가상의 XML 뷰를 기반으로 한 XQuery 질의 처리

차명훈,* 박영철**
영진전문대학,* 경북대학교**
mhcha@yeungjin.ac.kr, ycpark@knu.ac.kr

Query Processing of XQuery based on a Virtual XML View

Myung Hoon Cha,* Young Chul Park**
Yeungjin Junior College,* Kyungpook National University**

요 약

XML을 관계형 데이타베이스 기술과 접목시키는 하나의 방법으로 관계형 데이타베이스의 내용을 표현하는 가상의 XML 뷰를 제공하고 그 뷰에 대하여 XQuery의 질의를 요구하는 환경을 고려할 필요가 있다. 본 논문은 가상의 XML 뷰와 관계형 데이타베이스 사이의 매핑, 그 뷰를 기반으로 하는 XQuery 질의를 관계형 데이타베이스에서 처리할 수 있는 SQL 질의로 변환, 관계형 데이타베이스 시스템이 SQL 질의를 처리한 결과를 XML 문서로 변환하는 기법을 제시한다.

1. 서론

XML은 데이타를 표현하고, 교환하고, 검색하기 위한 표준화된 포맷(format)으로 성장하였다. 관계형 데이타베이스의 내용을 XML로 표현하게 되면, 지금까지 축적되어온 방대한 비즈니스 데이타들을 지속적으로 이용할 수 있게 된다. 관계형 데이타베이스의 내용을 가상의 XML로 표현한 상태에서 사용자들은 XQuery[2]를 이용하여 질의를 수행할 수 있다. 이러한 환경의 제공은 관계형 데이타베이스의 내용을 가상의 XML 뷰로 표현하고, 사용자는 그 뷰를 대상으로 XQuery 질의를 제기하는 것이다. 이 방법의 장점은 XQuery의 사용자는 자신의 질의를 SQL로 변환하지 않아도 되며 오직 XQuery와 가상의 XML 뷰만을 인식하면 된다는 것이다.

본 논문은 관계형 데이타베이스와 가상 XML 뷰의 매핑, XML 뷰를 기반으로 작성된 XQuery 질의를 관계형 데이타베이스에서 처리할 수 있는 SQL 질의로 변환, 그리고 관계형 데이타베이스 시스템이 처리한 SQL 질의 결과를 XML 문서로 변환하는 방법을 제시한다.

2. 관련 연구

XML 문서를 관계형 데이타베이스에 저장한 후 질의하는 방법에 관한 연구로는 [5,7]이 있고, XML과 뷰를 접목시키는 연구로는 [1,3,4,6]이 있다. XML 뷰를 기반으로 XQuery 질의를 처리하기 위한 일반적인 프레임워크와 그 질의를 효율적으로 처리하기 위한 방법이 제시된 시스템은 XPERANTO[1]가 대표적이다. XPERANTO는 관계형 데이타베이스 위에 XQuery를 이용하여 디폴트 XML 뷰라는 가상의 뷰를 정의하고, 그 뷰를 기반으로 사용자 정의 뷰를 정의하며, 사용자 정의 뷰를 대상으로 XQuery 질의가 제기된다. 시스템은 그 질의와 사용자 정의 뷰를 합성한 후 그 합성된 표현에서 SQL로 변환하는 부분과 XML 문서를 생성하기 위한 툴 역할을 하는 태거(tagger) 부분으로 구분하여 변환을 수행한 후 SQL로 변환된 질의를 실행한 결과를 태거와 결합하여 XML 문서를 생성한다.

Agora[4]에서는 이질적인 데이타 소스들이 각자의 지역 스키마로 표현되어 있고, 그 지역 스키마들이 하나의 XML 전역 스키마로 통합된다. 사용자는 XML 전역 스키마를 대상으로 XQuery 질의를 제기하고, 그 질의를 지역 데이타 소스들에 대한 SQL 질의로 변환하는 방법을 제시하였다.

본 논문은 관계형 데이타베이스에 저장되어 있는 데이타들을 하나의 가상 XML 뷰로 가정하며 사용자들은 가상 XML 뷰에 대하여 XQuery로 작성된 질의를 제기하는 환경을 모델로 하였다.

3. 시스템 구조

하나의 가상 XML 뷰를 배경으로 XQuery 질의가 시스템에 입력되면, 파서는 그 질의를 분석한 후 SQL로 변환하며 태거 정보를 생성한다. 데이타베이스 시스템은 SQL로 변환한 질의를 실행한 후 그 결과를 XML 문서 구축자에게 전달하고 XML 문서 구축자는 태거 정보와 SQL 질의 실행 결과를 결합하여 XML 문서를 생성한다. 전체 시스템의 구조는 그림 1과 같다.

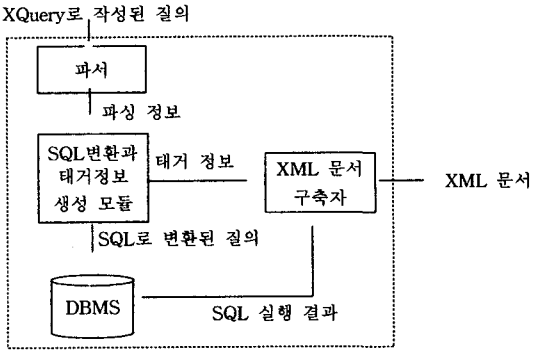

그림 1. 전체 시스템의 구조

가상 XML 뷰는 관계형 데이타베이스를 모델링한 것으로 XML 태그를 사용하여 여러 개의 데이타베이스들을 표현하며, 하나의 데이타베이스 요소는 여러 개의 테이블들, 하나의 테이블 요소는 여러 개의 행들, 하나의 행 요소는 여러 개의 칼럼 요소들을 포함한다. 가상 XML 뷰의 예는 그림 2와 같다.

```
<system>
  <database name="library">
    <table name="book">
      <row>
        <col name="title"> TCP/IP Illustrated</col>
        <col name="author">  100  </col>
      </row>
      <row>
        <col name="title"> UNIX network programming</col>
        <col name="author">  100  </col>
      </row>
    </table>
    <table name="author">
      <row>
        <col name="ssn"> 100 </col>
        <col name="name"> R. Stevens </col>
      </row>
      <row>
        <col name="ssn"> 200 </col>
        <col name="name"> D. Kruglinski </col>
```

†본 논문은 한국과학재단 목적기초연구(과제번호 : 2000-2-51200-002-3) 지원으로 수행되었음.

```
    </row>
  </table>
 </database>
</system>
```

그림 2. 가상 XML 뷰

가상 XML 뷰는 개념적인 것이며, 그 뷰의 정의가 데이타베이스에 저장되는 것은 아니다. 그림 3은 그림 2의 가상 XML 뷰가 표현하는 실제 테이블이다.

book

title	author
TCP/IP Illustrated	100
UNIX network programming	100

author

ssn	name
100	R. Stevens
200	D. Kruglinski

그림 3. library 데이타베이스의 테이블들

그림 4는 그림 2의 가상 XML 뷰에 대한 XQuery 질의의 예이다.

```
for $author in
view("default")//database[@name="library"]/table[@name="author"]/row
return
  <author name=$author/col[@name="name"]/text()>
    <books>
      for $book in view("default")//database[@name=
          "library"]/table[@name="book"]/row
      where $author/col[@name="ssn"] =
          $book/col[@name="author"]
      return
        <book>$book/col[@name="title"]/text()</book>
    </books>
  </author>
```

그림 4. XQuery 질의

4. XQuery 질의의 SQL 변환과 태거 정보의 생성

SQL 변환과 태거 정보 생성 모듈은 파서에 의해 생성된 태거 정보를 바탕으로 태거 정보 리스트라고 하는 하나의 리스트를 구성한다. 태거 정보 리스트를 생성하는 방법은 다음과 같다.

첫째, 태거 정보 리스트의 헤드(head) 노드를 생성한다. 헤드 노드는 태거 정보 리스트의 출발점이다.

둘째, XQuery 질의에 포함되어 있는 모든 원소들에 대하여 각 원소의 형(type)에 따른 노드를 생성하며 그를 이전 노드가 가리키도록 한다. XQuery 질의에 포함된 원소들은 XML 문서의 요소, 속성, pcdata, 변수, FLWR 구문으로 구분된다. 각 FLWR 문마다 return 문을 경계로 return 문 상위에 위치한 구문(for, let, where)을 SQL 노드라고 하는 하나의 노드로 표현한다. SQL 노드는 XQuery 질의를 SQL 질의로 변환하는 기본 단위로서 <하나의 스택, 변환될 SQL 구문을 저장할 SQL 필드, 다음 노드를 가리키는 포인터>의 필드들을 가진다. 요소는 요소 노드, 속성은 속성 노드, pcdata는 pcdata 노드, 향후에 값을 대입받을 변수 부분은 변수 노드, FLWR 구문은 SQL 노드로 생성하여 헤드 노드로부터 차례로 연결된 하나의 리스트를 구성한다.

셋째, XQuery의 질의의 끝에 도달하면 리스트의 마지막 노드의 다음 필드를 NULL로 설정한다.

그림 4의 질의에 대한 태거 정보 리스트와 후에 설명할 중첩_조인 트리는 그림 5와 같다. 그림 5의 태거 정보 리스트에서 SQL1은 XQuery 질의의 첫 번째 FLWR 구문인 for $author in view("default")//database[@name="library"]/ table[@name="author"]/row를, SQL2는 두 번째 FLWR 구문인 for $book in view("default")//database[@name="library"]/table[@name="book"]/row where $author/col[@name="ssn"]=$book/col[@name="author"]를 표현하는 SQL 노드들이다.

그림 5에서 $V1과 $V2는 XML 문서 구축자가 SQL 문을 실행한 결과에서 값을 인출한 후 그 값을 대입할 변수를 나타내는 노드들이다. 예를 들어, $V1은 $author/col[@name="name"]/text()를 나타낸다. $author는 view("default")//database[@name= "library"]/table[@name="author"]/row에 속하기 때문에 $V1이 SQL 문에서는 library 데이타베이스의 author 테이블의 name 칼럼의 값이라는 것을 알 수 있다.

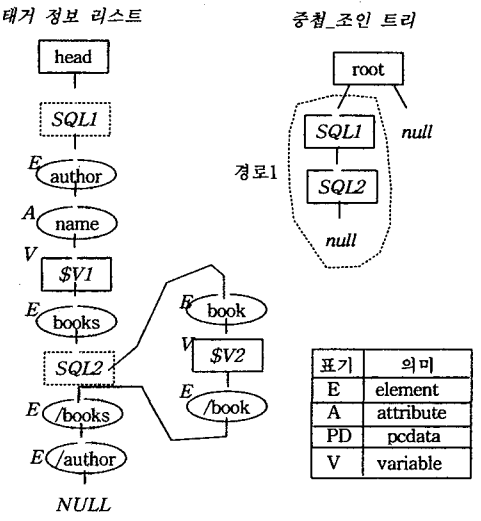

그림 5. 태거 정보 리스트와 중첩_조인 트리

그 변수 노드들을 SQL 문에서 사용될 테이블의 칼럼 이름으로 변환한 결과는 표 1과 같다.

표 1. return 문 하위 변수들의 변환결과

태거 정보 리스트의 변수 노드	SQL 문에 사용될 칼럼
$V1	author.name
$V2	book.title

SQL 노드가 가지는 스택은 XML 문서 구축자가 태거 정보 리스트의 노드들을 방문하면서 문서를 생성할 때 사용된다. 예를 들어, 태거 정보 리스트에서 SQL 노드 다음에 나오는 모든 XML 요소들의 시작 태그에 대하여 그 SQL 노드의 스택에 그 요소 정보를 푸쉬(push)하며 XML 요소의 종료 태그를 만나면 그 스택에서 팝(pop)한다. 태거 정보 리스트를 차례로 순회하는 도중 새로운 SQL 노드를 만나면 그 SQL의 스택에 대하여 위의 과정을 반복한다. 따라서, 그림 5의 SQL1의 스택에 <author>와 <books> 요소가 차례로 푸쉬된 후, SQL2의 스택에 <book>이 푸쉬되며, </book> 요소에 의하여 대응되는 <book>을 스택에서 팝한다. SQL2 노드의 스택이 비게 되면 SQL2 이하에 부속된 XML 구조에 대한 처리는 종료되며 그 이하에 나오는 </books>와 </author> 요소에 의하여 SQL1의 스택의 내용이 차례로 팝된다. SQL1의 스택이 완전히 비게 되고 리스트의 끝에 도달하면 리스트의 순회가 종료된다.

그림 5의 태거 정보 리스트에서 SQL1은 SQL2를 구조적으로 포함하고 있으며 SQL1과 SQL2 사이에 조인 관계가 존재한다. 이와 같이 중첩과 조인 관계가 동시에 성립하는 SQL 노드들의 정보를 하나의 리스트로 생성하여 중첩_조인 트리라고 하는 트리에 둔다.

XML 문서 구축자는 중첩_조인 트리에 존재하는 각 단말 노드로 부터 루트에 이르는 각 경로에 대하여 하나의 SQL 문을 생성한다. 그림 5의 중첩_조인 트리에는 하나의 경로(SQL2->SQL1->root)만이 존재한다. 중첩_조인 트리의 각 경로에 개하여 그 경로에 포함된 모든 SQL 노드들이 표현하는 for, let, where 문들을 하나의 SQL 문으로 변환하는 방법은 그림 5의 예를 대상으로 설명하면 다음과 같다.

① 사용할 데이타베이스 이름을 추출한다.

```
view("default")//database[@name="library"]
    => set schema 'library';
```

② SQL 문의 from 절에서 사용할 테이블 이름을 추출한다.

```
view("default")//database[@name="library"]/table[@name="
author"]    -> author
view("default")//database[@name="library"]/table[@name="
book"]      -> book
```

=> *from author, book*

③ 표 1에서 SQL 문에서 사용되도록 변환된 칼럼(name, title)들과 조인 조건에 사용되는 칼럼(ssn)을 유니온(union)하여 SQL select 절을 생성한다. 단, 조인 조건에 사용된 칼럼 중 중복되는 칼럼은 한번만 표현한다.

=> *select ssn, name, title*

④ XQuery의 where 절을 SQL where 절로 변환한다.

where $author/col[@name="ssn"] =
$book/col[@name="author"]

=> *where author.ssn = book.author*

⑤ 경로에서 부모, 자식 관계를 고려하여 left outer join이 되도록 ④의 결과 값을 변형한다.

where author.ssn = book.author

=> *where author.ssn = book.author(+)*

⑥ 조인 조건으로 사용된 칼럼을 기준으로 order by 절을 추가한다. 만일, XQuery에 명시적으로 sortby 절이 사용된 경우에는 그 절의 칼럼들을 SQL 문의 order by 절에 차례로 추가한다. 단, 조인 조건으로 사용된 칼럼이 order by 절의 선두에 위치한다.

=> *order by ssn*

생성된 SQL 질의문은 표 2와 같다.

표 2. 생성된 SQL 질의문

```
set schema 'library';
select ssn, name, title
from author, book
where author.ssn = book.author(+)
order by ssn;
```

표 2를 실행함으로써 생성되는 결과 집합 RS1은 표 3과 같다.

표 3. SQL 문의 실행 결과 (결과 집합 이름: RS1)

ssn	name	title
100	R. Stevens	TCP/IP Illustrated
100	R. Stevens	UNIX network programming
200	D. Kruglinski	

마지막으로, 생성된 결과 집합에 대한 정보를 해당 경로에 연계하여 XML 문서 구축자가 문서 생성 과정에서 사용할 수 있도록 표 4의 구조에 저장한다.

표 4. 경로와 결과 집합의 연계 테이블

경로 이름	결과집합 이름	현재 커서 위치	조인 칼럼
경로1	RS1	1	ssn

하나의 XML 문서 내에 포함된 FLWR 문들 사이에 조인 관계가 성립하지 않는 경우도 존재할 수 있다. 이 경우, 중첩_조인 트리에 다수 개의 경로가 존재한다. 예를 들어, 그림 6에는 SQL2->SQL1->root, SQL3->root, 그리고 SQL4->SQL1->root의 3개의 경로들이 존재한다. 이들은 3개의 SQL 문들로 변환된다.

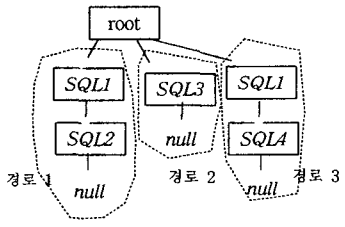

그림 6. 경로가 3개인 중첩_조인 트리

5. XML 문서 생성 알고리즘

XML 문서 구축자가 태거 정보 리스트, 중첩_조인 트리, 그리고 경로와 결과집합의 연계 테이블을 바탕으로 XML 문서를 생성하는 알고리즘은 다음과 같다.

태거 정보 리스트의 각 노드를 헤드에서부터 차례로 하나씩 방문하면서 그 리스트의 마지막인 널(null) 링크에 도달할 때까지 방문한 노드의 형에 따른 작업을 다음과 같이 수행한다.

방문한 노드가 루트 노드에 가장 인접한 SQL 노드일 경우에는 중첩_조인 트리에서 그 노드가 포함된 경로를 찾고, 그 경로에 대응되는 결과 집합의 정보를 경로와 결과 집합의 연계 테이블에서 읽은 후, 해당 결과 집합의 첫 번째 행에서부터 마지막 행까지 커서의 위치를 하나 씩 증가하면서 각 행마다 현재의 SQL 노드의 아래쪽에 연결된 노드들을 반복적으로 처리한다. 각 반복의 수행 범위는 그 SQL 노드의 스택에 요소 노드들을 푸쉬해서, 그 요소 노드들이 모두 팝될 때까지이다. 변동되는 커서의 위치는 표 4의 경로와 결과 집합의 연계 테이블의 "현재 커서 위치" 필드에 저장하여 그 상태를 유지한다.

방문한 SQL 노드가 루트 노드에서 가장 인접한 SQL 노드가 아닐 경우에는 즉, 그림 6의 경로 1의 SQL2(또는 경로 3의 SQL4) 노드일 경우에는 처리할 SQL 노드(SQL2)가 다른 SQL 노드(SQL1)에 포함되고 동시에 조인 조건이 만족될 경우(경로 1)로서 해당 결과 집합(RS1)에서 조인 칼럼(표 4의 ssn) 값이 동일한 행들에 대해서 각 행마다 SQL2 이하에 연결된 노드들을 SQL2의 스택이 팝될 때까지 태거 정보 리스트를 따라 내려가면서 처리한다. 또한, 그림 6의 경로 1과 경로 3에서 SQL1은 중복되어 있으므로, 태거 정보 리스트에서 방문하는 첫 번째 SQL 노드(SQL1)에 대한 결과 집합을 선택할 때는 중첩_조인 트리의 첫 번째 경로인 경로1의 결과 집합을 사용한다. 그 후, SQL4를 방문하면, 경로1의 결과 집합의 커서 위치에 대응되는 행의 조인 칼럼 값을 가진 행을 경로3의 결과 집합에서 검색한 후 그 행의 조인 칼럼과 동일한 값을 가지는 경로3의 결과 집합의 행들에 대해서 위의 과정을 반복한다. 표 4에서 현재 커서 위치는 중첩_조인 트리에서 루트에 가장 근접하면서 자식을 가지고 있는 SQL 노드들 중 첫 번째 경로에 해당하는 결과 집합에 대해서만 의미가 있다. 즉, 표 4에서 경로 1의 RS1에 대해서만 현재 커서의 위치가 처리할 행의 시작점을 나타낸다. 그러나, 다른 결과 집합들에 대해서는 현재 커서 위치 정보를 항상 1로 설정함으로써, 다른 경로와 관계가 없는 독립적인 결과 집합의 경우에는 그 결과 집합의 첫 번째 행부터 검색하고, 다른 경로와 관계가 있는 결과 집합의 경우에는 해당 행을 찾기 위한 출발점을 첫 번째 행부터 시작한다는 사실을 표현한다.

방문한 노드가 변수 노드이면 결과 집합의 커서 위치의 행에 대하여 대응되는 칼럼 값을 출력한다. 방문한 노드가 XML 문서의 요소, 속성, pcdata이면 그 노드의 값을 출력한다.

6. 결론

본 논문은 하나의 가상 XML 뷰를 설정하고 사용자는 그 뷰에 대해서 XQuery 질의를 함으로써 뷰가 여러개인 환경에서 나타나는 문제점인 질의의 복잡도를 단순화하였으며, 하나의 경로를 하나의 SQL 문으로 변환함으로써 질의 처리의 단순함을 도모하였다.

참고문헌

[1] J. Shanmugasundaram, J. Kiernan, E. Shekita, C. Fan, J. Funderburk, "Querying XML Views of Relational Data", Proceedings of the 27th VLDB Conference, pp.261-270, 2001.

[2] W3C, "XQuery 1.0: An XML Query Language", http://www.w3.org/TR/xquery/.

[3] S. Cluet, P. Veltri, D. Vodislav, "Views in a Large Scale XML Repository", Proceedings of the 27th VLDB Conference, pp.271-280, 2001.

[4] I. Manolescu, D. Florescu, D. Kossmann, "Answering XML Queries over Heterogeneous Data Sources", Proceedings of the 27th VLDB Conference, pp.241-250, 2001.

[5] J. Shanmugasundaram, K. Tufte, H. Gang, C. Zhang, D. DeWitt, J. Naughton, "Relational Databases for Querying XML Documents: Limitations and Opportunities", Proceedings of the 25th VLDB Conference, pp.302-314, 1999.

[6] 최규원, 정채영, 김영옥, 김영균, 강현석, 배종민, "관계형 데이터베이스에서 XML 뷰 기반의 질의 처리 모델", 2002 정보과학회 춘계 학술발표 논문집, pp.16-18, 2002.

[7] 박명제, 민준기, 윤정희, 안재용, 정진완, "관계형 데이터베이스와 XQuery를 이용한 XML 문서의 저장 및 검색 시스템", KDBC 2002 학술발표논문집, pp. 283-290, 2002.