XML 데이터베이스 기반 멀티미디어 데이터 관리 기법의 개선

안민영「,김한준
서울시립대학교 전자전기컴퓨터공학부
{cuzluvu「, khj}@uos.ac.kr

Improving Multimedia Data Management Technique based on XML –native Databases

요약

빠른 인터넷의 확산과 다양한 정보의 등장으로 효율 적인 정보 관리와 저장 시스템에 대한 요구는 증가하고 있다. W3C 에서는 데이터를 표현 및 관리함에 있어서 보여주기 위한 데이터가 아닌 정보 공유를 위한 데이터에 중점을 두는 XML 을 제안하였고 빠르게 세계의 표준이 되어 가고 있다. 멀티미디어의 발전에 따른 멀티미디어의 용량 거대화 추세로 멀티미디어 데이터에 대한 저장의 요구도 커지고 있는 것이다. 멀티미디어의 용량은 커지고 있고 멀티미디어의 저장소로서 데이터베이스의 역할은 증대되고 있다. 하지만 기존의 관계형 데이터베이스 관리 시스템으로는 XML 문서와 멀티미디어 데이터에 대한 저장 및 관리의 효율성이 높지 않다. 그래서 본 논문에서는 XML 문서의 관리가 용이한 XML 데이터베이스를 활용함으로써 멀티미디어 데이터의 저장과 관리의 효율성이 크다는 것을 제시하여 이를 실험을 통해 확인하였다.

1. 서론

초고속 인터넷의 보급과 인터넷 속도의 향상으로 온라인에서 보여지기 위한 동영상, 이미지 파일, 음악 파일 등의 멀티미디어는 더욱 화려해지면서 보는 이의 눈을 즐겁게 해준다. 이에 따른 멀티미디어의 용량의 증대는 필연적이며 이 멀티미디어의 저장소로서의 데이터베이스에 대한 요구도 증가하고 있다. 기존의 관계형 데이터베이스 관리 시스템(이하 데이터베이스)은 SQL 을 이용한 데이터의 저장과 검색이 목적이기 때문에 이러한 멀티미디어 데이터의 저장소로써 기능이 만족스럽지 못하다.

또한 현재 XML (eXtensible Markup Language)은 데이터를 공유하는 표준안으로서 자리잡아 가고 있으며, 앞으로 정보처리 표준이 될 것으로 전망한다. 최근 마이크로소프트사의 새로운 버전의 오피스 프로그램은 XML 을 사용하여 문서 데이터를 저장하는 기법을 선보였다. 흔히 멀티미디어 데이터와 XML 문서를 저장하기 위해서 관계형 데이터베이스 관리 시스템 (Relational Database Management System)을 활용한다. 이는 관계형 데이터베이스가 가진 대중성과 간편성, 그리고 성능의 안정성이 보장되어 있기 때문이다. 관계형 데이터베이스에서는 XML 문서를 관리하는 것은 한계를 지니고 있다. 관계형 데이터베이스에서 XML 문서는 멀티미디어 데이터의 형태로 저장되는데 문서를 읽어 들일 때 DOM(Document Object Model)이나 SAX(Simple API for XML) 객체 형태로 문서를 재구성해야 하기 때문에 여러 단계의 프로세스와 그에 따른 메모리 사용이 필요하다.

그래서 본 논문에서는 XML 데이터베이스를 이용하여 멀티미디어 데이터의 저장과 효율적 관리를 위한 연구를 보여주고자 한다.

2. 관련 연구

XML 은 1998 년 W3C 에서 정식표준으로 제정이 되었고 사용이 간편하고 재사용성 및 확장성이 뛰어나다는 장점을 가지고 있다. 이러한 장점으로 인하여 XML 은 전자거래, 전자민원서비스 등 많은 분야에서 활용되면서

웹 상에서 유통되는 수많은 정보자원들과 메시지들을 XML 전자문서로 생성한다. 따라서 업체 및 조직에서는 이런 XML 전자문서들을 효과적으로 저장, 관리할 수 있는 데이터베이스가 필요하다고 인식하게 되었다

2.1 XML의 문서 관리 방식

XML 문서는 크게 데이터중심의 XML 문서와 문서중심의 XML 문서로 나눌 수 있다. 데이터 중심의 XML 문서는 데이터 전송을 위해 XML 문서를 사용하는 경우를 말하며 전자거래를 위한 구매주문서, 온라인상에서의 민원처리를 위한 신청서, 부처간 데이터교환을 위한 메시지 등과 같은 것을 포함한다. 문서중심의 XML 문서는 주로 학술분야의 논문 및 연구보고서, 사용자 매뉴얼, 마케팅 브로셔 등과 같이 웹을 통해서 최종사용자에게 보여주기 위해 작성된 XML 문서를 말한다. 기존의 XML 문서를 관리하는 방법으로 통합 관계형 데이터베이스 기반방식, 분할 관계형 데이터베이스 기반방식이 사용되어 왔다.

2.2 XML 문서의 검색 기술

XPath 는 XML 문서의 주소들을 표현하기 위한 언어이다. XML 문서에 대한 탐색을 통해 해당 엘리먼트 또는 애트리뷰트에 대한 위치 정보를 찾기 위한 방법으로 제시되었다. XPath 는 검색의 기준이 되는 컨텍스트 노드의 위치를 문서 또는 트리 구조의 애트리뷰트나 엘리먼트 같은 특정 노드에서의 검색이 가능하도록 지원하고 있다.

XQuery 는 XML 데이터의 컬렉션에 질의할 수 있는 질의 언어이다. 관계형 데이터베이스 시스템의 SQL 언어와 기본적으로 같은 기능을 가지고 있다. 현재 1.0 버전까지 개발되었다. XML 데이터나 XML 로 보이는 데이터 소스를 추출하고 다루는데 사용된다. XML 문서의 특정 위치를 알기 위해서 XPath 를 사용한다.[2]

2.3 멀티미디어 데이터

멀티미디어 데이터는 BLOB(Binary Large Object)의 형태로 생성되는데 BLOB 이란 integer, vchar, text 등과 같이 데이터베이스에 저장되는 파일 형식의 하나로 문자가 아닌 2 진 데이터 덩어리로 데이터 베이스에 저장되기 때문에 일반 데이터베이스처럼 내용물 검색이나 자료 관리는 불가능하고, 오직 그 크기와 위치만을 알 수 있다. 이미지, 비디오, 사운드 등과 같은 멀티미디어 객체들이 이에 해당한다

3. XML 데이터 베이스를 이용한 멀티미디어 데이터의 관리

사실 데이터베이스에 멀티미디어 자료 처리의 기능을 지원하는 연구는 현재 널리 사용되고 있는 관계형 데이터베이스 시스템분야에서부터 시작되었다. 관계형 데이터베이스 시스템은 기본적으로 여러 속성으로서 가진 튜플들이 모여 테이블 형태를 이루고 있다. 이때 각각의 속성으로서 기존에는 문자 자료형과 이들 자료형은 정해진 크기를 지니고 있어 가변적인 길이를 가지는 멀티미디어자료로의 적용이 부적합하다.

3.1 기본 동기

통합 관계형 데이터베이스(Integrated Relational Database) 기반 방식은 XML 문서를 BLOB(Binary Large Object)의 형태나 CLOB(Character Large Object)의 형태로 저장한다. 문서의 저장 및 추출 속도가 빠르나, 문서의 부분 수정이나 검색에 많은 시간이 필요하다는 단점이 있다. XML 문서를 수정하기 위해서는 관계형 데이터베이스에서 읽어온 문서를 다시 파싱하여 XML로 변환한 다음 수정을 한 다음에 다시 데이터베이스에 넣어야 하는 번거로움이 있다.[1]

분할 관계형 데이터베이스(Divided Relational Database) 기반방식은 XML 문서를 구성하는 각 요소 단위로 분할하여 테이블의 필드에 저장하는 것으로, 문서의 부분 수정이나 검색을 빠르게 수행할 수 있으나, 문서 추출 시에 여러 테이블에 대한 조인 과정을 거치므로 속도가 매우 느리다는 단점이 있다. 또한 XML을 분해/변환하는 과정에서 문서의 원래 구조를 잃어버린다. 엘리먼트의 순서나 애트리뷰트의 정보, 노드들의 관계 정보가 사라진다. XML 데이터가 많거나 문서가 많을 때에는 많은 조인 과정이 일어나므로 XML 문서 본

연의 기능이 상실되고[6] 프로세스의 성능 저하가 필연적으로 일어나게 된다. 순서 정보를 보존하기 위해서는 새로운 칼럼을 만들어야 하는데[7] 칼럼이 추가되고 용량은 더 커지게 된다.

통합형 관계형 데이터베이스에서 XML문서를 저장할 때는 XML 문서를 파싱해야 하는 문제점이 있고 분할 관계형 데이터베이스에 저장하기 위해서는 불규칙적인 XML 데이터를 저장할 때, 공간의 낭비와 비효율성 의 문제점을 안고 있다.

그래서 XML문서를 검색, 저장하기 위해서는 XPath, XQuery의 XML Query 질의의 표준을 사용하는 XML 전용 데이터베이스(Native XML Database)가 필요하다[3]. XML 데이터베이스는 XML 데이터에 대해서 더 빠른 검색과 저장을 가능하게 해주기 때문이다.

3.2 SQL 쿼리와 XQuery의 비교

데이터베이스에 저장되는 멀티미디어 데이터에 대한 메타데이터는 author, date, keywords, filename 등의 정보를 가질 수 있다. 그림 1의 A)는 통합 관계형 데이터베이스에서 멀티미디어 데이터로 저장된 XML 파일을 테이블에서 선택하여 바이트 스트림으로 읽어서 다시 XML 문서로 파싱 작업을 해서 문서를 읽는 과정이다. 그림 1의 B)는 분할 관계형 데이터베이스에 여러 테이블에 나눠진 엘리먼트들을 조합하여 SQL 문을 이용하여 쿼리해야 함으로 속도의 저하가 발생하는 과정을 보여준다. 그림2의 XML 데이터베이스는 XML 문서에 대해서 직접 쿼리를 해서 시간이 단축되고 update 를 할 경우를 보여준다. 이 때는 서버에 접속하여 바로 수정할 수 있는 장점이 있다.

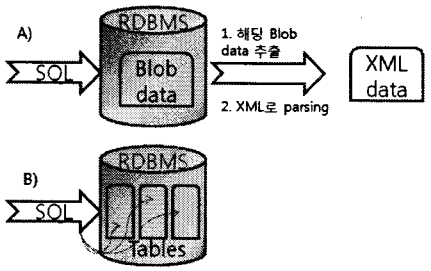

그림 1 : 관계형 데이터베이스의 쿼리 방식

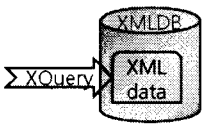

그림 2: XML 데이터베이스의 쿼리 방식

3.3 XML 데이터베이스를 이용한 멀티미디어 데이터의 관리

본 연구에 사용한 데이터베이스는 Wolfgang Meier 가 2000년 말에 개발 되기 시작한 eXist 라고 불리는 Open source Native XML 데이터베이스이다[4][5]. XML을 수정하기 위한 XUpdate와 XML을 query하기 위한 XPath, XQuery 를 지원한다. XQuery 를 지원하는 많지 않은 open source 데이터베이스 이고, 높은 안정성과 개발 환경을 제공해준다.

관계형 데이터베이스에서 BLOB으로 저장하거나 테이블에 노드로 나누어서 질의를 하는 것은 XML 데이터베이스를 사용하여 쿼리를 하는 것보다 복잡한 과정을 요구하고 속도의 성능에서 많이 뒤떨어진다. 또한 다수의 유저가 사용하게 될 데이터베이스의 경우에 계속 늘어나는 XML 데이터에 대해서 멀티미디어 데이터를 늘리거나 많은 테이블에 XML 문서를 분리하게 되면 관리하기 매우 복잡해지고 데이터의 보수관리 측면에서 일관성을 유지하기가 어렵다. 또한 테이블의 수를 늘리거나 크기를 키우는 것은 관리해야 할 XML 데이터가 많거나 다수의 유저가 사용할 경우 부적당하다. 관리 영역이 넓어져 프로세스의 성능 저하와 의존성 문제가 필연적으로 일어나기 때문이다[8].

하지만 XML 데이터베이스의 경우에는 XML 파일을 이용하여 메타 데이터 매니지먼트를 만들어서 하나의 파일에 update를 가함으로써 수정 정보에 대한 비대적 팽창을 예방할 수 있다. 또한 XQuery 는 테이블 에 질문을 하여 정보를 가져오는 관계형 데이터베이스와 비교하여 더 다양한 표현성을 가져서 더 다양한 정보의 검색을 가능케 해준다. 또한 XUpdate 를 사용하여 XML 데이터베이스 에 직접 접속하여 메타데이터 매니지먼트의 정보를 수정할 수 있다[4]. 관계형 데이터베이스는 XML 데이터를 읽어와 파싱을 하거나 테이블의 조인을 이용하여 수정하여 업데이트 하거나 여러 테이블을 수정해야 하는데 비해서 훨씬 능률적이다. 이러한 장점들은 멀티미디어 데이터의 관리를 더 체계적이고 효율적으로 할 수 있게끔 해준다.

표 1- 데이터베이스 입력 속도 측정

DB \ MB	1	3	5	7	10	30	70	150	300
PostgreSQL	138	333	670	713	1181	5431	11138	23113	241464
	117	341	518	850	1284	4839	11047	28087	263125
	118	401	696	757	1482	4279	11788	31885	207586
eXist	330	587	799	1148	3525	10917	28513	82399	357868
	242	575	897	1186	4481	19186	41612	124187	446366
	504	567	772	1026	3206	13906	41603	122439	392411

4. 실험

실험은 관계형 데이터베이스와 XML 데이터베이스의 멀티미디어 데이터의 입력 속도에 관한 실험이다. 첫 번째 실험을 통해서 높은 수준까지 개발된 관계형 데이터베이스와 비교하여 개발 초기인 XML 데이터 베이스에 대한 멀티미디어 데이터의 저장 적합성 여부를 측정할 수 있다. 두 번째 실험에서는 입력된 멀티미디어 데이터들을 검색하는 실험을 한다. 이 실험을 통해서 멀티미디어 데이터들에 대한 관리적 측면을 예측할 수 있다

4.1 실험 환경

기존 의 관계형 데이터베이스의 오퍼레이션과 XML 데이터베이스를 비교하기 위해 안정성이 높고 대중적인 PostgreSQL 을 사용하였다. PostgreSQL 은 현재 8.2 버전까지 출시되었으며, MySQL 과 비교하여 더 큰 용량의 서버로서 적합하다고 평가를 받아서 선정하였다.

연구에서 개발된 프로그램은 메타데이터 매니지먼트 API로써 XML을 이용한 멀티미디어 데이터 매니지먼트 프로그램이다. XML을 이용하여 저장되는 멀티미디어 데이터에 대한 정보들을 입력하여 관리함으로써 멀티미디어 데이터를 관리한다. 멀티미디어 데이터를 데이터베이스에 입력할 때에 XML에 데이터에 대한 author, date, keyword, filename등을 입력하게 됨으로써 각 데이터에 대한 메타 데이터를 관리한다. 이러한 메타 데이터 관리는 관계형 데이터베이스에서 테이블을 만들어서 할 수 있지만 XML DB에서 메타 데이터 매니지먼트는 관리 프로그램 안에 담긴 멀티미디어 데이터가 XML이라면 이를 다시 access하여 조작할 수 있는 이점이 있다.

메타데이터 매니지먼트는 4가지의 operation 으로 구현되어 있다. 파일을 등록하는 registerObject 와 내용을 검색하는 search, XML 데이터를 수정하는 update, 마지막으로 Object를 삭제하는 DeleteObject 이다. XML 메타데이터 File을 만들어서 등록되고 수정되는 멀티미디어 데이터 의 정보를 입력한다. 구현 언어는 JAVA로 구현되었다.

1. registerObject operation: Blob 데이터를 저장할 때, 입력 데이터의 정보를 XML 메타 데이터 file에 author, date, keyword, filename의 총 4가지로 구분하여 입력한다.
2. Search operation: 검색은 XML 메타데이터 File에 기록된 정보를 검색한다. XML 메타데이터 파일에서 입력된 XQuery 나 해당 단어에 대해서 데이터베이스에 접속하여 query 를 수행하고 결과를 리턴한다.
3. Update operation: 해당 blob file의 정보의 수정 사항을 XUpdate를 이용하여 XML 메타데이터 파일에 update한다.
4. Delete operation을 할 때는 XML 메타 데이터 file에서 해당 멀티미디어 데이터의 정보를 삭제하고 동시에 저장된 파일도 삭제한다.

4.2 입력 속도의 측정 실험

PostgreSQL 에 멀티미디어 데이터가 입력될 테이블을 구축하고 JAVA 언어로 메타데이터 매니지먼트 모듈을 구현하였다. 그리고 JDBC를 이용하여 데이터베이스에 접속하고 멀티미디어 데이터를 입력한다. 멀티미디어 데이터의 용량은 1MB, 3MB, 5MB, 7MB, 10MB, 30MB, 70MB, 150MB, 300MB 로 나누어서 입력한다. eXist 에도 마찬가지로 JAVA 언어로 메타데이터 매니지먼트를 구현하고 XML 메타데이터 파일을 만

표 2 - 관계형 데이터베이스와 XML 데이터베이스의 데이터 읽기 (Y:Yes N:No)

Db \ Mb	1	3	5	7	10	30	70	150	300
PostgreSQL	Y	Y	Y	Y	Y	N	N	N	N
eXist	Y	Y	Y	Y	Y	Y	Y	Y	Y

들어서 멀티미디어 데이터를 저장할 때 메타데이터의 정보를 저장한다. 9종류의 용량으로 나누어진 멀티미디어 데이터가 각 데이터베이스 에 입력되는 속도를 비교하여 멀티미디어 데이터에 대한 데이터베이스의 적합성을 측정한다. 표 1과 같이 입력되는 속도는 계형 데이터베이스인 PostgreSQL 가 전체적으로 나은 속도를 보였다. 관계형 데이터베이스인 PostgreSQL 는 8.2 버전이고 오래 전부터 개발되었지만 XML 데이터베이스인 eXist 는 버전이 1.2에 불과하기 때문에 아직 속도의 측면에서는 기존의 관계형 데이터베이스 관리 시스템보다 느리다. 하지만 XML 데이터베이스 관리 시스템이 버전 업이 된다면 속도의 향상이 될 것이다.

그림 3은 관계형 데이터베이스와 XML 데이터베이스의 평균 입력속도를 보여준다. 멀티미디어 데이터의 크기가 300MB일 때, XML 데이터베이스가 1.6배 정도 느리다.

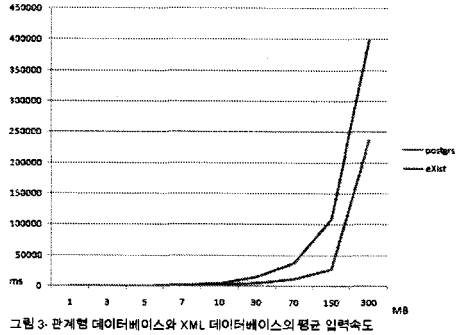

그림 3. 관계형 데이터베이스와 XML 데이터베이스의 평균 입력속도

4.3 저장된 멀티미디어 데이터 검색 실험

관계형 데이터베이스에는 Large Object의 형태로 멀티미디어 파일이 테이블에 저장된다. 마찬가지로 다시 읽어오기 위해서는 테이블에 저장된 Large Object 형태를 읽어서 다시 바이트 스트림으로 읽어야 한다. 하지만 XML 데이터베이스 관리 시스템에서는 형태를 변환할 필요가 없기 때문에 저장된 멀티미디어 데이터에 대한 더 빠른 검색이 가능하다. 테이블에 저장되는 것이 아니라 컬렉션에 바로 저장을 하기 때문이다.

표 2는 입력되는 멀티미디어 파일들의 순서에 따라서 파일을 읽어온 결과이다. 관계형 데이터베이스의 경우에는 10MB의 멀티미디어 파일이 입력되었을 때 까지는 테이블에서 읽어왔으나 30MB의 파일이 입력되었을 때부터는 검색을 할 때 시스템 지연이 심해지고 10분이 지나도록 읽어오지를 못했다. 그 다음 용량의 파일에 대해서는 검색의 기준 중에서 하나인 속도를 만족시키지 못함으로 검색 여부를 N으로 처리하였다. 하지만 XML 데이터베이스의 경우에는 300MB 파일이 입력되었을 때도 문제없이 컬렉션의 데이터 리스트를 표시해주었다.

5. 분석

아직 초기 개발 단계인 XML 데이터베이스의 트랜잭션 처리나 검증된 안정성 면에서는 기존의 관계형 데이터베이스 에 비해서 약하다. 하지만 본 연구에서는 앞으로 표준이 될 XML과 XML 저장소로서 XML 데이터베이스의 미래성에 초점을 두어 연구를 진행하였다. Open source XML DB의 종류로는 Xindice, DBXML, eXist가 있는데 eXist는 XML 데이터베이스로서 가장 개발이 활발하고 많이 사용되는 데이터베이스 관리 시스템 중의 하나이다. 기존 관계형 데이터베이스를 사용하는 방법에 비해서 XML 데이터베이스인 eXist 를 사용하는 것이 멀티미디어 데이터의 관리가 더 용이한 것을 알 수 있다. 또한 XML 메타데이터 에 대해서 직접 질의를 함으로써 관계형 데이터베이스에서 이루어져야 할 번거로운 파싱 작업이 생략되는 것으로 질의 속도의 향상을 보여주었다. 하지만 개발 시점의 차이 때문에 멀티미디어 데이터에 대한 입력 속도가 관계형 데이터베이스에 비해서 느린 것은 앞으로 개선될 사항이다. XML 데이터베이스를

사용한 XML 데이터에 대한 수정, 접근성이 높아져서 멀티미디어 데이터의 정보 관리에 유용하지만 클라이언트가 XQuery 의 사용법을 알고 있어야 하는 단점이 있다.

6. 결론

인터넷의 보급으로 인한 멀티미디어의 발전은 빠르게 진행되고 있고 멀티미디어의 저장소로서 데이터베이스의 역할이 점점 커져가고 있다. 또한 XML은 정보공유의 표준으로써 자리 잡아가고 있다. 하지만 기존의 관계형 데이터베이스는 XML과 멀티미디어의 저장소로서 만족할만한 성능을 보이지 못하고 있다. 현재 개발되고 있는 XML 데이터베이스를 활용하여 XML을 활용도를 높이고 멀티미디어 데이터에 대한 관리를 향상시킬 수 있는 방법을 연구했다. 입력 속도의 측면에서는 XML 데이터베이스가 아직 만족할 정도는 아니지만 XML의 활용과 멀티미디어 데이터 저장소로서 적합함을 본 논문에서 보여주었다.

7. 참고문헌

[1] W3C, http://www.w3.org/xml/query

[2] 류광택, " XML을 이용한 데이터베이스 연동방안 연구", 한국전산원.

[3] 장희철, "관계형 스키마에서 XML DTD로의 변환에 관한 연구", 연세대 공학대학원, 2002.

[4] 송정석, "XML 문서의 RDB로의 매핑을 위한 의미적 제약 조건 보존과 확장 색인 기법", 광운대 대학원, 2005.

[5] Wikipedia, http://en.wikipedia.org/wiki/XML_데이터베이스

[6] exist, http://exist-db.org

[7] Wolfgang Meier. eXist: An Open Source Native XML 데이터베이스. 노드 2002 Web- and 데이터베이스-Related Workshops, Erfurt, Germany, October 2002.

[8] Jim Fuller, "XML and 관계형 데이터베이스: 10 years on", simple-talk.xom, 2006