XML 문서 저장기능에 관한 DBMS 접근방법의 성능 비교

이말순°, 이재윤, 강석훈
대전대학교 컴퓨터공학과
e-mail: jamkong@zeus.dju.ac.kr

Performance Comparison of DBMS Access Methods on XML Document Storage Functions

Mal-Soon Lee, Jae-Yun Lee, Suk-Hoon Kang
Dept. of Computer Engineering, Daejeon University

요 약

인터넷 기반의 정보교환 표준으로 급부상한 XML에 의해 대용량의 XML 문서를 효율적으로 저장하고 관리하는 방법이 필요하게 되었다. 이에 따라 XML 문서를 저장하기 위한 백엔드 데이터베이스 시스템으로 RDBMS와 ORDBMS 그리고 Native XML 등의 XML 저장 시스템이 출현하였다. 그러므로 이들에 대해 유형별로 용도와 비용대비 성능에 관한 저장, 조회 및 관리 차원의 비교는 시급하다. 본 논문에서는 XML 문서를 저장, 관리하기 위한 RDBMS와 Native XML 데이터베이스에서의 XML 문서 접근방법에 대하여 비교 분석한다. 유형별 각 시스템 접근방식의 장단점을 비교 한 후 효과적인 XML 문서의 저장 기능 개발을 위해 상용 DBMS의 유형을 대표하는 Oracle 8i와 Tamino를 선택하여 성능을 비교한다.

1. 서론

다양한 형태의 정보들이 인터넷상에 기하급수적으로 급증하면서 많은 사용자들로부터 기존 정보의 공유와 호환성, 편의성을 효과적으로 사용하고자 하는 연구가 진행되었다. 이러한 요구에 부흥하기 위하여 HTML의 편리한 사용성과 SGML의 확장성 등의 장점을 취합한 XML (eXtensible Markup Language)이 표준 마크업 언어로 채택되었고 시스템 통합이라는 전기를 맞아 급속한 발전을 이루게 되었다.

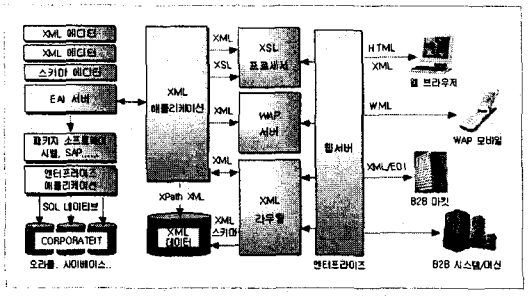

그림 1. 엔터프라이즈 컴퓨팅 환경에서의 XML 사용

기업에서도 그림 1에 도시한 바와 같이 XML 문서들이 많아짐에 따라 XML 문서들을 통합적인 관리 또한 필요하게 되었고 DBMS 들은 이러한 추세에 맞추어 한 단계 발전하게 되었다.

급속히 발전하고 있는 웹 상에서 또는 컴퓨터상의 각각 서로 다른 종류의 다양하고 많은 양의 정보를 제공하기 위해서는 문서들을 전산화하는 것과 다량의 전자 문서를 저장 관리하고 검색을 할 수 있는 시스템이 필요하다. 이러한 문서 저장 관리기를 통해 대량의 문서를 저장하고 저장된 문서 중 사용자가 원하는 부분에 대해 빠른 검색을 지원하며, 저장된 문서의 일부분을 추가, 삭제 및 변경할 수 있어야 한다.

기존의 DBMS는 문서의 내용정보에만 중점을 두어 문서를 저장 관리하였다. 그러나 최근의 XML과 같은 구조 문서는 문서의 내용뿐만 아니라 그러한 내용이 무엇과 관련되어 있는지에 대한 구조정보도 문서에 포함하고 있기 때문에, 이러한 문서를 처리해줄 수 있는 DBMS를 필요로 하게 되었다. XML 문서 저장이 가능한 백엔드 데이터베이스 시스템으로는 RDBMS와 OODBMS 그리고 ORDBMS 등이 있지만 이러한 시스템들은 엘리먼트를 테이블로 표현 시 많은 조인들을 유발하므로 성능 저하를 유

발하는 문제점을 갖는다[1].

이러한 성능 저하 문제를 해결하기 위하여 Native XML DBMS에서는 XML 문서를 테이블로 표시하지 않고 XML 형태를 유지하여 저장함으로서 기존 시스템들에서의 성능 저하 문제를 개선하는 접근 방법을 채택하였다.

Native XML DBMS를 이용할 경우, 계층적인 조인을 이용한 것보다 빠르게 처리할 수 있고, XML 문서를 Native하게 저장함으로서 대용량의 데이터에 대한 질의를 안정성 있게 처리할 수 있다는 장점을 가진다. 그러나 관계형 DBMS는 현재 사용자들의 과반수 이상을 차지하는 제품으로서, 새로운 XML DBMS로의 이전을 하기 위해서는 추가비용이 요구되는 부담이 있다. 그러므로 본 논문에서는 가장 많이 구현된 방식으로서 XML의 계층 구조를 관계형 데이터베이스의 관계로 표현하며 XML의 각 요소와 속성을 테이블의 필드로 표현하는 관계형 데이터베이스 맵핑형과 XML 문서 처리의 강력한 기능을 가지는 새로운 저장 기법으로 등장한 Native XML DBMS를 비교 평가한다.

본 논문의 구성은 다음과 같다. 2절에서는 연구와 관련된 XML 문서저장 방식들과 해당 DBMS에 대하여 살펴보고, 3절에서는 RDBMS와 Native XML DBMS의 저장 방식에 대해서 기술하고, 4절에서는 본 DBMS와 XML 가능 DBMS(RDBMS)와 비교하고 평가한다. 마지막으로 5절에서 결론 및 향후 연구방향을 제시한다.

2. 관련 연구

과거 저장 모델과 저장 시스템에 대한 연구가 활발히 진행되어 왔으며, 최근 XML은 차세대 저장 모델로 대두되어 왔다. 이번 절에서는 XML 처리 방법과 DBMS 연계에 대하여 장·단점을 분석한다.

2.1 기존의 XML 문서 처리 방식

2.1.1 부자연스러운 처리 방식

비구조적인 XML 문서와 컨텐츠들은 일반 파일시스템에 저장하여 관리하고 구조적인 관계형 데이터와 메타 데이터는 관계형 데이터베이스에서 관리하는 방법이다. 이는 XML을 미드티어(Mid-Tier)레벨의 애플리케이션에서 모두 처리되며, 대표적인 것으로 XML 문서에 대한 파싱이나 XSL-T을 이용한 각 디바이스에 표현될 수 있도록 변환하는 과정, 기타 문서의 유효성 검증 등을 들 수 있으며 개발자로서 고려해야 할 사항들이 많은 환경이다. 그러나 이 방법은 XML 문서를 일반 파일시스템의 특정 디렉토리에 저장하기 때문에 XML 문서를 데이터베이스에 저장하려는 의도나 업무보다 기존의 레거시 관계형 데이터베이스로부터 데이터를 추출하여 XML 문서를 생성한 다음 데이터를 요청한 클라이언트 또는 대상자에서 전달하기 위한 환경에 적합하다.

2.1.2 데이터와 컨텐츠 서버 분리 방식

XML 전용 저장소(Repository)를 채택하여 구조적인 관계형 데이터와 완전히 분리시켜 XML 문서나 컨텐츠 또는 메타 데이터를 전용 저장소에서 관리하는 방법으로 개발자나 사용자 입장에서는 XML 전용 저장소가 제공하는 기능으로 손쉽게 문서의 저장 및 검색을 할 수 있다는 장점이 있으나 문서 내의 데이터들은 RDBMS에서 추출되거나 XML 문서 내의 데이터를 필요시 RDBMS내에 저장해야 한다. 이때 일종의 분산 DB(Distributed DBMS)와 같은 방법으로 처리해야 하며 데이터의 일관성을 고려해야 하는 불편함이 있다. 또한 XML 전용저장소는 XML 문서의 저장, 검색 기능이 탁월하나 대용량 문서 저장에 따른 동시 사용자들에 대한 서비스는 성능과 가용성 측면에서 설계에 반영되지 못했으며 별개의 두 데이터베이스를 관리해야 하기 때문에 시스템 성능을 유지하기 위해 데이터베이스 관리자로서의 역할이 대단히 중요성을 갖는다.

2.1.3 Native 처리 방식

RDBMS 엔진과 XML DB 엔진이 통합되어 하나의 엔진으로 XML 문서와 관계형 데이터를 함께 관리할 수 있다. XML 문서와 컨텐츠, 그리고 메타 데이터와 관계형 데이터가 하나의 데이터베이스에서 관리되고 XML 저장소를 보유하고 있기 때문에 XML 문서의 저장, 스키마 등록, 파싱, 변환, 그리고 XML 문서에 대한 검색이 SQL (Structure Query Language) 기반으로 쉽게 처리될 수 있다. 따라서 개발자로 하여금 높은 생산성을 제공한다.

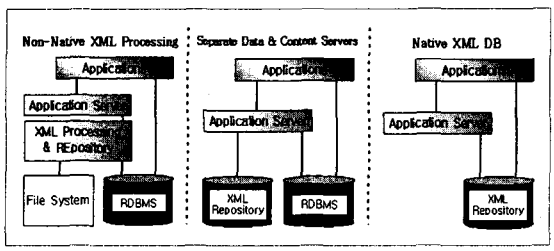

그림 2. XML 처리 방법에 따른 분류 비교

2.2 상용DBMS의 XML 문서 처리 방식

현재 대부분의 DBMS 회사들은 XML 문서를 수용할 수 있게 확장하고 있다. ObjectStore를 확장해서 만든 eXcelon[2]은 XML 문서를 개별적인 XML 엘리먼트로써 객체들로 저장한다. 오라클사의 오라클 9i 서버[3]와 함께 사용되는 오라클 XML DB는 XML 데이터를 컬럼의 데이터타입으로 저장 할 수 있도록 XML Type을 지원하고, XML 데이터 및 문서의 관리를 위해 XML Repository라는 인터넷 저장소를 제공한다. 또한 XML 전용 DBMS인 AG의 Tamino[4]는 컬렉션 형태로 여러 개의 문서 형태로 구성되어 관리되어진다.

기존의 데이터베이스 기반의 XML 문서 저장 시스템 연구는 크게 XML 문서를 관계 또는 객체지향 데이터 모

델로 변환하여 저장하는 모델링 연구와 효율적인 검색을 위한 인덱스 및 검색 구조 설계 연구로 나뉜다. 기존의 시스템은 XML 문서를 데이터 모델로 변환하는 과정에서 XML의 구조적 특징과 내용을 문서 독립적으로 모델링하여 저장하지 않으므로 XML 문서 구조의 갱신이 발생할 경우, 이에 따른 모든 구조정보가 수정되어야 했고, DTD 정보를 저장함에 있어 DTD 트리의 순차적인 탐색을 기반으로 리스트 형태의 정보를 데이터베이스에 저장하므로, 검색시의 효율이 떨어지거나 임의 위치의 엘리먼트 탐색이 어렵다.

이에 비해 Native XML 저장방법을 이용하는 경우 XML 엘리먼트를 맵핑하는 과정이 필요 없이 그대로 저장하므로 저장 시 부하가 최소화 되고, 조회 시 계층 구조에 최적화 되어 있고 XML 데이터를 있는 그대로 가져오므로 조회 시 부하가 경감되며 XML 구조 변경 시 데이터베이스 스키마 변경이 쉬운 장점을 가지는 반면 Native XML DBMS에 저장 시 질의 결과가 XML 문서라는 면과 관계형 DB에서 ODBC나 JDBC와 같은 표준화된 API가 개발되지 않았다는 개발 환경상의 단점을 가진다[5].

3. XML 문서 저장 방식의 비교

RDBMS와 Native XML DBMS에서의 XML 문서의 저장 방식을 비교하고 두 저장 방식의 특징을 찾아내어 XML 문서에 따른 적합한 저장 방식을 제시한다.

3.1 Native XML에서의 문서 저장방식

Native XML 데이터베이스에서는 XML문서를 기본단위로 가지는 스토리지(storage)라는 저장 개념을 이용하며 두가지 아키텍쳐로 분류한다. 먼저 텍스트 기반 스토리지는 전체 XML 도큐먼트를 텍스트 형태로 저장하고(이는 파일시스템 내의 파일 일수도 있고, RDBMS의 BLOB 또는 독점적인 텍스트 포맷일수도 있다.) 도큐먼트를 액세스할 수 있는 몇 가지 종류의 데이터베이스 기능을 제공한다. 두번째 모델 기반 스토리지는 XML 도큐먼트(document)를 텍스트로 저장하기 보다는 도큐먼트로부터 내부 객체를 생성하고 이 모델을 저장한다. 즉, DOM 또는 DOM의 변종과 같은 도큐먼트의 바이너리 모델을 기존의 또는 커스텀 데이터 저장소에 저장하는 것이다[6].

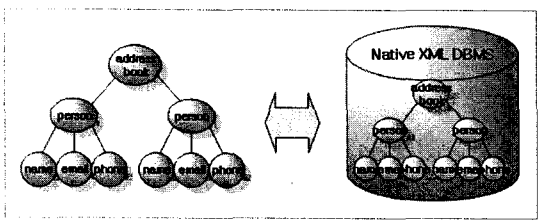

그림 3. Native XML에서의 문서 저장 방식

3.2 RDBMS에서의 XML문서 저장방식

그림 4와 같이 RDBMS에서 XML 문서를 저장하기 위해서는 매핑단계를 거친다. 매핑을 위해서는 미들웨어를 사용하여 매핑을 하거나 XML 가능 데이터베이스와 미들웨어를 사용하여 저장한다. 매핑하는 방법은 Table-Based Mapping과 Object-Relation Mapping으로 분류된다. 테이블 기반 매핑 방법은 소프트웨어를 사용하여 column에 child elements와 attributes 데이터로 저장되고 각 element나 attribute는 XML 문서에서의 태그들의 이름으로 사용된다. 테이블 기반 매핑은 연속적인 관계 데이터를 나타내기에 유용하지만 형식에 맞지 않는 XML 문서에는 사용할 수 없는 단점을 가진다. 객체 관계 매핑은 XML 문서를 객체 트리처럼 데이터를 저장하는 모델로 엘리먼트와 애트리뷰트 type, element content, complex element type이 클래스화 되어 테이블에 매핑된다.

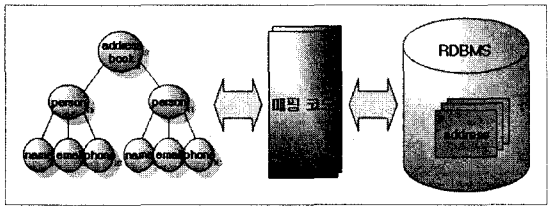

그림 4. RDBMS에서의 XML문서 저장방식

표 1. Native XML DBMS와 RDBMS의 비교

구분	Native XML DBMS	RDBMS
저장	-XML 문서 맵핑 없이 자체를 그대로 저장 -XML 데이터를 있는 그대로 저장하므로 저장 시 부하가 없음 -XML 객체나 요소를 원래의 형태대로 저장	-XML 태그를 각 테이블의 컬럼으로 맵핑 -RDBMS 테이블의 특성상 정규화 과정 필요 -XML 문서의 계층구조가 깊어질수록 테이블 구조가 복잡해짐 -XML의 속성을 표현할 방법이 없음
조회	-조회시 계층(hierarchy) 구조에 최적화 됨 -XQL,XPath,X_Query등 XML을 위한 쿼리 사용 -조회하는 쿼리가 간단함 -XML데이터를 있는 그대로 가져오므로 조회 시 부하가 없음	-조회시에는 쪼개진 테이블을 join해서 가져와야 하기 때문에 부하 발생 -조회하는 SQL 문장이 복잡함
관리	-XML 구조 변경시 데이터베이스 스키마 변경이 쉬움	-XML 구조 변경시 데이터베이스 스키마 변경이 어려움

4. 상용DBMS를 이용한 성능비교 : Oracle 8i vs. Tamino

DTD 독립적인 문서와 의존적인 XML 문서의 평가를 위해 DTD를 설계하고, 데이터 중심 문서와 도큐먼트 중심의 예제를 들어 Native DBMS와 관계형 DBMS에 각각 저장, 검색하고 수행시간을 측정하였다.

```
<!ELEMENT Shopping_mall (#PCDATA)>
<!ELEMENT SHOPPING (member, auction, categories, ……)>
<!ELEMENT member (Shopping_mall)>
<!ELEMENT auction (Shopping_mall)>
<!ELEMENT categories (Shopping_mall)>
                    ⋮
```

그림 5. 쇼핑몰 DTD

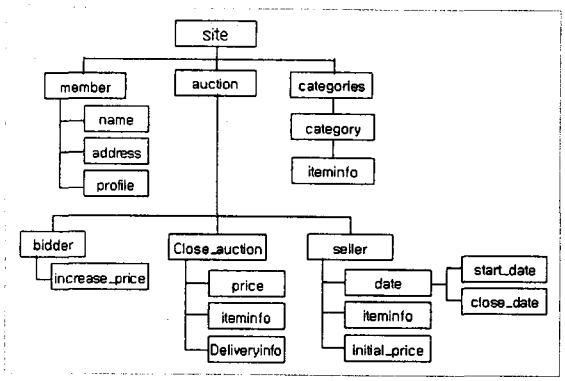

그림 6. 쇼핑몰 Site 구조

4.1 실험환경

본 시스템은 Windows 2000 환경에서 구현되었다. 실험에 사용된 XML 문서는 xmlgen을 사용하여 생성하였다.

4.2 성능평가 기준

성능 평가는 앞서 제시한 XML 문서 저장 시스템하에서 조건을 달리하여 성능차이를 비교하였다. 본 시스템 하에서는 DTD 독립적인 XML 문서와 의존적인 문서간 저장시간과 검색시간의 차이를 비교해 보며, 저장시스템을 RDBMS와 Native XML로 했을때의 차이를 비교하였다.

4.3 성능평가

표 2에서는 문서의 크기를 xmlgen 프로그램을 측량값으로 넣어서 생성한 결과를 나타낸다.

표 2. Data Size와 Blockload time

System \ Size & Time	Size	Blockload time
RDBMS	224MB	374s
RDBMS(DTD)	238MB	453s
Native XML	290MB	103s
Native XML(DTD)	307MB	155s

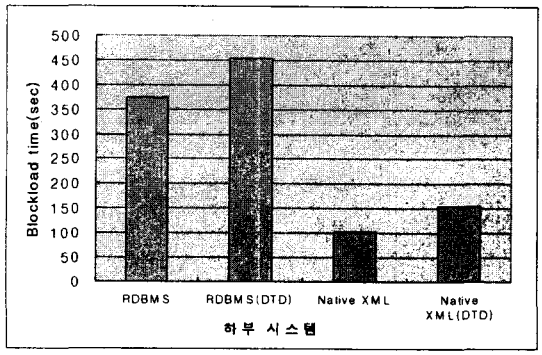

그림 7. Data Size와 Blockload time

표 3에서는 몇 가지 질의를 통해 DTD의 유무와 DBMS를 달리했을 때의 결과를 보여준다. 첫째, DTD 의존인 경우보다 DTD 독립인 경우 처리 시간이 적게 소요된다. 둘째, 질의에서 단순질의에서는 성능의 차이가 발생하지 않으나 조인 질의 시 Native XML의 성능이 확연히 우월함을 볼 수 있다.

표 3. 하부시스템과 DTD 유무에 따른 검색시간 비교

Query \ System	RDBMS	RDBMS (DTD 포함)	Native XML	Native XML (DTD 포함)
Qurey_1	221s	237s	196s	204s
Qurey_2	331s	509s	298s	336s
Qurey_3	427s	741s	287s	408s
Qurey_4	10189s	32534s	5994s	13698s

Query_1 : DTD 독립의 단순질의 Query_2 : DTD 의존의 단순질의
Query_3 : DTD 독립의 복잡한 조인 연산이 포함된 질의
Query_4 : DTD 의존의 복잡한 조인 연산이 포함된 질의

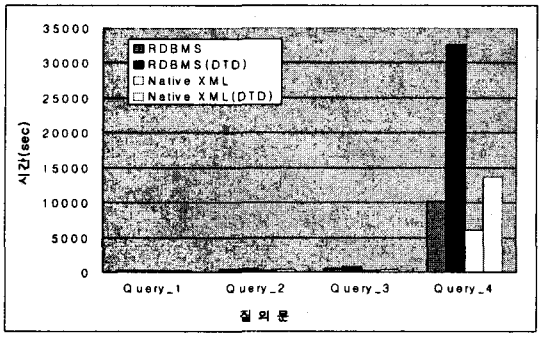

그림 8. 하부시스템과 DTD 유무에 따른 검색시간 비교

5. 결론

본 논문에서는 각광받고 있는 XML 문서를 저장하기 위한 접근 방법들을 분석하고 비교 평가하였다. XML 문서 관리를 위해 데이터베이스의 선택은 XML로 다루고자 하는 데이터의 형식과 데이터의 활용도에 따라 선택되어져야 한다. 기존에 관계형 데이터베이스를 이용하여 데이터를 관리하였다면 반드시 Native DB를 선택하는 것만이 큰 이점을 가져오는 것은 아니다. 그러나 복잡한 설계 도면과 같은 정보를 관계형 데이터베이스로 계속 관리하는 것은 비효율적일 수 있다. 따라서 XML 데이터를 대량으로 관리하고 이용하는 경우에는 Native DBMS를 권장하고, 기존시스템을 EDI 등에서 XML로 이전하는 경우에는 관계형 데이터베이스를 사용해도 무방하듯이 응용분야별로 적절한 데이터베이스를 선택하여 데이터를 저장하고 관리하는 것이 효과적으로 판단된다.

참고문헌

[1] 박민경, 홍의경, "XML 문서저장시스템의 성능평가", 정보기술연구소 논문집 제4집, pp. 29~36. Aug. 2002
[2] http://www.datec.co.kr eXcelon
[3] http://otn.oracle.com/tech/xml/content.html
[4] http://www.tamino.co.kr/taminoserver/jsp/index.jsp
[5] http://www.xmlone.co.kr/ttn/jsp/content/content.jsp
[6] http://www.rpbourret.com/xmldbms/index.htm