XML 스키마로 부터 관계형 스키마 추출 기법

김은욱, 민미경
서경대학교 컴퓨터과학과

Extraction of Relational Schema from XML Schema

Eun-Wook Kim, Mee-Kyung Min
Dept. of Computer Science, SeoKyeong University

요 약

데이터로서 XML의 의미가 중요해짐에 따라 XML 문서를 저장하는 방법들에 대한 연구가 활발히 진행되고 있다. 그 중 하나가 스키마를 이용하여 XML 문서를 관계형 데이터베이스에 저장하는 것으로서, 지금까지 DTD를 중심으로 연구가 이루어져 왔으나, XML 스키마의 등장으로 DTD의 단점을 보완하고, 기존 관계형 데이터베이스와 보다 유사하게 표현 할 수 있게 되었다. 본 논문에서는 XML 스키마에서 관계형 스키마를 추출하는 기법을 제시한다. 제시된 기법은 DTD에서 관계형 스키마를 추출하는 기법을 기반으로 하여, DTD에서 표현할 수 없는 사용자 정의 데이터형을 추가로 제공하는 등, XML 스키마의 속성과 요소에 따른 여러 특성을 표현할 수 있다.

1. 서론

XML은 W3C에 의해 웹을 기반으로 하는 구조화된 문서를 기술하는 표준으로 지정되면서 데이터로서의 역할이 중요해지게 되었다. 따라서 XML 문서의 효율적인 저장 방법이 필요하게 되었고, XML 구조를 정의하는 스키마를 통해 기존 데이터베이스에 저장하는 연구가 이루어지고 있다. 기존 시스템 중 관계형 데이터베이스는 현재 가장 보편화 되어있는 시스템이다.

XML에서 주로 쓰이는 스키마로는 DTD와 W3C에서 2001년 5월에 표준으로 지정된 XML Schema로 크게 나눌 수 있다. 스키마의 사전적 의미로서는 DTD를 포함하지만 앞으로 나오는 스키마란 W3C에서 정의한 스키마를 말한다. 스키마는 DTD의 많은 단점을 항상 시켰으며 다양한 데이터 타입을 표현한다.

지금까지 DTD를 중심으로 관계형 스키마에 사상시키는 많은 기법들이 제시되어 왔으나 본 논문에서는 XML 스키마를 이용하여 관계형 데이터 베이스 시스템에 XML 문서를 저장하기 위한 스키마 추출 기법을 제안한다.

본 논문 2장에서는 관련연구를 살펴보고, 3장에서는 스키마 추출기법을 제안하며, 4장에서 XML 문서의 저장 예를 보인다. 끝으로 5장에서 결론을 맺는다.

2. 관련연구

관계형 모델을 사용하여 XML 문서를 저장하는 방법은 다음과 같이 크게 두 가지로 나눌 수 있다.

첫 번째로 스키마를 분석하지 않고, 문서 자체를 분석해서 관계형 데이터 모델로 변환하는 방법이 있다[2]. 두 번째로 DTD가 주어졌을 때, DTD를 분석하여 XML 문서구조를 관계형 스키마로 사상시키는 방법이 있다[3][4].

DTD를 이용한 방법에는 XML을 Element, Attribute, Text, Path 네 가지 테이블 스키마로 나누어 저장하는 방법[4], DTD에서 각 요소를 하나의 테이블 스키마로 사상하고, 요소의 속성을 테이블 필드로 두는 방법[3], 가능한 적은 수의 테이블 스키마로 사상시키는 Hybrid Inlining방법[3] 등이 있다.

[4]의 경우는 IDREFS를 고려하지 않고 있다. [3]에서와 같이 요소마다 테이블을 사상하는 방법은 엘리먼트와 테이블간의 직접적인 사상으로 문서의 과도한 단편화를 초래한다. [3]의 Hybrid Inlining방법은 가능한 요소의 자손들을 한 테이블 스키마의 필드로 정의함으로써 단편화를 해결하고 가능한 적은 수의 테이블로 XML 문서를 사상시키기 위한 방법이다.

본 논문은 이중 Hybrid Inlining방법을 기반으로 DTD가 아닌 XML 스키마에서 관계형 스키마로 사상시키는 방법을 제시한다.

3. XML 스키마에서 관계형 스키마로 사상 방법

XML 문서는 속성과 요소로 이루어져 있고, XML 스키마는 XML 문서의 요소와 속성들의 구조를 정의한다. 따라서 XML 스키마에서 관계형 스키마로의 사상 방법은 다음과 같은 규칙을 따르며 크게 속성과 요소의 경우로 분류할 수 있다.

(1) 속성

XML 스키마의 속성은 관계형 스키마에서 필드가 된다.

(2) 요소

요소는 내용 정의에 따라 텍스트만을 가지는 단순형 요소와 자식 요소와 속성을 가지는 복잡형 요소로 분류할 수 있다.

(2.1) 단순형

(2.1.1) 단순형 요소

단순형 요소는 부모나 조상요소에 대한 관계형 스키마에서 필드가 된다. 그림 1은 XML 스키마의 단순형 요소를 관계형 스키마로 사상한 것이다. XML 스키마의 Customer 요소는 complex type 이며, 3개의 simple type 요소인 FirstName, MiddleInitial, LastName을 자식으로 가지고 있다. 따라서 3개의 요소는 Customer 요소의 필드가 된다.

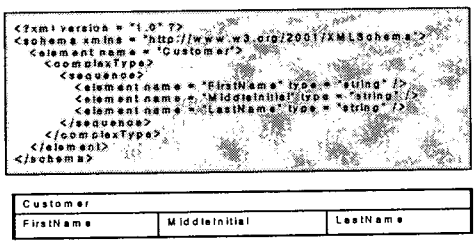

그림 1 단순형 요소의 사상

(2.1.2) 유도된 단순형

유도된 단순형으로는 여러 가지가 있으나 다음 2개의형에 따라 새로운 테이블이 생성된다.

(2.1.2.1) 리스트형(list) - 기초 데이터형의 값들이 공백으로 구분된 리스트가 된다. 리스트형은 새로운 테이블로 생성된다. 그림 2는 리스트형을 쓰는 XML 스키마를 관계형 스키마로 사상시킨 것이다. 리스트형에는 외래키가 추가되고, 이 외래키는 decimal키와 함께 기본키가 된다.

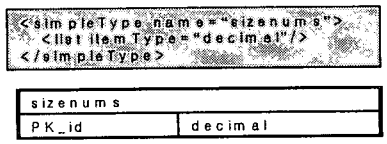

그림 2 리스트형(list) 요소의 사상

(2.1.2.2) 열거형(enumeration) - 데이터형의 value space에 허용되는 값들에 대한 선택적인 제한들을 제약 패싯(constraining facet)이라 하고, 제약 패싯중 열거형은 새로운 테이블을 생성 한다. 그림 3은 열거형을 쓰는 XML 스키마를 관계형 스키마로 사상시킨 것이다. 열거형은 스키마에 나타나 있는 값이 생성된 테이블에서 인스턴스값이 된다. 이러한 테이블의 각 튜플들은 다른 테이블에서 참조하여 사용한다.

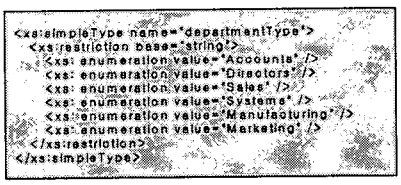

그림 3 열거형(enumeration) 요소의 사상

(2.2) 복잡형

(2.2.1) 지명 복잡형(named complex type) - 지명 복잡형을 갖는 complexType은 하나의 테이블 되고, 그 complexType을 가지는 요소들은 부모 테이블의 필드가 되어 complexType 테이블의 튜플을 참조한다. 그림 4는 지명 복잡형의 XML 스키마를 관계형 스키마로 사상시킨 것이다. Address 테이블의 기본키를 이용하여 Addresses 테이블에서 참조하여 사용한다

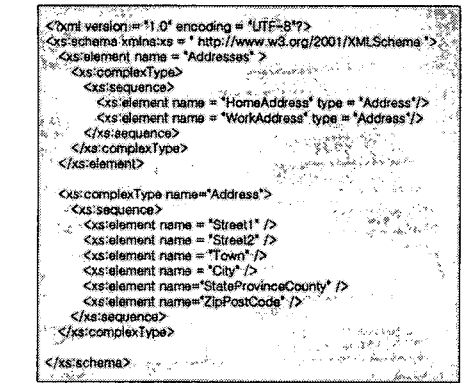

그림 4 복잡형 요소의 사상

(2.2.2) 익명 복잡형(anonymous complex type) - 복잡형이 요소 선언 내부에 중첩된 경우를 말한다. 익명 복잡형이 가지는 단순형 요소들은 조상 요소의 필드가 된다. 그림 5의 XML 스키마는 익명 복잡형으로 구성되어 있다. 이 방법은 복잡형 요소마다 테이블을 사상시키는 방법과 비교했을 때 테이블의 단편화를 줄일 수 있다.

(2.3) 다중성

다중으로 나타낼 수 있는 요소는 새로운 테이블을 생성 한다. (maxOccurs 속성 값이 2이상일 때) 예를 들면 그림 5의 XML 스키마에서 DeliveryReceipt 요소는 Customer 요소와 Items 요소의 자식요소를 가지고 있다. Items 요소는 minOccurs 속성 값이 1이고, maxOccurs 속성 값이 unbounded인 값을 가지고 있다. 따라서 Items요소는 새로운 테이블로 생성 한다. 이 방법은 요소들의 중복된 값을 피할 수 있게 된다.

(2.4) 외부참조

(2.4.1) 부모-자식 참조

자식 요소중에 새로운 테이블이 만들어지게 되면 새로운 테이블에는 부모테이블의 primary key를 참조하는 foreign key가 추가된다. 예를 들면 그림 5의 Items 테이블은 부모테이블을 참조하는 foreign key 필드를 추가한다.

(2.4.2) IDREF와 Keyref 요소

IDREF나 Keyref 요소는 새로운 테이블이 된다. 새로운 테이블에는 부모테이블의 ID나 Key를 참조하는 foreign key가 추가된다.

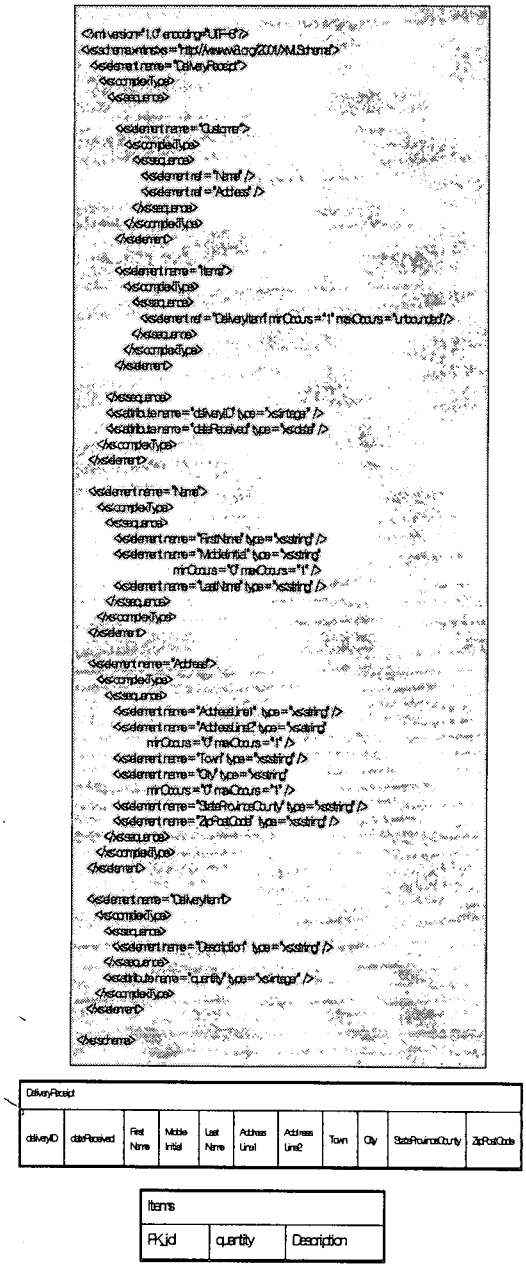

그림 5 다중성 요소와 외부참조 요소의 사상

4. XML 문서의 저장

Key와 ID 속성이 없는 테이블은 primary key 필드

를 생성 한다. 여러 XML 문서의 공유되는 요소를 위해 관계형 테이블에 XML문서의 이름을 넣는 필드를 추가한다. 마지막으로 XML문서에 대한 구조정보를 가지고 있는 테이블을 생성 한다. 이 테이블은 데이터베이스에 저장되어 있는 데이터에서 XML문서로 복원시 원래 구조를 유지할 수 있게 해준다.

앞에서 제시한 방법을 토대로 실제적인 문서가 저장되는 예를 보면 다음과 같다. 그림 5의 XML 스키마를 따르는 XML문서가 그림 6과 같이 있을 때, 이 문서가 관계형 테이블로 저장된 내용은 그림 7과 같다.

```
<?xml version="1.0" ?>
<DeliveryReceipt deliveryID="4215" dateReceived="2001-04-16"
 xsi:noNamespaceSchemaLocation="http://localhost/schemaTestUtil/Ch01/DeliveryReceipt.xsd"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <Customer>
   <Name>
    <FirstName>Ray</FirstName>
    <MiddleInitial>G</MiddleInitial>
    <LastName>Bayliss</LastName>
   </Name>
   <Address>
    <AddressLine1>10 Elizabeth Place</AddressLine1>
    <AddressLine2></AddressLine2>
    <Town>Paddington</Town>
    <City>Sydney</City>
    <StateProvinceCounty>NSW</StateProvinceCounty>
    <ZipPostCode>2021</ZipPostCode>
   </Address>
  </Customer>
  <Items>
   <DeliveryItem quantity="2">
    <Description>Small Boxes</Description>
   </DeliveryItem>
  </Items>
</DeliveryReceipt>
```

그림 6. XML 문서(문서명 : DeliveryReceipt.xml)

DeliveryReceipt

id	Doc	deliveryID	dateReceived	First Name	Middle Initial	Last Name	Address Line1	Address Line2	Town	City	StateProvince County	ZipPostCode
1	DeliveryReceipt.xml	4215	2001-04-16	Ray	G	Bayliss	10 Elizabeth Place		Paddington	Sydney	NSW	2021

Items

id	doc	FKid	quantity	Description
1	DeliveryReceipt.xml	1	2	Small Boxes

Info

id	개체번호	요소	속성
1	1	DeliveryReceipt	deliveryID
2	2	Customer	
... ...	... ...	...	...
16	4	Description	

그림 7 XML문서에 대한 관계형 테이블

5. 결론

XML 스키마가 표준으로 등장하면서 기존 DTD에 비해 XML은 데이터로서의 많은 장점을 가지게 되었다. XML 스키마는 XML 문법으로 작성되기 때문에 스키마 작성법을 배우기 위해 새로운 문법을 배울 필요가 없다는 점과 DTD보다 다양한 데이터형을 지원한다는 장점이 있다. 따라서 본 논문은 지금까지 연구된 DTD를 관계형 스키마로 사상시키는 방법을 기초로 XML 스키마를 관계형 스키마로 추출하는 기법을 제시하였다. 특히 XML에서 주로 사용되는 요소와 속성을 중심으로 규칙들을 제시하였으며, DTD에서 표현할 수 없는 사용자 정의 데이터형에 대한 추출 기법이 추가되었다. 또한, XML 문서와 XML 스키마에서 관계형 테이블을 추출하여 저장하는 예를 보였다. 본 논문은 XML 문서의 스키마를 이용해 현재 보편화 되어있는 관계형 데이터베이스 시스템에서 XML 문서를 효과적으로 관리하는데 적용될 수 있다.

[참고문헌]

[1] W3C Recommendation, "XML Schema", *http://www.w3.org/XML/Schema*, 2001.

[2] Alin Deutsch, M.F ernandex and D.Suciu, "Storing Semi-structured Data with STORED", *Proceedings of ACM SIGMOD Conference,* Philadelphia, Pennsylvania, May 1999.

[3] J. Shanmugasundaram, H. Gang, K. Tufte, CZhang, D. J. DeWitt, and J. F. Naughton, "Relational Databases for Querying XML Documents: Limitations and opportunities", *VLDB* '99, Edinburgh, Scotland, 1999.

[4] T. Shimura, M. Yoshikawa, and S. Uemura "Storage and Retrieval of XML Documents Using Object-Relational Database", *DEXA* 1999.

[5] Kevin Williams, *PROFESSIONAL XML Databases*, 2001.

[6] Jon Duckett, *PROFESSIONAL XML Schemas*, 2002.