관계 데이터를 XML로 변환하는 단순한 방법

이동진, 신병주, 진 민
경남대학교 컴퓨터공학과

A Simple Method for Publishing Relational Data in XML

Dong-Jin Lee, Byung-Ju Shin, Min Jin
Dept. of Computer Engineering, Kyungnam University

요 약

XML 데이터 사용이 급속도로 증가함에 따라 대용량의 XML 데이터를 저장 관리하는 기술이 요구되고 있다. XML과 관계 데이터베이스의 구조적 불일치로 인해, 관계 데이터베이스에 저장된 XML 데이터에 대한 질의를 처리하여 XML 문서를 생성하기 위해서는 별도의 처리과정이 요구된다. 본 논문은 패스테이블에 표현된 스키마 구조 정보를 이용하여 XQuery 질의에서 요구하는 데이터를 위한 SQL 문을 만들고, 질의 결과 XML에 대한 구조 정보를 추출하여 SQL에 의해 출력된 결과를 XML 문서로 생성해 주는 단순한 방법을 제안한다.

1. 서론

XML 데이터 사용이 급속도로 증가함에 따라 XML은 클라이언트 시스템의 복잡한 데이터 처리와 웹상에서 데이터의 작성 및 관리의 표준으로 중요한 위치를 자리잡고 있다. 이에 따라, 전 세계적으로 XML과 관련된 많은 연구들이 진행중이며, 그 중에서도 관계형 데이터베이스 관리 시스템에 XML 데이터를 저장 및 질의 처리를 위한 연구가 가장 활발히 진행되고 있다. 그러나, XML 데이터를 관계형 데이터베이스 시스템에 저장 및 질의 처리하기 위해서는 별도의 처리과정이 필요하다. 왜냐하면, XML의 계층적 구조와 관계 데이터베이스의 평면적인 스키마 구조가 불일치하기 때문이다. 따라서, XML 문서를 관계 데이터베이스에 효율적으로 처리하기 위해서는 별도의 추가적인 과정이 필요하다[7].

관계 데이터베이스에 저장된 XML 데이터에 질의 처리하여 XML 형태로 문서화하는 기존의 방법은 XML 뷰를 사용하는 방법과 사용하지 않는 방법으로 분류될 수 있다[2][3][4][8]. XML 뷰를 이용하는 방법은 사용자가 XML 뷰를 정의해야 하는 문제점이 있었다. 따라서, 본 논문에서는 XML 뷰를 사용하지 않고 XML 질의 언어인 XQuery를 SQL로의 효율적인 변환 방법을 제안한다.

본 논문의 구성은 다음과 같다. 2장에서는 관계 데이터베이스에 저장된 XML 데이터에 대해 질의 처리하는 방법 중 XML을 뷰를 사용하는 방법과 사용하지 않는 방법에 대해 소개하고, 3장에서는 XML 질의 처리 시스템에 대해 간략히 소개한다. 4장에서는 XML을 생성과정 중, Path 테이블을 사용하는 방법과 XQuery의 질의 유형을 SQL로 효율적으로 변환하는 방법 및 알고리즘에 대한 효과적인 XML 문서 생성 방법을 제안한다. 마지막으로 5장에서는 결론을 맺는다.

2. 관련연구

릴레이션 간의 평면 구조로 이루어진 관계 데이터베이스에 저장된 XML 데이터에 대한 질의 처리 후, XML 문서를 생성해 내기 위한 방법으로는 크게 XML 뷰를 이용하는 방법과 XML 뷰를 이용하지 않는 방법이 있다. XML 뷰를 이용한 질의 처리 미들웨어로서는 대표적으로 SilkRoute[1]와 XPERANTO[10][11]가 있다. 또한, XML 뷰를 이용하지 않는 방법

본 연구는 정보통신부의 정보통신학술기초연구지원사업(정보통신연구진흥원)으로 연구되었음

에도 효율적인 질의 처리를 위해 다양한 방법들이 제안되었다[12]. SilkRoute는 XML 뷰를 처리하기 위해서 선언적인 질의어인 RXL(Relational to XML Transformation Language)을 사용하여 XML 뷰를 기술한다. RXL은 기존의 관계 데이터베이스 질의어인 SQL과 XML 질의어인 XML-QL로 이루어진 질의어이다. 이는 블록 구조, 중첩된 질의, Skolem 함수를 지원하는 특징을 가지고 있다. SilkRoute에서는 DTD와 관계 스키마를 참조하여 RXL로 작성된 XML 뷰가 정의된다. RXL 뷰 질의어와 사용자 질의어인 XML-QL이 결합하여 새로운 RXL이 생성된다. 그리고, 이 RXL은 질의 변환 과정을 통해 SQL로 변환되어 관계 데이터베이스에 질의를 하게 된다[1].

XML 뷰의 지원없이 관계 테이블의 XML 데이터를 XML 문서로 변환하는 방법이 있다. 여기서는, 관계 테이블의 XML 데이터의 XML 문서로의 변환시 그림 1과 같이 중첩된 구조와 태그를 생성하는 시기와 방법에 따라 다양한 방법들을 제시하고 있다. 그리고 이 방법들은 각각 중첩된 구조와 태그를 생성하는 과정을 관계 데이터베이스 엔진 내에서 처리하는지, 엔진 외부의 응용 프로그램에서 처리하는지에 따라 성능을 평가하였다[12].

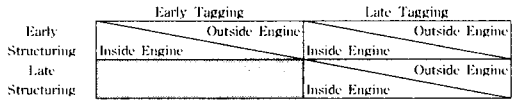

그림 1. 관계 데이터의 XML 문서로의 변환 방법 분류

3. XML 질의 처리 시스템

그림 2는 관계 데이터베이스를 이용한 XML 질의 처리 시스템의 구성도이다. XML 질의 처리 시스템은 크게 XQuery 변환기, 데이터 추출기, XML 구조화, XML 생성기 등으로 구성되어져 있다. 관계 데이터베이스의 여러 테이블에 분산 저장된 XML 데이터에 대한 질의 처리를 위해 사용자는 XQuery로 작성된 질의를 사용한다. 그러나, 기존의 관계 데이터베이스 시스템은 XML 표준 질의 언어인 XQuery를 지원하지 못하기 때문에 XQuery로 작성된 사용자 질의는 SQL로 변환한다. 변환된 SQL을 통해 관계 데이터베이스로부터 XML 데이터들을 추출한 후, 사용자가 원하는 형태의 XML 문서를 생성하게 된다.

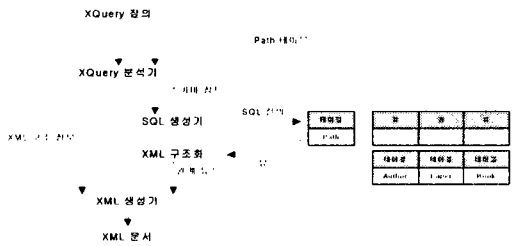

그림 2. XML 질의 처리 시스템 구성도

4. XML 생성

4.1 Path 테이블

본 논문에서는 다양한 XML 표준 질의 언어들 중 XQuery를 사용한다. XQuery에서는 구조화된 XML 문서의 특정 정보를 가리키기 위해서 패스 표현식을 적용한다[13]. 하지만, XQuery는 기존의 관계 데이터베이스에서 지원하지 못하는 이유로 XQuery의 효율적인 SQL로의 변환 과정이 필요하다. 이 과정을 위해서는 XQuery에서 사용되는 패스 표현식의 처리를 위해 발생할 수 있는 모든 패스 표현식을 Path 테이블에 저장한다[5][6][9]. 그림 3은 그림 4의 DTD에서 표현될 수 있는 모든 패스 정보를 저장한 예이다.

Path

PathID	PathExp	Table	Column	ParentCode
1	#/Paper	NULL	NULL	NULL
2	#/Paper#/PaperTitle	Paper	PaperTitle	NULL
3	#/Paper#/Authors	NULL	NULL	NULL
4	#/Paper#/Authors#/Author	NULL	NULL	NULL
5	#/Paper#/Authors#/Author#/Name	Author	Name	1
6	#/Paper#/Authors#/Author#/Country	Author	Country	1
7	#/Paper#/Authors#/Author#/University	Author	University	1
8	#/Paper#/Reference	NULL	NULL	NULL
9	#/Paper#/Reference#/Book	NULL	NULL	NULL
10	#/Paper#/Reference#/Book#/BookTitle	Book	BookTitle	1
11	#/Paper#/Reference#/Book#/Authors	NULL	NULL	NULL
12	#/Paper#/Reference#/Book#/Authors# Author	NULL	NULL	NULL
13	#/Paper#/Reference#/Book#/Authors#/Author#/Name	Author	Name	2
14	#/Paper#/Reference#/Book#/Authors#/Author#/Country	Author	Country	2
15	#/Paper#/Reference#/Book#/Authors#/Author#/University	Author	University	2
16	#/Paper#/Reference#/Paper	NULL	NULL	NULL
17	#/Paper#/Reference#/Paper#/PaperTitle	Paper	PaperTitle	1
18	#/Paper#/Reference#/Paper#/Authors	NULL	NULL	NULL
19	#/Paper#/Reference#/Paepr#/Authors#/Author	NULL	NULL	NULL
20	#/Paper#/Reference#/Paepr#/Authors#/Author#/Name	Author	Name	1
21	#/Paper#/Reference#/Paepr#/Authors#/Author#/Country	Author	Country	1
22	#/Paper#/Reference#/Paepr#/Authors#/Author#/University	Author	University	1

그림 3. Path 테이블

```
<!ELEMENT Paper ( PaperTitle, Authors, Reference? )>
<!ELEMENT PaperTitle ( #PCDATA )>
<!ELEMENT Authors ( Author* )>
<!ELEMENT Author ( Name, Country?, University? )>
<!ELEMENT Name ( #PCDATA )>
<!ELEMENT Country ( #PCDATA )>
<!ELEMENT University ( #PCDATA )>
<!ELEMENT Reference ( Book*, Paper* )>
<!ELEMENT Book ( BookTitle, Authors )>
<!ELEMENT Authors ( Author* )>
<!ELEMENT Author ( Name, Country, University )>
<!ELEMENT Name ( #PCDATA )>
<!ELEMENT Country ( #PCDATA )>
<!ELEMENT University ( #PCDATA )>
```

그림 4. DTD 예

4.2 XQuery 질의 유형

XQuery 분석기는 XQuery로 작성된 사용자 질의를 파싱을 통해 분석한다. 이 때, XQuery 분석기는 데이터 추출을 위한 스키마 정보와 XML 문서를 생성하기 위한 구조 정보를 분리하여 얻는다. 데이터 추출을 위한 스키마 정보는 SQL 생성기로 전달하여 사용자가 원하는 XML 데이터를 추출하기 위한 SQL 문 생성에 사용되고 XML 구조 정보는 XML 생성기에 전달되어 구조화된 관계 튜플과 함께 XML 문서 생성에 사용된다. 질의 변환시에 필요한 XQuery의 경로

```
for $p in document("paper.xml")//Paper
return
  <Paper>
    <PaperTitle>{$p/PaperTitle/text()}</PaperTitle>
    <Authors> {
        for $a in $p/Authors/Author
        return
          <Author><Name>{$a/Name/text()}</Name></Author> }
    </Authors>
    <Reference>
      for $b in $p/Reference/Book
      return
        <Book>
          <BookTitle>{$b/BookTitle/text()}</BookTitle>
          <Authors> {
              for $ba in $b/Author
              return
                <Author><Name>{$ba/Name/text()}</Name></Author> }
          </Authors>
        </Book>
      for $pp in $p/Reference/Paper
      return
        <Paper>
          <PaperTitle>{$pp/PaperTitle/text()}</PaperTitle>
          <Authors> {
              for $ppa in $pp/Authors/Author
              return
                <Author><Name>{$ppa/Name/text()}</Name></Author> }
          </Authors>
        </Paper>
    </Reference>
  </Paper>
```

그림 5. XQuery로 작성된 질의

표현식은 약간의 변환과정이 필요하다. 그림 3의 Path 테이블에서 보는 것처럼, 경로 표현식은 "#/"로 각 노드들을 구분하고 있다. 이는 경로 표현식에서 나타날 수 있는 "//"과 같은 경로 표현식에 대한 처리를 위해 사용하고 있다. 즉, XQuery로 작성된 질의에서 "/"는 "#/"로, "//"는 "#%/"로 변환한다. 이 변환과정과 SQL 문의 LIKE 문을 이용하면 간편한 질의 변환이 가능하게 된다[5]. 그림 5는 XQuery로 작성된 사용자 질의이다.

4.3 XML 구조 정보

XQuery 분석기에서 데이터 추출을 위한 스키마 정보를 획득하는 과정 외에도 XML 구조 정보를 추출한다. 이는 XQuery로 작성된 사용자 질의에 의해 추출된 XML 데이터를 XQuery의 return 절에 의해 사용자가 정의한 XML 구조로 XML 문서를 생성하기 위해 사용된다. 즉, XML 생성기의 입력 값이 되는 것이다. 그림 6은 그림 5의 XQuery로 작성된 질의에 대한 XML 구조 정보를 추출한 형태의 일부분이다. 이 XML 구조 정보는 XML 생성기로 전달되어 추출된 관계 튜플들과 결합되어 XML 문서를 생성하게 된다.

0	1	2	3	4	5	6
For	<Paper>	<PaperTitle>	Data(1,PaperTitle)	</PaperTitle>	<Authors>	For

7	8	9	10	11	12	13
</Author>	<Name>	Data(2,Name)	</Name>	</Author>	EndFor(6)	·····

:

43	44	45	46	47	48	49
·····	</Authors>	</Paper>	ForEnd(31)	</Reference>	</Paper>	ForEnd(0)

그림 6. XML 구조 정보

XQuery의 return 절에 사용자가 정의한 XML 구조가 없을 경우에는, 그림 3의 path 테이블을 이용하여 모든 XML의 구조를 표현하여 return절에 추가시켜서 XML 구조 정보를 추출한다.

4.4 생성 알고리즘

XML 생성기는 XQuery 분석기에서 추출된 구조정보와 XML 구조화 과정에서 생성된 관계 튜플을 결합하여 최종적인 XML 문서를 생성한다. 그림 7은 XML 문서를 생성하는 알고리즘을 보여준다. 그림 7의 알고리즘에서 보는 것처럼, 구조정보와 관계 튜플이 입력되어 최종적인 XML 문서가 생성된다.

XQuery에서 표현된 XML 구조 정보를 저장한 배

열을 순차적으로 순회하면서 XML 문서를 생성한다. XML 구조 정보를 저장한 배열에서 태그는 태그 생성을 위해서 사용되고 XML 데이터가 필요한 경우에는 추출된 관계 튜플의 Type 속성을 비교하여 순차적으로 데이터를 삽입한다.

```
XMLGenerator( list_pointer XQuery, recordset record_set ) {
  boolen inner_for = false; list_morpheme w; int front = 1, rear = 1;
  for( i = 0; i < MAX_MORPHEME; i++ ) {
    w = XQuery[i];
    if( w->morpheme == "FOR" || w->morpheme == "LET" ) {
      if( inner_for == true && front != rear ) queue_output( &front, rear );
      else inner_for = true;
    }
    else if( w->morpheme == "DATA" ) {
      if( w->tuple_type == record_set("type") ) {
        if( front != rear ) queue_output( &front, rear );
        printf("%s", record_set(*att_name));
      }
      else {
        record_set.movenext;
        if( w->tuple_type == record_set("type") ) {
          if( front != rear ) queue_output( &front, rear );
          printf("%s", record_set(*att_name));
        }
        else {
          i = break_for( i );
          front = rear;
        }
      }
    }
    else if( w->morpheme == "FOREND" || w->morpheme == "LETEND")
        || w->start_for == 1;
    else {
      if( inner_for == false ) printf("%s", w->morpheme);
      else {
        rear = rear + 1;
        queue[rear] = *morpheme;
      }
    }
  }
}
void queue_output( int *front, int rear ) {
  while( *front <= rear ) {
    *front = *front + 1;
    printf("%s", queue[*front]);
  }
}
int break_for( int i ) {
  while( XQuery[i]->morpheme != "FOREND" &&
                                XQuery[i]->morpheme != "LETEND" ) i++;
  return i;
}
```

그림 7 XML 문서 생성을 위한 알고리즘

5. 결론

XML 문서와 관계 데이터베이스의 구조적 불일치로 인해 XML 표준 질의 언어인 XQuery는 기존의 관계 데이터베이스에서 지원해 주지 못하고 있다. 본 논문에서는 관계 데이터베이스에 저장된 XML 데이터에 대해 XQuery로 표현된 질의를 처리하고 결과를 XML 문서로 출력해 주는 단순한 방법을 제안하였다. 질의 처리 부분은 Path 테이블의 정보를 이용하여 XQuery의 경로 표현식을 SQL로 변환하고 XQuery 질의로부터 질의 결과 XML에 대한 구조 정보를 추출하여 질의 결과를 XML로 출력한다.

[참고문헌]

[1] M. Fernandez, W. Tan, D. Suciu, "SilkRoute : Trading Between Relations and XML", Proceedings of the International World Wide Web Conference, 2000

[2] M. Carey, D. Florescu, Z. Ives, Y. Lu, J. Shanmugasundaram, E. Shekita, S. Subramanian, "XPERANTO : Publishing Object-Relational Data as XML", WebDB Workshop, Dallas, 2000

[3] M. Fernandez, W. Tan, D. Suciu, "SilkRoute : Trading Between Relations and XML", Proceedings of the International World Wide Web Conference, 2000

[4] J. Shanmugasundaram, J. Kiernan, E. Shekita, C. Fan, J. Funderbu가, "Querying XML Views of Relational Data", Proceedings of the 27th VLDB Conference, Roma, 2001

[5] M. Yoshikawa, T. Amagasa, "XRel : A Path-Based Approach to Storage and Retrieval of XML Documents Using Relational Databases", ACM Transactions on Internet Technology, Vol. 1, No. 1, August 2001

[6] World Wide Web Consortium, "XML Path Language(XPath) Version 1.0", W3C Recommendation, November 1999. http:/www.w3c.org/TR/xpath

[7] J. Shanmugasundaram, K. Tufte, G. He, C. Zhang, D. DeWitt, J. Naughton, "Relational Databases for Querying XML Documents : Limitations and Opportunities", VLDB, Edinburg, Scotland, 1999

[8] J. Shanmugasundaram, E. Shekita, R. Barr, M. Carey, B. Lindsay, H. Pirahesh, B. Reinwald, "Efficiently Publishing Relational Data as XML Documents", VLDB, 2000

[9] World Wide Web Consortium, "XQuery 1.0: An XML Query Language", W3C Working Draft, November 2002. http://www.w3c.org/TR/xquery

[10] M. Carey, D. Florescu, Z. Ives, Y. Lu, J. Shanmugasundaram, E. Shekita, S. Subramanian, "XPERANTO : Publishing Object-Relational Data as XML", WebDB Workshop, Dallas, 2000

[11] J. Shanmugasundaram, J. Kiernan, E. Shekita, C. Fan, J. Funderburk, "Querying XML Views of Relational Data", Proceedings of the 27th VLDB Conference, Roma, 2001

[12] J. Shanmugasundaram, E. Shekita, R. Barr, M. Carey, B. Lindsay, H. Pirahesh, B. Reinwald.

"Efficiently Publishing Relational Data as XML Documents", *VLDB*, 2000

[13] 신병주, 송청, 진민, "관계 데이터베이스를 이용한 XML 문서 저장 및 검색", 한국멀티미디어학회 추계학술발표논문집 제5권 제2호, 2002