regular path queries. In: Proc. of the ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems (PODS), 1999

10 Calvanese D, et al. View-based query processing for regular path queries with inverse. In: Proc of the ACM SIGACT--SIGMOD-SIGART Symposium on Priciples of Database Systems (PODS), 2000

11 Challenger J, Iyengar A, Dantzig P. A Scalable System for Consistently caching Dynamic Web Data. In: Proc. of INFOCOM'99, March 1999

12 Deutsch A, et al. XML-QL: A Query Language for XML. In: Proc. of the Query Languages workshop (QL98), Cambridge, Mass., December 1998. http: // www.w3.org/TR/1998/NOTE-xml-ql-19980819/

13 Deutsch A, et al. A Query Language for XML. In: Proc. of 8th Int World Wide Web Conference, WWW8, Toronto, Canada, May 1999. www8.org/fullpaper.html

14 Florescu D, Kossmann D. Storing and quering XML data using an RDBMS. IEEE Data Engineering Bulletin, 1999, 22(3): 27～34

15 Florescu D, Levy A, Mendelzon A. Database Techniques for the World-Wide Web: A Survery. SIGMOD Record, 1998, 27(3)

16 Fernandez M, Suciu D. Optimizing regular path expressions using graph schemas. In: Proc. of the Intl. Conf. on data engineering. 1998. 14～23

17 Goldman R, McHugh J, Widom J. From Semi-structured Data to XML: Migrating the Lore Data Model and Query Language. In: Proc. of the 22nd Intl. Workshop on the Web and Database (WebDB' 99), Philadelphia, Pennsylvania, June 1999

18 Goldman R, Widom J. DataGuides: Enabling query formulation and optimization in semistructured databases. In: Proc. of the 23th Intl. conf. on VLDB, Athens, Greece, August 1997. 436～445

19 Labrinidis A, Roussopoulos N. Web View Materialization. ACM SIGMOD, 2000(5): 367～378

20 Maier D. Database desiderata for an xml query language. http://www.w3.org/TandS/QL/QL98/pp/maier.html

21 McHugh J, et al. Lore: A database management system for semistructured data. SIGMOD Record, 1997, 26(3): 54～66

22 McHugh J, et al. Indexing semistructured data. [Technical report]. Stanford university database group, 1998. Available at: ftp: // db.stanford.edu/pub/papers/semindexing98.ps

23 Milo T, Suciu D. Index structures for Path Expressions. 1999

24 McHugh J, Widom J. Query Optimization for XML. In: Proc. of the 25th VLDB Conf. Edinburgh, Scotland, 1999. 315～326

25 McHugh J, Widom J. Query Optimization for semi-structured data: [Technical report]. Stanford University Database Group, February 1999. Available at: ftp: // db.stanford.edu/pub/papers/qo.ps

26 Robie J, Lapp J, Schach D. XML Query Language (XQL). In: Proc. of the Query Language workshop, Cambridge, Mass., Dec. 1998. http: // www.w3.org/TandS/QL/QL98/pp/xql.html

27 Robie J. The design of xql. 1999. http: // www.texcel.no/whitepapers/xql-design.html

28 Shanmugaundaram J, et al. Relational Databases for Querying XML Documents: Limitations and Opportunities. In Proc. of the 25th VLDB conference, Edinburgh, Scotland, 1999

29 Suciu D. Semi-structured Data and XML: [Technical report]. AT&T Labs, 1999

30 Tian F, et al. The Design and Performance Evaluation of Alternative XML Storage Strategies. Wisconsin University, 2000

31 W3C XSL Working Group. The Query Language Position Paper of the XSL Working Group. In: Proc. of the Query Language Workshop, Cambridge, Mass., Dec. 1998. http: // www.w3.org/TandS/QL/QL198/pp/xsl-wg-position.html

32 Proceddings of 3d WWW caching Workshop (Manchester, England). June 1998

33 Widom J. Data Management for XML: Research Directions. Bulletin of the IEEE Computer Society Technical Committee on Data Engineering, 1999