XQuery의 SQL:2003으로의 효율적인 변환

김송현[0†] 박영섭[††] 이윤준[††]
[†]공군사관학교 전산통계학과 [††]한국과학기술원 전산학과
shkim[0†]@afa.ac.kr, {yspark[††], yjlee[††]}@dbserver.kaist.ac.kr

An Efficient XQuery Translation into SQL:2003

Song Hyon Kim[0†] Young Sup Park[††] Yoon Joon Lee[††]
[†]Department of Computer Science and Statistics, Korea Air Force Academy
[††]Department of Computer Science, Korea Advanced Institute of Science and Technology

요 약

XML이 다양한 장점으로 인해 인터넷 기반 환경에서 데이터의 표현 및 교환의 표준으로 자리잡으면서 XML 데이터의 효율적인 저장 및 질의 처리에 대한 연구가 활발히 진행되었다. XML 데이터를 저장하는 방법 중에서 강력한 질의 처리 및 데이터 관리 기능을 제공하는 관계형 데이터베이스 시스템에 저장하는 것은 많은 이익을 가져온다. 그러나, 이 방법을 사용하기 위해서는 XML 질의를 SQL 질의로 변환해야 한다. 본 논문에서는 대표적인 XML 질의 언어인 XQuery 질의를 SQL:2003 질의로 변환하는 방법을 제안한다. 최근 XML 데이터와 관계형 데이터베이스의 상호 운용에 대한 요구가 증대되면서 SQL:1999를 대체하는 SQL:2003에는 XML을 지원하기 위한 표준을 포함하고 있으므로, SQL:2003을 지원하는 관계형 데이터베이스 시스템을 기반으로 한다면, XML 질의를 보다 쉽게 SQL 질의로 변환할 수 있다. 본 논문에서는 SQL 템플릿을 기반으로 XQuery 질의를 SQL 질의로 변환하는 방법을 제안한다.

1. 서 론

XML(eXtensible Markup Language)[1]이 데이터의 표현 및 교환의 표준으로 자리 잡으면서 많은 새로운 데이터들이 XML 형식으로 작성되고, 기존의 데이터들이 XML 형식으로 변환되어, XML 데이터의 양이 크게 증가하고 있다. 따라서, 대량의 XML 데이터에 대한 효율적인 저장 및 질의 처리가 매우 중요하게 인식되고 있으며, 최근 몇 년 동안 XML 데이터의 저장 및 질의 처리 기법에 대한 연구가 활발하게 진행되고 있다.

XML 데이터의 저장 및 질의 처리 기법 중에서 기존의 데이터베이스 시스템을 기반으로 한 XML 저장 및 질의 처리 시스템에 관한 연구가 활발히 진행되고 있는데, 이는 기존 시스템들이 오랜 기간 동안의 연구, 개발 및 운영을 통해 안정화되고 최적화된 성능을 가진 다양한 기능을 제공하고 있기 때문이다.

최근 XML과 관계형 데이터베이스의 상호운용에 대한 요구가 증대되면서 SQL:1999 표준을 대체하는 SQL:2003에는 XML을 지원하기 위한 표준을 포함하고 있다. 그러므로, SQL:2003을 지원하는 관계형 데이터베이스 시스템을 기반으로 한다면, XML 질의를 보다 쉽게 SQL 질의로 변환 할 수 있으며, 관계형 데이터베이스 시스템에서의 질의 결과를 XML 형태로 구조화하고 태깅(tagging)하던 후처리 과정을 생략하고, 관계형 데이터베이스 시스템에서 바로 XML 인스턴스를 결과로 얻을 수 있는 장점이 있다.

본 논문은 XML을 지원하기 위한 표준을 포함하고 있는 SQL:2003을 준수하는 관계형 데이터베이스 시스템을 기반으로 하여 XML 데이터가 저장되어 있을 때 발생하는 XML 질의의 SQL 변환 문제를 다룬다. 본 논문의 전제조건이 SQL:2003을 준수하는 관계형 데이터베이스 시스템이므로, 우리는 쉽게 XML 데이터 타입과 XML 출판(publishing) 함수를 사용할 수 있다[2].

본 논문의 구성은 다음과 같다. 제 2장에서는 관련 연구로서 기존 연구에서의 XML 질의의 SQL 변환과 SQL:2003 표준에서의 XML 지원 특징에 대해 설명한다. 제 3장에서는 SQL 템플릿을 기반으로 한 XML 질의의 SQL 변환에 대해 설명한다. 마지막으로 제 4장에서 결론을 맺는다.

2. 관련 연구

2.1 XML 질의의 SQL 변환

그 동안 XML 질의를 SQL로 변환하여 처리하고자 하는 노력이 많이 진행되어 왔다. 일반적인 XQuery를 SQL로 변환하고자 하는 연구는 [3], [4]에서 진행되었다.

[3]과 [4]는 XML 스키마를 기반으로 하지 않는 저장 방법인 에지[5] 방법으로 XML 데이터를 관계형 데이터베이스 시스템에 저장하고, XQuery 질의에 대한 SQL 변환은 사전에 정의된 SQL 템플릿을 이용하여 단일 SQL로 변환한다. 그러나, XML 데이터를 관계형 데이터베이스에 저장하는 방법을 비교한 논문[6]에서 에지 매핑 방법은 스키마를 기반으로 한 저장 방법과 비교했을 때, 떨어진 성능을 보이고 있으며, [3]과 [4]에서 제안한 방법은 모든 XML 데이터를 하나의 테이블에 저장하고 있으므로, XML 데이터의 양이 많은 실제 환경에서는 적용하기 힘들다. 특히, [3]에서는 많은 정수 조건에 의한 조인을 수행하게 되는데, 이는 일반적인 관계형 데이터베이스 시스템을 이용할 시 성능저하를 가져오게 된다[7].

2.2 SQL:2003 표준

SQL:2003은 SQL:1999를 대체하게 되는데 SQL:1999에 있던 버그들을 수정하고, SQL:1999의 내용 중 중복되거나 불필요한 부분을 제외하고 새롭게 8개의 부분(part)으로 정비하였으며, XML과 관련된 SQL/XML(part 14)[2]을 추가하였다.

SQL/XML은 네이티브(native) XML 데이터 타입, SQL과 XML의 매핑 규칙, XML 출판(publishing) 함수 등을 정의한다. XML 출판 함수들을 정리하면 다음과 같다.

- XMLELEMENT() : 태그 이름과 관계 데이터를 전달받아서 XML 엘리먼트를 생성한다.
- XMLATTRIBUTES() : 관계 데이터의 각 칼럼 이름을 애트리뷰트의 이름으로 사용하여 여러 칼럼으로부터 XML 애트리뷰트들을 생성한다. 주로, XMLELEMENT()와 함께 사용된다.
- XMLROOT() : XML 인스턴스의 루트 엘리먼트를 생성한다.
- XMLFOREST() : 여러 칼럼의 데이터를 전달받아서 XML 엘리먼트들을 생성한다. 이때, 명시적으로 태그 이름을 전달 받지 않으면,

관계 데이터의 칼럼명이 태그 이름으로 사용된다.
- XMLCONCAT() : 관계 데이터 결과에서 한 행의 여러 칼럼에 대한 XML 엘리먼트 리스트를 전달받아서 XML forest를 생성한다.
- XMLAGG() : 관계 데이터 결과에서 여러 행의 한 칼럼에 대한 XML 엘리먼트 리스트를 전달받아서 XML forest를 생성한다.

SQL/XML에는 XML 데이터 타입 인스턴스에서 XML 데이터를 추출하는 함수를 정의하고 있지 않기 때문에, XML 데이터 타입을 활용하는데 한계가 있으므로 사용자 정의 함수를 정의하여 사용한다.

- XT() : XML 데이터 타입 인스턴스와 XPath 표현식을 전달받아서 XML 단편이 포함된 XML 데이터 타입 인스턴스를 반환한다.
- XTV() : XML 데이터 타입 인스턴스와 XPath 표현식을 전달받아서 결과 노드의 스칼라 값을 반환한다.
- XS() : XML 데이터 타입 인스턴스를 전달받아서 최상위-레벨 엘리먼트가 하나의 튜플을 이루는 XML 데이터 타입 튜플 집합을 반환한다.

3. SQL 템플릿을 기반으로 한 질의 변환

본 장에서는 SQL 템플릿을 기반으로 XQuery 질의를 SQL 질의로 변환하는 방법에 대해 설명한다. 제 3.1절에서는 XQuery 질의를 처리하기 위하여 XQuery 구문을 핵심 구문과 보조 구문으로 구분하는 것에 대해 설명하고, 제 3.2절에서는 제 3.1 절에서 정의한 핵심 구문에 대응하는 SQL 템플릿을 정의한다. 제 3.3절에서는 정의된 SQL 템플릿을 이용하여 XQuery 질의를 SQL 질의로 변환하는 과정을 설명한다.

그림 1은 본 논문에서 사용할 XQuery 질의로서 XMark[8]에서 가져온 것이다. 질의의 내용은 경매가 종료된 경매 정보에서 사람마다 몇 개의 품목을 샀는가를 묻는 것이다.

```
1: for $p in doc("auction.xml")/site/people/person
2: let $a := for $t in doc("auction.xml")/site/closed_auctions
   /closed_auction
3:           where $t/buyer/@person = $p/@id
4:           return $t
5: return <item person="{$p/name}">{count($a)}</item>
```

그림 1 XQuery 질의

3.1 XQuery 구문 정의

XQuery 구문은 XQuery 질의 내에서의 역할과 SQL 템플릿의 필요성에 따라, 핵심 구문과 보조 구문으로 분류한다.

정의 1. 핵심 구문(Core Syntax)은 XQuery에서 중추적인 역할을 하며, SQL 템플릿을 필요로 하는 구문이다.

핵심 구문은 XQuery 질의를 SQL로 변환했을 때, XML 데이터 타입 인스턴스를 입력으로 받아 XML 데이터 타입 인스턴스를 반환할 수 있는 구문을 말하며, FLWOR 표현식의 ForClause, LetClause, WhereClause, OrderByClause, ReturnClause, 함수호출의 FunctionCall, 생성자인 DirElemConstructor 등이 그 예이다.

정의 2. 보조 구문(Assistance Syntax)은 XQuery에서 보조적인 역할을 하며, 대응하는 특정 데이터 구조를 가지는 구문이다.

보조 구문은 SQL 템플릿을 가지지 않고, 특정 데이터 구조에 저장되어 핵심 구문이 SQL 템플릿으로 치환될 때, 필요한 데이터를 제공한다.

3.2 SQL 템플릿

핵심 구문에 대응하는 SQL 템플릿은 *FunctionCall, ForClause, LetClause, WhereClause, OrderByClause, ReturnClause, DirElemConstructor* 등이다(SQL 템플릿의 이름은 대응하는 XQuery 구문의 이름을 이탤릭체로 표현함).

■ *ForClause*

XQuery의 for 절은 XML 인스턴스에서 Path 표현식에 해당하는 XML 부분을 반환하는데 사용된다. 그림 2는 ForClause의 SQL 템플릿을 나타낸다. *ForClause* 템플릿은 변수 이름(v)과 Path 표현식(p)를 전달인자로 전달받아, XML 타입 인스턴스를 반환한다.

라인 5는 변환되는 SQL의 별명(alias)을 기술한다. 이는 XQuery의 각 핵심 구문이 WITH 절의 서브 블록으로 표현되기 때문에 모든 SQL 템플릿은 별명을 가지게 되고, CBN(Current Block Name)은 자동적으로 부여된다. 라인 7의 SELECT 절에서는 XT 함수를 사용해서 XML 인스턴스를 추출한다. 그리고, SELECT 절에서의 XML 인스턴스에 대한 칼럼명을 xvalue로 통일하기 위해 " as xvalue" 라고 기술해 준다.

라인 8의 FROM 절에는 p.VN이 사용된다. p.VN은 추상구문트리를 재구성하여 SQL로 변환할 때, 바로 이전의 서브 질의 블록을 가리킨다(그림 7 참조). 즉, 그림 1의 예제에서 라인 1의 ForClause를 SQL로 변환할 때, p.VN은 doc 함수에 대응하는 SQL 서브 블럭을 가리키게 된다. 따라서, doc 함수에 의한 XML 타입 인스턴스를 for 절에서 사용할 수 있게 되는 것이다. 라인 8~9를 살펴보면, 관계 데이터베이스에서의 연산의 단위가 튜플이기 때문에 XT 함수를 사용하여 해당 path 표현식의 데이터들을 추출하고, 이것을 XS 함수를 사용하여 최상위-레벨 엘리먼트들의 집합으로 만든 다음, 이를 table 함수를 사용하여 임시 테이블(v)을 만든다.

```
1: ForClause:
2: ( var: VarName,      // 변수 이름
3:     p: PathExpr )    // Path 표현식
4: ⇒
5: CBN
6: as
7: SELECT XT(value(v), 'p.LSk.QN') as xvalue
8: FROM p.VN v0, table(XS(XT(v0.xvalue,
9: 'p.LS0.QN + ... + p.LSk.AX + p.LSk.AT + p.LSk.QN'))) v
```

그림 2 ForClause의 SQL 템플릿

XQuery는 변수를 선언하여, 질의에 포함된 다른 구문 내에서 선언된 변수를 사용할 수 있다. XQuery 구문 중에서 for 절과 let 절에서 새로운 변수를 선언할 수 있다. 그림 1의 예제 XQuery에서는 바깥쪽 FLWOR 표현식의 for 절(라인 1)에서 선언한 변수(p)가 안쪽 FLWOR 표현식의 where 절(라인 3)에서 사용되고 있다. 따라서, XQuery 질의를 SQL 질의로 변환하기 위해서는 이런 변수 정보를 유지할 필요가 있다.

표 1 질의 변수 정보

VarName	Element Name	CBN
a	paper	bn1
t	author	bn3
...	...	...

우리는 표 1과 같이 정보를 유지한다. 유지하는 정보는 변수 이름(VarName), 변수에 해당하는 최상위 엘리먼트 이름(Element Name), 그리고, 그 변수에 대해 가장 마지막에 변환된 서브 질의 블록 이름(CBN)이다. 가장 마지막에 정의된 서브 질의 블록 이름을 유지하는 이유는 다음과 같다. 그림 1의 XQuery 질의에서 라인 2~4에 걸쳐 변수 t가 사용되고 있다. 라인 2의 for 절에서 변수 t가 선언되면, 변수 t의 초기값은 라인 2의 for 절을 변환한 서브 질의 블록이 된다. 그러나, 라인 3의 where 절에 의해 변수 t가 가리키는 XML 인스턴스가 등호 조건에 의해 필터링 되므로, 이제 t의 값은 where 절에 의해 필터링 된 XML 인스턴스이고, 이는 where 절을 SQL로 변환한 서브 질의 블록이 된다. 따라서, 라인 4의 return 절에서 사용하는 변수 t는 라인 2의 for 절에서 변환된 서브 질의 블록을 사용하는 것이 아니라, where 절에 의해 필터링 된 서브 질의 블록을 사용하게 된다. 이렇게 하기 위해 같은 변수에 대해 가장 마지막에 정의된 서브 질의 블록 이름을 유지하는 것이다.

```
bn2
( SELECT XT(value(v), 'person') as xvalue
  FROM bn1 v0, table(XS(XT(v0.xvalue,
  '/site/people/person'))) v
)
```

그림 3 ForClause의 SQL 변환 결과

그림 3은 그림 1의 예제에서 라인 1의 ForClause가 SQL로 변환된 결과이다. 다른 SQL 템플릿들도 위에서 설명한 방법과 유사한 논리와 구조를 가지고 구성할 수 있다. 본 논문에서는 지면의 제약으로 인해 다른 SQL 템플릿에 대한 설명은 생략한다.

3.3 XQuery to SQL 변환

추상구문트리는 Transformer에 의해 핵심 구문을 중심으로 재구성된다. Transformer는 추상구문트리를 전위 순회방식으로 순회하면서, 보조 구문을 가진 노드를 특정 데이터 구조로 재구성하고, 이를 핵심 구문을 가진 부모나 조상 노드의 속성에 포함시켜서 추상구문트리의 노드가 핵심 구문만으로 이루어 지도록 재구성한다. 즉, 그림 4에서 보는 것처럼, 핵심 구문1,2와 보조 구문1,2,3을 가진 추상구문트리를 핵심 구문1과 핵심 구문2가 트리를 구성하고, 보조 구문1,2,3은 각각 부모 노드의 속성으로 변환한다.

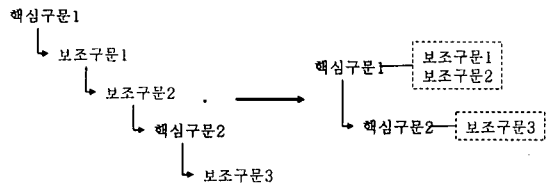

그림 4 추상구문트리(왼쪽)와 재구성된 추상구문트리(오른쪽)

추상구문트리를 재구성하는 알고리즘이 그림 5에 나타나 있다. transformAST 함수는 재구성하고자 하는 추상구문트리(a_1)와 재구성된 추상구문트리가 저장될 데이터구조(t_1)를 전달받아서 알고리즘을 수행한다. 이 알고리즘은 a_1이 자식 노드를 가질 때(라인 5) 수행된다. a_1이 자식 노드를 가지면, 각각의 자식 노드에 대해 그 노드가 핵심 구문인지를 검사하고(라인 7), 핵심 구문이면 라인 8~11을 수행한다. 그리고, a_2가 핵심구문이 아닐 때는 a_2의 자식 노드 a_3에 대해 핵심 구문을 만날 때 까지 특정 데이터 구조 d를 생성한다(라인 15).

```
1:  procedure transformAST(transformed AST t1, AST a1) {
2:    // t1,t2 : node of transformed AST
3:    // a1,a2,a3 : node of AST
4:    // d : specific data structure
5:    if (a1 has child node) {
6:      for each a1's child node a2 {
7:        if(a2 is Core Syntax) {
8:          Generate transformed AST t2
9:          Add t2 to attribute of d
10:         Add t2 to child node of t1
11:         transformAST(t2, a2)
12:       } else {
13:       for each a2's child node a3 {
14:         while (a3 is not Core Syntax) {
15:           Generate d
16:         }
17:         transformAST(t1, a3)
18:       } } } } }
```

그림 5 추상구문트리를 재구성하는 알고리즘

그림 1의 XQuery 질의를 핵심 구문을 중심으로 재구성하면, 그림 6과 같다(보조 구문에 해당하는 데이터 구조는 지면 제약상 표현하지 않음).

재구성된 추상구문트리는 전위 순회 방식으로 각각의 노드를 탐색을 하면서 각각의 구문을 대응하는 SQL 템플릿으로 치환된다. 즉, 그림 6의 추상구문트리는 각 노드의 괄호 안에 표시된 번호 순서대로 SQL 템플릿으로 치환되어, 최종적으로 단일 SQL 질의가 생성된다. SQL로 변환하는 알고리즘은 그림 7에 나타나 있다. generateSQL함수는 재구성된 추상구문트리 t_1을 전달 받아 SQL을 생성한다. t_1이 자식을 가지고 있을 때, 그 노드들을 전위 순회 방식으로 각 노드를 방문 하면서, 노드 이름에 따라 해당하는 SQL 템플릿을 적용한다(라인 7~9).

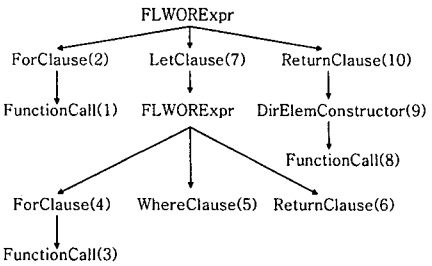

그림 6 재구성된 추상구문트리

```
1: procedure generateSQL(transformed AST t1) {
2:   if(t1 has child node) {
3:     for each t1's child node t2 {
4:       generateSQL(t2)
5:     }
6:     s ← node name of t2
7:     if(s equals 'ForClause')
8:       Use SQL template of Fig. 2
9:     else if .... } }
```

그림 7 SQL 생성 알고리즘

4. 결 론

XML은 플랫폼 독립적이고, 반구조적인 특징으로 인해 여러 분야에서 활용되어 XML 데이터의 양이 많아졌다. 이에 따라, 대량의 XML 데이터를 저장하고 질의하는 방법에 대한 연구가 활발히 이루어졌다. XML 데이터를 저장하는 방법으로 관계형 데이터 베이스 시스템을 이용하는 방법이 널리 연구되고 있고, 관계형 데이터베이스에 대한 표준 질의 언어인 SQL도 XML을 지원하기 위하여 SQL:2003 표준에서는 SQL/XML이라 불리는 XML을 지원 특징을 표준으로 포함하고 있다.

본 논문에서는 XML 데이터가 SQL:2003 표준을 지원하는 관계형 데이터베이스 시스템에 저장되어 있을 때, 이에 대한 XQuery 질의를 처리하기 위하여 SQL:2003 표준에서 정의하고 있는 SQL/XML과 사용자 정의 함수를 사용하여 XQuery 질의를 SQL 질의로 효율적으로 변환 방법을 제안하였다.

참고 문헌

[1] World Wide Web Consortium, Extensible Markup Language (XML) 1.0 (Third Edition), W3C Recommendation, Feb. 2004 http://www.w3.org/TR/2004/REC-xml-20040204/

[2] ISO/IEC 9075:2003 Information technology - Database languages - SQL (15 December 2003).

[3] D. DeHaan, D. Toman, M.P. Consens, and M.T. A Oszu. A Comprehensive XQuery to SQL Translation using Dynamic Interval Encoding. In Proc. of the 22nd Int'l ACM SIGMOD Conference on Management of Data, pages 623{634, San Diego, California, USA, June 2003.

[4] T. Grust, S. Sakr, J. Teubner. XQuery on SQL Hosts Proceedings of the 30th Int'l Conference on Very Large Databases (VLDB 2004), Toronto, Canada, August/September 2004.

[5] D. Florescu, D. Kossman, Storing and Querying XML Data using an RDMBS. IEEE Data Engineering Bulletin 22(3), 1999.

[6] Tian, et al., Design and Performance Evaluation of Alternative XML Storage Strategies, SIGMOD Record, March 2002.

[7] R. Krishnamurthy, R. Kaushik, and J. Naughton. XML-to-SQL Query Translation Literature: The State of the Art and Open Problems. In Proc. of the 1st Int'l XML Database Symposium (XSym), pages 1-18, Berlin, Germany, September 2003.

[8] XMark - An XML benchmark project. http://wwww.xml-benchmark.org.