유효성이 유지되는 XML Schema 의 저장 모델 설계

탁성수*, 이언배
한국방송통신대학교 정보과학과
e-mail : graynote@freechal.com* lub@mail.knou.ac.kr

Design of A Storage Model for Preserving XML Schema Validation

Sung-Su Tark *, Eun-Bae Lee
Dept. of Computer Science, Korea National Open University

요 약

본 논문에서는 관계형 데이터베이스를 이용하여 XML Schema 문서를 저장 및 갱신 시 유효성이 유지되는 저장 모델을 설계하였다. 기존의 XML Schema 저장 모델은 갱신 시 유효성 검증을 위해 XML Schema 를 파일로 추출하여 변경한 후 XML Parser 를 이용하는 복잡한 과정을 필요로 한다. 이 논문에서 설계한 모델은 슈퍼타입• 서브타입을 이용한 모델링과 관계형 데이터베이스에서 제공하는 참조 키, NOT NULL, Default, Check(Rule), Trigger 등의 제약 조건을 사용하므로 다른 응용 프로그램의 의존 없이 저장 및 갱신 데이터의 유효성을 간단하게 검증할 수 있다. 그러므로 이 저장 모델을 이용하면 XML Schema 문서가 데이터베이스에 저장이 되어 있는 그 자체로 유효한 스키마임을 보장할 수 있어 저장 및 갱신의 효율을 높일 수 있는 장점이 있다. 본 논문에서 설계한 저장 모델은 상용 관계데이터베이스인 SQL 서버 2000 에 적용하여 테이블과 제약조건을 설정한 후 데이터 갱신과정을 실험하였으며, 실험결과 XML Schema 의 작성규칙에 위배되는 변경 시도는 설정한 제약조건에 의해 사전 방지되어 데이터의 유효성이 유지됨을 확인하였다.

1. 서론

최근 XML 데이터를 질의하기 위해서 데이터베이스에 저장하는 방식이 많이 일반화되고 있다[1][2]. 또한 근래에는 데이터의 변경이 이루어지는 경우에 같이 저장되어 있는 DTD 및 XML Schema 에 의한 유효성을 검증하는 방법에 대한 연구가 차츰 진행되고 있다[3][4]. 하지만 이러한 시스템에서는 DTD 및 XML Schema 자체의 변경이 있을 경우에 XML Parser 로 재검증을 하여야만 유효성을 보장할 수 있기 때문에 간단한 수정이라도 XML Schema 파일로 재 추출해야 하는 문제점이 있다. 더욱이 XML Schema 저장 시 자동생성 아이디를 이용하거나 참조제약조건에 의해서 XML 데이터가 연결되어 있으면 상당히 복잡한 갱신 과정을 거쳐야 하는 어려움이 있기에 이러한 문제를 해결하기 위한 새로운 저장모델이 필요하다.

본 논문에서는 XML Schema 갱신 시 데이터베이스에 갱신되는 것 자체로 유효성이 보장되는 저장 모델을 제안한다. 우선 유효성을 보장하기 위해서 사용된 방법들을 제시하고 실제로 XML Schema 의 타입 별로 적용한 저장모델을 설계한다. 제안된 방식은 스키마 독립적인 저장모델로써 모든 XML Schema 데이터에 적용이 가능하고 유효성 검증을 위한 다른 응용 프로그램에 의존할 필요가 없는 장점이 있다.

본 서론에 이어 2 장에서는 XML DTD 및 Schema 저장 및 XML 데이터 유효성 검증에 관련된 연구를 소개하고, 3 장에서는 본 논문에서 제안하는 XML Schema 저장 모델 및 유효성 검증방법을 제시하고, 상용 데이터베이스인 SQL2000 에 적용한 사례를 보여준다. 4 장에서는 본 논문의 결론과 향후 연구과제에 대해 언급한다.

2. 관련연구

XML 데이터의 검증을 위해서 DTD 나 XML Schema 를 같이 저장하는 방식이 많이 연구되고 있다. 그 중

가장 단순한 방식은 네임스페이스에 따른 저장으로 XML Schema 의 targetNamespace 별로 저장하되 스키마 전체를 하나의 스트링으로 저장되고 특정 노드의 스키마 정보를 추출하기 위해서는 데이터 노드 경로 상의 노드들이 정의 된 스키마를 차례로 메모리로 읽어들여 파스 트리를 만들고 이를 검색함으로써 해당 스키마를 추출하게 된다[3]. 이러한 저장 방식에서의 XML Schema 의 갱신은 전체 내용을 파일로 추출하여 검증 절차 후 다시 저장해야 하는 단점이 있다.

XML 저장방식인 Edge approach[1]를 이용한 방법은 XML Schema 를 element, complexType, attribute 와 같은 엘리먼트들과 name, type 과 같은 애트리뷰트들을 각각 엘리먼트 테이블과 애트리뷰트 테이블에 저장하는 것이다. 하지만 이러한 저장 상태에서는 XML Schema 의 갱신 시 엘리먼트와 애트리뷰트 테이블 전체를 읽어서 조합하는 작업을 거쳐서 XML Schema 파일로 생성하여 검증하는 복잡한 절차가 필요하다.

XML Schema 의 추출시의 효율성을 고려한 최근 연구는 엘리먼트 이름별로 테이블을 생성하는 기본 Binary approach[1]에서 각각의 엘리먼트 별로 관련된 일부 엘리먼트들과, 애트리뷰트들을 해당 엘리먼트 테이블에 인라인닝하고, XML 데이터 갱신에 대한 유효검증 시 사용 될 스키마 정보를 Pathtable 기법을 통해 추출하는 방법이다[3]. 그렇지만 이 연구에서는 XML 데이터의 유효성이 유지되는 갱신에 대한 것일 뿐, XML Schema 자체의 변경에 대해서는 자체적으로 검증방법을 가지지 못하므로 유효성을 보장할 수 없다.

데이터베이스의 제약조건을 이용한 설계방식은 기존에 XML 데이터의 의미론적 제약조건을 유지하기 위한 스키마 의존적인 모델에서 제시된 바 있다[5].

3. 유효성이 유지되는 XML Schema 저장 모델

3.1 제안하는 저장 모델의 개요

XML Schema 를 데이터베이스에 저장하는 목적은 주로 XML 데이터에 대한 구조적인 제약조건을 확인하기 위해서이다. 간단한 XML Schema 가 데이터베이스에 저장되어 있는 경우는 XML Schema 파일로 추출하여 변경 후 XML Schema 자체 검증 이후 다시 정해진 규칙에 의해 저장을 할 수는 있겠지만 XML 데이터와의 참조 무결성 제약조건이 사용되고 있거나, XML Schema 를 저장할 때 자동 생성 아이디를 쓰고 있는 경우는 많은 부가적인 작업을 거쳐야 한다. 그러므로 XML Schema 의 수정은 데이터베이스에 저장이 되어 있는 상태에서 이루어지는 것이 바람직하고, 이 상태에서 XML Schema 의 유효성 검사가 이루어지는 것이 효율적이다.

하지만 XML Parser 을 사용하지 않고 데이터베이스에 저장되어 있는 XML Schema 의 유효성 검증은 쉬운 일이 아니다. 그러므로 가장 최선의 방법은 XML Schema 를 저장할 데이터베이스에 적절한 모델링을 하고 테이블 생성시 관계형 데이터베이스의 여러 제약조건을 이용하여 유효성을 위배되지 않도록 설계하는 것이다.

제안하는 모델은 슈퍼타입• 서브타입을 이용한 모델링과 관계형 데이터베이스에서 제공하는 참조 키, NOT NULL, Default, Check(Rule), Trigger 등의 제약 조건을 사용하여 XML Schema 타입별로 적용하였다.

3.2 저장 모델에 적용할 제약조건

유효성을 유지하기 위해서 사용된 방법들과 XML Schema 의 타입별 적용에 관한 것은 다음과 같다.

⑴ 슈퍼타입과 서브타입

XML Schema 는 element, attribute, complexType, simpleType 의 범위성, 즉 전역(Global)과 지역(Local)에 따라 가질수 있는 속성들이 달라진다. 예를 들어 전역 엘리먼트 선언시는 빈도 횟수를 나타내는 minOccurs 속성과 maxOccurs 속성을 사용할 수 없다. 만일 기존 저장 방식처럼 각 타입별로 같은 테이블에 저장이 된다면 불필요한 속성값이 정의되는 것을 제한할 수 있는 방법이 없다. 그러므로 각 타입은 아이디와 공통된 속성을 갖는 슈퍼타입과 고유 속성을 갖는 서브타입으로 테이블을 분리 생성되어야 한다.

<표 1>은 XML Schema 타입별로 범용적 슈퍼타입에 쓰일 공통된 속성과 전역, 지역의 서브타입에서만 가지는 고유 속성을 보여준다.

구 분	속 성
Element	Element(name) GlobalElement(substitutionGroup) LocalElement(minOccurs, maxOccurs, ref)
Attribute	Attribute(name) LocalAttribute(use, default, fixed, ref)
SimpleType	GlobalSimpleType(name)
ComplexType	GlobalComplexType(name)

<표 1> 슈퍼타입과 서브타입으로 가질 속성

⑵ 참조에 의한 무결성

XML Schema 를 작성할 때 많이 참조되는 Element 와 Attribute 의 경우는 전역으로 선언하고 이를 지역에서 참조하는 방식이 사용된다. 하지만 기존의 저장 방식에서는 전역이 아닌 지역 Element 와 Attribute 를 참조하게 되는 것과 같은 잘못된 참조를 방지하지 못한다. 그러므로 본 논문에서는 전역과 지역으로 테이블을 분리하고 전역 테이블의 Primary Key 를 참조하는 지역 테이블의 'ref' 속성을 Foreign Key 컬럼으로 작성하여 참조 무결성이 유지되도록 정의하는 방법을 제안한다. <표 2>는 XML Schema 타입별로 참조키 무결성 제약조건이 필요한 항목의 예이다.

속 성	참조키 제약
Local Element 의 ref 속성	Global Element 의 Id
Local Attribute 의 ref 속성	Global Attribute 의 Id
Attribute 의 type 속성	SimpleType 의 Id

<표 2> 참조키에 의한 무결성 제약의 예

⑶ NOT NULL 에 의한 필수값 지정

XML Schema 의 타입별로 생성 및 수정 시 필수와 선택요소가 존재한다. 그러므로 각 타입별 저장 테이블에 필수요소를 저장하는 컬럼에는 관계형 데이터베이스의 Not Null 제약조건으로 반드시 값이 정의되도록 강제시킨다. <표 3>에 나열된 각 구성요소 별 필수 속성들은 Not Null 제약조건을 사용한다.

구 분	필수 속성

Element	Element(name)
Attribute	Attribute(name)
SimpleType	SimpleType(childElement), GlobalSimpleType(name)
ComplexType	ComplexType(childElement), GlobalComplexType(name)

<표 3> XML Schema 에서의 필수 속성

⑷ Default 제약 조건

XML Schema 에는 선택된 속성의 값이 명시적으로 지정되지 않으면 기본값을 가지는 항목들이 있다. 이러한 규칙은 관계형 데이터 베이스의 Default 제약조건을 이용하여 값이 명시되지 않은 경우에 기본값을 부여하면 쉽게 구현될 수 있다. <표 4>는 Default 값이 존재하는 항목들로서 Element 와 Choice 의 minOccurs, maxOccurs 는 1 을, Attribute 의 use 속성은 optional 을 기본 값으로 가진다.

구 분	기본값을 가지는 속성
Element	LccalElement(minOccurs, maxOccurs)
Attribute	LocalAttribute(use)
Choice	Choice(minOccurs, maxOccurs)

<표 4> XML Schema 에서의 기본 값을 가지는 속성

⑸ Check 와 Rule 제약 조건

XML Schema 는 일부 Element 나 Attribute 로서 가질 수 있는 값이 정해져 있는 규칙이 있다. 예를 들어 심플타입 정의 시 사용되는 자식 엘리먼트는 restriction, list, union 만이 사용가능하고, 여러 개를 동시에 기술할 수 없는 제약조건이 있다. 이러한 규칙을 만족시키기 위해서 해당되는 컬럼에 관계형 데이터 베이스의 CHECK 나 Rule 제약조건을 사용하여 데이터의 값을 제한 하는 방법을 제안한다. <표 5>는 속성별로 가질 수 있는 값이 정해진 경우로서 CHECK 제약조건을 사용해야 되는 항목을 나타낸다.

구 분	Check 조건
Attribute 의 use 속성	optional, required
SimpleType 의 compositor	restriction, list, union

<표 5> check 조건이 필요한 속성

⑹ Trigger 적용

이 외의 복잡한 XML Schema 규칙에 대해서 Trigger 을 사용하여 유효성을 유지할 수 있다. 예를 들어 애트리뷰트의 default 값은 use 속성이 optional 일때만 값이 지정되도록 제약하는 데 사용될 수 있다.

3.3 XML Schema 저장 모델

3.2 절에서 제안된 방법을 사용하여 XML Schema 저장 모델을 설계하면 다음과 같다. 본 논문에서는 XML Schema 을 구성하는 여러 타입 중에 가장 핵심인 element, attribute, simpleType, complexType 로 범위를 국한한다.

⑴ 엘리먼트(element)

전역 엘리먼트는 빈도 횟수를 나타내는 minOccurs 속성과 maxOccurs 속성을 사용할 수 없고, 로컬 엘리먼트에서는 기본값으로 '1' 을 가진다. 로컬 엘리먼트에서의 참조는 전역 엘리먼트만이 가능하다.

컬럼	Key	Null	Default	용 도
EId	PK	N		엘리먼트 Id
EName		N		엘리먼트 이름
SCType		N		심플, 컴플렉스 타입구분
GLType		N		전역, 지역 구분
nillable		Y		Nillable 여부
SId	FK	Y		SimpleType 의 Id
CId	FK	Y		ComplexType 의 Id

<표 6 > Element 슈퍼타입 테이블

컬럼	Key	Null	Default	용도
EId	FK	N		엘리먼트 Id

<표 7 > Global Element 서브타입 테이블

컬럼	Key	Null	Default	용도
EId	FK	N		엘리먼트 Id
minOccurs		Y	1	최소 발생횟수
maxOccurs		Y	1	최대 발생횟수
Sord		N		엘리먼트 저장 순서
ref	FK	Y		전역 엘리먼트의 참조 Id
PCId	FK	N		상위 ComplexType 의 Id

<표 8 > Local Element 서브타입 테이블

⑵ 애트리뷰트(attribute)

전역 속성은 'use' 속성을 사용할 수 없고, 로컬 속성에서 참조는 전역 속성만이 가능하다. 속성에서의 데이터 타입은 반드시 simpleType 만 가능하다.

컬럼	Key	Null	Default	용도
AId	PK	N		애트리뷰트 Id
AName		N		엘리먼트 이름
GLType		N		전역, 지역 구분
SId	FK	N		SimpleType 의 Id

<표 9 > Attribute 슈퍼타입 테이블

컬럼	Key	Null	Default	용도
AId	FK	N		애트리뷰트 Id

<표 10 > Global Attribute 서브타입 테이블

컬럼	Key	Null	Default	용도
AId	FK	N		애트리뷰트 Id
use		Y	optional	use 속성값
default		Y		default 속성값
fixed		Y		fixed 속성값
ref	FK	Y		전역 애트리뷰트의 참조 Id
Sord		N		애트리뷰트의 저장순서
PCId	FK	N		상위 ComplexType 의 Id

<표 11 > Local Attribute 서브타입 테이블

이외에 애트리뷰트의 use 속성의 값을 제한하기 위해서 아래와 같이 CHECK 제약 조건을 사용한다.

use check(value in ("optional", "required"))

⑶ 심플타입(simpleType)

simpleType 정의 시 사용되는 자식 엘리먼트는 값으로서 restriction, list, union 가 올 수 있는데, 여러 개를 동시에 기술 할 수는 없다. 또한 Local simpleType 타입에는 name 속성을 가질 수 없고, 전역 심플 타입은 반드시 필요하다.

컬럼	Key	Null	Default	용 도
SId	PK	N		심플타입의 Id
childElement		N		자식 엘리먼트
GLType		N		전역, 지역 구분
BUType		N		빌트인, 사용자타입

<표 12 > simpleType 슈퍼타입 테이블

컬럼	Key	Null	Default	용 도
SId	FK	N		심플타입의 Id
GSName		N		심플타입 이름

<표 13 > Global simpleType 서브타입 테이블

컬럼	Key	Null	Default	용 도
SId	FK	N		심플타입의 Id

<표 14 > Local simpleType 서브타입 테이블

이외에 심플타입의 자식 엘리먼트는 유효성을 검증하기 위해 아래와 같이 CHECK 제약 조건을 사용한다.

childElement check(value in ("restriction", "list", "union"))

(4) 컴플렉스타입(complexType)

지역 컴플렉스 타입은 name 속성이 없고 전역 컴플렉스 타입은 반드시 필요하다.

컬럼	Key	Null	Default	용 도
CId	PK	N		complexType 의 Id
childElement		N		자식 엘리먼트
GLType		N		전역, 지역 구분

<표 15 > complexType 슈퍼타입 테이블

컬럼	Key	Null	Default	용 도
CId	FK	N		complexType 의 Id
CSName		N		complexType 이름

<표 16 > Global complexType 서브타입 테이블

컬럼	Key	Null	Default	용 도
CId	FK	N		complexType 의 Id

<표 17 > Local complexType 서브타입 테이블

3.4 스키마 갱신 및 데이터 유효성 실험 및 평가

제안된 모델을 이용하여 XML Schema 를 저장하고 갱신 시 유효성을 검증하는 실험을 하였다. 실험대상으로는 XML Schema Part 0: Primer Second Edition[6]에서 제공한 The Purchase Order XML Schema 을 택했다. 실험에 사용된 쿼리문은 XML Schema 를 DOM 으로 갱신한 후 그 내용을 토대로 생성한 것으로 가정한다.

(1) 스키마 갱신 시 유효성 유지검사

XML Schema 의 유효성에 위배되는 갱신이 실제로 일어났을 때 작업이 실패하는 지를 알아보기 위해서 전역 엘리먼트에 minOccurs 와 maxOccurs 속성을 입력하였다. (그림 1)은 이러한 열 이름이 존재하지 않는 메시지와 함께 갱신작업이 실패되었음을 보여준다.

```
Declare @iErrorCode int
Begin Transaction
  Insert Into XElement (Eld, SCType, EName, GLTYpe,NSId, SId)
  Values (18, 'S', 'orderDate', 'G', 1, 9)
  Select @iErrorCode = @@Error

  Insert Into XGlobalElement (Eld, minOccurs, maxOccurs)
  Values (18, 1, 1)
  Select @iErrorCode = @@Error

If @iErrorCode = 0    Commit Transaction
Else  Rollback Transaction
```

```
서버: 메시지 207, 수준 16, 상태 1, 줄 1
열 이름 'minOccurs'이(가) 잘못되었습니다.
```

(그림 1) 불필요한 속성 추가 실험 결과

또한 (그림 2)과 같이 전역 엘리먼트(EId : 5)를 참조하는 로컬엘리먼트를 다른 로컬엘리먼트(EId : 7)로 참조하고자 했을 때 외래키 제약조건에 의해서 실패로 종료되었다.

```
Update XLocalElement Set Ref = 7 Where Eld = 5
```

```
서버: 메시지 547, 수준 16, 상태 1, 줄 1
UPDATE 문이 COLUMN FOREIGN KEY 제약 조건 'FK__XLocalEleme__Ref'과(와) 충돌되었습니다. 충돌은 'XSchema' 데이터베이스, 'XGlobalElement', column 'Eld' 테이블에서 발생했습니다. 문이 종료되었습니다.
```

(그림 2) 참조 무결성 위배 실험 결과

실험 결과와 같이, 제안된 모델은 XML Schema 를 변경하고자 할 때 갱신 이전에 유효 검증을 수행하고 유효한(Valid) 경우에만 적용이 된다는 것을 확인하였다. 이처럼 제안하는 XML Schema 저장모델은 데이터베이스에 저장된 상태에서 검증이 이루어지므로 다른 응용 프로그램을 통하지 않고 가장 적은 비용으로 갱신이 이루어지는 효율적인 방식이다.

4. 결론 및 향후 연구과제

XML 데이터의 구조적인 검증을 위해서는 XML Schema 도 같이 데이터베이스에 저장되어야 한다. 하지만 기존의 저장방식으로는 XML Schema 의 갱신 필요성이 발생할 경우에 다시 파일로 재 추출하여 검증하는 비효율적인 절차를 거쳐야 하기에 이를 해결하기 위한 새로운 XML Schema 저장모델을 제안하였다.

본 논문에서 제안한 모델은 XML Schema 를 구성하는 여러 타입 중에서 가장 핵심이라 할 수 있는 element, attribute, simpleType, complexType 을 대상으로 하였다. 효율적인 유효성 검증을 위해서 데이터베이스의 슈퍼타입• 서브타입을 이용한 모델링과 참조 키, NOT NULL, Default, Check(Rule), Trigger 등의 제약 조건을 사용하여 XML Schema 의 갱신이 일어났을 때도 데이터베이스에 저장된 상태에서 수정이 가능하다. 이로 인해 다른 검증절차를 거치지 않고 언제나 유효한 XML Schema 임을 보장할 수 있으므로 가장 적은 비용으로 스키마 갱신이 가능한 이점이 있다.

향후 과제는 DOM 을 이용한 XML Schema 변경 시 제안한 저장모델에 적용될 수 있는 쿼리문 자동 시스템을 구현하고, 스키마 갱신내용에 대응하는 XML 데이터의 유효성 검증절차에 대한 연구이다.

참고문헌

[1] Daniela Florescu, Danald Kossmann "Storing and Querying XML Data using an RDMBMS", Bulletin of the IEEE Computer Society Technical Committee on Data Engineering, 1999

[2] Jayavel Shanmugasundaram, Kristin Tufte, Gang He, Chun Zhang, David DeWitt, Jeffrey Naughton, "Relational Databases for Querying XML Documents: Limitation and Opportunities", Proceedings of the 25th VLDB, 1999

[3] 이지현(2004), "XML Schema 에 대한 유효성을 보장하는 ORDBMS 에 저장된 XML 데이터 갱신 기법", 석사 학위 논문, 한국과학기술원

[4] 김상균(2001), "XML 데이터베이스 변경 연산의 부분 즉시 검증 메커니즘", 공학석사 학위 논문, 충북대학교

[5] Lee, D., Chu, W. W., "Constraints-preserving Transformation from XML Document Type Definition to Relation Schema", In: Int'l Conf. on Conceptual Modeling (ER). Springer, LNCS 1920, Salt Lacke City, UT.

[6] XML Schema Part 1 : Structures Second Edition, W3C, http://www.w3.org/TR/2004/REC-xmlschema-1-20041028