XML Schema를 위한 관계형 스키마 자동생성기의 개관

김정섭[0] 박창원 정진완
한국과학기술원 전자전산학과 전산학전공
{jskim[0], cwpark, chungcw}@islab.kaist.ac.kr

An Overview of an Automatic Relational Schema Generation System for the XML Schema

Jung-Sup Kim[0] Chang-Won Park Chin-Wan Chung
Department of Electrical Engineering and Computer Science
Division of Computer Science
Korea Advanced Institute of Science and Technology

요 약

XML이 널리 확산되면서 XML문서들을 관계형 데이터 베이스에 저장하기 위한 관계형 스키마의 생성이 더욱 중요해 지고 있다. 기존의 DTD기반 관계형 스키마 생성 기법은 XML Schema에 적용될 수 없는데, 그 이유는 XML Schema에는 다양한 데이터 타입, 상속, 다형성과 같은 DTD에 존재하지 않는 새로운 특징들이 많기 때문이다. 이 논문은 XML 문서들의 구조를 정의한 XML Schema로부터 자동으로 관계형 스키마를 생성해 내는 XML Schema를 위한 관계형 스키마 자동생성기의 개관을 설명한다.

1. 서 론

HTML과 SGML의 단점을 보완하기 위해 제안된 XML은 이미 전자 상거래, 각종 문서의 인코딩, 그리고 멀티미디어 데이터 인코딩 등을 위한 각종 어플리케이션에서 사용되고 있다. 이와 함께, XML문서를 이미 많은 사용자를 확보하고 있는 관계형 데이터 베이스에 저장하고 검색하기 위한 연구가 활발히 수행되고 있다. 그러나, 계층적 구조를 가진 XML문서를 평평한 구조의 관계형 데이터 베이스에 저장하는 작업은 쉽지 않다. 무엇보다도 관계형 테이블로는 계층 구조를 표현하기 어렵기 때문이다.

본 논문은 XML Schema[1]로부터 관계형 스키마를 자동으로 생성하는 XML Schema를 위한 관계형 스키마 자동 생성기의 개관에 대하여 설명한다. XML Schema는 데이터 중심의 XML문서를 위하여 사용될 수 있도록 DTD와는 다른 많은 새로운 요소들을 포함하고 있다. 다양한 데이터 타입, 상속, 다형성 등과 같은 요소들은 XML Schema를 보다 강력한 스키마로서의 역할을 할 수 있도록 해 주지만, 한편으로는 DTD를 이용하는 인라인 기법이 더 이상 XML Schema에 적용될 수 없는 결과를 초래하였다. 본 논문은 XML Schema의 다양한 각각의 요소들에 대해서 관계형 스키마를 어떻게 생성할 것인가를 간략하게 제시하고 있으며, 다양한 휴리스틱 방법들을 통하여 보다 성능 향상된 XML Schema 인라인 기법이 가능함을 간략하게 설명한다.

2. 관련 연구

2.1. XML문서의 저장

XML문서를 저장하고자 할 때, DTD와 같은 스키마 정보를 이용하는 방법으로는 [4]와 [5]가 있다. [4]의 방법에서는 DTD에서 표현될 수 있는 내용들 중에서 실제 관계형 테이블들을 만드는데 사용될 수 있는 정보들만을 이용하여 DTD Graph와 Element Graph를 만들고, Element Graph를 깊이 우선 순회하면서 방문하는 엘리먼트들을 하나의 관계형 테이블에 인라인 시키는 방법을 사용하고 있다. 구체적으로 살펴 보면, 그래프의 루트 엘리먼트에 해당하는 테이블을 생성한 다음 루트 엘리먼트의 자손 노드들을 모두 이 테이블의 어트리뷰트로 인라인 시키게 되는데 다음과 같은 예외가 있다. 첫째는 같은 엘리먼트가 여러 번 나타나는 경우, 둘째는 그래프 내에서 사이클을 만나는 경우이다. 이 경우에는 새로운 테이블을 생성하여 그 엘리먼트의 자손들을 인라인 시키게 된다. 이러한 과정은 그래프의 모든 노드를 방문할 때까지 반복된다.

그러나, [4]에서 제안하는 방법들은 DTD를 위한 내용들이라서 XML Schema에는 적용되기 어렵다. 무엇보다 엘리먼트가 문서 구조의 주축을 이루는 DTD와는 달리, XML Schema는 타입과 엘리먼트라는 구성 요소가 주축을 이루고 있고 타입들은 상속과 다형성의 속성을 갖기 때문에 DTD Graph와 Element Graph를 그리는 것이 불가능하다.

2.2. XML Schema

지금까지 DTD가 XML 문서의 구조를 정의하는 수단으로 사용되었다. 그러나, DTD는 여러 가지 문제점들을 내포하고 있는데, 이를 해결하기 XML Schema[1]가 새롭게 제안되었다.

첫째, XML Schema는 XML의 syntax를 따르고 있다. 즉, XML Schema자체가 하나의 XML 문서가 되며, 따라서 이미 존재하고 있는 XML문서 처리 기술을 이용해서 XML Schema를 처리할 수 있다.

둘째, XML Schema는 다양한 데이터 타입을 제공한다. string, byte, integer, date를 비롯한 40가지 이상의 Built-in 타입들을 제공하고 있으며(이들은 simple type이라고 불린다), 그 외에도 사용자는 자신이 원하는 타입들을 새로이 정의하여 사용할 수 있다.

셋째, XML Schema는 namespace를 지원한다. namespace를 이용하면 이미 만들어져 있는 다른 XML Schema로부터 타입과 전역 엘리먼트들을 재사용할 수 있는 장점이 있다. 또한, 여러 개의 XML Schema를 정의하여 하나의 XML문서를 작성하거나 문서의 타당성(validity)를 검증할 수도 있다.

넷째, XML Schema는 기본적으로 타입들로 구성되는데, 타입은 다시 simple type과 complex type으로 구분된다. Complex type은 어트리뷰트 또는 엘리먼트들을 이용하여 정의가 된다. 이러한 타입들은 상속이 가능하며, 추상 타입을 정의할 수 있는 점 등은 객체 지향 프로그래밍 언어와 같다.

3. 관계형 스키마 자동 생성 시스템

그림 1은 시스템의 구성을 나타내고 있다. 우선 XML Schema가 시스템에 입력되면 타입 분리기가 XML Schema 내의 구성 요소들을 분류하여, 엘리먼트 처리기, 데이터 타입 변환기, 스키마 그래프 생성기 등을 거쳐서 최종적으로 관계형 스키마 생성기에 의하여 관계형 스키마를 생성하게 된다. 이 장에서는 각 부분들에 대해서 간략히 설명하도록 한다.

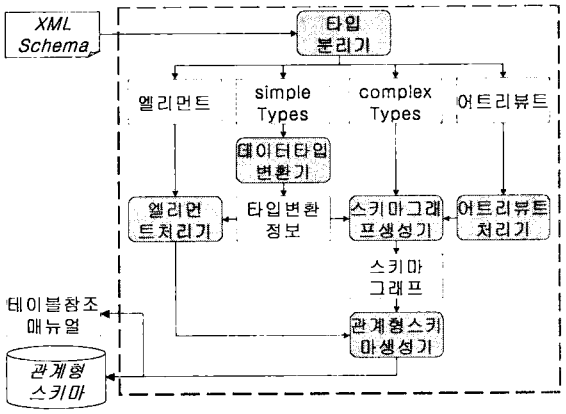

그림 1 시스템 구성도

3.1. 타입 분리기

일반적으로 XML Schema의 구성은 전역 엘리먼트, 타입, 전역 어트리뷰트, 엘리먼트 그룹, 어트리뷰트 그룹으로 구성되어 있다. 타입 분리기가 하는 일은 이들을 구분하여 필요로 하는 다른 모듈에게 전달하는 것이다. 예를 들어, 전역 엘리먼트 선언을 만나게 되면, 이를 엘리먼트 처리기에게 전달한다. 타입의 경우에는 simple type과 complex type으로 구분하여 처리한다. 엘리먼트 그룹과 어트리뷰트 그룹은 각각 엘리먼트와 어트리뷰트로 처리한다.

3.2. 데이터 타입 변환기

XML Schema는 string, byte, integer, date를 비롯한 40가지 이상의 다양한 Built-in 데이터 타입을 지원할 뿐 아니라, Built-in 데이터 타입으로부터 상속을 하여 새로운 데이터 타입을 만들어 낼 수 있다.

그러나, 일반적으로 관계형 데이터 베이스 시스템은 XML Schema에서 정의된 타입(simpleType) 들을 100% 지원하지 못한다. 예를 들어 오라클 DBMS의 경우, 문자열 타입을 위한 VARCHAR2, 또는 CHAR타입, 숫자 타입을 위한 NUMBER타입, 그리고 DATE를 비롯한 몇몇 Built-in타입이 있을 뿐이다. 따라서, XML Schema를 관계형 스키마로 매핑시키기 위해서는 XML Schema에서 제공하는 타입들은 모두 관계형 데이터 베이스 시스템에서 지원하는 타입들로 단순화되어야 한다. 다음으로는 사용자가 XML Schema의 Built-in타입들을 상속 받아 새로이 정의한 simple Type들의 경우에는 XML Schema의 Built-in타입과 관계형 데이터 베이스 시스템의 Built-in타입간의 매핑 정보, 그리고 사용자 정의 타입의 상속 정보를 이용하여, 관계형 데이터 베이스 시스템의 데이터 타입으로 매핑시켜야 한다.

3.3. 엘리먼트 처리기

엘리먼트 처리기로 전달되는 것은 크게 세 가지로 구분할 수 있다. 첫째는 XML문서의 루트 엘리먼트에 해당하는 엘리먼트이다. 이 엘리먼트는 나중에 관계형 스키마 생성기에 의하여 독립된 테이블을 생성하게 된다. 둘째는 전역 엘리먼트로서 complex type의 선언 때, 구성 요소로 참조된다. 셋째는 엘리먼트 그룹으로 둘째의 경우와 같이 complex type의 선언 때, 구성 요소로 참조된다. 한편, 엘리먼트가 XML문서의 루트 엘리먼트에 해당하는지 또는 단순한 전역 엘리먼트로만 사용되었는지의 여부는 스키마 그래프가 완성된 후에 파악이 가능하다.

3.4. 스키마 그래프와 스키마 그래프 생성기

본 논문에서는 XML Schema의 구조를 그래프로 나타내기 위해 스키마 그래프와 타입 그래프를 제시한다. 이 그래프들의 가장 큰 특징은 기본 요소로 타입과 엘리먼트를 동시에 다루고 있다는 점이다

스키마 그래프에는 여러 가지 형태의 데이터들이 나타나고 있는데 먼저, 사각형은 타입을 나타내며, 타원은 엘리먼트를 나타낸다. 실선으로 된 화살표는 엘리먼트의 내포관계를 나타내는데, BookType이라는 타입에는 title, section, year, author라는 엘리먼트들이 내포되어 있음을 보여주고 있다. 이 실선 위에는 발생 횟수(cardinality)를 나타내는 값이 올 수 있는데, *는 maxOccurs = "unbounded"로서 제한이 없음을 나타내며, 아무런 숫자도 표시되어 있지 않는 경우는 그 값이 1로서 생략된 경우이다. 한편, 화살표가 없는 실선은 엘리먼트의 타입 정보를 나타낸다. 예를 들어, title이라는 엘리먼트의 타입은 string이며, section의 타입은 SectionType임을 위의 스키마 그래프에서 확인할 수 있다. 점선으로 표현된 화살표는 상속관계를 나타내는 것으로 화살촉이 있는 쪽이 부모 타입이 되고, 반대편이 자식 타입이 된다. 예를 들어 AuthorType은 PersonType으로부터 상속을 받았으며, SpecialAuthorType은 AuthorType으로 상속받았음을 알 수 있다.

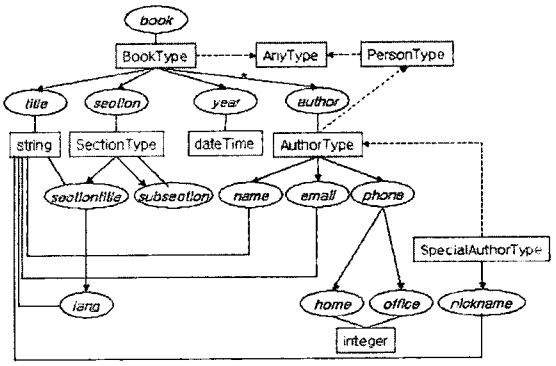

그림 2 스키마 그래프

타입들 중에서 string, anyType, SectionType과 같이 상속관계 트리의 루트에 있는 타입들을 루트 타입이라고 하고, NameType, SpecialAuthorType, SectionType과 같이 단말에 위치한 타입들을 단말 타입이라고 하며, 루트 타입을 포함하여 단말 타입이 아닌 타입들을 비단말 타입이라고 한다.

3.5. 관계형 스키마 생성기

스키마 그래프를 순회하면서 관계형 스키마를 만들 때, 테이블을 생성하는 기준들은 다음과 같다. 첫번째는 [4]의 Element Graph에서 * 연산자를 만나는 것과 같은 경우이다. 즉, 주어진 타입에 내포되어 있는 엘리먼트의 maxOccurs의 값

이 2이상 일 때 해당하는 엘리먼트는 독립적인 테이블로 만들게 된다. 둘째는, 사이클이 형성되어 있는 경우에는 독립적인 테이블이 만들게 된다. 그림 2의 스키마에서는 SectionType을 독립된 테이블로 생성하게 된다. 셋째는 엘리먼트의 타입이 추상 타입으로 선언되어 있는 경우, 그 추상타입으로부터 상속된 구체 타입들은 독립된 테이블을 생성한다. 한편, 현재 생성중인 테이블에는 하나의 칼럼을 추가하여, 인스탄스 XML문서에서 어떤 구체 타입을 사용하는지, 즉 어떠한 테이블에 데이터가 저장되었는지를 표시한다. 이렇게 추상 타입으로 선언된 경우에 자식 타입들을 독립적인 테이블로 구성하는 이유는 XML문서를 보지 않은 이상, 서브 엘리먼트들의 구조를 알 수 없기 때문이다. 한편, 추상 타입으로 선언된 경우의 엘리먼트는 XML 문서 상에서 xsi:type이라는 어트리뷰트의 값을 통해서 서브 엘리먼트들의 구조를 지정하게 된다. 넷째는 루트 엘리먼트에 해당하는 경우에는 테이블을 생성한다. 다섯째, 엘리먼트의 타입이 비단말 타입으로 선언된 경우에, 그 비단말 타입으로부터 상속된 자식 타입들은 독립된 테이블로 구성한다. 비단말 타입의 경우에도 xsi:type을 사용하여 XML문서에서 선언된 타입의 자식 타입을 사용할 수 있기 때문이다.

3.6. xsi:type의 처리

앞에서 설명한 것과 같이 xsi:type은 추상타입에 뿐만 아니라, 구체 타입일지라도 비단말 타입 노드인 경우에 사용될 수 있다. 이처럼, 스키마에서 정의된 타입과는 다른 타입이 사용될 수 있다는 사실은 XML Schema만을 보고서는 정확히 어떠한 데이터가 어떠한 테이블에 저장되는지를 결정할 수 없음을 의미한다. 하지만, 다행히 xsi:type에 올 수 있는 타입은 스키마에서 정의된 타입으로부터 상속을 받은 타입들만 올 수 있다. 따라서, 우리는 스키마에서 선언된 타입의 자식 타입들을 위한 테이블들을 미리 생성해 둘 필요가 있게 된다.
이와 함께, 어떤 타입이 XML문서에서 사용되었는지를 저장해 두어야 하는데, 이 값은 xsi:type이라는 칼럼을 두어서 처리한다. 그런데, 이 칼럼을 어디에 두느냐는 경우에 따라서 조금 다르다. 만약 선언된 타입의 maxOccurs가 1이어서 인라인 되는 경우에는 인라인 되는 테이블에 xsi:type이라는 칼럼을 추가한다. 만약 maxOccurs가 unbounded이면, xsi:type칼럼은 새로이 생성되는 테이블에 추가가 된다.

3.7. 휴리스틱 방법

첫번째, 반 인라인 기법(counter inlining technique)으로 이름이 말하고 있듯이, 타입 그래프를 순회하면서 관계형 테이블을 만들어 나갈 때, 비록 엘리먼트의 maxOccur가 1일지라도 성능 향상의 목적상, 또는 관리의 목적상, 인라인 시키지 않는 것이 바람직하다고 볼 때에는 엘리먼트를 인라인 시키지 않고 독립된 테이블로 만드는 것이다. 둘째는, DTD의 경우 데이터의 발생 회수(cardinality)가 0또는 1, 0이상과 같이 매우 한정적이지만, XML Schema의 경우에는 구체적으로 몇 번이 발생하는지를 명시할 수 있다. 즉, 엘리먼트의 어트리뷰트 maxOccurs에 값을 할당하면 된다. 이 값을 이용해서 기준 값보다 작은 경우에만 인라인 시키고 기준 값보다 클 경우에는 독립적인 테이블로 만드는 방법이 있다.

3.8. 테이블 참조 매뉴얼

시스템의 구현과 함께, 시스템에서 생성된 테이블들의 구조와 테이블들간의 관계를 쉽게 파악할 수 있도록 해주는 도구가 필요한데, 이러한 목적을 위하여 사용되어 질 수 있는 테이블 참조 매뉴얼도 자동으로 생성된다. 테이블 참조 매뉴얼은 브라우징이 가능한 형태의 웹 문서로 되어 있다. 그림 3은 웹 브라우저를 통해서 본 테이블의 구조를 보여주고 있다. 웹 문서는 3개의 프레임으로 구성되어 있는데, 상단의 프레임에는 시스템을 통해서 생성된 모든 테이블들이 나타나 있다. 오른쪽 하단의 프레임에는 사용자에 의하여 선택이 된 테이블의 칼럼들이 나타나 있는데, 칼럼의 이름은 기본적으로 col이라는 영문자에 일련번호를 붙여서 생성하였고 그 칼럼이 나타내는 XML문서의 엘리먼트를 경로로서 나타내고 있다. 왼쪽 하단에는 선택된 테이블의 주키(primary key)를 외래키로 갖고 있는 테이블들의 목록이 나타나 있다.

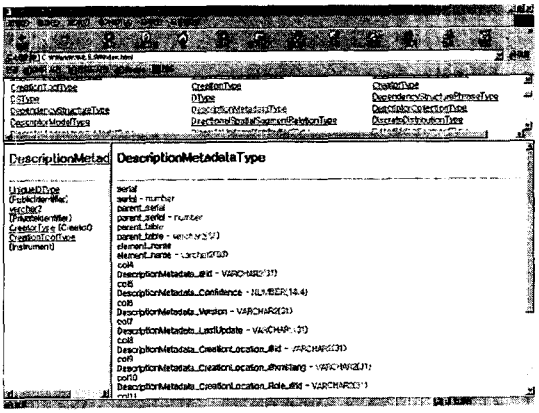

그림 3 테이블 참조 매뉴얼

4. 결 론

본 논문에서는 XML 문서들의 구조를 정의한 XML Schema로부터 자동으로 관계형 스키마를 생성해 내는 XML Schema를 위한 관계형 스키마 자동생성기의 개관을 설명하였다. 본 논문을 통해 첫째로 XML Schema로부터 관계형 테이블들을 생성하기 위해 필요한 정보들로 구성된 스키마 그래프와 타입 그래프를 제시하고, 둘째로 XML Schema의 데이터 타입, 상속, 다형성 등과 같은 다양한 각각의 요소들에 대해서 관계형 스키마를 어떻게 생성할 것인가를 제시하였으며, 셋째로 시스템의 성능을 향상시킬 수 있는 휴리스틱 기법들과 생성된 관계형 스키마의 이용을 돕기 위한 테이블 참조 매뉴얼을 제시하였다.

참고 문헌

[1] XML Schema Part 0: Primer, http://www.w3.org/TR/xmlschema-0/
[2] XML Schema Part 1:Structure, http://www.w3.org/TR/xmlschema-1/
[3] XML Schema Part 2:Datatypes,http://www.w3.org/TR/xmlschema-2/
[4] Jayavel Shanmugasundaram, Kristin Tufte, Gang He,Chun Zhang, David DeWitt, Jeffrey Naughton, "Relational Databases for Querying XML Documents : Limitations and Opportunities," Proceedings of VLDB 1999, pp. 302-314.
[5] Gerti Kappel, et Al, "X-Ray - Towards Integrating XML and Relational Database Systems," Proceedings of ER 2000, pp. 339-353.