

Oracle8i XDK를 이용한 XML 문서의 저장*

0

하상호, 이강석, 백인천
순천향대학교 정보기술공학부

Storing XML Documents using Oracle8i XDK

0

Sangho Ha, Gangseog Lee, Incheon Paek

Dept. of Information Technology, Soonchunhyang University

요 약

XML은 웹 상에서 데이터의 원활한 교환을 위해서 HTML을 보완하여 설계된 차세대 인터넷 문서 작성용 언어이다. XML 문서와 같은 반구조(semistructured)의 특성을 갖는 데이터를 효과적으로 다루기 위한 새로운 데이터 모델과 질의어가 제안되어 오고 있지만, 여기서는 관계형 데이터베이스에 XML 문서를 효과적으로 저장하는 방법에 관해서 논의한다. 먼저, 도서를 표현하는 XML 문서를 위한 DTD를 제시하고, 이 DTD를 관계 테이블로 변환하는 방법을 논의한다. 다음에는 Oracle에서 지원하는 XDK를 이용하여 XML 문서를 Oracle8i DB에 저장하는 방법에 대해서 논의한다.

1. 서론

XML[1]은 웹 문서 기술을 위한 표준과 모든 유형의 정보에 대한 데이터 기술 언어로서 점차 그 사용 폭을 넓혀가고 있어서, 앞으로 전자적 데이터 교환 방식이 이에 의존하게 될 것이라는 것이 예고된다.

현재 사용중인 데이터베이스가 XML을 지원할 경우에 기업 외부 즉, 인터넷에 산재해 있는 수많은 XML 또는 HTML 형태의 데이터를 마치 기업 내부의 데이터처럼 사용할 수 있어 장기적으로 지식관리 시스템 구축이 용이할 뿐만 아니라 공통된 포맷 지원으로 각기 다른 DBMS 사이의 데이터 교환도 자유롭게 된다. 또한, DBMS의 검색 방식이 애트리뷰트 또는 테이블 별로 색인을 미리 만들어 다계층 또는 관계 데이터베이스 질의어를 통해서 원하는 데이터를 추출할 수 있는 장점을 갖는다[2].

XML 문서를 어떻게 저장할 것인가는 그 문서의

성격에 달려 있다. 즉, 전자상거래에서 사용되는 도서를 표현하는 XML 문서와 같은 경우에 그 문서는 매우 정규적인 구조를 갖으며, 데이터의 크기가 비교적 작으며, 또한 이러한 데이터는 자주 갱신될 필요가 있다. 이러한 데이터 중심의 XML 문서를 저장하고 검색하는 데는 관계나 객체지향 중심의 데이터베이스를 사용하는 것이 효과적이다. 반면에, XML 문서가 비정규적인 구조를 가지며, 데이터의 크기가 비교적 크며, 거의 갱신되지 않는 문서는 텍스트 파일 형태로 간단히 저장할 수 있다[3].

XML 문서를 데이터베이스에 저장하는 방법에 대해서 다양하게 연구되어 왔는데[4,5], 여기서는 Oracle사에서 지원하고 있는 XDK(XML Developer's Kit)[6]를 사용하여 XML 문서를 Oracle8i DB에 저장하는 방법에 대해서 논의한다.

Oracle 사는 데이터베이스 응용 프로그램에서 XML 기술이 갖는 장점을 이용할 수 있는 여러 가지 요소들을 XDK 안에서 제공하고 있는데, 이러한 요소에는 XML 문서를 구문 분석하여 파스 트리를 생성

* 본 연구는 정보통신부의 대학 S/W 연구센터 지원 사업에 의해 수행된 것임.

해 주는 XML 파서, XSL 스타일 쉬트를 사용하여 주어진 XML 문서를 원하는 다른 형태의 XML이나 HTML 문서로 변환해 주는 XSL 변환 프로세서(XSLT), 질의(query)를 데이터베이스에 전달하고, 그 결과로부터 XML 문서를 생성할 수 있고, XML 문서를 데이터베이스 테이블에 저장할 수 있는 기능을 제공하는 XML SQL Utility, XML 문서에 포함된 질의문을 처리하고 그 결과를 XML 문서로 출력하는 XSQL Servlet 등을 들 수 있다. 이러한 요소 모두 Java나 Java Servlet으로 작성된 버전을 가진다.

논문에서는 Java나 Java Servlet으로 작성된 XDK 요소들을 사용하며, JDBC를 사용하여 Oracle8i와 연동한다. 따라서 이 방법은 특정 플랫폼에 독립하여 사용될 수 있다.

논문의 다음 순서로, 2장에서는 주어진 XML 문서의 DTD를 명세하고, 이로부터 관계 테이블을 생성하는 방법에 대해서 논의한다. 3장에서는 XDK의 요소들을 사용하여 주어진 XML 문서를 생성된 관계 테이블 상에 사상하는 방법에 대해서 논의한다. 마지막으로 4장에서는 결론을 언급한다.

2. DTD로부터 테이블 생성

여기서는 XML 문서의 구조 정보를 표현하는 도서에 관한 DTD를 제시하고, 이 DTD로부터 관계 테이블을 생성하는 방법에 대해서 다룬다. 다음은 도서 DTD의 일부이다.

```
<!ELEMENT books (content_info, type_info)>
<!ATTLIST books id ID #REQUIRED
      subject CDATA #REQUIRED>
<!ELEMENT content_info (basic_info, selling_info)
<!ATTLIST content_info id ID #REQUIRED
      ref_parent IDREF #REQUIRED>
<!ELEMENT basic_info (title, page, isbn, price, picture?,
      book_size, author+, contents?, summary?)>
<!ATTLIST basic_info id ID #REQUIRED
      ref_parent IDREF #REQUIRED>
<!ELEMENT title (#PCDATA)>
<!ELEMENT page (#PCDATA)>
<!ELEMENT price (#PCDATA)>
<!ATTLIST price_unit (won | dollar | yen | pound)
      #REQUIRED>
<!ELEMENT isbn (#PCDATA)>
<!ELEMENT picture (#PCDATA)>
<!ELEMENT book_size (#PCDATA)>
<!ATTLIST book_size size_unit (centimeters | inches)
      #IMPLIED>
<!ELEMENT contents (#PCDATA)>
<!ELEMENT summary (#PCDATA)>
<!ELEMENT author (name, info?)>
<!ATTLIST author id ID #REQUIRED
```

```
link CDATA #IMPLIED
ref_parent IDREF #REQUIRED>
<!ELEMENT name (#PCDATA)>
<!ELEMENT info (#PCDATA)>
<!ELEMENT selling_info (selling_price, status?)>
<!ATTLIST selling_info id ID #REQUIRED
      ref_parent IDREF #REQUIRED>
<!ELEMENT selling_price (#PCDATA)>
<!ATTLIST selling_price dis_rate CDATA #IMPLIED>
<!ELEMENT status (#PCDATA)>
<!ATTLIST status inventory CDATA #IMPLIED>
<!ELEMENT type_info (additional_type*)>
<!ATTLIST type_info id ID #REQUIRED
      ref_parent IDREF #REQUIRED>
<!ELEMENT additional_type (#PCDATA)>
<!ATTLIST additional_type best_seller (yes | no) #IMPLIED
      new_book (yes | no) #IMPLIED
      recomment (yes | no) #IMPLIED
      prize CDATA #IMPLIED
      time CDATA #IMPLIED
      place CDATA #IMPLIED
      occasion CDATA #IMPLIED
      ref_parent IDREF #REQUIRED>
```

위에서 제시된 DTD로부터 도서 정보는 크게 내용 정보(content_info)와 유형 정보(type_info)로 구성되어 있고, 내용 정보는 다시 기본 정보(basic_info)와 판매 정보(selling_info)로 구성되어 있음을 알 수 있다. 이러한 도서의 구조 정보를 유지할 수 있는 관계형 테이블을 생성하고자 한다.

DTD로부터 관계 테이블의 생성 원칙은 다음과 같다. DTD를 트리 T로 표현했다고 볼 때,

- ① T의 루트 노드나 중간 노드는 자신의 테이블로 생성된다. 이 테이블은 자신의 기본 키(primary key)와 이 노드의 부모 노드에 대해서 생성된 테이블의 기본 키를 외부 키(foreign key)로 갖는다. 기본 키 값은 ID의 속성으로부터, 외부 키 값은 IDREF의 속성으로부터 각각 가져온다. 또한, 집합 값(set-value)을 갖지 않는 자식 노드는 생성된 테이블의 애트리뷰트가 된다.
- ② 집합-값을 갖는 잎 노드는 별도의 테이블로 생성된다. 이 경우에 생성된 테이블은 그 부모 노드의 기본 키 값을 외부 키로 갖는다.
- ③ 엘리먼트(ELEMENT)의 각 속성(ATTRIBUTE)은 그 엘리먼트가 속하는 테이블의 한 애트리뷰트가 된다.

다음은 위의 원칙에 의해서 도서 DTD로부터 생성된 테이블과 그 애트리뷰트를 보여준다.

books

id	subject
----	---------

content_info

id	ref_parent
----	------------

type_info

id	ref_parent
----	------------

basic_info

id	ref_parent	title	page	isbn	price
price_unit	picture	book_size	size_unit	contents	summary

authors

id	ref_parent	link	name	info
----	------------	------	------	------

selling_info

id	ref_parent	selling_price	dis_rate	status	inventory
----	------------	---------------	----------	--------	-----------

add_type

id	ref_parent	best_seller	recomm	new_book	prize	time	place	ocassion
----	------------	-------------	--------	----------	-------	------	-------	----------

①의 원칙에 의해서 DTD의 문서 구조 정보가 테이블 상에 표현되고 있으며, ②의 원칙에 의해서 각 테이블의 애틀리뷰트 값은 단일의 스칼라 값을 가지며, ③의 원칙에 의해서 엘리먼트의 속성들이 해당 테이블의 애틀리뷰트가 되는 것을 알 수 있다. 애틀리뷰트의 타입으로는, number, varchar2, date 등이 적절히 사용될 수 있다. 또한, summary(요약)나 contents(목차)의 크기가 클 경우에는 CLOB의 타입을 사용할 수 있으며, 표지 그림 picture에 대해서는 BLOB의 타입이 사용된다. 이러한 엘리먼트의 값은 다른 엘리먼트와는 달리 데이터보다는 문서적인 특성을 갖으며, 이러한 값을 텍스트 파일이나 그래픽 파일 형태로 별도의 공간에 저장할 수 있다.

3. Oracle8i에서의 XML 문서 저장

여기서는 Oracle XDK의 한 요소인 XSQL Servlet[7]를 사용하여 주어진 XML 문서를 명세된 데이터베이스의 테이블에 저장하는 방법에 대해서 다룬다. 이 XSQL Servlet은 Java를 위한 XML Parser와 XML SQL Utility를 사용하여 XML 문서에 대한 저장뿐만 아니라 검색 등 다양한 질의 문에 대해서 연산을 수행한다. XSQL Servlet은 Java Servlet으로 구현되어 있다.

XSQL Servlet를 사용한 XML 문서의 저장은 그

림 1의 과정에 의해서 수행된다. 첫째, XML 문서는 XML Parser에 입력되고, XML 파서는 입력된 문서를 해당 DTD 문법과의 유효성을 판단한다. 둘째, 저장하고자 하는 XML 데이터를 해당 데이터베이스 테이블의 구조에 일치시킨다. 이 과정은 XSLT 프로세서에 의해서 수행된다. 임의의 주어진 XML 문서를 저장할 테이블의 구조와 일치한 XML 문서로 변환해주는 XSL 변환을 생성할 수 있음을 유의해야 한다. XSL 변환은 XSLT 프로세서가 XML 문서와 XSL 파일을 입력받아 수행한다. 셋째, XSQL Page Processor는 전달된 XML 문서에서 태그들 사이의 값들을 추출하여 SQL문장을 구성하고, 이 SQL 문장들을 XML SQL Utility에 전달한다. 마지막으로, XML SQL Utility는 Oracle8i 데이터베이스에 SQL 질의문을 전달하여 데이터를 저장토록 한다.

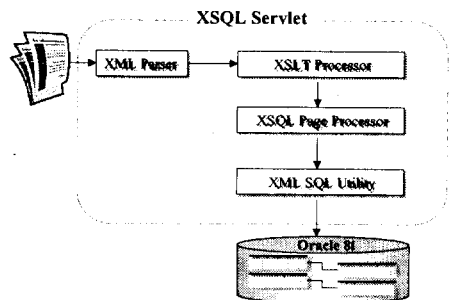


그림 1 Oracle 8i에서의 XML 문서 저장

다음은 XSQL Servlet을 사용하여 XML 문서를 Oracle8i DB에 저장하는 한 예로, books.xml의 XML 문서를 basic_info 테이블에 저장하는 방법을 설명한다. 다음은 books.xml의 저장을 XSQL Servlet에 요청하는 명령어 라인 상의 명령을 나타낸다.

```
# /xsql/bin/xsql insert_basic_info.xsql posted-xml
= books.xml
```

위에서 명세된 insert_basic_info.xsql 문서(이를 XSQL page라 한다)는 다음과 같이 작성된다.

```
<!--insert_basic_info.xsql-->
<xsql:insert-request xmlns:xsql="um:oracle-xsql"
table="basic_info"
transform="insert_basic_info.xsl"
connection="demo"/>
```

위의 xsql 문서에는 저장할 테이블 basic_info, 전달된 XML 문서를 basic_info의 테이블 구조와 일치되게 변환하는데 사용될 XSL 문서 insert_basic_info.xsl이

명세되어 있으며, 그리고 데이터베이스와의 연동을 위한 연결 정보가 connection 상에 표현되어 있다.

다음은 위의 xsql 문서상에 명세된 insert_basic_info.xml 문서이다.

```
<!--insert_basic_info.xml-->
<ROWSET xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
        xsl:version="1.0">
  <xsl:for-each select="/books/content_info/basic_info">
    <ROW>
      <id><xsl:value-of select="@id"/></id>
      <ref_parent><xsl:value-of select="@ref_parent"/></ref_parent>
      <title><xsl:value-of select="title"/></title>
      <page><xsl:value-of select="page"/></page>
      <isbn><xsl:value-of select="isbn"/></isbn>
      <price><xsl:value-of select="price"/></price>
      <price_unit><xsl:value-of
        select="price/@price_unit"/></price_unit>
      <picture><xsl:value-of select="picture"/></picture>
      <book_size><xsl:value-of select="book_size"/></book_size>
      <size_unit><xsl:value-of
        select="book_size/@size_unit"/></size_unit>
      <contents><xsl:value-of select="contents"/></contents>
      <summary><xsl:value-of select="summary"/></summary>
    </ROW>
  </xsl:for-each>
</ROWSET>
```

위의 xsl 문서에서 <ROWSET>은 테이블 basic_table을 명세하며, <ROW>는 그 테이블의 한 레코드를 명세한다. 가령, title 애트리뷰트의 값은 XML 문서에서 경로 /books/content_info/basic_info의 자식 노드 title 상에 위치한 값이라는 것을 명세한다. <xsl:value-of> 태그는 XML 문서에서 해당 데이터의 참조를 나타내며, select는 해당 엘리먼트의 값을 가져오게 한다. 엘리먼트의 속성은 애트리뷰트로 변환하기 위해서 @를 사용하여 그 속성을 명세한다. Oracle XDK는 XML 문서에 명세된 속성 값을 직접 저장하는 방법을 갖지 않는다. 따라서 속성 값을 저장하기 위해서 엘리먼트와 동일하게 애트리뷰트로 변환하는 것이 필요하다.

위와 같은 방식으로 books.xml 문서는 XSL 변환을 통해서 여러 개의 문서로 분리되어 해당 테이블에 저장된다.

4. 결론

본 논문은 Oracle 사에서 지원하고 있는 XDK를 사용하여 XML 문서를 Oracle8i DB에 저장하는 방법을 다루었다. 이를 위해서 먼저, 도서의 논리적 정보

를 나타내는 DTD를 기술하였고, 이 DTD로부터 관계 테이블을 생성하였다. 다음에, XSQL Servlet을 사용하여 주어진 XML 문서를 명세된 데이터베이스 테이블 상에 저장하는 방법을 기술하였다.

향후 연구 과제로는, XDK를 사용하여 저장된 XML 문서를 다양한 방식으로 검색할 수 있는 시스템을 구축하는 것이다. 그리고 이 시스템이 XML 문서를 위한 질의어 XQL[8]이나 XML-QL[9]을 효과적으로 지원하는 방법에 대해서 연구한다.

참고문헌

- [1] "Extensible Markup Language (XML)", <http://www.w3.org/XML/>, 2000.
- [2] Ronald Bourret, "XML and Data Base", <http://www.informatik.tu-darmstadt.de/DVS1/staff/bourret/xml/XMLAndDatabases.htm>, 1999.
- [3] Jennifer Widom, "Data Management for XML", <http://www-db.stanford.edu/~widom/xml-whitepaper.html>, September 1999.
- [4] Deutsch, M. Fernandex, D. Suci, "Storing Semistructured Data With STORED", Proc. of ACM SIGMOD, 1999.
- [5] J. Shanmugasundaram, et al, "Relational Databases for Querying XML Documents: Limitations and Opportunities", Proc. of VLDB, 1999.
- [6] "Using XML in Oracle Database Applications", http://technet.oracle.com/tech/xml/info/htdocs/otnwp/about_oracle_xml_products.htm, 1999.
- [7] Steve Muench, "Oracle XSQL Pages and the XSQL Servlet", http://technet.oracle.com/tech/xml/xsql_servlet/, 2000
- [8] "XML Query Languages(XQL)", <http://www.w3.org/TandS/QL/QL98/pp/xql.html>, 1998.
- [9] "XML-QL: A Query Language for XML", <http://www.w3.org/TR/1998/NOTE-xml-ql-19980819>, 1998