文章编号: 1007- 2985(2007)04- 0046- 04

XML 文档规范化和反规范化*

黄海燕
(中南大学信息科学与工程学院, 湖南 长沙 410083)

摘 要: 从数据库角度分析了 XML 文档中存在的间接函数依赖、传递函数依赖、多值依赖, 阐述了间接函数依赖和传递函数依赖规范化所对应的第 1 范式、第 2 范式及多值依赖规范化所对应的第 3 范式, 提出了 4 个文档规范化规则, 这些规则分别消除了间接函数依赖、传递函数依赖、多值依赖, 同时从文档规范化的性能因素方面探讨了反规范化的有用性.

关键词: XML; 数据库; 函数依赖; 规范化; 反规范化
中图分类号: TP301.2 文献标识码: A

XML(eXtensible Markup Language) 已经成为 Internet 上主要的数据表示和交换标准之一. 作为一种半结构化数据表示模型, 正如关系数据库有自己的平面关系模式一样, XML 文档也有自己的模式, 例如 XML-Data、XML-Schema 和 XML 文档类型定义 DTD(Document Type Definition) 等. 而在实际应用中, DTD 是 XML 文档使用最多也是应用最成熟的一种模式, 而 XML-Schema 是 DTD 的扩展和延伸. 但是由于 XML 的模式 DTD 存在设计上的缺陷, DTD 中可能存在一些异常的数据依赖, 从而导致数据冗余和更新、删除和插入异常. 因此, 对 XML 文档中普遍存在的函数依赖[1] 问题和 XML 文档进行规范化[2] 设计进行研究非常必要. 在对 XML 数据进行有效存储和查询的研究中, 由于现有关系数据库取得了巨大的成功, 用关系模式进行有很大的优势[4] . 但是, 由于规范化产生了许多表, 而每个表有相对较少的列, 并且这些列必须使用主码/外码关系连接, 这就降低了数据库的性能. 所以, 根据业务的需要, 可在规范化的基础上再进行反规范化[3], 即修改表结构以允许存在一些冗余数据从而提高数据库的性能.

现在对 XML 文档的规范化方面的研究进行得很多. 在文献[1- 2, 4- 5] 中都研究或涉及了函数依赖, 在文献[6] 中研究了 XML 文档的规范化算法, 在文献[7] 中研究了多值依赖的规范化, 在文献[7] 中研究了从 DTD 映射到关系模式(一种保持数据依赖的映射方法), 已经注意到了文档碎片, 在这里笔者引入了文献[3] 中关系数据库的反规范化概念.

1 XML 函数依赖

下面是一个描述课程(course) 和学生(student) 之间关系的 DTD D1:[8]

```
<! ELEMENT courses (course* )>
<! ELEMENT course (title, takenby)>
    <! ATTLIST course cno CDATA # REQUIRED>
<! ELEMENT title (# PCDATA)>
<! ELEMENT takenby (student* )>
<! ELEMENT student (sname, teacher)>
    <! ATTLIST student sno CDATA # REQUIRED>
<! ELEMENT sname (# PCDATA)>
<! ELEMENT teacher (tname)>
    <! ATTLIST teacher tno CDATA # REQUIRED>
<! ELEMENT tname (# PCDATA)>
```

* 收稿日期: 2007- 05- 11
基金项目: 国家自然科学基金资助项目(60173041)
作者简介: 黄海燕(1979-), 女, 湖南永州人, 中南大学信息科学与工程学院研究生院硕士研究生, 主要从事 XML、函数依赖、数据库研究.

D1 对应的XML 文档树如图 1 所示.

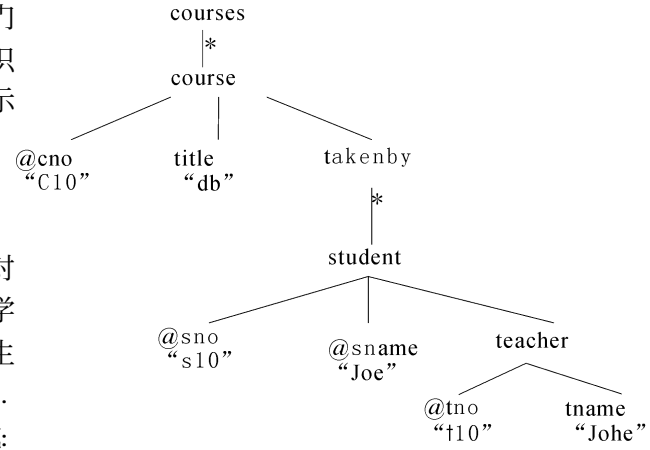

图 1　符合 $D1$ 的 XML 文档

假定要表示这样一个函数依赖: 对于一门确定的课程(course) , 学生的学号(sno) 能唯一标识该XML 文档中的一个学生节点(student) . 则可表示为

(courses. course, [courses. course. takenby. student. @ sno] → [courses. course. takenby. student]) .

上述函数依赖不仅有它成立的前提条件, 即对于一门确定的课程; 而且还涉及到节点的比较, 即学生的学号能唯一地标识该 XML 文档中的一个学生节点(student) . XML 函数依赖的定义参考文献[5] . 给定 DTD D, 在 D 上的函数依赖 φ具有如下形式: $(S_h, [S_{x1}, \cdots, S_{xn}] \rightarrow [S_{y1}, \cdots, S_{ym}])$. 其中: $S_h \in$ paths(D) 叫做函数依赖 φ的头部路径, 定义了 φ在 D 上成立的范围; $[S_{x1}, \cdots, S_{xn}]$ 叫作 φ的左部路径; $[S_{y1}, \cdots, S_{ym}]$ 叫作 φ的右部路径.

DTD $D1$ 中有

$F1$: [courses. course. @ cno] → [courses. course] ,

$F2$: (courses. course, [courses. course. takenby. student. @ sno] → [courses. course. takenby. student]) ,

其中: 函数依赖 $F1$ 表示在整个图 1 所示的XML 文档中, 课程编号(cno) 能唯一地决定一门课程(course 节点) ; $F2$ 表示对于确定的一门课程, 学生的学号(sno) 能唯一地决定一名学生(student 节点) .

间接函数依赖(IFD) 的定义参考文献[8] . 其直观解释是: 当把语义上非直接联系的元素类型组织成树状结构时, 就会有间接函数依赖, 这样, 在 XML 文档中就会有数据冗余.

在DTD $D1$ 中, 存在 IFD:

(courses. course, [courses. course. takenby student. @ no] → courses. course. takenby. student. teacher. @ tno)

传递函数依赖[8] 的定义如下: 对于具有形如 $(S_h, [S_{x1}. б_{x1}, \cdots, S_{xn}. б_{xn}] \rightarrow S_y. б_y,)$ 的 FD $\varphi \in (D, \Sigma)^+$, 称 φ为传递函数依赖(TFD) . 如果存在 FD $\varphi 1 \in (D, \Sigma)^+ : (S_h, [S_{x1}. б_{x1}, \cdots, S_{xn}. б_{xn}] \rightarrow [S_{y1}. б_{y1}, \cdots, S_{ym}. б_{ym}])$ 和 $\varphi 2 \in (D, \Sigma)^+ : (S_h, [S_{y1}. б_{y1}, \cdots, S_{ym}. б_{ym}] \rightarrow S_y. б_y)$, 且FD($S_h, [S_{y1}. б_{y1}, \cdots, S_{ym}. б_{ym}] \rightarrow [S_{x1}. б_{x1}, \cdots, S_{xn}. б_{xn}]$) $\notin (D, \Sigma)^+$, 满足条件 $S_y. б_y \notin [S_{y1}. б_{y1}, \cdots, S_{ym}. б_{ym}]$. 其中 S 都是 D 中的路径, $S_h \supseteq$ path S'_h, б 是属性或者具有形式 τ. S(τ 是元素类型) .

在 $D1$ 中存在 FD:

$F3$: courses. course. @ cno → courses. course. takenby. student. teacher. @ tno, $F4$: courses. course. takenby. student. teacher. @ tno → courses. course. Takenby. student. teacher. tname. S, $F5$: courses. course. @ cno → courses. course. takenby. student. teacher. tname. S 且 FD courses. course. Takenby. student. teacher. @ tno → courses. course. @ cno 不成立. 那么根据定义可知, $F5$ 是TFD.

2　XML 规范化

2. 1 XML 的范式

针对非平凡的间接函数依赖和传递函数依赖的规范化, 文献[8] 给出了 XML 第 1 范式的定义: 给定 DTD D 和 $\Sigma \subseteq$ FD(D) , 如果(D, Σ) 是规范化的, 并且在(D, Σ)$^+$ 中不存在非平凡的间接函数依赖和传递函数依赖, 那么(D, Σ) 是第 1 种 XML 范式(记为 XNF-1) , XNF-1 是关系模式 3NF 的泛化.

XML 第 1 范式的基础上更进一步提出了第2 范式: 给定 DTD D 和 $\Sigma \subseteq$ FD(D) , 如果(D, Σ) 是规范化的, 并且对每一个非平凡的FD $\varphi \in (D, \Sigma)^+$, 如果具有下述形式: $(S_h, [S_{x1}, \cdots, S_{xn}] \rightarrow S_y. @a)$ 或者$(S_h, [S_{x1}, \cdots, S_{xn}] \rightarrow S_y. S)$, 都有$(S_h, [S_{x1}, \cdots, S_{xn}] \rightarrow S_y) \notin (D, \Sigma)^+$, 那么(D, Σ) 是第 2 种 XML 范式(记为XNF-2) .

2. 2 规范化规则

规则 1 是用来消除XML 文档中的间接函数依赖的. 它的基本思想是通过在XML 树中构造一个新的节

点来消除IFD,从而达到消除由IFD所引起的数据冗余的目的.

规则1 给定DTD $D=(E,A,P,R,r)$[8],假定存在IFD $\varphi\in(D,\Sigma)^+:(S_h,[S_1.\sigma_1,\ldots,S_n.\sigma_n]\rightarrow S_y.\sigma_y)$,易知存在FD $\varphi'\in(D,\Sigma)^+:(S_h',[S_h.S_{h1}.\sigma_{h1},\ldots,S_h.S_{hm}.\sigma_{hm}]\rightarrow S_y.\sigma_y)$.其中 $i\in[1,n]$,使得 $S_y\supseteq \text{path}\ S_i\supset \text{path}\ S_h$,$\sigma$ 是属性或者具有形式 $\Gamma.S$(Γ 是元素类型).

下面来消除IFD φ,从而消除由它所引起的数据冗余.通过创建一个新的元素类型节点 Vnew 来构造一个新的DTD $D'=(E',A,P',R',r)$,使得 lab(Vnew) 成为 last(S_h) 的子节点,如图2所示.

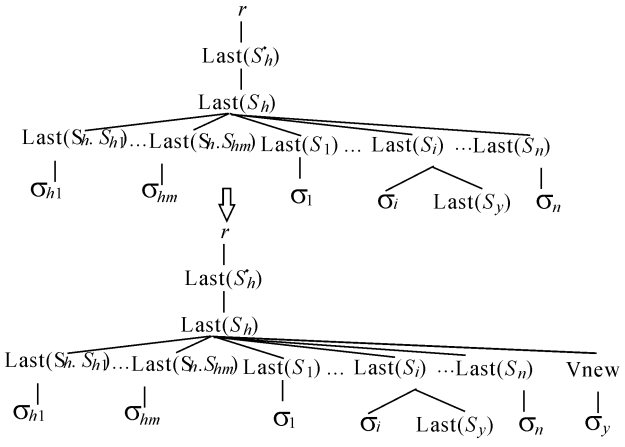

图2 规则1变换示意图

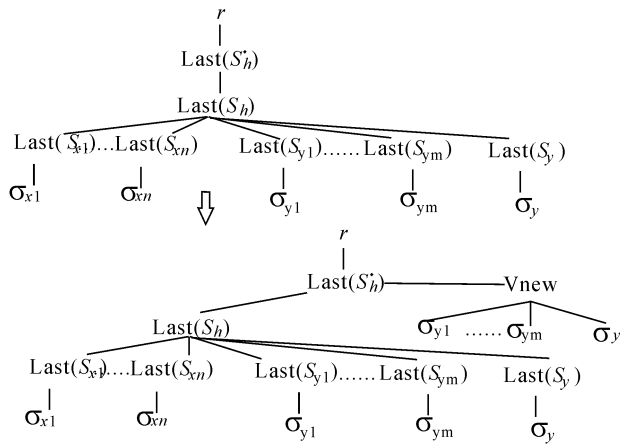

图3 规则2变换示意图

规则2是用来消除XML文档中的传递函数依赖的,它的基本思想是通过构造一个新的元素类型来消除TFD,从而达到消除由TFD所引起的数据冗余的目的.

规则2 给定DTD $D=(E,A,P,R,r)$,假定存在TFD $\varphi\in(D,\Sigma)^+:(S_h,[S_{x1}.\sigma_{x1},\ldots,S_{xn}.\sigma_{xn}]\rightarrow S_y.\sigma_y)$,易知存在FD $\varphi1\in(D,\Sigma)^+:(S_h,[S_{x1}.\sigma_{x1},\ldots,S_{xn}.\sigma_{xn}]\rightarrow[S_{y1}.\sigma_{y1},\ldots,S_{ym}.\sigma_{ym}])$ 和 $\varphi2\in(D,\Sigma)^+:(S_h',[S_{y1}.\sigma_{y1},\ldots,S_{ym}.\sigma_{ym}]\rightarrow[S_y.\sigma_y)$,其中 S 都是 D 中的路径,$S_h\supseteq\text{path}\supset S_h'$,$\sigma$ 是属性或者具有形式 $\Gamma.S$(Γ 是元素类型).

下面来消除TFD φ,从而消除由它所引起的数据冗余.通过创建一个新的元素类型 Vnew 来构造一个新的DTD $D'=(E',A,P',R',r)$,相应的,D' 上成立的FD集合是 Σ',如图3所示.

规则3的基本思想是通过在XML树中构造一个新的节点来消除函数依赖引起的数据冗余.

规则3 给定DTD $D=(E,A,P,R,r)$,假定存在FD $\varphi\in(D,\Sigma)^+:(S_h,[S_{x1}.\sigma_1,\ldots,S_{xn}.\sigma_n]\rightarrow S_y.\sigma_y)$,其中 $\sigma_{x1},\ldots,\sigma_{xn}$ 和 σ_y 是属性或者具有形式 $\Gamma.S$(Γ 是元素类型).通过创建一个新的元素类型节点 Vnew 来构造一个新的DTD $D'=(E',A,P',R',r)$,如图4所示.

规则4的基本思想是通过移动元素的属性来消除由函数依赖所引起的数据冗余,也就是把其它元素类型的属性作为最和它相关的元素类型的一个属性来看待.

规则4 给定DTD $D=(E,A,P,R,r)$,$p.@a$,$q\in\text{paths}(D)$,且 $\text{last}(q)\in E$.DTD $D'=D\{p.@a:=q.@a\}=(E,A,P,R,r)$ 是通过把 $\text{last}(p)$ 的属性 $@a$ 转换成 $\text{last}(q)$ 的一个属性构造得来的.如图5所示.

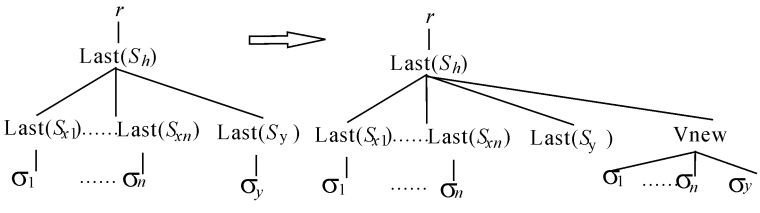

图4 规则3变换示意图

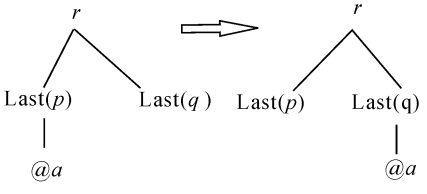

图5 规则4变换示意图

3　XML 反规范化

由于规范化产生了许多表, 而每个表有相对较少的列, 并且这些列必须使用主码/外码关系连接, 这就降低了数据库的性能, 反规范化的唯一原因是要提高性能. 一个反规范化的数据库并不同于一个没有被规范化的数据库, 反规范化是在规范化基础上对数据库进行的考虑. 反规范化是规范化的派生物. 所以, 根据业务的需要, 可在规范化的基础上再进行反规范化, 即修改树(表)结构以允许存在一些冗余数据以提高数据库的性能. 反规范化也和规范化一样, 是有代价的, 即须考虑如何管理冗余数据.

4　结语

在对 XML 文档规范化和反规范化中间有一个中间限度, 这需要对实际数据和有关的特殊业务需求进行彻底的了解.

参考文献:

[1] 吕　腾, 闫　萍, 王真星. XML 的函数数依赖 [J]. 小型微型计算机系统, 2005, 26(6): 864- 868.
[2] 谈子敬, 施伯乐. DTD 的规范化 [J]. 计算机研究与发展, 2004, 41(4): 594- 600.
[3] 何玉洁, 武　欣, 邓一凡, 等. 数据库设计 [M]. 北京: 机械工业出版社, 2001.
[4] 胥正川, 宫学庆, 李　明, 等. XML 文档在关系数据库中的规范化存储 [J]. 小型微型计算机系统, 2008, 24(10): 1 753- 1 758.
[5] 吕　腾, 闫　萍, 王真星. XML 函数依赖及其与键的关系 [J]. 小型微型计算机系统, 2005, 26(9): 1 677- 1 680.
[6] 张忠平, 王　超, 朱扬勇. 基于约束的 XML 文档规范化算法 [J]. 计算机研究与发展, 2005, 42(5): 755- 764.
[7] 何盈捷, 王　珊. 从 DTD 映射到关系模式: 一种保持数据依赖的映射方法 [J]. 计算机研究与发展, 2004, 41(5): 868- 873.
[8] 吕　腾. XML 文档的规范化问题研究 [D]. 上海: 复旦大学研究生院, 2003.

XML Documents of Normalization and Anti-Normalization

HUANG Hai-yan
(School of Information Science and Engineering, Central South University, Changsha 410083, China)

Abstract: On view of database, the article analyzes the indirect functional dependency, transitive functional dependency and MVD in XML, elaborates the XNF-1, XNF-2 and XNF-3, which are corresponded by the normalization of the indirect functional dependency, transitive functional dependency and MVD respectively, and puts forward four principles of file normalization. Taking the property elements into consideration, the article also probes into the feasibility of anti-normalization.

Key words: XML; database; functional dependency; normalization; anti-normalization

(责任编辑　陈炳权)