濟州大學校 尖端技術硏究所 論文集, 第13卷, 1號, pp.85-90, 2002. 6
Cheju Nat'l Univ. Res. Inst. Adv. Tech. Jour., Vol.13, No.1, pp.85-90, 2002. 6

XML 문서의 관계형 데이터베이스 게이트웨이 설계 및 구현

홍 동 현* · 김 정 희** · 곽 호 영***

The Design and Implementation of Gateway between XML Document and Relation Database

Dong-Hyun Hong*, Jeong-Hee Kim** and Ho-Young Kwak***

ABSTRACT

In this paper, using the characteristics of XML document, the method that separately stores the structures and contents of XML documents, is proposed for the relational database(RDB), which is representative of the industrial world. The design of the storage architectures to efficiently store XML documents and easily reconstruct the original documents in RDB, is based on E-R modeling. In order to practically store XML documents in RDB, XML Parser and Inserter that handle XML documents according to the proposed storage architecture, were implemented. And when the structures and the contents of XML documents are separately queried, to turn out a complete result, Extractor and XML Composer were also implemented. Moreover, the convenient user interface was designed to easily store and query XML documents.

Key Words : XML(eXtensible Markup Language), relational database, E-R modeling

Ⅰ. 서 론

인터넷의 발전이 진전되면서 이 기종간의 시스템에서 작성된 문서에 대한 데이터베이스의 구축과 검색 그리고 상호 교환이 중요성이 높아지고 있다. 이에 따라, 다양한 형식으로부터 원하는 정보를 효율적으로 관리, 공유하기 위해서는 문서를 일관성 있게 구조화하는 기술의 필요성이 대두되었고, 1986년 ISO (International Organization for Standardization)에서는 SGML(Standard Generalized Markup Language)이라는 문서의 논리구조를 표현하는 국제적인 표준안을 마련했다[1].

하지만 SGML은 다양한 기능에도 불구하고, 그 구성이 너무 복잡하다는 단점을 가지고 있고, 이를 해결하고자 HTML(Hypertext Markup Language)이 제기되었지만 이는 제한된 태그로 인해 한계를 가지고 있어서 사용이 부적당하였다. 이에 따라 W3C에서는 일반화된 마크업, 복합구조, 검증의 특성을 그대로 지원하는 한편 사용자에 의한 확장성을 가지고 있는 XML

* (주) 필링크 연구원
Assistant Research Engineer, Feelink Co. Ltd.
** 제주대학교 대학원
Graduated School, Cheju Nat.l Univ
*** 제주대학교 통신·컴퓨터공학부, 첨단기술연구소
Faculty of Telecommunication & Computer Eng., Research Institute of Advanced Technology, Cheju Nat'l Univ.

(eXtensible Markup Language)을 제안하였다[2-4].

그래서, 최근의 웹(Web) 또는 디지털 전자 도서관 시스템, CALS(Commerce At the Light Speed), 수학 분야, 채널 기술의 CDF(Channel Definition Format), 이동 통신에서의 HDML(Handheld Device Markup Language)들과 같은 환경에서 많은 문서들이 XML 마크업 언어를 적극 활용하고 있으며 이러한 언어로 문서들을 표현함으로써 문서의 논리적인 구조를 표현할 수 있고, 그럼으로써 문서들을 데이터베이스에 저장하고 검색 할 수 있는 필요성들이 대두되고 있다[5-8].

XML문서에서는 문서의 구조적 특성이 적절한 태그(Tag)에 의해서 분리되고 재현 양식에 관한 정보는 별도의 스타일 문서에 의해 제공되므로 마치 데이터베이스에 질의를 가하는 것처럼 문서를 검색할 수 있으며 트랙잭션 단위의 인트라넷 데이터의 표현도 용이하다는 장점을 가지고 있기 때문에 웹 상에서의 다양한 형식이 전자문서를 표시하기에 매우 적합하다. 또한 기존 HTML을 확장·보완하였기 때문에 HTML을 그대로 사용하면서 더욱 복잡하고 다양한 구조적 문서를 작성할 수 있다[9].

XML문서는 SGML의 다양한 구조의 문서를 작성할 수 있는 특징과 HTML처럼 웹 상에서 쉽게 사용할 수 있다는 장점으로 인하여 향후 인터넷 문서의 표준으로 사용될 것으로 예측되며 다양하고 방대한 문서들이 생산될 것으로 보인다. 그에 따라 XML문서의 양이 방대해지고 사용자는 원하는 XML문서를 쉽게 검색하고 관리할 수 있기를 원한다. 그러기 위해서는 XML문서를 데이터베이스에 저장하여 관리하여야 한다.

II. 게이트웨이 설계

2.1. 파서

본 시스템에서는 SAX 파서를 사용한다. 파서의 역할은 XML 문서를 순차적으로 읽어 내려가면서 각각의 요소들의 연관관계를 데이터베이스에 입력하는 과정을 반복하게 된다. SAX 파서를 사용하는 것은 DOM 파서를 사용할 때 메모리 사용량과 속도의 효율성이 떨어지는 문제점을 해결하기 위함이다. 그러나 SAX는 이와 달리 DOM 생성 작업이 단순하지 않다. 실제로도 DOM 파서는 SAX 파서를 이용하되, DOM 생성부분을 추가하고 있는 경우가 대부분이다. SAX는 단순하게 어떤 요소를 읽었는지 등의 정보를 줄 뿐, 현재 이 요소가 어떤 요소의 일부인가 등의 문맥정보를 자동으로 유지해주지 않는다. 이는 사용자가 문서 구조로부터 정보를 추출하기 위해 보다 많은 일을 해야 함을 의미한다. 결국 SAX를 사용하는데 있어 성패는 SAX의 단점을 보완할 만한 도구를 적절히 만들 수 있느냐에 달려있다.

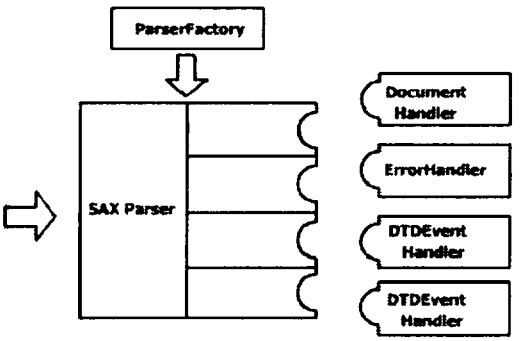


Fig. 1. Structure of SAX parser.

2.2. XML 문서 설계

XML 문서는 트리 구조이며 기존의 관계형 데이터베이스는 선형구조를 하고 있다. 따라서 XML의 내포구조를 선형구조로 나타내기 위해서는 XML문서를 어떻게 저장할 것인가에 대한 모델설계가 필요하다. 우선 XML 문서를 데이터베이스에 저장하기 위해서는 DTD문서가 어떤 유형으로 존재하고 있는지 조사해야 한다. 그 유형은 아래와 같다.

- ▶ DTD가 별도로 존재하는 문서
- ▶ DTD가 XML문서에 내포되어 있는 문서
- ▶ DTD가 없는 문서

여기서 DTD를 이용해 XML문서의 유형을 나누기는 하지만, DTD는 문서가 가진 특성을 규정한 문서이기 때문에 어떤 특별한 의미를 가지는 것이라고 보

기는 어렵다. 단지 XML 문서의 적합 여부만 가려내는 문법이다. 하지만 본 연구에서는 XML 문서가 관계형 데이터베이스에 저장이 되었을 경우, XML 문서의 원형 복구뿐 만 아니라 DTD의 유형에 의한 문서 생성도 고려하고 있기 때문에, DTD의 저장방식도 유형에 따라 달라져야 한다. DTD가 별도로 존재하는 경우에는 DTD의 이름정보를 저장하도록 하였고, DTD가 XML 문서 내에 내포되어 있는 경우에는 DTD 자체를 저장하며, DTD가 없는 경우는 특별한 처리를 해주지 않도록 하였다. DTD가 처리된 후에는 XML문서 자체에 대한 처리를 해주어야 되는데 XML 문서의 구조화 특성을 이용하여 PI, 엘리먼트, 태그, 속성들을 순서대로 처리되도록 하였다.

2.3. XML 문서의 맵핑 요소 모델

XML문서와 관계형 데이터베이스와 맵핑하는 것은 두 가지 모델이 다른 줄기를 가졌기 때문에 몇 가지 문제가 일어날 수 있다. E. FCodd가 제안한 관계형 데이터베이스 모델은 XML 데이터 모델과 상당히 다르다. 이것은 기본적으로 3계층 모델이다. 데이터베이스는 레코드의 구성으로 된 테이블의 집합이고, 레코드들은 몇 개의 필드로 구성되어 있다. 레코드 필드의 값은 값들의 리스트에 속하면서 하나만 가져야 한다. 반면 XML 문서는 불명확한 깊이를 가진 트리 구조이다. 따라서 XML 관점을 관계형 모델로 표현하는 것은 더욱 더 어려운 것이다. 그러나 XML 문서가 트리 구조임에도 불구하고 무난한 네임스페이스를 가지고 있어서 같은 네임스페이스안에 엘리먼트 타입이 한번이상 선언되지 않으며, 모든 엘리먼트는 같은 영역에 선언되기 때문에 다른 이름을 가지게 된다. 엘리먼트 타입은 애트리뷰트를 선언할 수 있고 다른 엘리먼트들이 일반적으로 애트리뷰트 네임으로 쓰이게 된다. 따라서 모든 테이블 네임은 다르고 같은 필드이름은 다른 테이블이어야만 하는 관계형 모델의 제한(restriction)과 비슷하다. 그렇기 때문에 테이블은 엘리먼트에 필드값에 맵핑시킬 수 있다고 볼 수 있다. 따라서 테이블의 각각의 행은 한 엘리먼트에 일치시키고 애트리뷰트의 값은 필드의 값과 일치시키는 것이 가능하게 된다.

XML 문서를 관계형 데이터베이스에 맵핑하기 위한 요소 모델은 다음과 같다.

▶ XML 선언부 요소 : XML 문서의 전반적인 정보를 담는 선언부에 대한 개체.

- Encoding정보, 버전(Version), DTD를 구성하는 형태에 대한 정보인 RMD(Required Markup Declaration)를 가지게 된다.

▶ File 요소 : XML 문서가 관계형 데이터베이스에 저장하기 이전 문서를 만들기 위한 Key를 제공하게 된다.

▶ DTD 요소 : XML문서가 관계형 데이터베이스에 저장된 후에 동일한 DTD에 의한 문서를 추출하기 위해 필요한 테이블이다.

▶ PI요소 : 엘리먼트 속성에 의해서 위치가 결정된다. 이 요소를 Index값에 의해 유일하도록 하며 내용과 처리지시를 따로 저장되도록 한다.

▶ 엘리먼트 요소 : XML문서는 모두 엘리먼트에 의해서 취급되게 된다. File값은 엘리먼트가 속해있던 file의 인덱스를 가리키며, DTD 요소는 DTD의 인덱스를 가리킨다. 다만 DTD가 필요 없는 경우이면, null값을 가지게 된다. 그리고 관계형 데이터베이스에서는 트리 구조로 표현되기 때문에 깊이 값으로 처리된다.

▶ 속성요소 : 엘리먼트가 가리키는 것의 속성을 정의하도록 한다.

▶ 속성내용 : 속성값이 가질 수 있는 것은 제한이 없다. 속성값과 같이 묶어버릴 경우, 이 점을 처리하기가 어려워진다. 그래서 따로 테이블을 만들어 부여함으로서, 추가, 삭제, 편집이 보다 쉽게 되도록 한다.

▶ 내용요소 : 한 엘리먼트에 속하는 데이터들은 여러 개가 있을 수 있기 때문에 어떤 엘리먼트와 관계되어 있는지를 나타내는 엘리먼트 요소와 실질적인 데이터를 표현하도록 한다.

2.4. 테이블 설계 및 관계설정

XML문서를 저장하기 위해서는 구조적 정보를 데이터베이스에 저장할 수 있어야 하고 저장된 문서에 대해 탐색과 엘리먼트에 대한 질의, 구조적 질의가

가능하게 설계되어야 한다. 따라서 트리에 대한 구조 질의를 위해서 리스트를 이용하여 자식 노드를 가리키는 일반적 트리 구성 방식과 특정 트리를 기준으로 구조적 질의를 위한 DFS(Depth First Search) Numbering 방식을 이용하여 트리 구조를 데이터베이스에 함께 표현하도록 한다.

XML 문서를 데이터베이스에 저장하기 위해서는 먼저 XML 문서를 데이터베이스에 어떻게 저장하는가에 대한 모델설계가 필요하다.

모델설계는 XML문서의 기본구조를 살피고 그 기본구조에 따라 XML 문서를 표현하기 위한 개체정의와 개체간의 관계를 규정짓는 과정이 이루어진다.

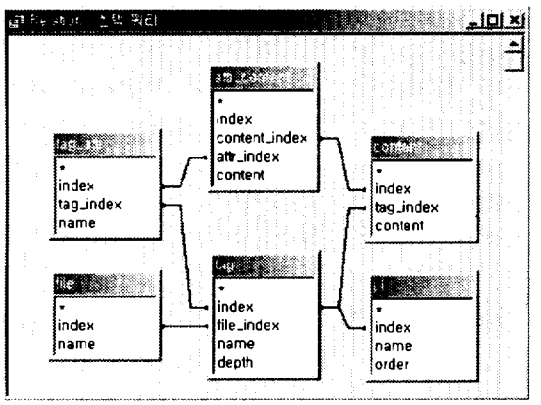


Fig. 2. Relation of table.

각각의 내용을 테이블로 만들어 보면 다음과 같다.

▶ file : index, name

▶ tag : index, file_index, pre_tag, name, depth

▶ tag_attr : index, tag_index, name

▶ attr_content : index, content_index, attr_index, content

▶ content : index, tag_index, content

▶ pi : index, name, order

▶ File 테이블

- index : 파일을 나타내는 유일한 값
- name : 파일 이름

▶ File 테이블

- index : 파일을 나타내는 유일한 값
- name : 파일 이름

▶ Tag 테이블

- index : 동일 DTD에서 유일한 element를 나타내는 값. 같은 name값을 가지고 있지만 depth가 다르면 다른 element이므로 다른 값을 부여받게된다.
- file_index : 어떤 XML파일의 element를 나타내기 위한 값
- pre_tag : 바로 이전의 element 값
- name : element의 이름
- depth : 각각의 element들의 깊이 값.

▶ Tag_attr 테이블

- index : 인덱스 값
- tag_index : 어떠한 element의 attribute인지 나타내준다.
- name : attribute값

▶ Attr_content 테이블

- index : 인덱스 값
- content_index : content 값마다 다른 attribute를 가질 수 있기 때문에 content에 종속된다.
- attr_index : tag_attr테이블과 연결하기 위한 값
- content : attribute값

▶ Content 테이블

- index : 인덱스 값
- tag_index : element의 인덱스
- content : 실질적인 내용

▶ Pi 테이블

- index : 인덱스 값
- name : Pi 값
- order : Pi값을 실행시킬 외부 처리기 값

III. 시스템 구현

XML 문서가 입력으로 들어오면 문서를 한 줄씩 읽으면서 테이블을 튜플 단위로 분류한다. 먼저 XML문서 선언부를 읽어서 XML 선언부 테이블에 해당하는 정보를 배열에 저장한다. XML 문서형식 정의부(DTD)가 있는지 확인하고 같이 들어있다면 DTD 부분을 분리해 낸다. 이를 파일로 만들어서 따로 저장한다.

다음으로 문서부(D1)의 모든 줄을 차례차례 읽는데 이것이 엘리먼트인 경우에는 필요한 구조 정보를 구조 테이블 배열에 저장하고, 이 중 구성이 엘리먼트_요소 테이블에 해당하는 정보를 엘리먼트 테이블 배열에 저장한다. 하지만 동일한 DTD나 File이 있을 때에는 추가를 하지 말아야 할 것이다. 또한 동일한 엘리먼트일 경우 무조건 추가하는 것이 아니라 다른 깊이에 존

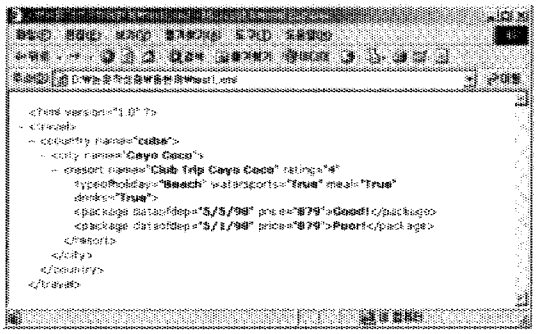


Fig. 3. XML document.

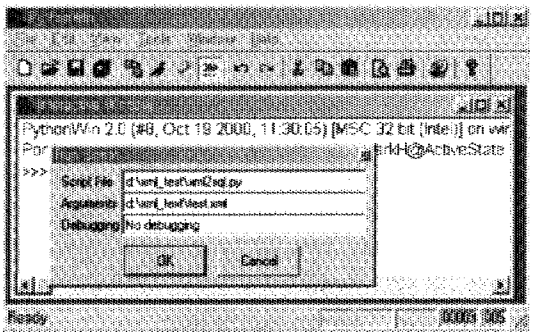


Fig. 4. Execution.

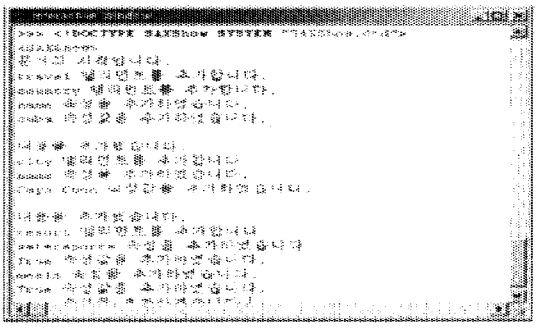
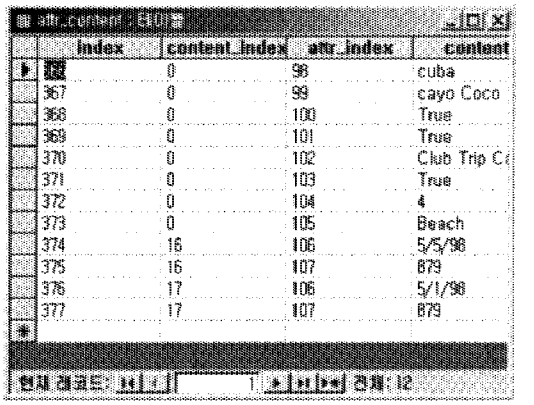
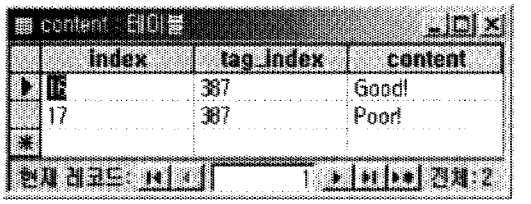


Fig. 5. Results.



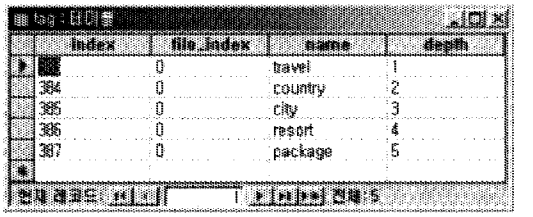
index	content_index	attr_index	content
1	0	98	cuba
367	0	99	cayo Coco
368	0	100	True
369	0	101	True
370	0	102	Club Trip C
371	0	103	True
372	0	104	4
373	0	105	Beach
374	16	106	5/5/98
375	16	107	879
376	17	106	5/1/98
377	17	107	879

Fig. 6. Attr_content table.



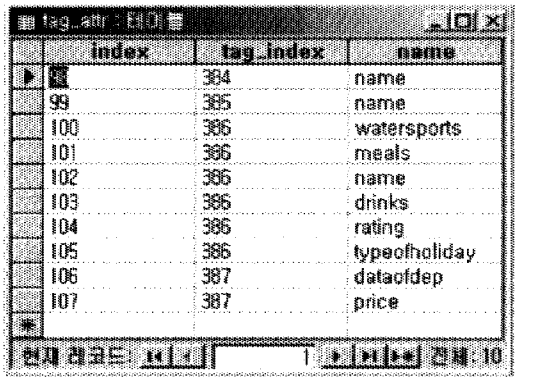
index	tag_index	content
16	387	Good!
17	387	Poor!

Fig. 7. Content table.



index	file_index	name	depth
1	0	travel	1
384	0	country	2
385	0	city	3
386	0	resort	4
387	0	package	5

Fig. 8. Tag table.



index	tag_index	name
98	384	name
99	385	name
100	386	watersports
101	386	meals
102	386	name
103	386	drinks
104	386	rating
105	386	typeofholiday
106	387	dataofdep
107	387	price

Fig. 9. Tag_attr table.

재할 경우에만 추가한다. 만약에 엘리먼트가 속성들을 가지고 있으면 속성_리스트_값 테이블에 해당하는 정보를 속성_리스트_값 테이블에 저장한다.

SAX파서의 동작에 따라서 문서부(DI)를 모두 읽어서 처리하는 것이 아니라 엘리먼트별로 한 줄씩 처리하여 데이터베이스에 맵핑한다.

Ⅳ. 결 론

XML 문서는 이기종 시스템간의 데이터 이동과 같은 미들웨어 등 다양한 응용분야에서 수요가 증가함에 따라 효율적인 저장과 관리가 필요하게 되었다. 이를 위해 XML 문서를 데이터베이스에 저장해야 하며, 본 논문에서는 XML 문서를 구성요소별로 분리하여 관계형 데이터베이스에 XML문서를 저장하고 검색, 수정할 수 있는 방안에 대하여 연구하였다. 또한 XML 문서를 테이블에 완전히 분리 저장하여 데이터의 중복이 없도록 하였고, XML 문서를 다시 재구성할 수 있도록 하기 위해 구체적인 모델링 단계를 거쳐서 테이블 스키마를 작성하였다. XML 문서를 저장함에 있어 기존연구에서는 내용에 해당하는 부분을 Long Type이나 BLOB으로 저장함에 따라 데이터베이스의 LIKE와 같은 문자 비교를 할 수 없으므로 내용검색을 하기 위해 별도의 정보검색기를 구축하여야 한다. 그에 반해 본 연구에서는 엘리먼트의 내용부분을 한 줄 단위로 저장함으로 데이터베이스의 LIKE와 같은 문자 비교를 가능하게 하였다. 그러므로 내용 검색을 간단하고 정확하게 수행할 수 있도록 하였다.

향후 연구과제로는 XML에 제공하는 링크의 기능과 외부로 보여지기 위한 포맷팅 처리를 지원하는 XML 저장 시스템의 설계와 구현이 진행될 필요가 있다.

참고문헌

1) 손정한, 이희주, 장재우, 심부성, 주종철, 1998, "구조화된 문서를 위한 정보검색시스템의 설계 및 구현", '98 동계 데이터베이스 학술대회 논문집 제14권 1호, pp.102-106.

2) 연재원, 장동준, 김용훈, 이강찬, 이규철, 1999, "효율적인 검색 지원 SGML 저장 관리기의 설계 및 구현", '99 한국 데이터베이스 학술대회 논문집 15권 1호, pp.136-143.

3) 유재수외 8명, 1999, "전자도서관 표준문서관리를 위한 XML 저장관리기 기술 개발", 케이오텍 최종보고서.

4) Charles L. A. Clarke, Gordon V. Cormack, Forbes J. Burkowski, 1995, "An Algebra for Structured Text Search and a Framework for its Implementation. The Computer Journal Vol.38, No.1, pp.43-56.

5) Dongwook Shin, Hyuncheol Jang, and HongLan Jin, 1998, "Bus : An Effective Indexing and Retrieval Schema in Structured Documents", ACM, pp.235-243.

6) Francois, 1996, "Generalized SGML repositories: Requirements and Modeling", Computer Standards & Interfaces.

7) Tuong Dao, Ron Sacks-Davis, James A.Thom, 1997, "An indexing scheme for structured documents and its implementation", Proceedings of the 4th International Conference on DATABASE Systems for Advanced Applications, Melbourne, Australia, pp.125-135.

8) 맹성형, 주종철, 1998, "문서 구조화와 정보 검색", 정보과학회지, 제16권, 제8호.

9) 이종설, 강형일, 손충범, 강승현, 정연수, 민영수, 박종관, 유재수, 1999, "XML 저장관리시스템 설계 및 구현", Journal of the Research Institute for Computer and Information Communication Vol.7, No.2.