XML 저장 및 검색 시스템에서의 XML 문서 추출 서브시스템의 설계 및 구현

곽민성[0], 홍석진, 이석호
서울대학교 전기컴퓨터공학부
{cuty[0], jinny}@db.snu.ac.kr, shlee@cse.snu.ac.kr

Design and Implementation of XML Publishing Subsystem in the XML Storage and Retrieval System

Minsung Kwak[0], Seokjin Hong, Sukho Lee
School of Electrical Engineering and Computer Science, Seoul National University

요약

관계형 데이타베이스에 저장되어 있는 데이타를 XML 문서로 추출하기 위해서는 테이블에 저장된 데이타를 XML 문서 형식에 맞는 계층적인 관계를 가지도록 구조화하는 과정이 필요하다. 본 논문에서는 관계형 데이타베이스를 사용하는 XML 저장 및 검색 시스템인 eXDM(embedded XML Database Management System)의 내부에서 구현한 XML 문서 추출 서브시스템에 대하여 기술한다. 이 시스템은 XML 문서에 대한 질의(XQuery)의 RETURN 절과 SORTBY 절을 분석하여, 관계형 데이타베이스에 저장되어 있는 데이타를 구조화하여 계층적인 구조를 가지는 XML 문서 형태로 추출한다.

1. 서론

XML(eXtensible Markup Language)[1]은 그 확장성과 유연성으로 인해 데이타의 표현과 교환의 표준으로 자리잡고 있다. XML은 이미 TV-Anytime[2], MPEG-7, ebXML, 웹 서비스등의 다양한 환경에서 사용되고 있다. XML 문서의 사용 범위가 넓어짐에 따라, XML 데이타를 효율적으로 저장하고 검색하는 기법에 대한 다양한 연구가 진행되고 있다.[3][4]

eXDM[5][6]은 관계형 데이타베이스를 기반으로 하여 XML 형태의 데이타를 저장하고 검색하는 시스템으로, [7]에서 제시한 방법과 유사한 번호 부여 기법을 사용하여 XML 문서를 테이블에 저장한다. 질의 언어는 2002년 4월에 수정된 XQuery 1.0[8]의 핵심 부분 집합을 사용하며, XQuery형태의 질의를 내부적으로 SQL 문으로 변환하여 처리하고 결과를 XML 형태로 반환한다. 본 논문에서는 eXDM을 구성하는 여러 서브시스템 중 XML 문서를 추출하는 서브시스템의 구조와 기능 및 구현에 대하여 기술한다.

본 논문의 구성은 다음과 같다. 2장에서 eXDM 내부의 XML 문서 추출 서브시스템의 개요를 살펴보고, 3장에서는 내부 구현에 대해 설명한 후, 4장에서 결론을 맺는다.

2. XML 문서 추출 서브시스템의 개요

2.1. eXDM의 전체 구조

eXDM의 전체 구조를 살펴보면 그림 1과 같다.

XML 데이타 로더는 XML 문서를 관계형 데이타베이스에 저장하고, XML 타입 처리기는 미리 정의된 데이타 타입 정보를 바탕으로 XML 데이타 로더나 XQuery 처리기에 각 노드의 데이타 타입을 알려준다. XQuery 처리기는 XQuery를 두 단계로 나누어서 처리한다. XQuery 변환기는 FOR 절과 WHERE 절을 기반으로 SQL 문을 구성한다. 해당 SQL 문은 하위 관계형 데이타베이스에 의해 수행되어 릴레이션 형태의 중간 결과를 반환한다. XML 문서 추출 서브 시스템은 반환된 중간 결과, XQuery의 RETURN 절, SORTBY 절을 분석하여 최종 결과를 추출한다.

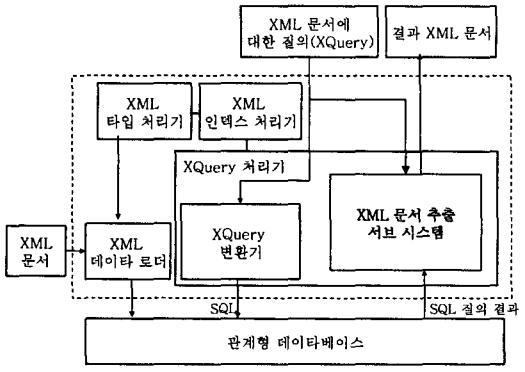

그림 1. eXDM 전체 구조

2.2. XML 문서 추출 서브시스템

XML 문서 추출 서브시스템은 SQL 문을 수행한 중간 결과, XQuery의 RETURN 절, SORTBY 절을 입력 데이타로 받아서 해당 XQuery의 결과를 출력한다.

XQuery 변환기가 생성한 SQL문은 하위 관계형 데이터베이스에 의해 수행되며 테이블 형태의 중간 결과를 반환한다.

중간 결과 테이블의 열은 XQuery의 FOR 절에서 지정한 변수들로 구성되며, 각 레코드는 XQuery의 WHERE 절을 만족하는 XML 부분 문자열로 구성된다. XML 문서 추출 서브시스템에서는 중간 결과 테이블의 레코드 하나씩을 읽어서 RETURN_XPath 리스트를 구성한 후 이 리스트로부터 하나 또는 하나 이상의 XML 레코드를 생성한다. 이 XML 레코드는 결과 테이블 레코드가 하나씩 처리될 때마다 XML 레코드 리스트로 모아지고, 정렬 및 태그 추가 과정을 거쳐 결과 XML 문서로 출력된다.

3. XML 문서 추출 서브시스템의 구현

3.1. 자료 구조

RETURN_XPath 리스트는 RETURN 절에서 지정한 XPath의 값을 저장하며, RETURN 절에서 지정한 XPath의 개수만큼 RETURN_XPath 리스트가 생성된다. RETURN_XPath 리스트에 대한 카디션 프로덕트 과정을 거쳐 XML 레코드를 생성하게 된다.

XML 레코드는 XQuery의 RETURN 절에서 지정한 XPath들과 SORTBY 절에서 지정한 XPath들의 실제 값들로 구성된 레코드이다. XML 레코드 리스트는 XML 레코드를 리스트 형태로 모아 놓은 것으로, SORTBY 절이 명시되어 있는 경우, 전체 리스트를 지정한 XPath의 값에 의해 정렬한다. XML 레코드 리스트는 그대로 테이블 형태로 반환되거나, 태그를 붙이는 과정을 거쳐 XML 문서 형태로 반환되는데 사용된다.

그림 2는 이러한 XML 문서 추출 서브시스템의 내부 자료구조에 대한 그림이다.

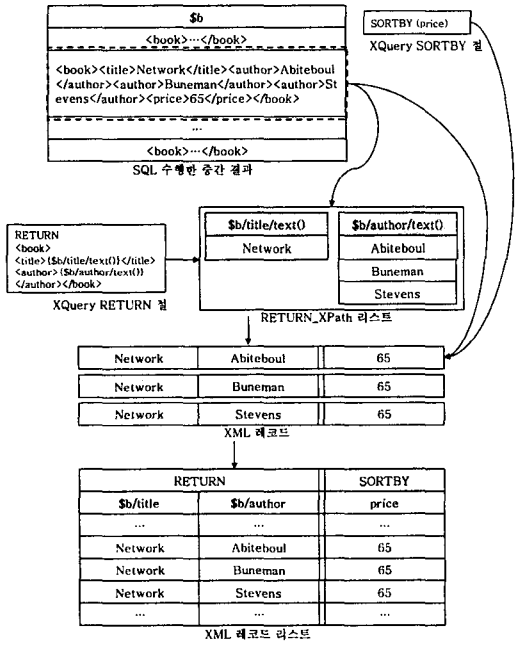

그림 2. XML 문서 추출 서브시스템의 내부 자료구조

3.2. XML 문서 추출 과정

3.1의 자료구조를 이용하여 XML 형태로 결과를 추출하는 과정은 다음과 같다.

1. SQL 중간 결과에서 한 레코드를 읽는다.
2. RETURN 절의 각 XPath 마다 중간 결과 레코드의 XML 부분 문자열로부터 해당 값을 읽어내어 XPath 개수만큼의 RETURN_XPath 리스트를 구성한다.
3. RETURN_XPath 리스트들 간에 카티션 프로덕트 (Cartesian Product) 연산을 수행한 후, 결과로 얻어지는 레코드마다 SORTBY 절의 XPath에 대한 값을 붙여서 XML레코드를 만든 다음, XML 레코드 리스트에 추가한다.
4. SQL 중간 결과의 모든 레코드에 대해 1~3의 과정을 수행한다.
5. XML 레코드 리스트의 SORTBY 값을 기준으로 해서, XML 레코드 리스트를 정렬한다.
6. XML 레코드 리스트의 각 레코드에 대해 RETURN 절의 XPath 부분을 XML 레코드에 저장되어 있는 실제 값으로 대체시켜 RETURN 절에서 요구하는 형태의 XML 형태의 결과로 반환한다.

3.3. XML 문서 추출 서브시스템의 수행 예

3.3에서는 그림 3의 XML문서를 이용하여 3.1과 3.2에서 설명한 자료 구조를 이용하여 XML 문서 추출 서브시스템이 수행하는 처리 과정의 예를 들어보겠다.

```
<bib>
    <book year= "1992">
        <title>Network</title>
        <author>Stevens</author>
        <price>65</price>
    </book>
    <book year= "1994">
        <title>Unix</title>
        <author>Stevens</author>
        <price>65</price>
    </book>
    <book year= "2000">
        <title>Database </title>
        <author>Abiteboul</author>
        <author>Buneman</author>
        <price>39</price>
    </book>
</bib>
```

그림 3. 예제 XML 문서

다음 질의문 Q는 그림 3의 XML 문서에 질의하기 위해 작성한 간단한 XQuery이다.

Q :
```
FOR $b in //book
WHERE $b/@year > 1993
RETURN
<book><title>{$b/title/text()}</title>
<author>{$b/author/text()}</author></book>
SORTBY (price)
```

XQuery 변환기에서 변환한 SQL문을 하위 관계형 데이타베이스에 수행하면, 그림 4와 같은 중간 결과가 얻어진다.

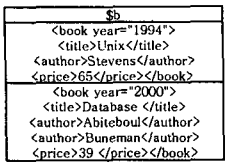

그림 4. SQL 문 수행 후의 중간 결과

테이블 형태의 중간 결과에서 레코드를 하나씩 읽어와서 XML 레코드를 구성하는데, 예제에서는 그림 4 테이블의 두 번째 레코드를 읽어온 후에 XML 문서 추출 서브시스템의 처리 과정에 대해서 살펴보겠다. 먼저, 중간 결과에서 가져온 XML 부분 문자열로부터 RETURN 절에서 지정하는 2개의 XPath, $b/title와 $b/author에 해당하는 값을 읽어내어 각 XPath에 대해 RETURN_XPath 리스트를 구성한다. XQuery Q의 RETURN 절은 그림 5와 같이 2개의 RETURN_XPath 리스트를 구성한다.

$b/title	$b/author
Database	Abiteboul
	Buneman

그림 5. RETURN_XPath 리스트

그 다음에는 RETURN_XPath 리스트 간에 카티션 프로덕트(Cartesian Product) 연산을 수행하여 그 결과로 얻어지는 레코드마다 SORTBY 절의 XPath, $b/price와 $b/author에 대한 값을 붙여서 그림 6과 같은 XML레코드를 만든다.

Database	Abiteboul	39
Database	Buneman	39

그림 6. XML 레코드

그림 6에서 만들어진 XML 레코드를 XML 레코드 리스트에 추가하면 중간 결과의 두 번째 레코드에 대한 작업이 끝난 것이다. 중간 결과의 마지막 레코드를 읽어올 때까지 XML 레코드는 계속 만들어지고, 생성된 XML 레코드는 XML 레코드 리스트에 추가된다. 그림 4 중간 결과의 모든 레코드를 처리하고 난 후에는 그림 7의 (a)와 같은 XML 레코드 리스트가 만들어진다.

RETURN		SORTBY
$b/title/text()	$b/author/text()	$b/price
Unix	Stevens	65
Database	Abiteboul	39
Database	Buneman	39

(a) 정렬 전

RETURN		SORTBY
$b/title/text()	$b/author/text()	$b/price
Database	Abiteboul	39
Database	Buneman	39
Unix	Stevens	65

(b) 정렬 후

그림 7. XML 레코드 리스트

XML 레코드 리스트가 만들어지면, XML 레코드 리스트의 SORTBY 값을 기준으로 해서, XML 레코드 리스트를 정렬한다. 그림 7의 (b)는 (a)를 "SORTBY(price)"에 맞추어 정렬을 한 그림이다.

정렬을 마친 XML 레코드 리스트를 이용하여 RETURN 절의 XPath를 그림 7에 (b)의 XML 레코드에 저장되어 있는 실제 값으로 대체시켜 그림 8과 같이 RETURN 절에서 요구하는 형태의 XML 형태의 결과로 반환한다.

```
<book>
<title>Database</title>
<author>Abiteboul</author></book>
<book>
<title>Database</title>
<author>Buneman</author></book>
<book>
<title>Unix</title>
<author>Stevens</author></book>
```

그림 8. XQuery 최종 결과

4. 결론

본 논문에서는 관계형 데이타베이스를 사용하여 XML 문서를 저장하고 검색하는 시스템인 eXDM의 내부 모듈로 구현한 XML 문서 추출 서브시스템에 대하여 기술하였다. 이 시스템은 XQuery의 RETURN 절과 SORTBY 절을 분석하여, 관계형 데이터베이스에 저장되어 있는 데이타를 구조화하여 계층적인 구조를 가지는 XML 문서 형태로 추출한다.

참고문헌

[1] Tim Bray, Jean Paoli, C.M. Sperberg-McQueen and EveMaler, "Extensible Markup Language(XML) 1.0 second edition W3C recommendation" , Technical Report REC-xml-2001006, World Wide Web Consortium, October 2000

[2] TV Anytime Forum, http://www.tv-anytime.org

[3] Surajit Chaudhuri, Raghav Kaushik and Jefrey F. Naughton, "On Relational Support for XML Publishing: Beyond Sorting and Tagging" , SIGMOD 2003.

[4] J. Shanmugasundaram, E. Shekita, R.Barr, and M. Carey. , "Efficiently publishing Relational data as XML documents" , VLDB 2001.

[5] 권준호, 권동섭, 홍석진, 곽민성, 임우규, 신효섭, 이석호, "내장형 XML 저장 및 검색 시스템의 구현", 한국정보과학회 춘계 학술 발표논문집 2003

[6] Joonho Kwon, Dongseop Kwon, Hyoseop Shin, and Sukho Lee, "Development of Embedded System for storing and retrieving XML data", CAiSE 2003

[7] Q. Li and B. Moon, "Indexing and Querying XML data for regular Path expressions", VLDB 2001

[8] XQuery 1.0 : An XML Query Language, http://www.w3.org/TR/xquery