관계형 데이터베이스에서 XML 뷰 기반의 질의 처리 모델

정 채 영[†]·최 규 원[†]·김 영 옥[††]
김 영 균[†††]·강 현 석[††††]·배 종 민[†††††]

요 약

본 논문은 XML 기반의 데이터베이스 통합 방법론 중에서 관계형 데이터베이스 모델에 대한 랩퍼 시스템의 질의어 처리에 대하여 논한다. 관계형 데이터베이스의 내용은 W3C에서 제안된 XML Schema로 표현되며, 사용자는 XML Schema에 대하여 XML 질의어인 XQuery로써 질의를 한다. 그리고, 개발된 랩퍼 시스템은 사용자가 정의한 XML 뷰를 지원한다. XML 뷰 정의 언어는 XQuery이다. 이러한 환경에서 본 논문은 새로운 XML 질의 처리 모델을 제시한다. XML 뷰와 사용자 질의어의 합성 알고리즘, XQuery를 SQL로 변환하는 알고리즘, 그리고 XML 문서 생성을 위한 템플릿 구성 알고리즘을 제시한다.

A Query Processing Model based on the XML View in Relational Databases

Chai-Young Jung[†] · Kyu-Won Choi[†] · Young-Ok Kim[††]
Young-Kyun Kim[†††] · Hyun-Syug Kang[††††] · Jong-Min Bae[†††††]

ABSTRACT

This paper addresses the query processing component of a wrapper system for a relational database model based on the XML view in integrating databases. The schema of a relational database is represented as XML Schema that is proposed by W3C. Users submit a query using the XML query language XQuery over the XML Schema. The wrapper system to be developed supports an user-defined XML view. XQuery is also used as the view definition language. In this environment, this paper suggests a new XML query processing model. We propose the composition algorithm of an XML view with an user query, the translation algorithm of XQuery into SQL, and the XML template construction algorithm for generating XML documents.

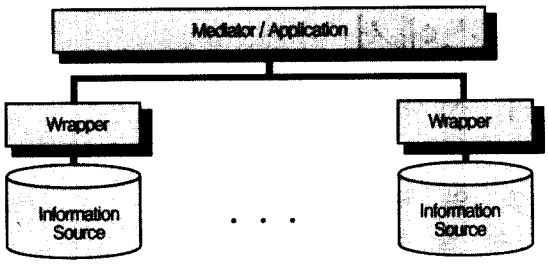
키워드 : XML, 관계형 데이터베이스(Relational Database), XML 뷰(XML View), XQuery, SQL

1. 서 론

사용자는 원하는 정보를 얻기 위하여 다수의 데이터베이스에 접근 해야할 경우가 있다. 이 때, 사용자는 많은 데이터베이스에 각각 접근해야 하고, 자료를 스스로 처리해야 하는 불편함이 있다. 이러한 문제를 해결하기 위하여 서로 연관된 내용을 가진 이질(Heterogeneous)의 데이터베이스들을 물리적으로 혹은 가상적으로 통합함으로써, 사용자에게 단일 데이터베이스 관점을 제공할 필요가 있다. 즉, 데이터베이스를 통합하여 사용자에게 단일화된 인터페이스를 제공할 필요가 있다.

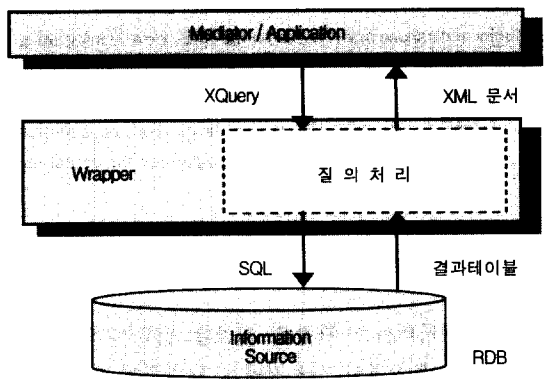
데이터베이스 통합론[7, 9, 15, 17] 중에서 미디에이터(Mediator)[9, 18] 기반의 데이터베이스 통합 방법은 미디에이터가 가상적으로 통합된 뷰를 유지하도록 하고, 사용자가 통합된 뷰를 통해서 질의를 할 수 있도록 하는 것이다. 이를 위해 각 지역 데이터베이스와 직접 대화하는 자료 저장소 랩퍼(Wrapper)[8, 11]가 필요하다. 즉, 랩퍼는 각 지역 데이터베이스마다 하나씩 존재하여, 지역 데이터베이스에 대한 내용을 미디에이터에게 전달함으로써, 미디에이터가 통합된 뷰를 구성할 수 있도록 정보를 제공하고 미디에이터로부터 받은 사용자 질의어를 지역 데이터베이스에게 전달하여 그 결과를 미디에이터에게 넘겨주는 역할을 한다.

※ 본 연구는 한국전자통신연구원의 "클러스터 기반 통합 멀티미디어 DBMS 개발" 사업의 일부로 수행된 결과임
† 준 회 원 : 경상대학교 대학원 컴퓨터과학과
†† 정 회 원 : 경상대학교 대학원 컴퓨터과학과
††† 정 회 원 : 한국전자통신연구원
†††† 종신회원 : 경상대학교 컴퓨터과학과 교수
††††† 종신회원 : 경상대학교 컴퓨터과학과 교수, 교신교수
논문접수 : 2002년 7월 24일, 심사완료 : 2003년 1월 6일



(그림 1) 미디에이터-랩퍼 시스템

이러한 미디에이터-랩퍼 시스템에서 이질적인 정보 시스템들을 통합하기 위해서는 각 지역 데이터베이스의 내용을 표현할 수 있는 하나의 중립적인 모델이 필요한데, XML 기반의 모델이 중요한 대안으로 인정받고 있다. 이질적인 데이터를 쉽게 표현할 수 있는 XML은 웹 상에서 문서 데이터 교환을 위한 표준이 되었다. XML은 기업간 비즈니스 어플리케이션을 통합할 수 있고 시스템간의 데이터를 교환할 수 있다. 그리고, 구조적인 검색을 할 수 있으며, 또한 여러 유형의 데이터를 통합 관리할 수 있기 때문에 본 논문에서는 XML 기반으로 이질적인 데이터를 통합한다. XML 기반의 통합이라 함은 각 지역 데이터베이스의 내용을 XML 스키마로 표현하고, 사용자 질의어는 XML 질의어를 사용하며 질의 결과는 XML 문서임을 말한다. 즉, 사용자 입장에서는 실제 지역 데이터베이스를 바탕으로 질의를 하는 것이 아니라 XML 관점에서 질의를 하고 결과를 얻는다.



(그림 2) 랩퍼 시스템의 질의 처리

XML 기반의 랩퍼 시스템은 지역 데이터베이스의 내용을 하나의 통일된 관점으로 표현하고 관리하며, 사용자 질의어를 지역 데이터베이스가 처리할 수 있는 질의어로 변환해야 한다. 이 중에서 본 논문은 XML 기반 랩퍼 시스템의 질의 처리에 관하여 논한다. XML 기반 랩퍼 시스템의 질의 처리는 두 가지 기능을 가지고 있어야 한다. 첫째, 미디에이터에서 받은 XML 질의어를 지역 데이터베이스에 실행시킬 수 있는 질의어로 변환하는 기능을 가지고 있어야 한다. 둘째, 변환된 질의어를 지역 데이터베이스에 실행시켜 나온 결과를 XML 문서로 변환하는 기능을 가지고 있어야 한다. 본 논문에서는 XML 질의어로 현재 W3C에 워킹 드래프트(Working Draft)로 제안되어 있는 XQuery 1.0[4, 5, 16]를 사용하고 지역 데이터베이스는 관계형 데이터베이스를 대상으로 한다. 따라서, 본 논문은 (그림 2)에서 XQuery를 관계형 데이터베이스에서 실행시킬 수 있는 질의어인 SQL로 변환[1, 2, 12, 13]하는 방법과 SQL을 실행시켜 나온 결과 테이블을 XML 문서로 변환하는 것에 대한 랩퍼 시스템의 질의 처리를 연구한다.

사용자 질의어인 XQuery를 SQL로 변환할 때 사용자 질의어만으로는 SQL로 변환할 수 있는 정보를 추출할 수가 없다. 또한 사용자 질의어만으로는 어떤 형태의 XML 문서 결과를 원하는지 알 수 없다. 이것은 사용자 질의가 무엇을 기반으로 만들어지는지 살펴보면 그 이유를 찾을 수 있는데, 그 전에 다음과 같은 정의가 필요하다. 관계형 데이터베이스의 스키마를 XML 뷰로 표현한 것을 '기본 XML 스키마'라고 정의한다. 따라서 사용자는 기본 XML 스키마를 기반으로 데이터베이스 내용을 파악한다. 그런데, 관계형 데이터베이스에 대하여 사용자가 뷰 테이블을 정의할 수 있듯이, 기본 XML 스키마에 대해서도 사용자가 새로운 XML 뷰를 정의할 수 있다. 이를 '응용 XML 뷰'라고 정의한다. 이 때, 사용자는 응용 XML 뷰를 기반으로 질의를 한다. 따라서 SQL로 변환하고 XML 문서를 생성하기 위해서는 응용 XML 뷰와 사용자 질의어의 합성 과정이 필요하다. 합성의 필요성과 원리는 3장에서 다시 자세히 살펴본다. 본 논문은 질의 처리를 위하여 사용자 질의어와 응용 XML 뷰에 대한 새로운 모델을 제시하고, 제시된 모델을 바탕으로 합성하여 XQuery를 SQL로 변환하는 알고리즘과 질의 결과를 XML 문서로 변환하는 알고리즘을 제시한다.

논문의 구성은 다음과 같다. 2장에서는 XML 뷰 기반으로 질의 처리가 이루어지는 기존의 연구들에 대하여 알아본다. 3장에서는 질의 처리가 이루어지는 기본 원리를 설명하고, 4장에서는 질의 처리를 위한 새로운 모델을 제시한 다음 합성과 SQL 변환 알고리즘에 대하여 설명한다. 5장에서는 실제 질의 처리 과정을 예를 들어 살펴본다. 6장에서는 질의 트리를 순회함으로써 XML 문서가 생성됨을 보인다. 7장에서 구현 결과 및 분석을 제시하고, 마지막으로 8장에서 결론 및 향후 연구 과제에 대해서 논한다.

2. 관련 연구

본 논문과 유사하게 XML 뷰 기반으로 질의 처리가 이루어지는 시스템의 대표적인 것으로 SilkRoute 시스템과 XPERANTO 시스템이 있다.

우선 SilkRoute 시스템[1, 3]은 관계형 데이터베이스 환경에서 설계된 미들웨어 시스템이다. 이는 XML 뷰를 처리하기 위해서 자체적으로 정의한 선언적인 질의어인 RXL(Relational to XML)을 사용하여 XML 뷰를 기술한다. RXL은

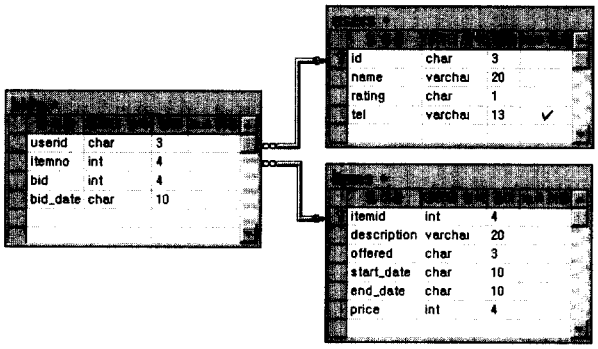
SQL(from과 where 절) 부분과 XML-QL (construct 절)의 조합으로 이루어진 질의어이다. 이는 블록 구조, 중첩된 질의, 그리고 Skolem 함수를 지원하는 특징을 가지고 있다. SilkRoute 시스템에서 질의어 관리는 먼저 약속된 DTD와 관계 스키마를 참조하여 RXL로 작성된 XML 뷰가 정의된다. RXL 뷰 질의어와 사용자 질의어인 XML-QL과 합성되어 새로운 RXL이 생성된다. 그리고 합성된 RXL은 질의 변환 과정을 통해 SQL로 변환되어 관계형 데이터베이스에 접근하게 된다. 그런데, SQL로 변환하기 전에 합성된 결과를 최적화 하는 단계가 필요한데, 이는 어려운 문제로 남아 있어 변환된 SQL 문장에 불필요한 내용이 존재한다.

XPERANTO 시스템[2, 6, 14] 역시 관계형 데이터베이스 환경에서 설계된 미들웨어 시스템이다. 관계형 데이터를 XML 뷰로 정의하기 위해 XQuery를 사용하고 사용자 질의 또한 XQuery이다. 사용자 질의가 들어오면 먼저 파싱하여 XQGM (XML Query Graph Model)이라 불리는 중간 질의 표현으로 변환된다. XQGM은 QGM(Query Graph Model)이라 불리는 SQL 중간 질의 표현의 확장으로 볼 수 있고, XML 질의 대수의 개념이 추가되어 설계되었다. XML 뷰를 정의한 XQuery와 사용자 질의 XQeury는 각각 XQGM으로 변환되어 합성하게 된다. 합성된 결과도 중간 모델인 XQGM이고 이것은 SQL로 변환되는 부분과 XML 문서를 생성하기 위한 부분으로 분리되어 각각 실행된다. XPERANTO는 그래프 모델 기반의 질의 처리 시스템인데 반해 본 논문에서는 XML 뷰 트리 기반의 질의 처리 모델을 제시한다.

3. 질의 처리 기본 원리

3.1 응용 XML 뷰 기반의 사용자 질의

기본 XML 스키마를 정의하는 방법은 다양하게 있을 수 있는데, 본 논문에서는 XPERANTO[2, 14]에서 제시된 표현에 따른다. 이 절에서는 그 개념에 대하여 간단히 설명한다. (그림 4)는 관계형 데이터베이스에 (그림 3)과 같이 users, bids 그리고 items라는 세 개의 테이블로 구성되어 있을 때, 이를 XML 스키마로 표현한 것이다.



(그림 3) 관계형 데이터베이스 스키마

db 엘리먼트의 하위에 나타난 users, bids, 그리고 items 엘리먼트는 실제 데이터베이스의 테이블을 나타낸다. users 하위의 id, name, rating 그리고 tel 엘리먼트는 users 테이블의 실제 컬럼을 나타낸다. 또한 users 엘리먼트와 id 엘리먼트 사이에 tuple 엘리먼트를 두어서 실제 테이블상에서 튜플의 반복을 나타낸다. 즉, 관계형 데이터베이스의 스키마를 최하위 수준으로 표현한 기본 XML 스키마에서 실제 관계형 데이터베이스의 테이블 명과 컬럼 명을 알 수 있다.

```
...
<xsd : element name = "db">
  <xsd : complexType>
    <xsd : sequence>
      <xsd : element name = "users">
        <xsd : complexType>
          <xsd : sequence>
            <xsd : element name = "tuple" minOccurs = "0"
                           maxOccurs = "unbounded">
              <xsd : complexType>
                <xsd : sequence>
                  <xsd : element name = "id" ···/>
                  <xsd : element name = "name" ··· />
                  <xsd : element name = "rating" ··· />
                  <xsd : element name = "tel" ··· />
                </xsd : sequence>
              </xsd : complexType>
            </xsd : element>
          </xsd : sequence>
        </xsd : complexType>
      </xsd : element>
      <xsd : element name = "bids" ··· > ··· </xsd : element>
      <xsd : element name = "items" ··· > ··· </xsd : element>
    </xsd : sequence>
  </xsd : complexType>
</xsd : element>
...
```

(그림 4) 기본 XML 스키마

이 기본 XML 스키마를 기반으로 사용자가 원하는 관점으로 새롭게 XML 뷰를 정의할 수 있는데, 이를 '응용 XML 뷰'라고 정의하였다. 응용 XML 뷰를 표현하기 위한 정의어로서 XQuery를 사용한다. (그림 5)는 (그림 4)의 기본 XML 스키마를 기반으로 새롭게 정의한 응용 XML 뷰의 예이며, 이러한 응용 XML 뷰를 기반으로 사용자 질의가 이루어진다.

```
<Auction>
 FOR $x IN view("db")/users/tuple
 RETURN
      <Users>
       <Name ID = {$x/id/text()}> $x/name/text() </Name>
       { FOR $y IN view("db")/bids/tuple
         WHERE $y/userid = $x/id
         RETURN
       <Bids>
            <ItemNum> $y/itemno/text() </ItemNum>
            <Bid> $y/bid/text() </Bid>
       </Bids>          }
       <Rating> $x/rating/text() </Rating>
```

```
    </Users>
</Auction>
```

(그림 5) 응용 XML 뷰

3.2 질의어와 뷰의 합성

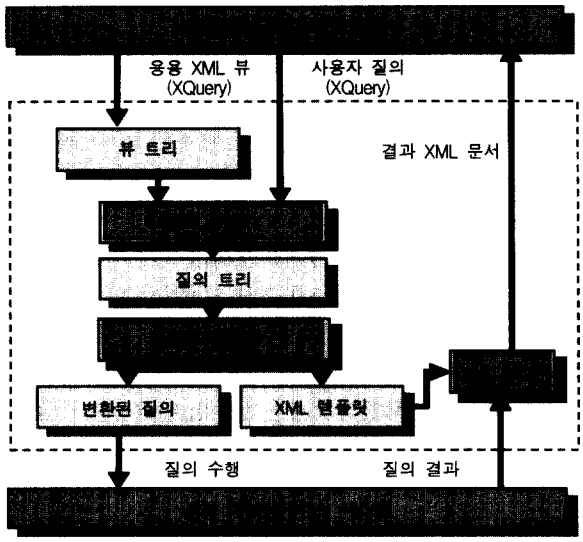
사용자 질의는 응용 XML 뷰를 기반으로 이루어지기 때문에 SQL로 변환하고, XML 문서를 생성하기 위한 템플릿을 구성하기 위해서는 응용 XML 뷰에서 정보를 얻어와야 한다. 즉, 응용 XML 뷰 기반의 사용자 질의어는 응용 XML 뷰와의 합성을 통해서 SQL로 변환 가능한 정보를 얻을 수 있는 기본 XML 스키마에 대한 질의어로 변경시킬 수 있다. 응용 XML 뷰에 나타난 경로식은 기본 XML 스키마의 엘리먼트와 대응되므로 응용 XML 뷰의 경로식에서 실제 관계형 데이터베이스의 테이블 명과 컬럼 명을 추출할 수 있다. 즉, tuple 엘리먼트를 기준으로 tuple의 부모 엘리먼트가 테이블 명이 되고 tuple의 자식 엘리먼트가 컬럼 명이 되어 SQL로 변환할 수 있게 된다. 또한, 응용 XML 뷰에 나타난 조건절은 사용자 질의어와 합성하여 SQL 문장을 생성할 때 적용되어야 하는 조건 정보가 된다. 그리고, 사용자 질의 결과는 응용 XML 뷰의 엘리먼트 구조의 영향을 받은 형태로 출력되어야 한다. 결국, 질의 처리를 위해서 사용자 질의어와 응용 XML 뷰의 합성이 필요하며, 합성 결과에서 SQL로 변환할 수 있는 정보를 추출할 수 있으며, SQL 질의어를 실행시켜 나온 결과 테이블을 어떤 형태의 XML 문서로 출력할 것인지에 대한 템플릿을 구성할 수 있다.

4. 질의 처리 모델

본 장에서는 질의 처리의 개략적인 모습과 세부 알고리즘을 제시한다.

4.1 질의 처리의 전체적인 구성

(그림 6)은 본 논문에서 제시하는 질의 처리에 대한 구성도이다. 먼저 미디에이터나 응용 프로그램에서는 지역 데이터베이스의 내용을 XML 문서의 관점에서 볼 수 있는 뷰가 필요하다. 사용자는 XQuery를 이용하여 이를 정의하는데, 이것을 응용 XML 뷰라 한다. 응용 XML 뷰를 바탕으로 XQuery로 정의된 사용자 질의는 질의 합성기에서 응용 XML 뷰 트리와 합성되고, 그 결과로 질의 트리가 구성되어진다. 합성된 질의는 기본 XML 스키마 상에서의 질의가 된다. 질의 변환기는 질의 트리로부터 지역 데이터베이스 시스템에서 수행하게 될 질의어를 생성한다. 변환된 질의어 즉, 관계형 데이터베이스의 경우, SQL이 지역 데이터베이스에서 수행되어 질의 결과를 반환하게 되고 반환된 질의 결과는 XML 태거에서 XML 문서로 변환된다. 이때 질의 결과를 XML 문서로 변환하기 위한 틀의 역할을 하는 템플릿을 이용하게 된다. 변환된 XML 문서는 질의 결과로서 사용자에게 전달된다.



(그림 6) 질의 처리 구성도

본 논문에서는 (그림 6)의 질의 트리가 합성 결과이고 SQL로 변환된 결과이면서 동시에 XML 문서를 생성하기 위한 템플릿이 됨을 보인다.

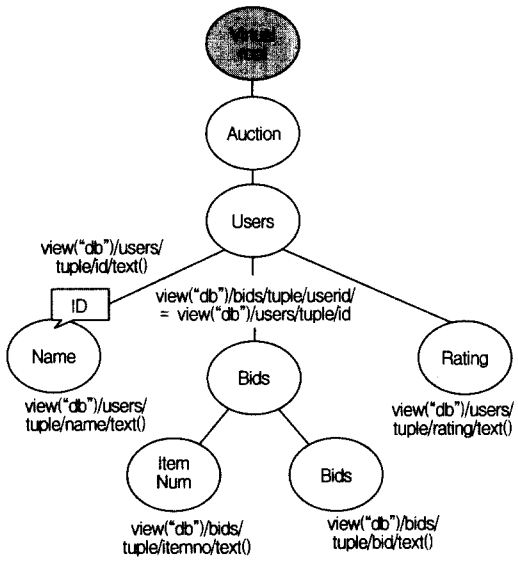
4.2 뷰 트리

4.2.1 뷰 트리의 정의

뷰 트리는 XQuery로 표현된 응용 XML 뷰를 트리 형태로 나타낸 것이다. 응용 XML 뷰에 나타난 모든 엘리먼트 구조대로 트리의 각 노드가 구성되는데 XQuery는 항상 루트 엘리먼트가 존재한다는 보장이 없기 때문에 가상 루트를 만든다. 나머지 응용 XML 뷰에 나타난 표현식들은 뷰 트리 노드 필드의 값으로 기억되어 그 정보가 유지된다.

4.2.2 뷰 트리의 구성 예

(그림 7)은 응용 XML 뷰인 (그림 5)에 대한 뷰 트리이다. 가상 루트를 만들고 응용 XML 뷰에 나타난 엘리먼트 구조대로 가상 루트 하위에 노드가 추가된다. 응용 XML 뷰의 엘리먼트 내용은 해당 노드의 '값' 필드에 기억된다. 예를 들어 Name 엘리먼트는 $x/name/text()와 같은 내용을 가지는데, 변수는 값으로 대체하여 저장한다. 즉, Name 노드의 값 필드에는 view("db")/users/tuple/name/text()가 들어간다. 트리에서 Name 노드 옆의 사각형은 애트리뷰트를 의미하고 본 논문에서는 애트리뷰트를 엘리먼트 노드의 필드로 취급한다. Users 노드와 Bids 노드 사이의 링크에 있는 레이블은 그 하위의 노드에 적용되어야 하는 조건을 의미한다. 이러한 뷰 트리는 응용 XML 뷰의 모든 정보를 트리로 유지하여 사용자 질의어와 합성된다.



(그림 7) 뷰 트리 1

4.2.3 뷰 트리 노드의 구조

〈표 1〉 뷰 트리 노드의 구조

필 드	내 용
이 름	엘리먼트 이름
타 입	비단말노드, 단말노드(텍스트), 단말노드(빈태그)
부모 노드	현재 노드의 부모 노드 포인터
첫 번째 자식 노드	현재 노드의 첫 번째 자식 노드 포인터
다음 형제 노드	현재 노드의 다음 형제 노드 포인터
값	엘리먼트 내용
조 건	XQuery의 WHERE절 내용
애트리뷰트	애트리뷰트 이름, 애트리뷰트 값

<표 1>은 뷰 트리 노드의 구조를 보여준다. '이름' 필드는 응용 XML 뷰에 나타나는 엘리먼트의 이름을 의미한다. '부모 노드' 필드, '첫 번째 자식 노드' 필드, 그리고 '다음 형제 노드' 필드는 현재 엘리먼트를 기준으로 부모 엘리먼트, 첫 번째 자식 엘리먼트, 그리고 다음 형제 엘리먼트를 의미하며 각각 해당 노드의 포인터 값을 가진다. '타입' 필드는 트리를 구성하는 각 노드가 단말 노드인지 비 단말 노드인지를 구별하는데 세 가지로 분류한다. 비 단말 노드, 값을 가지는 단말 노드, 그리고 값을 가지지 않는 단말 노드로 나눈다. '값' 필드는 단말 노드이면서 엘리먼트 내용이 있는 경우에 값을 가지게 되는 필드이다. 다음 '조건' 필드는 해당 노드가 출력되기 위한 조건으로 응용 XML 뷰상에서 WHERE 절의 내용이 들어간다. 마지막으로 '애트리뷰트' 필드는 엘리먼트에 애트리뷰트가 존재할 경우 값이 들어간다. 여기서 값은 애리트뷰트 이름과 애트리뷰트 값을 의미한다. 이러한 필드로 구성된 노드를 가지는 뷰 트리는 다음과 같은 역할을 한다.

첫째 : 뷰 트리는 실제 데이터베이스의 테이블 명과 컬럼 명을 가지고 있기 때문에 SQL 문장으로 변환할 수 있는 정보를 얻을 수 있다.

둘째 : 사용자 질의어에 나타나는 임의의 경로식에 적용되어야 하는 조건 정보를 파악할 수 있다.

셋째 : 사용자 질의어의 경로식에 내재된 구조적인 정보를 뷰 트리 상에서 쉽게 알 수 있다.

4.3 합성과 변환

사용자 질의어가 들어오면 이를 실제 데이터베이스에 실행시킬 수 있는 질의어로 변환하고 변환된 질의어를 실행시켜 그 결과를 XML 문서로 출력해야 한다. 이를 위해서는 사용자 질의어와 응용 XML 뷰를 합성하는 문제와 SQL로 변환하는 문제 그리고 XML 문서를 생성하기 위한 XML 템플릿을 구성하는 문제를 모두 고려한 질의 처리 방법을 모색해야 한다.

4.3.1 질의 트리의 정의

사용자 질의어를 뷰 트리와 합성하여 만든 트리를 질의 트리라 한다. 질의 트리는 사용자 질의어와 뷰 트리를 합성하는 문제와 SQL로 변환하는 문제 그리고 템플릿을 이용하여 XML 문서를 생성하는 문제를 모두 고려한 모델이다. 질의 트리가 구성되면 트리 내에 변환된 SQL 문장이 존재하고 질의 트리 자체가 템플릿이 되어 XML 문서를 생성할 수 있다.

4.3.2 XQuery 1.0 문법 제한

사용자 질의어인 XQuery를 SQL로 변환하기 위해서는 XQuery의 각 표현식이 SQL의 어떤 문법과 유사한 의미를 가지는지 알아야 한다. 하지만, XQuery는 계층적인 구조를 가지는 XML 문서에 대한 질의어이고 SQL은 이차원 구조를 가진 테이블에 대한 질의어이기 때문에 두 질의어는 근본적인 차이가 있다. 이러한 차이로 인해 변환할 수 없거나, 변환 자체가 큰 의미가 없는 XQuery 문법은 본 논문에서 제한을 두고 질의 처리를 하였다. XQuery 기능 중에 순서와 관련된 연산자, 데이터 타입 관련 표현식, 이름공간, 사용자 정의 함수 등은 지원하지 않는다.

4.3.3 질의 트리 노드의 구조

질의 트리를 구성하는 노드형으로는 가상 루트 노드, XQueryElement 노드 그리고 XQuerySQL 노드가 있다. 가상 루트 노드는 뷰 트리와 마찬가지로 모든 노드의 루트가 되는 가상적인 노드이다.

XQueryElement 노드는 사용자 질의어에 나타난 엘리먼트와 뷰 트리와의 합성으로 인해 뷰 트리에서 추가되는 엘리먼트에 해당한다. 이 노드는 XML 문서를 생성하기 위한 템플릿 역할을 하여 질의 결과인 XML 문서에 태그로 나타

난다. XQueryElement 노드는 <표 2>와 같은 필드로 구성된다.

〈표 2〉 XQueryElement 노드의 구조

필 드	내 용
이름	엘리먼트 이름
타입	1 (XQueryElement)
부모 노드	현재 노드의 부모 노드 포인터
첫 번째 자식 노드	현재 노드의 첫 번째 자식 노드 포인터
다음 형제 노드	현재 노드의 다음 형제 노드 포인터
값	엘리먼트 내용
위치	SQL의 SELECT절에서 컬럼 위치
애트리뷰트	애트리뷰트 이름, 애트리뷰트 값

XQueryElement 노드의 대부분의 필드는 뷰 트리의 노드에서 설명하였다. '위치' 필드는 SQL의 SELECT 절에 있는 컬럼 위치를 말하는데, XQueryElement 노드가 태거화(tagging)될 때 결과 테이블에서 해당 위치의 컬럼에 있는 값을 가져와 삽입한다.

〈표 3〉 XQuerySQL 노드의 구조

필 드	내 용
이 름	없다.
타 입	2 (XQuerySQL)
부모 노드	현재 노드의 부모 노드 포인터
첫 번째 자식 노드	현재 노드의 첫 번째 자식 노드 포인터
다음 형제 노드	현재 노드의 다음 형제 노드 포인터
SELECT 절	SQL의 SELECT절 내용
FROM 절	SQL의 FROM절 내용
WHERE 절	SQL의 WHERE절 내용
ORDER BY 절	SQL의 ORDER BY절 내용

XQuerySQL 노드는 변환된 SQL 문장이 기억되는 노드이다. 본 논문은 XQuery에서 하나의 FLWR (FOR-LET-WHERE-RETURN) 절에 대하여 하나의 SQL 문장을 생성하기 때문에 사용자 질의어에 나타나는 FLWR 절에 대하여 XQuerySQL 노드를 생성시켜 질의 트리에 추가한다. FLWR 절에서 SQL 문장으로 변환된 내용은 기본적으로 해당 XQuerySQL 노드에 추가된다. 단, FLWR 절에서 사용된 변수가 다른 FLWR 절에서 선언된 변수일 경우는 변환된 SQL 내용의 일부가 다른 XQuerySQL 노드 즉, 그 변수가 선언된 FLWR 절에 대응되는 XQuerySQL 노드에 추가된다. 그리고 XQuerySQL 노드는 기억되어 있는 SQL 문장을 실행시켜 나온 결과 테이블의 튜플 수만큼 하위 엘리먼트 노드들이 반복됨을 의미한다. XQuerySQL 노드는 <표 3>과 같은 필드로 구성되며, 하위 네 개 필드에 변환된 SQL 문장이 추가된다.

4.3.4 합성과 SQL 변환 알고리즘

```
void userXQueryProcessing(parsedUserQuery, viewTree) {
  queryTree = createXQueryNodeObj() ;
  makeVirtualRoot(queryTree) ;
  while(!endOfQuery) {
      token = getNextToken(parsedUserQuery) ;
      if(token == startTag) {
          q = createXQueryElementObj() ;
          append(queryTree, q) ;
          while(exist attributes) process attribute ;
      }
      else if(token == startOfFLWR) {
          sql = createXQuerySQLObj() ;
          append(queryTree, sql) ;
          token = getNextToken() ;
          do {
              if(token == "FOR" or "LET") {
                  (varName, varValue) = getVariable() ;
                  path = getPath(viewTree, varValue) ;
                  sql.setVar(varname, path) ;
                  process predicates ;
              }
              else if(token == "WHERE") {
                  do {
                      condition = getCondition() ;
                      sqlCondition = translateSQL(condition) ;
                      sql.setWhereClause(sqlCondition) ;
                  } until(endOfWhere)
              }
          } until(token == "RETURN")
      }
      else if(token == pathExpr) {
          targetNode = getNodePtr(viewTree, token) ;
          temp = makeQuerySubTree(targetNode) ;
          append(queryTree, temp) ;
          if(token == startOFVar) process VarPattern(() ;
          sqlcomp[] = convertSQLComponents(temp) ;
          sql.setSqlStatementObj(sqlcomp) ;
      }
      else if(token == endOfFLWR) {
          viewcond = getViewCondition() ;
          sqlcomp[] = convertSQLComponents(viewcond) ;
          sql.setWhereClause(sqlcomp) ;
      }
      else if(token == "SORTBY") {
          sqlsortby = translateSQL(sortbyClause) ;
          sql.setOrderByClause(sqlsortby) ;
      }
  }
}
```

(그림 8) 합성 및 변환 알고리즘

위의 알고리즘은 합성과 SQL로 변환하는 즉, 질의 트리를 생성하는 것으로 두 개의 인자를 받아 처리된다. 첫 번째 인자는 XQuery 파서로 파싱된 사용자 질의어이다. 두 번째 인자는 응용 XML 뷰에 대한 뷰 트리이다.

질의 트리 생성의 첫 단계로써 질의 트리에 대한 가상 루트 노드를 생성한다. 사용자 질의어에서 시작 태그를 만나면, XQueryElement형 객체를 생성하여 질의 트리에 추가한다. 그리고 시작 태그에 애트리뷰트가 존재하면 이를 처리한다.

다음으로 사용자 질의어에서 FLWR 절을 만나면 SQL 문장을 생성하기 위한 준비를 하고, FOR나 LET 절에서

제시된 변수나 조건을 처리한다. 이를 위하여 먼저 SQL로 변환하기 위한 정보를 모아둘 노드 구조인 XQuerySQL형 객체를 생성하여, 이를 질의 트리에 추가한다. FOR나 LET 절은 선언된 변수에 대한 값 즉, 경로식에 해당하는 목적 노드(Target Node)를 뷰 트리에서 찾아 그 노드의 위치를 XQuerySQL형 객체의 해시(Hash) 테이블에 기억시킨다. 그리고 WHERE 절을 만나면 조건에 대응하는 SQL 문의 일부를 생성시켜서, XQuerySQL형 객체에 기억시킨다.

사용자 질의어에서 경로식에 대한 참조가 발생하면, 뷰 트리에서 경로식에 해당하는 노드를 찾아서, 이에 대응하는 XQueryElement형 객체를 생성하여 질의 트리에 추가한다. 그리고 경로식에 해당하는 SQL 문장 요소를 XQuerySQL형 객체에 저장한다. 이 때, 경로식이 변수로 시작될 경우, 이 변수가 어디에서 선언되었는지 확인한 후 경로식으로 인해 변환되는 SQL 문장 요소는 그 변수가 선언된 XQuerySQL형 객체에 기억되는데, 자세한 내용은 다음 5장에서 예를 통해 살펴본다.

FLWR 절의 마지막에 도달했을 때는 뷰에서 전달되어온 조건들에 대한 SQL 문장 요소를 생성시켜서 XQuerySQL형 객체에 저장한다. 그리고, 사용자 질의어에서 결과를 정렬시키는 "SORTBY"가 있을 경우는 SQL의 "ORDER BY"로 변환된다.

5. 질의 처리 예

질의 처리의 예를 보이기 위하여, 자신의 FLWR 절에서 정의된 변수만 사용하는 경우와 자신의 FLWR 절에서 정의되지 않은 변수를 사용하는 경우로 나누어 설명한다.

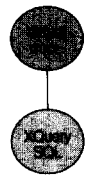
5.1 지역 변수만 사용된 경우

(그림 9)은 (그림 5)의 응용 XML 뷰를 기반으로 "등급이 A보다 큰 사람의 이름과 경매 내역을 검색"하는 사용자 질의어이다.

```
FOR $u IN view("auction")//Users
WHERE $u/Rating > "A"
RETURN
   <User>
      <Name> $u/Name/text( ) </Name>
      { $u/Bids }
   </User>
SORTBY(Name)
```

(그림 9) 사용자 질의어 예 1

(그림 5)의 응용 XML 뷰에 대한 사용자 질의어이므로 (그림 7)의 뷰 트리와 합성하여 어떻게 질의 트리가 생성되는지 과정을 살펴본다. 먼저 가상 루트를 만들고 처음에 FLWR 절로 시작되므로 (그림 10)와 같이 XQuerySQL 노드를 생성하여 질의 트리에 추가한다.



(그림 10) 질의 트리 생성 과정 I

FLWR 절에 FOR 절이 선언되어 있으므로 해당 변수를 처리하여 해시 테이블에 저장한다. 변수 이름인 $u가 해시 테이블의 키가 된다. $u에 바인딩되는 경로식 view ("auction")//user는 뷰 트리에서 해당 목적지인 Users 노드의 포인터를 해시 테이블의 값으로 저장한다. 사용자 질의어에 나타나는 모든 경로식은 응용 XML 뷰상의 엘리먼트를 가리키므로 뷰 트리에서 해당 경로식의 위치를 찾아 목적 노드에 대한 질의 처리를 한다. 다음으로 사용자 질의어의 WHERE 절에 있는 $u/Rating > "A"를 SQL의 WHERE 절로 변환하기 위해서는, $u/Rating이 실제 데이터베이스의 어떤 테이블의 컬럼인지 알아야 한다. $u/Rating은 경로식이므로 뷰 트리에서 해당 경로를 따라가 목적 노드를 찾는다. 이 때, 경로식이 변수로 시작되는데, $u는 지역 변수이므로 고려해야될 사항은 없다. $u/Rating은 $u에 대한 포인터인 Users 노드가 뷰 트리 상에서 경로의 시작점이 되고 Users 노드의 자식인 Rating 노드를 가리킨다. Rating 노드의 값 필드에서 테이블 명과 컬럼 명을 추출할 수 있다. 목적 노드인 Rating 노드에 대한 정보는 <표 4>와 같다.

<표 4> Rating 노드의 구조 (뷰 트리)

Rating 노드 필드	내 용
이름	Rating
타입	단말노드(텍스트)
부모 노드	Users 노드 포인터
첫 번째 자식노드	없다.
다음 형제 노드	없다.
값	view("base")/**users**/tuple/**rating**/text()
조건	없다.
애트리뷰트	없다.

<표 4>의 값 필드에서 tuple을 기준으로 users가 테이블 명이 되고 rating이 컬럼 명이 된다. 따라서 <표 5>와 같이 XQuerySQL 노드의 WHERE 절 필드에 SQL 문장이 추가된다.

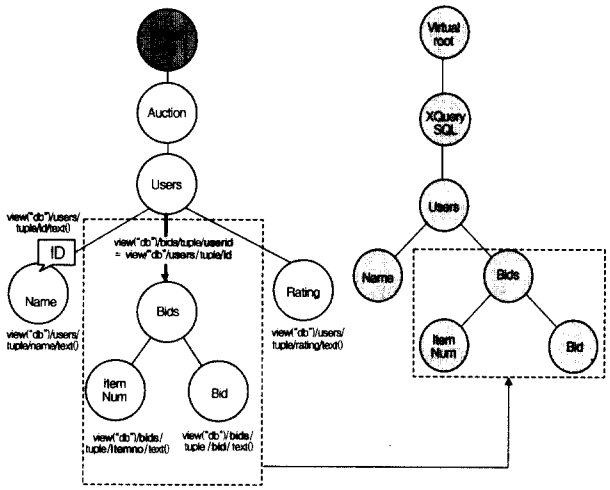
<표 5> XQuerySQL 노드의 내용 I

XQuerySQL 노드 필드	내 용
SELECT 절	없다.
FROM 절	**users**
WHERE 절	**users.rating > 'A'**
OrderBy 절	없다.

RETURN 절에 User 엘리먼트가 나타나므로 User 노드(XQueryElement)를 추가하고 다음 Name 엘리먼트에 대해서 Name 노드를 질의 트리에 추가한다.

Name 엘리먼트 내용에는 이름을 검색하는 경로식이 있다. 이 경로식에 해당하는 뷰 트리의 목적지인 Name 노드의 값 필드에서 테이블 명과 컬럼 명을 추출하여 XQuerySQL 노드의 SELECT 절 필드에 users.name을 추가한다.

경로식 $u/Bids는 뷰 트리에서 Bids 노드를 가리킨다. Bids 노드는 하위에 ItemNum 노드와 Bid 노드가 있으므로, 이 구조대로 질의 트리에 추가된다. 그리고 ItemNum과 Bid 노드로 인해 변환된 SQL 문장 요소를 XQuerySQL 노드에 기억시킨다. 또한 뷰 트리에 조건 정보가 있으므로 질의 트리에 적용되어야 한다. 즉, SQL 문장의 WHERE 절에 추가되어야 한다. (그림 11)에서 포인터가 Bids 노드를 가리키고 Bids 노드를 포함한 하위 노드들이 오른쪽의 질의 트리에 추가된다.



(그림 11) 질의 트리 생성 과정 2

사용자 질의어에서 $u/Bids에 의해 생성되는 SQL 문장은 <표 6>과 같다. 사용자 질의의 마지막에 SORTBY(Name) 절이 있다. 이는 Name을 기준으로 오름차순 정렬하는 것이므로, SQL의 ORDER BY users.name으로 변환한다.

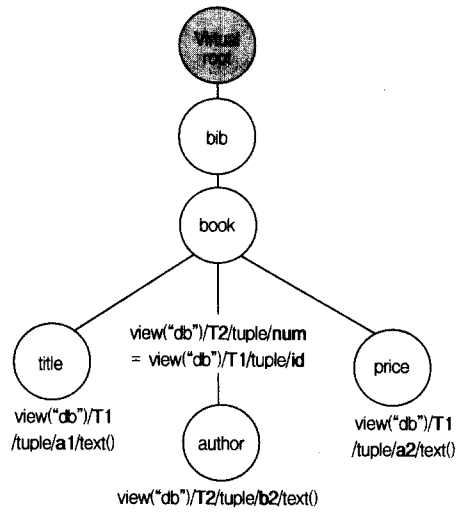
〈표 6〉 XQuerySQL 노드의 내용 2

XQuerySQL 노드 필드	내용
SELECT 절	users.name, **bids.itemno, bids.bid**
FROM 절	users, **bids**
WHERE 절	users.rating > 'A' **AND bids.userid = users.id**
OrderBy 절	**users.name**

(그림 11)의 오른쪽 그림이 (그림 9)의 사용자 질의어에 대한 완성된 질의 트리이고, 가상 루트의 아래에 있는 XQuerySQL 노드에 SQL 문장이 기억되어 있다.

5.2 비 지역 변수가 사용된 경우

여기서는, 다른 FLWR 절에서 선언된 변수를 사용하는 중첩된 FLWR 절을 SQL로 변환하는 예를 보인다. (그림 13)에 제시된 사용자 질의어에서 변수 $b의 사용이 그 예이고, (그림 12)의 뷰 트리에 대한 질의이다.



(그림 12) 뷰 트리 2

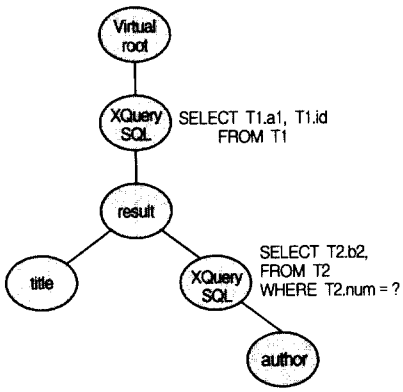
FLWR 절에서 사용된 변수가 다른 FLWR 절에 선언된 변수일 경우 그 변수에 의해 생성되는 SQL 문장 요소는 정적으로 결정될 수 없고, 그 이전에 실행된 임의의 SQL 문장에 대한 결과 값을 이용해서 동적으로 SQL 문장이 구성된다. 이를 SQL로 표현하기 위하여, 그 값을 필요로 하는 임의의 위치에 '?'를 둔다. 이는 JDBC의 PreparedStatement 객체를 이용해서 구현된다.

우선 그 변수가 어떤 테이블과 관련이 있는지 찾아내어, 생성된 SQL 문장에서 변수와 관련된 테이블에 해당하는 모든 컬럼들을 '?'로 바꾼다. 여기서 관련 있는 테이블이란 그 변수가 선언된 FLWR 절에 대한 SQL 문장에서 FROM 절에 선언된 테이블을 말한다. 그리고 '?'로 바뀌었던 SQL 내용은 그 변수가 선언된 곳의 FLWR 절에 대한 SQL 문장(SELECT 절)에 추가한다. 단, '?'가 SQL 문장의 SELECT 절에 추가되어야 할 경우는 그대로 SQL 내용을 넣고, 대신 WHERE 절에도 SQL 내용을 추가한 다음 '= ?'를 추가한다. 또한, 그 SQL 내용을 변수가 선언된 곳의 FLWR 절에 대한 SQL 문장(SELECT 절)에 추가한다. 구현시 SQL 문장의 SELECT 절에는 '?'를 사용할 수 없기 때문에 이와 같은 방법을 취한다.

변수 $b가 선언된 곳의 FROM 절에 T1이라는 테이블이 있으므로 $b와 관련 있는 테이블은 T1이 된다. 따라서 중첩된 FLWR 절에서 $b/author를 처리할 때 생성되는 SQL 문장 요소 T2.num = T1.id에서 T1.id 부분은 (그림 14)과 같이 '?'가 된다. 그리고 이 T1.id는 $b가 선언된 곳에 해당하는 SQL 문장의 SELECT 절에 추가된다.

```
FOR $b IN view("action")/bib/book
RETURN
        <result>
          { $b/title }
          { FOR $a IN $b/author
            RETURN $a      }
        </result>
```

(그림 13) 사용자 질의어 예 2

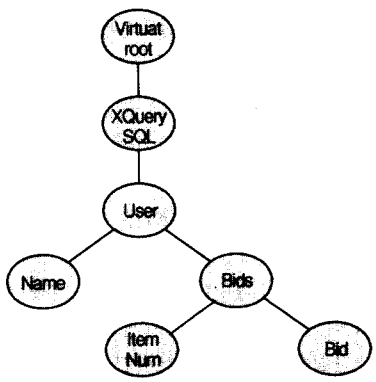


(그림 14) 질의 트리 1

6. XML 문서 생성

질의 트리는 사용자가 제시한 질의어의 결과에 대한 구조를 나타내는 것이기 때문에 그 자체가 XML 문서 생성을 위한 템플릿이 된다. 따라서 본 논문에서는 질의 트리를 순회함으로써 쉽게 XML 문서를 생성시킨다. 질의 트리의 XQueryElement 노드는 XML 문서로 구성되기 위한 엘리먼트와 애트리뷰트 정보를 담고 있다. 그리고 XQuerySQL 노드는 기억되어 있는 SQL 문장을 실행시켜 나온 결과 테이블의 값을 엘리먼트 내용(Element Content)으로 삽입하며, 결과 테이블의 튜플 수만큼 하위 트리 노드들을 반복 순회하여 XML 문서를 생성한다.

6.1 XML 문서의 생성



(그림 15) 질의 트리 2

(그림 15)의 질의 트리에 대한 XML 문서 생성 과정을 살펴본다. 가상 루트를 기준으로 깊이 우선 탐색을 하는데 처음에 XQuerySQL 노드를 만나므로 SQL 문장을 실행시킨다. SQL 문장을 실행시킨 결과 테이블이 <표 7>과 같을 때 다섯 개의 튜플이 있으므로 XQuerySQL 노드 하위의 노드들을 다섯 번 반복 순회한다.

〈표 7〉 SQL 실행 결과 테이블

name	itemno	bid
Tom	1	1200
Mary	3	10000
Jack	2	2550
Tony	3	2000
Jane	4	1400

6.1.1 질의 결과를 하나의 XML 문서로 생성

(그림 15)의 질의 트리에서 XQuerySQL 노드 다음 User 엘리먼트 노드가 나타나므로 <User> 시작 태그를 생성하고, User 엘리먼트의 자식으로 Name 엘리먼트 노드가 있으므로 <Name> 시작 태그를 생성한다. <표 8>에서 Name 엘리먼트 노드에 '위치' 필드 값이 존재하는데, 이는 해당 엘리먼트가 내용을 가지므로 결과 테이블에서 값을 가져와 XML 문서에 추가해야 함을 의미한다. Name 엘리먼트의 위치 필드 값이 1이므로 결과 테이블의 첫 번째 컬럼(name 컬럼)값 Tom을 가져와 삽입한다.

〈표 8〉 Name 노드의 구조(질의 트리)

Name 노드 필드	내 용
이 름	Name
타 입	1 (XQueryElement)
부모 노드	User 노드 포인터
첫 번째 자식 노드	없다.
다음 형제 노드	Bids 노드 포인터
값	view("db")/users/tuple/name/text()
위 치	1
애트리뷰트	없다.

계속해서 같은 방법으로 질의 트리를 깊이 우선 탐색하면 (그림 16)와 같이 하나의 XML 문서가 생성된다

```
<User><Name> Tom </Name>
   <Bids><ItemNum> 1 </ItemNum><Bid> 1200 </Bid></Bids></User>
<User><Name>Mary</Name>
   <Bids><ItemNum> 3 </ItemNum><Bid> 10000 </Bid></Bids></User>
<User><Name>Jack</Name>
   <Bids><ItemNum> 2 </ItemNum><Bid> 2550 </Bid></Bids></User>
<User><Name>Tony</Name>
   <Bids><ItemNum> 3 </ItemNum><Bid> 2000 </Bid></Bids></User>
<User><Name>Jane</Name>
   <Bids><ItemNum> 4 </ItemNum><Bid> 1400 </Bid></Bids></User>
```

(그림 16) 하나의 XML 문서로 생성된 질의 결과

6.1.2 커서를 이용한 XML 문서 생성

XML 문서를 생성할 때 (그림 16)와 같이 하나의 문서로 만들 수도 있지만 검색 결과의 크기가 큰 경우를 대비하여 데이터베이스의 커서(Cursor) 개념을 이용한다. 본 논문은 XQuery에서 최상위 수준의 FLWR 절을 커서 단위로 정의하여 아직은 한정된 범위에서만 지원을 하고 있다. 예를 들어 질의 트리 (그림 15)에 커서를 이용하여 XML 문서를 생성하면 User 엘리먼트 단위로 결과가 반환된다.

7. 구 현

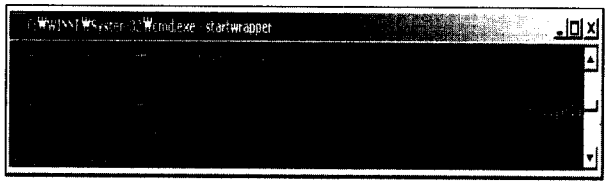
본 절에서는 본 논문에서 제안한 랩퍼 시스템의 질의 처리 모델을 구현한 내용을 제시한다.

7.1 구현 환경

랩퍼 시스템의 구현 환경은 다음과 같다. 운영체제는 Windows 2000이고, 데이터베이스는 MS-SQL Server 2000을 사용하였다. 구현 언어는 JDK 1.3을 사용하였고, MS-SQL 서버와 연동하기 위해서 SQL Server 2000 Driver for JDBC (Version 2.1)를 사용하였다. 그리고, XQuery 파서를 생성하기 위해서 JavaCC 2.0을 이용하였다.

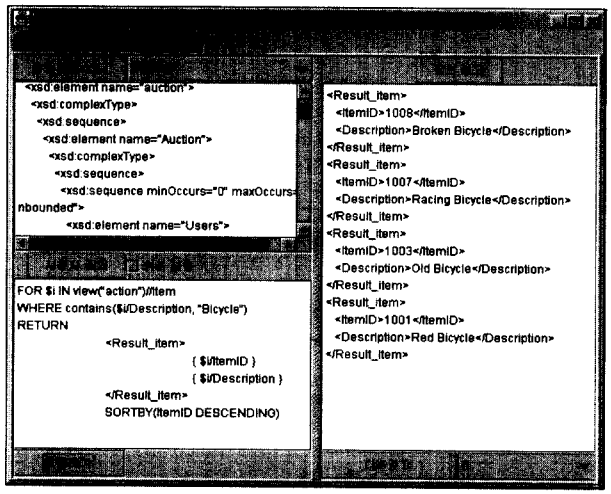
7.2 구현 결과

미디에이터-랩퍼 시스템은 클라이언트-서버 구조로 이루어지는데, 본 논문에서는 미디에이터와 랩퍼 사이의 통신을 위해서 자바 RMI를 이용한다. 랩퍼는 서버에 해당하므로 먼저 수행되어 있어야 한다. (그림 17)은 랩퍼를 실행시킨 화면이다.



(그림 17) 랩퍼 실행 화면

랩퍼가 정상적으로 동작하는지를 테스트하기 위해서 간단한 미디에이터 툴을 제작하였다. 미디에이터 툴은 랩퍼 시스템에 대한 클라이언트 프로그램으로서, 랩퍼를 통해 정해진 데이터베이스에 접근하여 기본 XML 스키마를 생성하고 사용자는 주어진 XML 스키마를 기반으로 응용 뷰를 생성, 수정, 삭제할 수 있으며, 사용자 질의를 수행하여 질의 결과를 얻을 수 있다. (그림 18)은 미디에이터 툴을 사용하여 질의를 수행한 결과 화면이다. (그림 18)의 왼쪽 상단은 응용 XML 뷰 스키마이며, 왼쪽 하단은 사용자 질의이다. 오른쪽은 사용자 질의를 SQL로 변환하여 DBMS로 보내어 수행시킨 후, 그 결과를 XML 문서로 변환한 결과이다.



(그림 18) 미디에이터 툴을 이용한 질의 수행 결과

7.3 평 가

랩퍼 시스템을 구현하여 작동시켜 봄으로써 본 논문에서 제안한 질의 처리 모델과 알고리즘이 정상적으로 수행됨을 확인하였다.

본 연구에서는 질의 처리를 하기 위하여 사용자 정의 뷰에 대해서는 뷰 트리를, 사용자 질의에 대해서는 질의 트리를 구성하였다. 사용자 정의 뷰와 질의를 트리로 표현하는 것은 XML과 유사한 구조를 가지므로 자연스러운 구성이 가능하다. 그리고, 뷰 트리는 사용자 질의의 경로식이 정확하게 드러나 있고, 합성에 필요한 정보와 질의 변환에 필요한 정보를 모두 유지하고 있기 때문에, 이 뷰 트리를 바탕으로 질의어에 따라서 질의 트리를 구성하는 과정이 합성하는 과정이면서, 동시에 XQuery가 SQL로 변환되는 과정이다. 즉, 합성과 질의 변환이 동시에 이루어진다. SilkRoute[1]와 XPERANTO[2]에서는 뷰와 사용자 질의를 합성한 후 합성된 결과를 가지고 질의 변환이 이루어지는데 비해 본 연구에서는 합성과 질의 변환이 동시에 일어나는 차이점이 있다. 이는 합성된 결과를 다시 읽어 들이는 부담을 줄일 수 있는 장점이 있다. 그리고, 질의 트리는 결과 XML 문서의 DOM 트리와 유사한 구조를 가지기 때문에 질의 결과를 XML 문서로 변환하기 위한 템플릿으로 활용된다. SilkRoute와 XPERANTO에서는 합성된 결과와는 별도로 템플릿을 유지하지만 본 논문의 질의 처리 모델에서는 그럴 필요가 없다.

질의 변환 결과와 결과 XML 문서를 생성시키는 방법에는 밀접한 관련이 있다. 합성된 XQuery 질의를 SQL로 변환하는 방법은 여러 가지가 있다[3]. 이것은 생성되는 SQL의 개수와 형태가 다양함을 의미하는데, 이는 결과 XML 문서를 생성하는 알고리즘에 영향을 미치게 된다. 본 논문의 질의 처리 모델은 6.1절에서 설명한 것처럼 결과 XML 문서 생성 알고리즘이 간단하다. 이것은 구현을 용이하게 하며, 주어진 질의의 정확한 의미를 반영한다. 그러나, 질의 트리를 깊이 우선 탐색하여 XML 문서를 생성할 때, XQuerySQL 노드를

만나면 SQL 문장을 실행시켜 반환된 결과 테이블의 튜플 수만큼 트리의 하위 노드를 반복 순회하여 XML 문서를 구성하는데, 하위 노드에 XQuerySQL 노드가 있다면 이 노드에 기억되어 있는 SQL 문장 또한 반복 실행된다. SQL 문장을 실행시키는 것은 데이터베이스의 접근을 의미하기 때문에 시스템에 부담을 주게 된다. 그러나, 커서 개념을 사용하여 SQL 문장이 한번에 다 실행되지 않고 사용자가 원하는 시점에 SQL 문장이 실행되도록 하는 방법을 사용하여 부담을 줄일 수 있다.

8. 결론 및 향후 연구 과제

본 논문은 관계형 데이터베이스에서 XML 뷰 기반의 랩퍼 시스템에 대한 질의 처리 모델을 설계, 구현하여 사용자 질의를 관계형 데이터베이스에서 실행될 수 있는 질의어로 변환하고, 최종 결과인 XML 문서를 얻었다. XQuery로 표현된 사용자 질의어를 SQL 질의어로 변환하기 위해서는 사용자 질의어와 응용 XML 뷰의 합성이 필요하다. 이를 위하여 본 논문은 XQuery로 작성된 응용 XML 뷰에 대한 트리 모델을 제시하였다. 이 뷰 트리는 구조적인 정보와 각 엘리먼트에 대한 조건 정보, 그리고 SQL로 변환할 수 있는 테이블 명과 컬럼 명에 대한 정보를 유지하여 사용자 질의어와 합성할 때 뷰 트리에서 효율적으로 정보를 얻을 수 있게 하였다. 사용자 질의어와 뷰 트리를 합성하여 질의 트리를 구성하였는데, 이는 합성 결과인 동시에 질의 트리 내에 변환된 SQL 문장이 기억되어 있으며, 질의 트리 자체가 XML 문서를 생성할 수 있는 템플릿 역할을 하는 모델이다. 즉, 본 논문에서 제시한 질의 트리는 합성하는 것과 SQL로 변환하는 것, 그리고 XML 문서를 생성하는 것에 대한 방법을 모두 고려한 결과물이다. 또한, 질의 트리를 순회하여 사용자가 원하는 정확한 형태의 XML 문서를 생성하고 검색 결과가 클 경우를 대비하여 데이터베이스의 커서 개념을 도입하여 사용하였다. 설계된 질의 처리 모델의 환경은 DBMS는 Microsoft SQL Server 2000을 사용하고 언어는 자바를 사용하여 구현하였다.

XQuery 질의로부터 SQL로 변환된 결과는 여러 가지 형태로 나올 수 있는데, 그 결과에 따라서 질의 처리의 효율성에 큰 영향을 미친다. 향후 과제로 질의어 최적화에 대한 연구가 더 이루어져야 한다. 그리고, XQuery의 모든 기능을 SQL로 변환하는데는 한계를 가진다. 본 논문에서도 XQuery의 제한된 범위에서만 SQL로 변환되도록 하였다. 앞으로 XQuery와 SQL 사이의 변환 능력에 대하여 더 연구하여야 한다.

XML 문서 생성시 좀더 세부적인 XML 결과 단위를 얻을 수 있는 커서를 정의하고, 계속해서 XQuery로 작성된 사용자 질의를 관계형 데이터베이스 외에 다른 이질적인 데이터베이스에 실행시킬 수 있는 질의로 변환하기 위한 질의 처리 알고리즘 연구가 과제로 남아 있다.

참 고 문 헌

[1] M. Fernandez, W. Tan and D. Suciu, "SilkRoute : Trading between Relations and XML," WWW9, pp.723-745, 2000.

[2] J. Shanmugasundaram, J. Kiernan, E. Shekita, C. Fan and J. Funderburk, "Querying XML Views of Relational Data," VLDB Conference, pp.261-270, 2001.

[3] M. Fernandez, A. Morishima and D. Suciu, "Efficient Evaluation of XML Middle-ware Queries," SIGMOD, pp.103-114, 2001.

[4] World-Wide Web Consortium, "XQuery 1.0 : An XML Query Language," http://www.w3.org/TR/2001/WD-xquery-20010607.

[5] World-Wide Web Consortium, "XML Query Use Cases," http://www.w3.org/TR/xmlquery-use-cases.

[6] M. Carey, D. Florescu, Z. Ives, Y. Lu, J. Shanmugasundaram, E. Shekita and S. Subramanian, "XPERANTO : Publishing Object-Relational Data as XML," Workshop on the Web and Databases (WebDB), pp.105-110, May, 2000.

[7] 이경하, 이강찬, 이규철, "XML 기반의 이질 정보의 통합 방법론", 한국정보과학회, 가을학술발표논문집(I), pp.96-98, 1999

[8] Y. Papakonstantinou, A. Gupta, H. Garcia-Molina and J. Ullman, "A Query Translation Scheme for Rapid Implementation of Wrappers," DOOD, pp.319-344, 1995

[9] C. Baru, A. Gupta, B. Lud ascher, R. Marciano, Y. Papakonstantinou, P. Velikhov and V. Chu, "XML-based Information Mediation with MIX," SIGMOD, pp.597-599, 1999.

[10] World-Wide Web Consortium, "XML Schema Part 0 : Primer," http://www.w3.org/TR/xmlschema-0/.

[11] V. Christophides, S. Cluet, J. Simeon, "On Wrapping Query Languages and Efficient XML Integration," SIGMOD Conference, Dallas, Texas, June, 2000.

[12] I. Manolescu, D. Florescu and D. Kossmann, "Pushing XML queries inside relational databases," Tech. Report no. 4112, INRIA, Available at www.caravel.inria.fr/Epublications. html, 2001.

[13] I. Manolescu, D. Florescu and D. Kossmann, "Answering XML Queries over Heterogeneous Data Sources," VLDB Conference, 2001.

[14] J. Shanmugasundaram, E. Shekita, J. Kiernan, R. Krishnamurthy, E. Viglas, J. Naughton, and I. Tatarinov, "A General Technique for Querying XML Documents using a Relational Database System," SIGMOD Record 30(3), pp.20-26, September, 2001.

[15] R. Domenig and K. Dittrich, "An Overview and Classification of Mediated Query Systems," SIGMOD Record, 28(3), pp.63-72, 1999.

[16] World-Wide Web Consortium, "http://www.w3.org/TR/query-semantics/."

[17] Z. Ives, D. Florescu, M. Friedman, A. Levy, D. S. Weld, "An Adaptive Query Execution System for Data Integration," Proceedings of the SIGMOD Conference, Phil-

adelphia, 1999.
[18] A. Levy, A. Rajaraman and J. Ordille. "Querying heterogeneous information sources using source descriptions," In Proceedings of the 22nd International Conference on Very Large Data Bases, 1996.
[19] A. Levy, "Logic-based techniques in data integration," In J. Minker, editor, Logic Based Artificial Intelligence. Kluwer Academic, 1999.

정 채 영

e-mail : wcdi@rtp.gsnu.ac.kr
1997년 경상대학교 전자계산학과 학사
2001년 경상대학교 대학원 컴퓨터과학과 석사
2001년~현재 경상대학교 대학원 컴퓨터과학과 박사과정
관심분야 : XML, WWW, 데이터베이스, 정보검색

최 규 원

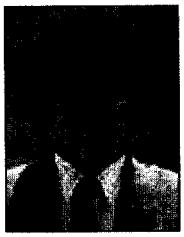
e-mail : jinsilkw@hotmail.com
2000년 경상대학교 컴퓨터과학과 학사
2002년 경상대학교 대학원 컴퓨터과학과 석사
2002년~현재 (주)보체웹닷컴 연구원
관심분야 : XML, VoiceXML, 데이터베이스 통합

김 영 옥

e-mail : yokim@kornet.net
1988년 울산대학교 전자계산학과 학사
1993년 동아대학교 산업대학원 컴퓨터공학과 석사
2002년 경상대학교 대학원 컴퓨터과학과 박사과정 수료
현재 진주제일병원 전산실장 재직
관심분야 : XML, 데이터베이스 통합, 데이터마이닝

김 영 균

e-mail : kimyoung@etri.re.kr
1991년 전남대학교 전산통계학과 학사
1993년 전남대학교 대학원 전산통계학과 석사
1995년 전남대학교 대학원 전산통계학과 박사
1995년~현재 한국전자통신연구원 컴퓨터소프트웨어 연구소 컴퓨터시스템연구부 선임연구원
관심분야 : 데이터베이스 시스템, 데이터베이스 통합, 데이터베이스 보안, 멀티미디어 정보 검색



강 현 석

e-mail : hskang@nongae.gsnu.ac.kr
1981년 동국대학교 전자계산학과 학사
1983년 서울대학교 대학원 계산통계학과 석사(전산학)
1989년 서울대학교 대학원 계산통계학과 박사(전산학)
1984~1993년 전북대학교 전자계산학과 부교수
1993년~현재 경상대학교 컴퓨터과학과 교수
관심분야 : 객체지향 데이터베이스, 전자문서 관리, 멀티미디어

배 종 민

e-mail : jmbae@nongae.gsnu.ac.kr
1980년 서울대학교 수학교육과 학사
1983년 서울대학교 대학원 계산통계학과 석사(전산학)
1995년 서울대학교 대학원 계산통계학과 박사(전산학)
1982년~1984년 한국전자통신연구소 연구원
1997년~1998년 Virginia Tech. 객원연구원
1984년~현재 경상대학교 전임강사, 조교수, 부교수, 교수
관심분야 : XML 데이터베이스 통합, 정보검색, 디지털라이브러리, 데이터마이닝