XML기반 공문서의 관계 데이터베이스 저장 모델

안만선*, 이언배*
*한국방송통신대학교 평생대학원 정보과학과
e-mail:amansun@samho.co.kr

A RDB storage model for XML-based public documents

Man-Sun An*, Eun-Bae Lee*
*Dept. of Computer Science, Korea National Open University

요 약

공문서는 정부가 제정한 공문서 표준 DTD 및 XML 문법에 따라 작성되고 있으나, 일반 문서와 동일하게 정부의 디렉토리시스템에 문서 단위로 저장 관리되고 있다. 그러나 구조 정보를 포함하고 있는 XML 문서를 보다 효과적으로 활용하기 위해서는 논리구조 단위로 정보를 저장 관리할 필요가 있다.

본 논문은 XML로 작성되는 공문서의 특성을 파악하여 데이터베이스로 저장할 때 적합한 모델을 제안한다. 대부분의 공공기관이 사용하고 있는 관계 데이터베이스시스템(RDBMS)을 사용하였고, 데이터 중심과 문서내용 중심의 성격을 동시에 가지는 공문서의 특성을 고려하였다. 제안하는 저장 모델은 메타데이터를 표현하는 부분은 정적인 테이블을 사용하여 구조정보와 내용을 함께 저장하고, 문서 내용 중심의 본문 부분은 분할하지 않고 저장하는 변형된 분할 저장 방식이다.

제안하는 저장 모델을 활용하면 기존 RDBMS로 개발된 여러 응용 시스템과의 연동이 가능하고, XML 데이터 저장/검색을 위한 새로운 데이터베이스시스템의 구입 없이 XML 전자문서를 효율적으로 관리할 수 있다는 장점이 있다.

1. 서 론

XML이 인터넷상의 정보 표현 및 데이터 교환의 표준 포맷으로 자리 잡아감에 따라 XML문서를 저장하는 방법에 대한 연구도 많이 진행되어 왔다. 그러나 현재까지 제안된 XML 데이터 관리 기법 및 시스템들은 각각이 고유한 장단점을 지니고 있어 아직 어느 방법이 최선이라고 말하기는 어렵다.

행정자치부는 XML을 전자문서 유통의 표준으로 확정하여 시행 중에 있다.[1] 공문서는 다른 분야의 문서에 비해 그 발생량이 많고, 재활용을 위한 보존기간이 월등히 길기 때문에 향후 XML로 작성된 공문서의 폭발적 증가가 예상된다. 따라서 이들 XML 전자문서들을 관리하기 위해 공문서 특성과 각 기관의 전산 환경을 반영한 효율적인 저장 기법에 대한 연구가 필요하다.

XML문서는 기존의 문서와는 다르게 트리 구조로 나타낼 수 있는 준구조적(semi-structured)인 형태이기 때문에, 이러한 특성을 고려한 시스템을 이용하여 저장하는 것이 이론적으로는 이상적이다. 그러나 XML을 저장하는 방법 중 관계 데이터베이스 시스템(RDBMS)에 저장하는 방법이 가장 많이 연구되고 있는 이유는 RDBMS가 전 세계적으로 가장 많이 사용되고 있으며 여러 기능들이 오랫동안 개발되어 왔기 때문이다.

현재 XML로 작성된 공문서는 정부의 디렉토리 시스템을 이용하여 문서 단위로 저장 관리되고 있다. 그러나 XML 문서를 보다 효과적으로 활용하기 위해서는 논리구조 단위로 정보를 저장 관리할 필요가 있다. XML 데이터 저장/검색 시스템을 위해 새로운 데이터베이스시스템을 구입하는 경우엔 기술적, 비용적 부담이 따르며, 기존 RDBMS를 기반으로 개발된 여러 응용 시스템과의 연동을 고려해야 한다. 따라서 본 논문에서는 XML기반의 공문서를 RDBMS에 저장하고자 할 때의 효과적인 RDB 스키

마를 제안 한다.

본 논문의 구성은 다음과 같다. 2장에서 저장 기법 및 RDBMS를 이용한 저장 방법에 대한 관련 연구를 알아보고, 3장에서 XML기반 공문서의 특성을 고려한 RDBM로의 저장 스키마를 제안하고, 프로토타이핑을 통해 제안된 기법이 적합한 저장 방식임을 파악해 보며, 마지막 4장에서는 결론과 향후 연구 방향에 대해 기술한다

2. 관련 연구

2.1 XML 저장 기법

XML 문서의 저장 관리에서 저장 공간 사용의 효율과 데이터의 검색 성능은 XML 데이터를 어떠한 형태로 저장하는 가에 따라 다르다. 현재까지 제안된 대표적인 XML 저장 관리 기법은 저장 미디어에 따라 파일시스템 이용 기법, 데이터베이스 시스템 이용 기법, XML 전용 관리 시스템 이용 기법으로 분류할 수 있다.[2]

이중 데이터베이스 기반의 XML 문서 저장 기법에 대한 연구는 XML 데이터를 테이블 형태로 저장하는 관계 데이터 모델이 주를 이루고 있다. 이는 RDBMS의 특성상 테이블과 튜플의 수가 많아 시스템의 성능이 저하되는 단점이 있지만, RDBMS가 현재 널리 사용되고 있어 전반적인 확산이 빠르다는 장점이 있기 때문이다. 한편, XML 데이터가 객체지향 데이터 모델과 많은 유사성을 가지기 때문에 객체지향 데이터베이스 시스템(OODBMS)를 이용하는 기법에 대한 연구도 최근 활발하지만, OODBMS은 체계적인 이론이나 표준의 미비 등으로 신뢰성과 안전성에서 아직 미흡하다.

2.2 RDBMS 기반의 저장 방법

XML 문서를 관계형 데이터베이스에 저장하는 방법은 XML 인스턴스를 엘리먼트 별로 나누어 저장하는 분할 저장 모델과 엘리먼트와 인스턴스를 한꺼번에 저장하며 영역대수기법을 사용하는 비분할 저장 모델로 구분할 수 있다.

분할 저장 모델은 인스턴스를 엘리먼트 별로 나누어서 저장하는 모델로 XML 문서의 구조 정보를 표현하는 DTD를 기반으로 데이터베이스의 스키마를 생성하여 저장하는 DTD 종속적인 방법[4]과, DTD를 사용하지 않고 미리 정해진 데이터베이스의 스키마에 따라 저장하는 DTD 독립적인 방법[5]으로 구분된다. 이 모델은 문서의 일부 내용들이 수정되었을 때 관계되는 노드들만 수정하면 되므로 문서의 편집 및 관리가 쉽다는 장점이 있지만, 문서의 내용을 추출하고자 할 때 각 단말 노드들을 순회하며 통합하는 과정에서 시스템의 성능을 저하시키는 문제가 발생한다.

비분할 저장 모델은 가상 분할 모델이라고도 불리며, XML 인스턴스 전체를 BLOB 형태로 저장한 다음, 각각의 단말 노드는 오프셋 정보를 가지고 접근하는 방식이다. 이는 인스턴스를 한꺼번에 저장하였기 때문에 통합 과정이 필요 없어 인스턴스 참조를 빨리 할 수 있지만, 내용의 일부만이 수정되었을 때도 인스턴스 전체를 재구성해야 한다는 큰 단점이 있다.

3. XML기반 공문서의 RDB 저장 모델

3.1 제안 모델의 개요

XML 문서의 데이터베이스로 저장 목적이 데이터 저장인지 또는 문서 자체 저장인지는 저장 모델에 중요한 영향을 미친다. 예를 들어, 데이터중심의 XML문서에서는 데이터 자체만 중요하게 다루면 되지만, 문서의 내용이 중요한 XML문서를 저장할 때는 문서의 물리적인 구조 정보의 저장도 매우 중요하게 다루어져야 한다.[3]

XML기반 공문서는 DTD와 인스턴스가 각각의 파일로 작성된 “유효한(Valid)” 문서이다. 또한 각 개별 공문서에서 공통으로 사용되는 구조 및 엘리먼트와 속성은 표준 DTD로 규정되어 있다. 따라서 DTD를 관계형 DBMS의 스키마로 매핑하고, XML 데이터를 DBMS의 테이블의 튜플로 저장하는 DTD 종속적인 분할저장기법의 적용이 가능하다.

그러나 공문서 표준 DTD는 엘리먼트 수가 많고 계층 단계가 깊기 때문에 DTD 종속적인 분할저장기법만을 이용하여 저장하는 경우에는 검색이나 재조합에 오히려 불리할 수 있다. 또한, 본문 부분은 문서 내용 중심으로 아래와 예와 같이 혼합 내용 요소 형식을 사용함에 따라 분할 저장 모델링에 적합하지 않다.

예) <!*ELEMENT a* (*#PCDATA* | *img* | *i* | *b* | *u* | *sub* | *sup*)*>

따라서 DTD 종속적이지만 정적인 구조의 테이블을 사용하며, 본문 부분은 분할하지 않고 저장하는 구조의 변형된 저장 모델을 제안한다.

본 논문에서 제안하는 저장 모델은 <표1>과 같은 5개의 테이블을 사용한다. 이 5개의 테이블은 DTD나 XML 인스턴스의 구조와 상관없이 사용될 수 있도록 설계되었다.

<표1> XML 저장을 위한 RDB Table

No.	테이블명	주요 저장 내용
①	Dtd_ID	DTD의 제목과 전체 내용
②	Doc_ID	XML 인스턴스의 제목과 본문 부분 내용
③	Ele_ID	DTD를 파싱하여 추출한 엘리먼트와 식별자
④	Path_ID	XML 인스턴스를 파싱하여 생성한 구조정보
⑤	ELEDB	XML 인스턴스를 엘리먼트 단위로 분할 저장

3.2 테이블 스키마 구조

XML 문서를 저장하기 위한 데이터베이스 스키마를 어떻게 생성하는가에 따라 전체 시스템의 성능에 많은 영향을 주므로 저장과 관리를 효율적으로 할 수 있는 스키마 생성방법이 요구된다. 본 논문에서는 DTD나 인스턴스의 수정, 갱신, 복원 등을 논하고 있지 않지만, 이러한 점이 충분히 고려된 스키마 구조를 갖도록 하였다.

제안하는 저장 모델에서는 <표1>의 테이블 표현 순서와 같이 XML 전자문서를 RDB에 저장하는 과정을 거친다. 각 테이블의 스키마 구조와 저장 과정은 다음과 같다.

① Dtd_ID Table

먼저 저장하고자 하는 XML 인스턴스가 참조하는 DTD가 Dtd_ID Table에 존재하는지 여부를 확인하여, 없는 경우 테이블에 저장한다. DTD의 전체 내용은 테이블에 텍스트 형태로 저장된다. DTDID는 여러개의 DTD가 존재할 때 구분하기 위한 것으로 고유한 값을 가진다.

<표2> Dtd_ID Table의 구성 요소

엔티티명	용도 및 저장 내용
DTDID	DTD를 구별하는 식별자
TITLE	DTD의 명칭
DTD	DTD 전체 내용

② Doc_ID Table

저장하고자 하는 XML 인스턴스의 본문 부분 내용 전체가 텍스트형태로 저장된다. 인스턴스의 기본적인 사항은 직접 입력하여 저장한다. 여기서도 각각의 XML 인스턴스를 구분하기 위해 인스턴스별 DOCID가 사용되며, 공문서 특성상 문서 개정시 원래 문서를 보존하여야 함에 따라 DOCREV를 사용하여 구분하도록 하였다.

<표3> Doc_ID Table의 구성 요소

엔티티명	용도 및 저장 내용
DOCID	XML 인스턴스를 구별하는 식별자
DOCREV	문서의 개정 번호
TITLE	문서의 명칭
FILENAME	문서의 파일명 및 위치
CONTENT	pubdoc/body/content 엘리먼트의 전체 내용
DTDID	사용하는 DTD 구분

③ Ele_ID Table

Doc_ID Table에서 저장하고자 하는 XML 인스턴스가 사용하는 DTD의 DTDID를 확인하고, 없다면 Dtd_ID Table의 DTD 내용을 읽어 들여 엘리먼트를 추출한다. 추출된 엘리먼트의 이름은 일련번호로 주어지는 엘리먼트 식별자(EID)와 함께 저장된다. 여러 개의 DTD에서 같은 이름의 엘리먼트가 있는 경우를 고려하여 해당 DTDID도 같이 저장되어야 한다.

<표4> Ele_ID Table의 구성 요소

엔티티명	용도 및 저장 내용
EID	엘리먼트를 구별하는 식별자 (00,01...ZZ)
Element	엘리먼트 명
DTDID	사용하는 DTD 구분

④ Path_ID Table

XML 인스턴스를 RDB에 저장하기 위해서는 먼저, 주어진 XML 인스턴스의 구조정보에 대해 유효성 검사가 선행되어야 한다. 이때, 인스턴스에서 사용되는 DTD의 엘리먼트 정보가 트리 형태의 구조정보로 생성되게 된다. 생성된 구조정보는 부모와 지식간 관계의 경로를 가진다.

<표5> Path_ID Table의 구성 요소

엔티티명	용도 및 저장 내용
EPID	XML 인스턴스의 엘리먼트 식별자
Element	엘리먼트 명
DTDID	엘리먼트가 생성된 DTD 구분

엘리먼트의 식별자인 EPID는 루트 엘리먼트부터 해당 엘리먼트까지의 경로를 나타낸다. 이는 Ele_ID Table의 EID를 조합한 형태로, 해당 엘리먼트의 직상위 부모 엘리먼트의 EPID를 상속받아 해당 엘리먼트의 EID를 끝에 추가하여 생성한다.

⑤ ELEDB Table

마지막 과정은 XML 인스턴스의 내용을 RDB에 저장하는 과정이다. 이 테이블에는 인스턴스의 엘리먼트별로 구조정보와 내용이 함께 저장 된다.

<표6> ELEDB Table의 구성 요소

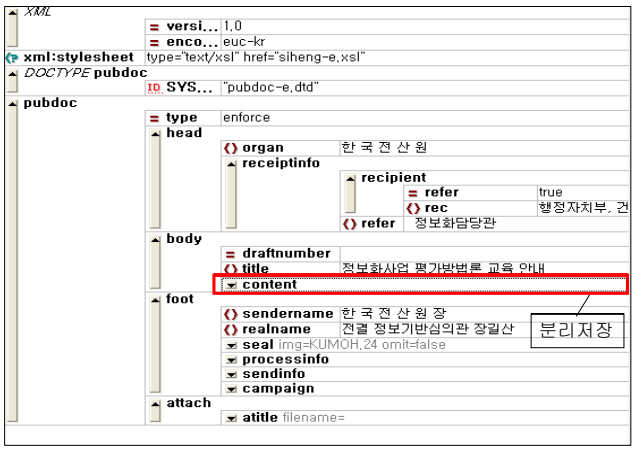
엔티티명	용도 및 저장 내용
SNO	XML 엘리먼트를 구별하는 식별자
DOCID	인스턴스 식별자
DOCREV	개정 번호
EPID	엘리먼트 식별자 (구조정보)
DTDID	엘리먼트가 생성된 DTD 구분
Element	엘리먼트명
Attribute	엘리먼트 속성과 그에 해당하는 값
Entity	엘리먼트가 가지는 값

각 엘리먼트는 엘리먼트명, 엘리먼트 속성, 엘리먼트 값이 각각 Element, Attribute와 Entity에 저장

된다. 여기서 엘리먼트명은 구조정보로 저장되는 EPID와 DTDID를 키로 Path_ID Table를 참조하면 알 수 있으므로 저장할 필요는 없지만, 질의처리 과정의 효율성을 고려하여 저장하도록 하였다. 그리고 DOCID와 DOCREV는 이 테이블을 여러 개의 인스턴스 저장에 공통적으로 사용하기 위해서 필요하다.

인스턴스에서 엘리먼트가 나타난 순서를 저장한 일련번호인 SNO는 저장되는 엘리먼트의 유일한 식별자로 인스턴스 내에서 반복되어 나타나는 엘리먼트에 대해 선·후행 관계를 표현한다. 이는 내용 중심의 본문 부분 외에도 데이터 중심과 문서내용 중심의 성격을 동시에 가지고 있는 부분이 있기 때문에 XML 인스턴스로 복원해야 할 경우에는 필요하다.

3.3 저장 결과 및 검토



<그림1> 행정기관의 XML 전자문서의 구조

<그림 1>과 같은 구조를 가진 공문서를 제안한 모델로 저장하면, 본문 부분은 Doc_ID Table에 별도로 저장되고, 나머지 메타데이터 부분은 인스턴스의 구조정보와 내용이 ELEDB Table에 <그림 2>와 같이 저장된다.

Table: ELEDB

SN	DD	DC	EPID	DT	Element	Attribute	Entity
1	10	1	01	17	pubdoc	type="enforce"	
2	10	1	0102	17	head		
3	10	1	010222	17	organ		한 국 전 산 원
4	10	1	01022206	17	receiptinfo		
5	10	1	0102220623	17	recipient	refer="true"	
6	10	1	010222062324	17	rec		행정자치부, 건~~
7	10	1	0102220625	17	refer		정보화담당관
8	10	1	0103	17	body	draftnumber=""	
9	10	1	010318	17	title		정보화사업 평~~
10	10	1	010319	17	content		
⋮	⋮	1	⋮	17	⋮	⋮	⋮

구조정보 | 속성 및 값

<그림2> 구조정보와 내용이 저장된 결과

이 모델은 엘리먼트 별로 테이블을 생성하지 않고 정적인 테이블을 사용하기 때문에 다양한 공공기관의 XML기반 공문서 저장에 공통적으로 활용할 수 있다. 또한 문서의 구조 정보와 내용이 한 테이블에 같이 저장되기 때문에 질의 처리 시 테이블간의 조인연산이 없다는 장점을 가지고 있다. 그리고 엘리먼트의 추가나 삭제 시 엘리먼트의 경로 정보를 가지고 있는 EPID에 간단한 조작을 가하면 쉽게 구현될 수 있는 점도 제안 모델이 갖는 부가적인 장점이다.

4. 결 론

공문서는 표준 DTD 및 XML 문법에 따라 작성되고 있기 때문에 DTD 종속적인 분할 저장 기법의 적용이 가능하다. 그러나 공문서 특성상 본문 부분은 문서 내용 중심의 성격을 가지므로 이 방법만을 이용하는 경우, 검색이나 재조합에 오히려 불리하다.

본 논문에서는 이러한 문제점을 개선한 변형된 분할 저장 모델을 제안하였다. 이 저장 모델은 메타데이터를 표현하는 부분은 정적인 테이블을 사용하여 구조 정보와 내용을 함께 저장하고, 문서 내용 중심의 본문 부분은 분할하지 않고 저장하여 통합관리하게 하는 방법이다.

제시된 저장 모델을 활용하면 메타데이터 저장 부분을 기존 관계 DBMS의 응용 시스템과 연동하여 활용하는 것이 가능하고, XML 데이터 저장/검색 시스템을 위해 새로운 데이터베이스 시스템 구입에 따른 추가적인 비용이 없어 공공기관에서 XML 전자문서를 효율적으로 관리할 수 있다.

향후 연구과제는 제안 모델을 이용한 저장 시스템의 구현이다.

참고문헌

[1] 행정자치부, 행정기관간 전자문서유통 표준, 2002.

[2] 민준기 외 3인, “다양한 저장소에서의 효율적인 XML 저장기법에 대한 연구”, 정보과학회 데이터베이스연구, 제19권 제1호, 2003.

[3] Ronald Bourret, "XML and Databases.," 2003. http://www.rpbourret.com/xml/XMLAndDatabases.htm

[4] J. Shanmugasundaram, K. Tufte, C. Zhang, H. Gang, D. J. DeWitt, and J. F. Naughton, "Relational databases for querying XML documents: Limitations and opportunities.," Proceedings of the 25th VLDB Conference, pp.302-314, 1999.

[5] D. Florescu & D. Kossmann,. "Storing and Querying XML Data using an RDBMS.," Bulletin of the IEEE Computer Society Thechnical Committee on Data Engineering. 22(3), pp. 27-34, 1999.