XQuery의 SQL 변환 연구

송청, 강민옥, 진민
경남대학교 컴퓨터공학과
e-mail:{thdcjd,minoklove,mjin}@kyungnam.ac.kr

A Study on Translation of XQuery Queries to SQL

Chung Song, Min-Ok Kang, Min Jin
Dept of Computer Engineering, Kyungnam University

요 약

관계형 데이터베이스를 이용한 XML 문서의 저장과 질의 처리와 관련한 연구가 활발히 진행되어 왔다. 그러나, XML 문서와 관계 데이터베이스는 구조적으로 일치하지 않기 때문에 XML 문서를 관계 데이터베이스에 저장 및 질의 처리를 하기에는 많은 어려움이 존재한다. XML 질의언어는 관계 데이터베이스에서는 직접 지원되지 않아 SQL로 변환되어 처리되어야 한다. 따라서, XQuery 질의를 SQL로 변환하여 처리하는 방법에 대해 많은 연구가 진행되고 있다. 본 논문에서는 지금까지 제안된 주요 방법들에 대해 비교 고찰한다.

1. 서론

XML은 내용과 표현이 분리되고 데이터 자체를 표현할 수 있고 확장 가능한 계층적 구조를 가지기 때문에 인터넷의 발전과 더불어 분산된 응용 프로그램 간의 데이터 교환을 위한 수단으로 점점 그 사용이 증대되고 있다.

계층적인 구조를 가지는 XML은 평면 구조를 가지는 관계데이터베이스와는 구조가 달라 관계데이터베이스에 XML 문서를 저장 및 질의 처리하기 위해서는 별도의 처리과정이 필요하다. 이를 위해 XPERANTO, XEAR, SilkRoute 등의 방법들이 제안되었다. 이 시스템들은 관계형 데이터베이스 환경에서 설계된 미들웨어 시스템이다. XQuery로 표현된 사용자 질의를 파싱하여 각각 XQGM(Query Graph Model)[2], XAT(XML Algebra Tree)[3][4], view forest[1]이라 불리는 중간 질의 표현으로 변환된다. 변환된 XML 표현들은 다시 SQL로 변환되어 질의를 처리한 후 질의 처리 결과에 대한 Tagging 작업을 거쳐 XML로 변환되어 사용자에게 결과가 전달된다. 본 논문은 XQuery로 표현된 질의를 SQL로 변환하여 처리하는 방법에 대한 주요 연구에 대해 각각의 특성을 비교 고찰한다.

2. XPERANTO

XPERANTO는 XML 뷰를 기반으로 관계 데이터베이스에 저장되어 있는 XML 데이터에 대한 질의 처리를 위한 미들웨어 시스템이다. XPERANTO 시스템은 자동적으로 default XML 뷰를 생성한다. default XML 뷰는 관계데이터베이스에 저장되어 있는 XML 데이터에 대한 뷰이므로 관계데이터베이스에 대한 정보와 XML 데이터에 대한 정보를 동시에 가지는 가상 뷰이다.

```
<order id="10">
  <customer>Smith
        Construction</customer>
  <items>
    <item description="generator">
      <cost>8000</cost>
    </item>
    <item description="backhoe">
      <cost>24000</cost>
    </item>
  </items>
  <payments>
    <payment due="1/10/01">
      <amount>20000</amount>
    </payment>
    <payment due="6/10/01">
      <amount>12000</amount>
    </payment>
  </payments>
</order>
<order id="9">
  ...
</order>
```

```
create view orders as (
  for $order in view("default")/order/row
  return
    <order id=$order/id>
      <customer>$order/custname</customer>
      <items>
        for $item in view("default")/item/row
        where $order/id = $item/oid
        return
          <item description = $item/desc>
            <cost>$item/cost</cost>
          </item>
      </items>
      <payments>
        for $payment in view("default")/item/row
        where $order/id = $payment/oid
        return
          <payment due = $payment/date>
            <amount>$payment/amount</amount>
          </payment>
        sortby(@due)
      </payments>
    </order>)
```

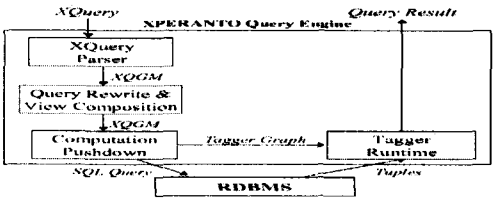
<그림1>원하는 XML 문서형태와 사용자정의 뷰

이 default XML 뷰를 이용해서 사용자 정의 뷰를

재정의하고 여기에 XQuery를 이용해 사용자 질의를 수행할 수 있다.

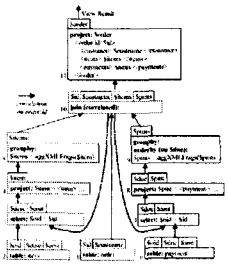
<그림1>과 같이 사용자 XML 뷰가 생성되면 아래와 같은 XQuery로 작성된 질의문을 수행하게 된다.

```
for $order in view("orders")
where $order/customer/text() like "Smith%"
return $order
```

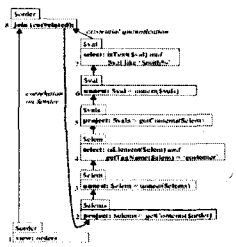


<그림2>XPERANTO시스템의 질의처리엔진구성도

<그림2>의 XPERANTO의 시스템은 사용자 질의가 들어오면 먼저 파싱하여 XQGM(XML Query Graph Model)이라 불리는 중간 질의 표현으로 변환된다. 이는 기존의 관계 데이터베이스가 아직 XQuery를 지원하지 못하기 때문이다. 따라서 XQGM의 변환은 XQuery를 SQL로 변환할 수 있도록 하고 대용량 데이터의 효율적인 질의 처리를 위해 질의 최적화를 할 수 있도록 함으로써 XML 데이터에 대한 질의 처리 문제를 해결하는 개념이라 할 수 있다. <그림3>과 <그림4>는 사용자 정의 XML 뷰와 사용자 질의문을 XQGM으로 표현한 것이다



<그림3> 사용자 정의 XML 뷰의 XQGM



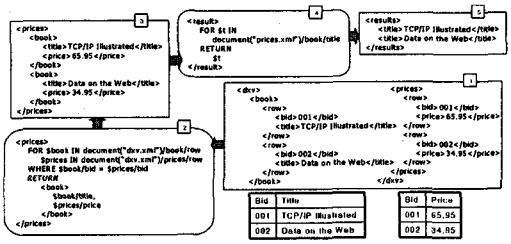
<그림4> 사용자 질의의 XQGM

XML 뷰를 정의한 XQuery와 사용자 질의 XQuery는 각각 XQGM으로 변환되어 결합되어진다. 결합되어진 XQGM은 반복적 순회 연산을 다른 연산과 결합시켜 불필요한 연산 작업을 최대한 줄여 중간 모델인 XQGM을 단순화시키는 과정이 view composition이다. 낭비적인 연산의 횟수를 줄이고 중첩 루프 조인 연산을 피하기 위해 SQL로 변환되는 부분과 XML 문서를 생성하기 위한 부분으로 분리되어 질의를 최적화 하는 과정이 Computation Pushdown 작업이다. 질의의 최적화가 이루어지면 관계 데이터베이스에 효율적으로 질의의 처리를 할 수 있도록 SQL 문을 생성하여 실행하고, 중간 형태의 XML 단편들을 생성하는 과정은 "Tagger Run-time" 모듈로 그 연산들을 넘겨주어 관계 데이터베이스에서 질의 처리된 결과들을 태그 정보에 의해 XML 문서를 생성하여 사용자에게 전달한다.

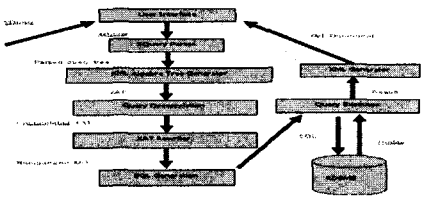
3. XEAR

XEAR은 WPI 의 Rainbow Project의 일부분으로, 관계 데이터베이스 관리 시스템과 XML 사이의 브릿지 역할을 수행하며, 사용자가 입력한 XQuery를 XAT(XML Algebra Tree)라 불리는 중간 질의 표현으로 변환한 후에 XAT를 최적화하여 SQL로 변환 생성하게 된다.

처음에 default view가 생성되는데, 직접적으로 관계 데이터베이스 안에 저장되어 있는 XML 데이터에 대한 정보를 표현한다. 이 default 뷰로부터, 사용자 뷰가 생성되며, 이 사용자 뷰에 XQuery로 사용자 질의를 수행 할 수 있다. 예를 들어 <그림5>을 들 수 있는데, 관계 데이터베이스에 저장되어 있는 데이터가 default 뷰로 <그림5-1>이 생성되고, 이 1번에 2번의 싱글 XQuery에 의해서 3번의 User view가 생성이 되며, 여기에 사용자 질의인 4번의 XQuery가 들어오게 되면 결과로 5번이 사용자에게 전달될 것이다.



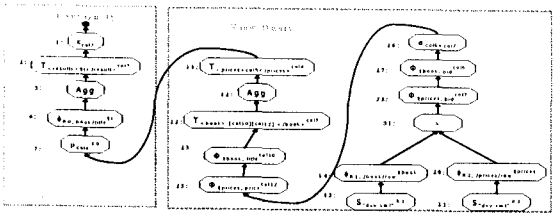
<그림5> (1:Default XML 뷰, 2:Mapping XQuery, 3:User XML 뷰, 4:User XQuery, 5:Result)



<그림6> XEAR 개념도

<그림6>의 XEAR 시스템에서 사용자는 XEAR 시스템에 XQuery를 사용해서 질의를 입력하고, 입력된 XQuery 질의는 파싱되어 XAT 생성기에 의해 XAT를 생성한다. <그림5-2>의 mapping 질의 또한

XAT로 변환 생성되며, 사용자 질의 XAT와 mapping 질의 XAT는 <그림7>과 같이 결합되어 진다. 이는 사용자 질의가 사용자가 검색하기를 원하는 값이나, 되돌려 받길 원하는 형식(tagging)의 정보만을 담고 있고, XML이 어떻게 DB에 어떻게 저장되어 있는지에 대한 정보는 포함되지 않기 때문이다. 이에 반해, mapping Query는 default view에 바탕을 두기 때문에 실제 XML 데이터를 유지하면서 정확한 관계모델에 대한 정보를 가지고 있으므로 이 두 질의를 결합(merge)함으로써 사용자 질의와 mapping 질의에 대한 정보를 공유할 수 있기 때문이다.



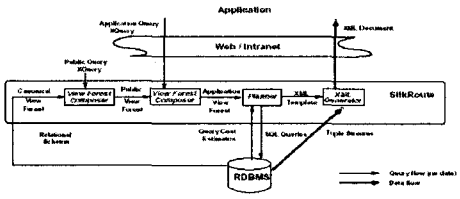
<그림7> XAT after Merging

질의 최적화를 위해 질의를 XAT로 변환 결합시킨 후 Query Decorrelation에서 중첩된 서브 질의를 동등한 선형적구조로 변환한다. 중간 모델 XAT의 최적화를 위해 XAT Rewriting과 XAT Cleanup를 실시한다. XAT Rewriting 단계의 목표는 중복을 제거하고 SQL-douable적인 것을 동등 룰에 의해 pushdown하게 된다. XAT Cleanup 단계에서는 XAT Table에서 사용되지 않는 컬럼들과 결과 도출에 사용되지 않는 연산자들을 제거한다. 왜냐면, SQL engine에서는 XQuery의 중복을 줄 일 수 없기 때문이다. 이는 Schema Cleanup에서 각 연산자의 produces, consumed, modified 와 같은 열(column)을 생성하여 minimum schema를 계산함으로써 여분의 Data를 줄일 수 있다. 결과의 도출에 사용되지 않는 연산자(unused operator Cutting)들의 삭제를 통해 여분의 tree도 삭제가능하다. 이를 위해 Cutting matrix를 생성하며 이 matrix의 컬럼들을 분석하여 연산자의 cutting을 한다. 이 XAT 최적화(Optimization)를 통해서 기존의 XEAR의 SQL구문의 생성의 효율이 증가하게 된다. 따라서 시스템 실행시 필요없는 연산과 연산자를 줄임으로써 효율성의 극대화를 이룰 수 있게 된 것이다.

XAT Rewriter에서 시스템의 효율성을 위해 SQL-doable 적인 것과 XML-specific 한 것으로 재편성한다. SQL-doable한 것은 SQL 질의로 변환가능 하기 때문에 시키고, XML-specific 한 태그들은 질의 결과들을 XML 문서를 구성하는 포맷 형식으로 태깅 작업을 해주기 때문에 트리의 상위에 위치시키게 된다. 재편성된 XAT는 Query Executor를 지나면서 데이터베이스에 SQL 질의를 실행하고 결과로서 생긴 튜플들은 XML 생성기를 거치면서 태깅 작업을 통해 XML 문서로서 사용자에게 전달된다.

4. SilkRoute

SilkRoute는 순환함수와 순서에 의존적인 XML data의 특징들을 제외하고 XQuery 표현을 SQL로 변환하여 질의 처리하는 미들웨어이다. SilkRoute에서 가장 두드러진 특징으로는 XQuery를 SQL로 변환해주는 방법으로 view forest를 이용한다는 것이다. 이때 view-forest composition algorithm(VFCA)이 사용되는데, 이 알고리즘은 XQuery 표현식들을 정규화 규칙들에 적용시켜 XQueryCore 표현식으로 정의하고 그 다음 view forest를 생성한다. view forest는 planner에 의해서 하나 또는 그 이상의 SQL 질의와 XML 템플릿으로 분해되어진다. SQL 질의들의 집합을 plan이라하며, 여러 가지 plan 중에서 최적의 SQL 질의들의 집합을 선택하기 위해서 SilkRoute에서는 공간탐색, greedy, cost-based 최적화 알고리즘을 제안하고 있으며 관계 엔진에 의해 실행되어진다.

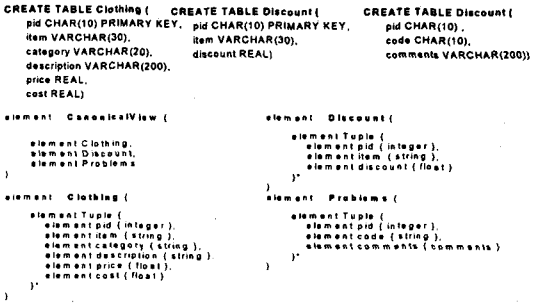


<그림8> SilkRoute 시스템 구조

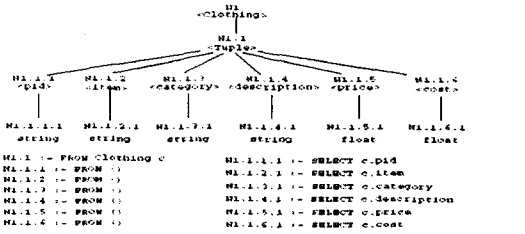
<그림8>은 SilkRoute의 시스템 구조를 나타낸 것으로 먼저 데이터베이스 관리자는 관계 테이블들을 canonical XML view로 정의하고, canonical XML view에 관하여 Public query를 이용하여 Public XML view를 정의한다. 이때 사용자들은 Application query를 이용해 Public XML view에 접근하게 되는데 Application query는 Public query와 조합되어 새로운 query를 만든다. Application 질의는 사용자가 원하는 결과에 대한 정보와 태깅 정보를 가지고, Public 질의는 관계 테이블에 대한 정보를 가지므로 이 두 질의를 결합하여 새로운 질의를 만들어낸다. 이때 Public view로부터 불필요한 표현식들과 중간 XML 결과들을 제거하고 관계 데이터베이스의 입력을 위해서 적절한 SQL 질의들을 산출한다.

SilkRoute에서 관계 데이터베이스는 가상으로

<그림9>과 같이 Canonical XML view로 표현되어진다. 그리고 <그림10>은 <그림9>에 있는 관계데이터베이스의 Clothing 테이블에 대한 Canonical view forest 단편을 나타낸 것이다.



<그림9>관계DB스키마(위), canonical XML 뷰(아래)



<그림10> Clothing에 대한 Canonical view forest

```
<supplier>
    <name>Acme Clothing</name>
    <discounted>
        { for $c in $CanonicalView/Clothing/Tuple,
          $d in $CanonicalView/Discount/Tuple
          where data($c/category) = "outerwear",
                data($c/pid) = data($d/pid),
                data($c/price) * data($d/discount) < 0.5 * data($c/price)
        return
            <product> { data($c/item) } </product>
        }
    </discounted>
</supplier>
```

<그림11> Composed Query

이 composed query는 planner에 보내지고, <그림12>과 같이 Planner는 view forest를 이용하여 하나 또는 그 이상의 SQL 질의와 XML 템플릿으로 분해하고, RDBMS에 의해 Query Cost에 대한 결과를 받아 최적의 plan를 선택한다.

```
SQL query :

SELECT    c.pid as pid, c.item as item
FROM      Clothing c, Discount d
WHERE     c.category = "outerwear",
          c.pid = d.pid,
          c.price * d.discount < 0.5 * c.price
ORDER BY  c.pid
```

```
XML template :

<supplier>
    <name>Acme Clothing</name>
    <discounted>
            <product>{ $item }</product>
    </discounted>
</supplier>
```

<그림12> SQL질의와 XML Template

Planner에서 선택되어 생성된 SQL 질의들은 관계엔진에 의해 실행되고 XML 생성기에서 결과 투플들과 XML 템플릿을 결합하여 사용자에게 XML 문서로 전달한다.

5. 결론

XPERANTO, XEAR, SilkRoute는 관계형 데이터베이스 환경에서 XML 문서 질의를 위해 설계된 대표적인 미들웨어 시스템이다.

XPERANTO와 XEAR은 관계형 데이터를 XML 뷰로 정의하기 위해 XQuery를 사용하고 사용자 질의 또한 XQuery로 작성한다. XPERANTO 시스템은 XQuery로 작성된 XML 뷰와 사용자 질의를 각각 XQuery으로 변환되어 결합한다. XEAR 시스템 또한 Mapping 질의와 사용자 질의는 각각 XAT로 변환되어 결합된다. SilkRoute 시스템은 관계 테이블들에 대한 정보를 나타내는 canonical XML view를 포함한 모든 뷰에 대해서 view forest로 변환해 주는 특징이 있다.

XPERANTO에서는 질의최적화를 위해 XQGM 그래프 상에서 중복되는 연산자들을 삭제하고 하나의 SQL문을 생성하여 수행한다. XEAR에서는 XAT cleanup과 Cutting Matrix를 통해 질의 결과 도출에 불필요한 연산자들과 컬럼들을 삭제하여 XAT를 최적화함으로써 효율적인 SQL문을 생성한다. 또, SilkRoute는 Planner에서 효율성을 검증하는 여러 알고리즘을 이용하여 최적의 SQL문을 생성하여 질의처리의 효율성을 높인다.

마지막으로 XQGM과 XAT는 절차적인 방법으로 복잡한 SQL에 대한 표현능력이 뛰어나고 SilkRoute의 view forest는 선언적인 표현으로 SQL변환에 대해 단순하게 접근할 수 있다는 장점이 있다.

참고문헌

[1] M. Feranadez, Y. Kadiyska, A. Morishima, D. Suciu and W.C. Tan "SilkRoute : A framework for Publishing Relational data in XML", ACM Transactions on Database Systems (2002) 438-4

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

[3] X. Zhang, B. Pielech and E. A. Rundensteiner, "XML Algebra Optimization", Technical Report WPI-CS-TR-02-25, Worcester Polytechnic Institute (2002)

[4] X. Zhang, et al "Honey, I Shrunk the XQuery! -An XML Algebra Optimization Approach", WIDM 2002 15-22