濟州大學校 尖端技術硏究所 論文集, 第14卷, 1號, pp.92-100, 2003. 6
Cheju Nat'l Univ. Res. Inst. Adv. Tech. Jour., Vol.14, No.1, pp.92-100, 2003. 6

data graph에 기반한 XML 인스턴스의 RDB 저장 모델

김 정 희* · 곽 호 영**

RDB storage model of XML instance based on the data graph

Jeong-Hee Kim* · Ho-young Kwak**

ABSTRACT

A RDB storage model based on the data graph is suggested for store the XML instance in relational databases(RDB). The XML instance being stored is represented by data graph based on the edge-labeled graph. data path table, element, attribute, and table index table values are extracted. Then database schema is defined, and extracted values are stored using the mapper. In order to support query, RDB storage model offers the translator translating XQL which is used as query language under XPATH. In addition, it makes us have DBtoXML generator restoring the stored XML instance. As a result, storage relationship between the XML instance and RDB structure can be expressed in terms of graph-based path, and it shows the possibility of easy search of random element and attribute information.

Key Words : RDB, XML, data graph, XQL, XPATH

Ⅰ. 서 론

인터넷의 발전이 진전되면서 이 기종간의 시스템에서 작성된 문서에 대한 데이터베이스의 구축과 검색 그리고 상호 교환의 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(Generalized Markup), 복합구조(Complex Structure), 검증(Validation)의 특성을 그대로 지원하는 한편 사용자에 의한 확장성(Extensibility)을 가지고 있는 XML(eXtensible Markup Language)을 제안하였다[2,3,4]. 그래서, 최근의 웹(Web) 또는 디지털 전자 도서관 시스템, CALS(Commerce At the Light Speed), 수학 분야, 채널 기술의 CDF(Channel

* 제주대학교 대학원
Graduate School, Cheju Nat,l Univ
** 제주대학교 통신 · 컴퓨터공학부, 첨단기술연구소
Faculty of Telecommunication & Computer Eng., Cheju Nat,l Univ., Res. Inst. of Adv. Tech.

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[5,6,7,8].

현재 관계형 데이터베이스상에 XML 문서를 저장하거나 추출하는 연구들이 진행 중[2]이며 이미 ADO (Active Data Object) 2.5와 SQL Server 2000에서는 각각 일차원적인 구조를 가진 레코드셋을 XML문서로 반환하거나 조인(Join)된 구조를 완벽하진 않지만 XML로 직접 추출해 내고 있다[9].

이에 본 논문에서는 XML 인스턴스들을 RDB에 저장하기 위한 모델을 제안하고 구현한다. 모델은 Edge-Labeled Graph에서 제공하는 Data Graph를 사용하여 요소(element)와 속성(attribute) 정보들을 추출한 후 데이터 경로(Data Path) 테이블과 요소와 속성 테이블을 생성하여 저장되도록 하였다. 또한 RDB에 대한 질의를 처리하기 위해, Query 변환, 그리고 RDB에서 XML 인스턴스로의 복원을 위해 DBto XML 생성기를 갖도록 하였다.

본 논문의 구성은 다음과 같다. 2장에서는 XML을 중심으로 기존 관련 연구를 살펴보고 3장에서는 제안하는 RDB 저장 모델을 설명하고, 4장에서는 구현 및 결과 그리고 5장에서는 결론 및 향후 연구 방향을 제시한다.

II. 관련 연구

XML 인스턴스 모델에 대한 기존의 연구 내용들을 살펴보면 DBMS의 활용과 문서 인스턴스의 저장 방식, 그리고 스키마 생성 방법에 따라 분류된다.

2.1 관계형 모델과 객체지향 모델

관계형 모델은 현재 가장 많이 사용하고 있는 관계형 데이터베이스를 기반으로 하고 있기 때문에 쉽게 접근할 수가 있어 사용자들의 전반적인 확산이 빠르다. 그러나, 관계형 데이터베이스의 특성상 문서의 구조에 대한 충분한 정보를 유지하기 위해 필요로 하는 테이블과 튜플의 수가 기하급수적으로 늘어날 수 밖에 없으며, 이에 따른 JOIN 연산으로 인한 시스템의 성능이 저하되는 단점이 있다. 객체지향 모델은 데이터베이스에서 지원하는 객체지향 개념을 이용할 수 있기 때문에 상속과 같은 객체지향 특성을 이용할 수 있으며, 엘리먼트 간의 전후 종속 관계를 클래스에 기반한 객체들간의 링크로 나타낼 수 있기 때문에 구조적인 문서를 모델링 하는데 적합하다 할 수 있다[2,10].

2.2 저장방식에 따른 분류

저장방식에 따른 분류에는 분할 저장 모델과 비분할 저장 모델, 그리고 혼합모델로 나누어진다. 분할 저장 모델은 XML 인스턴스를 엘리먼트별로 나누어서 저장한다. 이 모델은 문서의 일부 내용들이 수정되었을 때 관계되는 노드들만 수정하면 되므로 문서의 편집 및 관리가 쉽고, 동일한 내용을 갖는 노드들을 공유할 수 있다는 장점이 있지만, 문서의 내용을 추출하고자 할 때 각 단말노드들을 순회하며 통합하는 과정에서 시스템의성능을 저하시키는 문제가 발생한다. 비분할 저장 모델은 XML 인스턴스 전체를 BLOB 형태로 저장한 다음, 각각의 단말 노드는 오프셋 정보를 가지고 접근하는 방식이다. 이는 인스턴스를 한꺼번에 저장하였기 때문에 통합 과정이 필요없어 인스턴스 참조를 빨리 할 수 있지만, 내용의 일부만이 수정되었을 때도 인스턴스 전체를 재구성해야 한다는 큰 단점이 있다. 혼합모델은 분할 저장 모델과 비분할 저장 모델을 혼용하여 사용하는 모델로 각각의 모델에서 단점을 보완하고자 상대 모델의 특성을 일부 포함하였다. 하지만 혼합 모델의 단점인 저장 공간이 많이 소모된다는 문제점이 있다[11].

2.3 Edge-Labeled Graph

이는 XML 인스턴스를 반구조적 데이터(Semi-structured data)처럼 방향성 있는 그래프로 표현한 것이다. 엘리먼트는 객체(노드)로 표현되며, 각 객체

는 객체 식별자(Object Identifier)를 갖고 두 개의 객체(단순 객체, 복합 객체)로 구분된다. 그리고 객체들 간에는 간선이 존재하고, 각 간선마다 엘리먼트 이름으로 레이블이 있으며, 서브 엘리먼트를 표현하는 방향성이 있다. 또한 Edge-Labeled Graph에서는 Data Graph와 Schema Graph를 정의하는데 Data Graph는 XML 인스턴스의 모든 데이터가 표현되는 Edge-Labeled Directed Graph를 말하며, 이 Data Graph에서 깊이 우선 탐색 기법을 바탕으로 모든 경로가 단 한번만 표현되는 그래프를 Schema Graph가 된다[12]. Lore 시스템[13]의 DataGuide[14]처럼 스키마 그래프에서는 모든 레이블 경로가 유일하고(Concise), XML 인스턴스에 있는 모든 데이터는 표현되어야 하고(Accuracy), 각 노드의 구성이 어떻게 되어 있는지(Convenience) 알 수 있도록 하고 있다.

III. RDB 저장 모델

XML 인스턴스를 RDB로 저장하기 위한 본 논문의 제안 모델에서는 저장 시 필요한 물리적인 RDB의 구조를 정의하기 위해, 즉, 데이터베이스 스키마 정의를 위해 주어진 XML 인스턴스에서 다음과 같은 4가지의 주요 정보를 추출하여 데이터베이스 스키마 정의에 사용하게 된다.

3.1 요소의 경로와 값

요소의 경로(Element Path)는 주어진 XML 인스턴스를 Edge-Labeled Graph에 데이터 그래프(Data Graph)화 한 다음 리프(leaf) 노드까지 깊이 우선 탐색 기법을 적용하여 모든 데이터가 단 한번만 표현이 되도록 탐색하여 리프(Leaf) 노드까지의 전체 경로(Path)를 추출한다. 추출된 정보에는 리프(Leaf)까지의 엘리먼트 경로와 값이 포함되게 되고 이를 데이터 경로 테이블(Data Path Table)로 생성하도록 한다.

3.2 속성의 경로와 값

3.1절의 요소의 경로와 값을 구하는 방식으로 추출한다. 다만 속성은 필요에 의해 요소의 추가 정보를 지니면서 또한 요소 당 속성의 수는 가변적인 특징을 갖기 때문에 속성의 경로를 구할 때 이러한 점을 구분할 수 있도록 구별자를 갖도록 해야 한다. 값은 속성이 가지는 값이며 3.2절의 정보 역시 데이터 경로 테이블(Data Path Table)에 포함되도록 구성한다.

3.3 식별자(Object Identifier)

식별자는 요소와 속성을 구별하기 위해 사용된다. Data Path Table상의 경로 값(Value)이 요소인지 속성인지를 구별하며, 또한 검색의 효율성을 위해 추가로 생성되는 요소 테이블(Element Table)과 속성 테이블(Attribute Table)을 참조할 때 참조키(Reference Key)로 사용된다. 요소는 E, 속성은 A를 선두 문자로 사용한다.

3.4 문서 식별자 (Document Identifier)

문서 식별자는 여러 개의 XML 인스턴스를 구별하기 위한 식별자이다. 작업 영역 안에 처리할 XML 인스턴스가 여러 개 존재할 수 있기 때문에 이를 구별하기 위한 식별자이다. 문서 식별자는 XML 인스턴스의 URL 또는 URI를 사용하거나, 또는 이를 인덱스 하여 사용할 수 있도록 한다.

3.5 데이터 경로 테이블(Data Path Table)

데이터 경로 테이블은 XML 인스턴스를 분석하여 특정 요소 또는 속성에 대해 문서 식별자(DI), 경로(Path), 식별자(OID)를 튜플로 가지는 테이블이며 XML 인스턴스를 분석한 데이터 그래프(Data Graph)의 전체 내용이 삽입되게 된다. 또한 데이터 경로 테이블은 검색의 효율을 위해 추가로 생성되는 요소 테이블과 속성 테이블에서 질의에 대한 경로를 추출할 때 참조될 테이블이기도 하다.

3.6 요소 테이블 & 속성 테이블(Element Table & Attribute TAble)

이들은 데이터 경로 테이블(Data Path Table)에서

식별자가 요소값을 가지고 있으면 요소 테이블의 튜플로, 그리고 식별자가 속성값을 가지고 있으면 속성 테이블의 튜플로 경로 테이블을 요소와 속성에 따라 다시 재구성한 테이블이다. 요소, 속성 테이블을 별도로 두는 이유는 대부분의 XML 인스턴스에 대한 질의가 내용 검색과 구조 검색에 국한되기 때문에 질의 분석 결과에 따라 요소 또는 속성 테이블을 검색 대상으로 하는 것이 바람직하기 때문이다.

3.7 테이블 인덱스 테이블(Table Index Table)

여러 개의 XML 인스턴스를 관리하기 위해서는 XML 인스턴스간 구별하기 위한 방법이 요구되어지는데, 3.4절에 기술된 문서 식별자는 하나의 XML 인스턴스와 1대1의 관계로 존재하게 된다. 그리고 이는 실제 데이터베이스내에 저장될 때 테이블들의 이름들과도 서로 연관된다. 따라서 여러 개의 XML 인스턴스를 서로 구별하기 위해서는 문서 식별자, 데이터베이스내의 테이블 이름과의 관계를 관리하는 테이블이 필요하게 된다.

3.8 질의 분석

질의는 데이터베이스가 제공하는 기본 기능이된다. 따라서 본 논문에서도 저장된 XML 인스턴스에 대한 질의를 가능하도록 하였는데, 일반적으로 XML 인스턴스에 대한 검색은 크게 내용 검색, 구조 검색, 애트리뷰트 검색, 그리고 내용 + 구조 등의 혼합 검색으로 나눌 수 있으며, 다음은 설계한 시스템에서 지원되는 질의 형태와 질의 예를 보여준다.

▶ 내용 질의
예) "XML"이라는 단어를 포함하는 문서를 찾으시오.

▶ 구조 질의
- 특정 엘리먼트 질의
예) "XML"이라는 단어를 포함하는 첫 번째 Chapter를 찾아라.
- 특정 엘리먼트의 부모, 자식, 형제 엘리먼트를 질의
예) Chapter 엘리먼트의 부모 엘리먼트를 찾아라.
예) Chapter 엘리먼트의 두 번째 자손 엘리먼트를 찾아라.

▶ 혼합 질의
- 특정 키워드를 갖는 엘리먼트의 부모, 자식, 형제 엘리먼트 질의
예) "XML"이라는 단어를 갖는 Chapter의 자손 엘리먼트인 Section들 중"SGML" 이라는 단어를 갖는 것을 찾아라.

▶ 애트리뷰트 질의
- 엘리먼트에 나타날 수 있는 애트리뷰트 이름과 특성값에 대한 질의
예) 애트리뷰트 "check"가 "yes"인 엘리먼트를 찾아라.

3.9 XML 인스턴스 생성기

이는 데이터베이스내에 저장된 XML 인스턴스를 원래의 XML 인스턴스로 추출한다. 3.8절의 질의 결과에 대한 내용을 HTML형식과 XML형식으로 사용자에게 제공함과 동시에 데이터베이스에서 직접 추출하여 전체 XML 인스턴스 내용을 제공하는 방식을 지원하기 위해 필요하다.

IV. 시스템 구현

본 논문에서 구현한 시스템 모델은 하부 저장 시스템으로 ORACLE 9i를 사용하였으며, XML 인스턴스 분석기(Analyzer), Query 번역기, DBtoXML 생성기로 구성된다. 시스템의 전체 구조는 Fig. 1에서 보여준다.

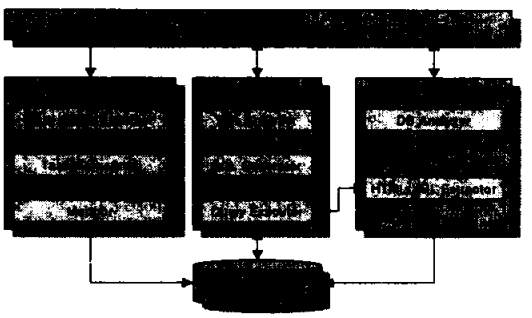

Fig. 1 System Architecture

4.1 XML Document Analyzer(XDA)

XDA는 데이터베이스내로 저장될 XML 인스턴스를 분석하여 저장하기 위해 필요한 스키마 구조의 기본 정보를 생성한다. 이를 위해 Edge-Labeled Graph에 기반 한 데이터 그래프(Data Graph)를 추출한 후 데이터 경로 테이블(Data Path Table)을 생성하고 또한 Element, Attribute Table, Table Index Table이 생성된다. 위의 과정이 끝나면 Mapper가 실제 데이터베이스내로 저장하게 된다. 세부 과정은 다음과 같다.

▶ Data Graph Extractor(DGE)

- DGE는 XML 인스턴스를 읽고 인스턴스내의 데이터에 대한 Data Graph를 추출한다. 이는 Data Path Table을 생성하기 위하여 사용될 기본 정보이며 Data Graph 생성 방법은 Edge-Labeled Graph에 기초하게 되며 단말 노드까지의 유일한 경로를 원칙으로 한다. Fig. 2는 Table 1에 대한 Data Graph를 보여준다.

Table 1. Example XML Instance

```
「 XML 문서 : ex.xml 」
<movie>
 <title>Citizen Kane </title> <year>1941</year>
 <director><full>Orson Welles</full></direcotr>
 <writer><full>Herman J. Mankiewicz</full></writer>
 <writer>
  <first>Orson</first> <last>Welles</last>
 </writer>
 <genre>Drama</genre>
 <cast num="1">
   <name>Orson Welles</name>
   <role>Charles Foster Kane</role>
   <award>Oscar</award>
   <category>Best Writing</category>
   <spouse> <name>Rita Hayworth</name>
   <occupation>divorced</occupation></spouse>
 </cast> <cast num="2"> . 이하 생략 . </cast>
 <language>English</language>
 <country>USA</country>
 <color>Black and White</color>
 <keywords>sled</keywords>
</movie>
```

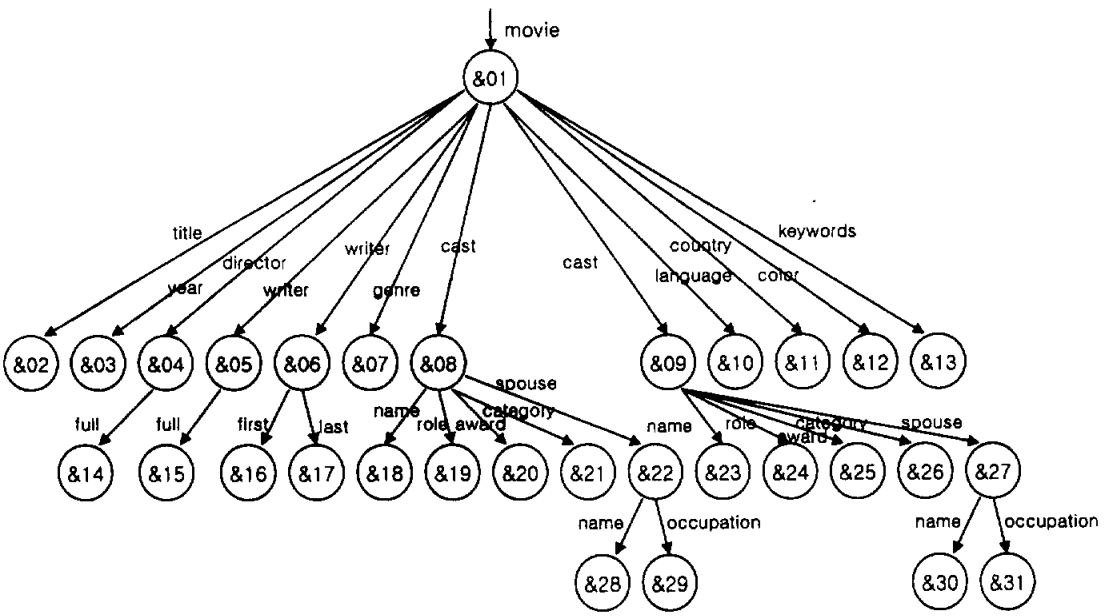

Fig. 2 Data Graph about the XML Instance

▶ Table Generator(TG)

- TG에서는 4개의 테이블이 생성된다. 먼저, Data Graph에 기반 한 Data Path Table이 생성되고 생성된 Data Path Table을 Element, Attribute Table로 분할한다. 또한 Table Index Table이 생성된다. Data Path Table에는 Data Graph를 기반으로 엘리먼트의 경로(path), 이에 대한 식별자(OID : 엘리먼트(E), 속성(A))와 문서 식별자(DID)를 부여한 후 테이블의 튜플로 삽입한다. Data Path Table은 Table 2에서 보여준다.

Table 2. Data Path Table

DID	ID	Path
D1	E_1	movie/title
D1	E_2	movie/year
D1	E_3	movie/director/full
D1	E_4	movie/writer/full
D1	E_5	movie/writer/first
D1	E_6	movie/writer/last
D1	E_7	movie/genre
...	...	...

- Data Path Table 정보를 이용하여 Element Table과 Attribute Table을 생성한다. 질의 수행의 효율을 위하여 Path Table에 대한 직접질의 보다는 이를 요소와 속성 정보로 분리하고 질의 분석을 통하여 해당되는 테이블 검색만 수행되도록 하는 것이 바람직하기 때문이다. 각각의 테이블 구조는 Table 3, 4와 같다.

- 또한 여러 개의 XML 인스턴스 처리를 위해 인스턴스에 대한 문서 식별자(DID)를 URL 또는 URI로 지원하기 위한 방법도 고려한다. 즉 URL 또는 URI에 대한 인덱스를 두는 테이블(Table 5)을 별도로 구축할 수 있도록 한다.

- Table Index Table(TIT)은 여러 개의 XML 인스턴스를 저장하게 됨으로 인해 데이터베이스 또한 여러 개의 테이블들을 가지게 된다. 따라서 생성되는 테이블들을 XML 인스턴스 별로 구별해야 하는 필요성이 제기된다. TIT는 이러한 요구를 처리하기 위해 지원하며, 그 구조는 문서 식별자(DID), Data Path Tabl(DPT)명, Element Table(IET)명, Attribute Table(AT)명, Document ID Index Table(DIDIT)명을 Table 6과 같이 생성한다.

Table 3. Element Table

DID	Element	Value
D1	E_1	Citizen Kane
D1	E_2	1941
D1	E_3	Orson Welles
D1	E_4	Herman J. Mankiewicz
...	...	...

Table 4. Attribute Table

DID	Attribute	Value
D1	A_1	1
D1	A_2	2
...	...	...

Table 5. Document ID Index Table

DID	URL_URI
D1	http://www.testcourse.com/ex.xml
D2	http://www.course.com/ex2.xml
...	...

Table 6. Table Index Table

DID	DPT	ET	AT	DIDIT

▶ Mapper

- Mapper는 위에서 생성된 테이블 구조를 기반으로 데이터베이스 내에 물리적인 구조를 생성하고 저장하는 기능을 담당한다.

4.2 Query Translator(QT)

QT는 XML 인스턴스에 대한 질의 요구를 처리하게 된다. 사용자가 사용하는 질의 방식은 XQL 또는 SQL이 가능하도록 하며, 상호간 변환이 가능하도록 한다. 최근에 XML 질의 언어로 표준화가 거의 확실

시 되는 XQuery를 사용하지 않고 XQL를 사용한 이유는 XQL로부터 XPATH가 유래되었고, XQuery는 SQL과 같이 섬세한 질의처리를 하도록 고안된 질의어이다. 따라서 두 질의어가 특성이 다르고 쓰임새도 다르다. XML에서 경로(Path) 질의는 매우 중요한 질의 처리 문제이며, XQuery도 경로(Path) 질의 처리를 지원하지만 구현과 질의 사용에 있어서 XPATH보다는 복잡하기 때문이다. 또한 XML 질의 처리에 있어서 XPATH를 축으로 하는 유형과 XQuery를 축으로 하는 유형이 현재 학계의 발전 추세이기도 하기 때문이다. 그리고 현재 XML 문서 필터링을 위해서 Continuous Query(CQ)를 이용한 시스템이 XPATH 질의를 수행하고 있기 때문이기도 하다. 따라서 본 논문에서는 XQL로 질의를 수행하도록 하였다. 일반적인 사용자의 질의 요구는 크게 내용 검색과 구조 검색으로 분석되며 본 논문에서는 다음과 같이 이를 구분하여 처리하도록 하였다.

▶ 내용 검색

- 내용 검색인 경우에는 XML 인스턴스내에 존재하는 텍스트(Keyword)에 대한 스트링 매칭(String Matching) 검색을 수행하게 되며, 이는 Element Table과 Attribute Table(속성 검색)의 Value 값을 기준으로 검색을 처리하게 된다. 검색 결과로는 Path에 근거한 형식으로 HTML 또는 XML 형식으로 사용자에게 보여준다.

▶ 구조 검색

- 구조 검색에서는 단순 구조 검색과 구조 + 내용 검색을 처리하게 된다. 단순 구조 검색은 계층간의 관계를 위해 조상, 자손의 관계가 있으며, 같은 계층내의 관계(Relationship)를 위해 형제 관계가 있으므로 각 엘리먼트들간의 순서에 따라 검색창의 인터페이스에 선후 관계(+, -, 숫자)를 입력할 수 있도록 하였으며, 또한 이들의 관계는 경로(Path) 형태로 구성되어 있으므로(A/B/C) 구분 문자(/)를 기준으로 처리가 수월하다. 그리고 구조 + 내용 검색에서는 단순 구조 검색과 내용 검색을 혼합한 형태이므로 두 가지 검색을 각각 내용과 구조로 수행한다. 알고리즘은 Fig. 3과 같다.

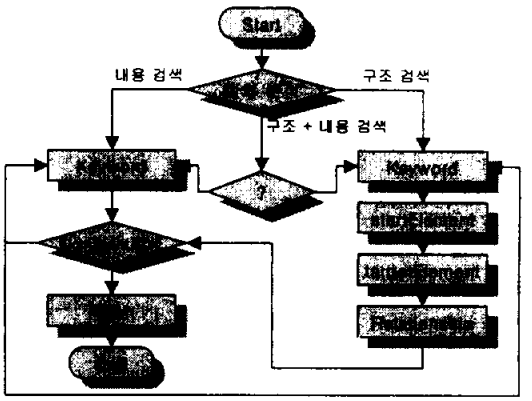

Fig. 3 Search Algorithm

▶ 질의 인터페이스

- Fig. 4는 위에서 설명한 내용 검색 및 속성 검색 그리고 구조 검색과 구조 + 내용 검색을 수행하기 위한 사용자 인터페이스 화면이다. 이는 「Citizen Kane를 포함하는 첫 번째 Title의 부모 엘리먼트를 찾아라」라는 구조 + 내용 질의 인터페이스를 보여준다.

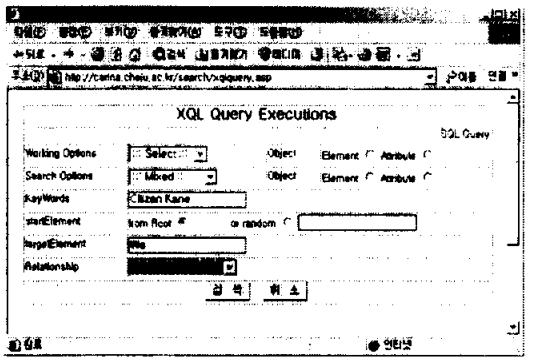

Fig. 4 XQL Query Interface

먼저, 첫 번째 Title에 포함된 키워드가 "Citizen Kane"이므로 시작 엘리먼트의 순위 값은 "1"이며 이는 첫 번째 구분문자(/) 앞에 해당되고, 시작 엘리먼트와 찾고자 하는 목적 엘리먼트의 관계가 부모이므로 관계 순위는 "1", 관계 값은 "조상"으로 한다. 관계 값과 엘리먼트간의 관계(Relationship)는 Table 7과 같다. 출력 방법은 엘리먼트 간의 계층구조와 해당 문서 전체에 대해서 출력이 가능한 링크를 연결하

였다. Fig. 5는 Flg 4의 질의에 대한 처리 결과이다.

Table 7. Relationship Between the Elements

관계	관계값	관계 순위
조상	조상	없음
부모	조상	1
자손	자손	없음
자식	자손	1
바로 앞 형제	형제	-1
앞 형제	형제	-
바로 뒤 형제	형제	+1
뒤 형제	형제	+

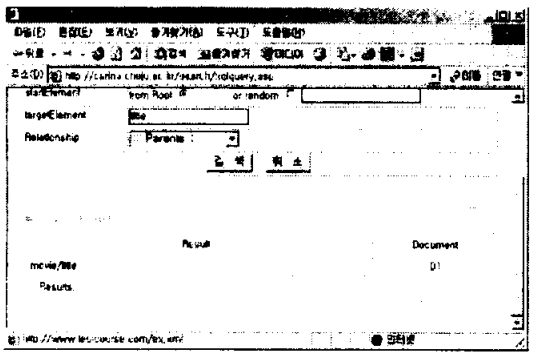

Fig. 5 Query Results

4.3 DBtoXML Generator(DXG)

DXG는 검색 결과에 대한 XML 형식의 추출과 데이터베이스내에서의 직접 추출 형식을 지원하도록 하였다. XML 형식의 추출은 4.2절의 검색 결과를 HTML 형식과 XML 형식으로 구분하여 출력하는 과정에서 수행되며, 여기에서는 직접 추출 과정을

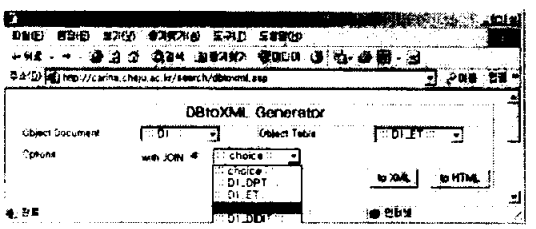

Fig. 6 DBtoXML Generator

Fig. 6과 같은 사용자 인터페이스를 이용하여 추출할 데이터베이스를 선택하며, 선택된 값에 따라 현재 데이터베이스내의 테이블 인덱스 테이블(TIT)를 조회하고 해당되는 테이블의 Element, Attribute 테이블과 이들의 Path 테이블을 JOIN하여 그 결과를 Fig. 7과 같이 보였다.

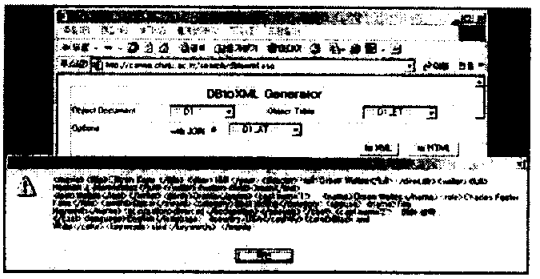

Fig. 7 XML Document

V. 결 론

본 논문에서는 대용량의 XML 인스턴스 관리 시스템을 개발하는데 있어서 기반 기술이 되고 있는 XML 저장 관리 시스템 모델을 제안하고 구현하였다. 이를 위해 Edge-Labeled Graph를 기반 한 Data Graph를 적용하였으며, 저장 모델에 필요한 각 프로세스의 세부적인 기능을 분석하고 정의하였다.

XML 인스턴스의 효율적인 저장과 관리를 위한 XML Document Analyzer는 XML 인스턴스를 분석하고 이를 데이터 경로 테이블(Data Path Table)과 Element, Attribute, Table Index Table을 생성하도록 구현 되였으며, 검색을 위한 Query에는 내용 검색(속성 검색 포함)과 구조 검색, 그리고 구조 + 내용 검색이 가능하도록 하였다. 또한 검색 결과에 대한 결과를 HTML과 XML 형식과 데이터베이스에서 직접 추출하는 형식을 지원하도록 검색 인터페이스를 구현하여 테스트하였다. 테스트 결과 Data Graph에 의한 경로(Path)기반으로 엘리먼트와 속성들의 구조 정보가 구축(계층 구조의 특성을 유지함)됨으로 인해 특정 엘리먼트를 쉽게 검색할 수 있었으며 다양한 XML 질의를 처리할 수 있는 가능성을 보였다. 특히, XML 인스턴스를 Data Graph의 경로(Path)를 기반

으로 한 RDB내의 저장을 보였다. 향후 연구 과제로는 XML 구조는 트리 형태의 계층적 구조로 표현될 수 있으며 이러한 원리에 의한 Data Path Graph와 현재 중점적으로 연구되고 있는 객체지향 데이터베이스와의 연동에 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) 연제원, 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 최종 보고서.

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 38(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 index-ing 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) 이석호, 2000, 데이터베이스 시스템" 정익사.

10) Brian Lowe, Justin Zobel, Ron Sacks Davis, 1997, A Formal Model for Representation and Query of Structured Documents. Journal of Systems Integration 7(1), pp. 31-46.

11) 이종설, 강형일, 손충범외 5, 1995, XML 저장관리시스템 설계 및 구현, Journal of the Structured Text, DASFAA, pp. 449-456.

12) 김성림, 윤용익, 2002, XML 문서에서의 엘리먼트 정보를 이용한 스키마 추출방법, 정보처리학회논문지, 제9-D권 3호.

13) J. McHugh, S. Abiteboul, R. Goldman, D.Quass, J. Widom, 1997, Lore: A Database Management System for Semistructured Data, SIG MOD Record, 26(3), pp. 54-66.

14) Roy Goldman, Jennifer Widom, , 1997, Data Guides : Enabling Query Formulation and Optimization in Semistructured Databases, In Proceedings of VLDB.