함수적 종속성을 반영한 XML 문서의 관계형 스키마 매핑 기법

A Mapping Technique of XML Documents into Relational Schema based on the functional dependencies

조 정 길*
Jung-Gil Cho

요 약

많은 기법들이 XML을 릴레이션으로 매핑(mapping) 하기위해 제안되었다. 그러나 대부분의 기법들은 XML 데이터의 의미(semantic)들을 고려하지 않았다. 이러한 의미들은 스키마를 설계하는 과정에 저장, 질의 최적화, 변경 이상 등을 체크하는 데에 매우 중요하다. 특히 함수적 종속성은 데이터베이스 이론의 중요한 부분이고, BCNF에서 관계형 테이블을 정규화하기 위한 기초를 형성한다. 이 논문은 XML 스키마 기반의 XML을 매핑하여 릴레이션으로 저장하기 위하여 함수적 종속성을 반영한 기법을 제공한다. 내용, 구조와 함께 함수적 종속성에 의해 기술된 제약조건들은 동시에 유지되며, 저장 중복성을 줄일 수 있다.

Abstract

Many techniques have been proposed for mapping from XML to relations, but most techniques did not negotiate the semantics of XML data. The semantics is important to validate storage, query optimization, modification anomaly in process of schema design. Specially, functional dependencies are an important part of database theory, also it is basis of normalization for relational table in BCNF. This paper propose a new technique that reflect functional dependencies to store relation mapped from XML based on XML Schema. The technique can reduce storage redundancy and can keep up content and structure with constraint described by functional dependencies.

1. 서론

XML은 인터넷에서 데이터 교환을 위한 포맷으로 널리 보급되었다. XML은 많은 이점을 갖고 여러 종류의 어플리케이션에서 사용되어졌다.

DTD와 XML 스키마[1,2]가 XML을 위한 스키마 언어로 연구되어 왔다. DTD는 현재까지도 사실상의 스키마 언어이다. 키와 외래키를 위한 DTD의 지원은 ID와 IDREF의 형식에 해당된다. 이러한 ID와 IDREF를 데이터베이스의 키 제약조건으로 사용함에는 많은 문제들이 있다. XML 스키마는 DTD의 부족분을 매우기 위하여 W3C (World Wide Web Consortium)에 의하여 제안되었고, 앞으로는 XML 스키마의 장점으로 인하여 급속도로 DTD를 XML 스키마로 교체하게 될 것이다[3,4]. XML 스키마는 형식 제약조건과 더욱 복잡한 출현지시자(cardinality) 제약조건의 특징을 제공한다. 그럼에도 불구하고 XML 스키마 언어는 함수적 종속성[5] 제약조건을 상술하기 위한 구조를 제공하지 않는다.

XML 데이터에서 키나 FD 등과 같은 XML 의미들이 없기 때문에, 많은 논문들[6~11]이 XML 제약조건을 여러 가지 표시법(notation)으로 제안하였다. XML을 데이터베이스 중심으로 보는 관점에서, [6, 8]은 XML의 문맥에 있는 함수적 종속성의 개념이 탐구되지 않는 것에 주목하였다.

* 정 회 원 : 성결대학교 공과대학 컴퓨터공학부
jkcho@sungkyul.edu
[2006/07/28 투고 - 2006/08/18 심사 - 2007/01/03 완료]

또한 [7, 9]는 함수적 종속성의 특별한 케이스인 키를 XML의 문맥으로 연구하였다. [11]은 XML 데이터의 의미들을 뽑아내어 XML에 대한 FD를 정의했기 때문에 이 논문에서 중요하다. 관계형 데이터에서 FD는 키들을 찾고, 좋은 설계를 위한 정규형과 갱신 이상 예방 등에 유용하다. 그렇기 때문에 XML에 대한 FD들은 관계형 데이터를 위해서 필요하고 매핑 절차에서 유지해야 한다. 따라서 XML FD에 따라 릴레이션으로 XML을 매핑하는 것은 가능하며, 동시에 내용과 구조와 같이 제약조건들을 유지할 수가 있다.

그러나 이러한 많은 논문들[6~11]은 DTD 기반의 XML 제약조건을 제안하였다. 따라서 이 논문에서는 형식 제약조건과 다양한 출현지시자 제약조건을 기술할 수 있는 XML 스키마[2] 기반에서 XML 데이터의 의미들을 추출하여 XML에 대한 FD를 정의하는 XML 제약조건에 대한 표시법을 제안한다.

이 논문에서 연구는 XML 데이터의 의미와 관계형 데이터의 의미 사이에서 관계성을 연관시킨다[12~14]. [12]는 하이브리드 인라이닝 기법[16]에 기반하여 키와 KeyRef에 의해 정의된 제약조건들을 연관시켜 XML을 릴레이션으로 매핑하는 기법을 제공한다. [13]에서는 미리 정의된 XML 키로부터 FD를 결정하는데 PTIME 알고리즘을 제공한다. 즉, 릴레이션 스키마는 수작업으로 변환하기 위한 변환 규칙을 상술한다. 그러한 규칙 기반의 알고리즘은 몇몇 스키마에서 주어진 FD를 XML 키들이거나 키가 아닌 것으로부터 끌어낼 수 있는지를 결정하는데 사용될 수 있다. [14]는 DTD로부터 제약조건들을 끌어내는데 CPI 알고리즘을 제공하며, 매핑에 사용된 기법은 역시 하이브리드 인라이닝 기법에 기반한다.

이 논문은 XML 스키마기반의 XML 문서를 XML FD에 따라 XML을 릴레이션으로 매핑하는 기법을 제공한다. 이 논문의 구성은 다음과 같다. 2장에서는 함수적 종속성의 표시법을 소개한다. 3장에서는 매핑하기 위한 자세한 기법을 기술하고 몇 가지 증명을 하였다. 4장에서는 기법을 확실하게 설명하기 위하여 예제를 들었다. 5장에서는 결론 및 향후 연구계획을 제시하였다.

2. 함수적 종속성

관계형 데이터 모델에서 함수적 종속성을 다음과 같이 정의 한다:

> 어떤 릴레이션 R에서 X와 Y를 각각 R의 애트리뷰트 집합의 부분 집합이라 하자.
> 애트리뷰트 X의 값 각각에 대해 시간에 관계없이 항상 애트리뷰트 Y의 값이 오직 하나만 연관되어 있을 때 Y는 X에 함수적 종속성이라 하고, $X \rightarrow Y$ 로 표기한다.

함수적 종속성을 FD라고 줄여서 사용한다. 애트리뷰트들의 집합 X를 FD의 LHS(Left-Hand-Side)이라 부르고, Y를 RHS(Right-Hand-Side)이라고 부른다.

관계형 데이터 모델의 정의는 평평하고 이차원적인 테이블에서 관계형 데이터를 사용하기 위하여 설계되었다. 그렇기 때문에 계층적 구조인 XML 데이터를 직접 적용할 수가 없다. 따라서 XML 데이터를 저장하기 위해서는 XML의 계층적 특성을 반영한 다른 표시법이 필요하다. 이러한 특징들은 처음 [11]에서 적용하였으며, 기술된 몇몇 표시법을 소개한다.

정의 1.

XML의 함수적 종속성 FD는 $(Q,[P_{x1}, P_{x2}, ..., P_{xn} \rightarrow P_y])$ 형식의 표시법이다 :

여기서, Q는 FD 헤더 경로이고, XML 문서의 루트로부터 경로가 시작되는 간단한 XPath[16] 표시법이다. P_{xi} $(1 \leq i \leq n)$는 엘리먼트 이름과 선택된 키 애트리뷰트들로 이루어져 있는 LHS 엔

티티 타입이다. 이것들은 키 애트리뷰트에 의해 유일하게 정의될 수 있다. P_y는 RHS 엔티티 타입이며, 엘리먼트 이름과 선택된 키 애트리뷰트 이름으로 이루어져 있다. XML FD ($Q,[P_{x1}, P_{x2}, ..., P_{xn} \rightarrow P_y]$)는 Q에 의해 인지된 어떤 두개의 서브트리에 대하여, 만약 모든 LHS 엔티티가 값인 $P_{x1}, P_{x2}, ..., P_{xn}$에 대하여 동의하면, RHS 엔티티 값인 P_y에 대하여 동의하여야 한다.

함수적 종속성 FD의 표시법은 XPath 표시법을 사용하여 만든다. 일반적으로 FD 헤더 Q는 범위를 한정하고 함수적 종속성이 유지된 노드 집합을 정의한다. 노드 집합은 FD가 유지되는 동안 "record"의 수집으로 볼 수가 있다. P_{xi} ($1 \leq i \leq n$)는 함수적 종속성의 LHS 엔티티 타입으로 간주하고, P_y는 RHS 엔티티 타입으로 간주한다. (P_{xi}, P_y)은 각각 전통적인 함수적 종속성 표시법의 LHS 와 RHS 표시법과 같은 뜻이다.

헤드 패스인 LHS 엔티티들과 RHS 엔티티들 사이에서 관계성에 의해 분류된 XML FD는 두 종류의 형식이 있다. 두 종류의 중요하고 의미 있는 형식은 잘 구조화된(well-structured) FD 와 평평한(flat) FD이다.

정의 2. DTD를 고려한다 :

<!ELEMENT H_1 (H_2)*>

. . .

<!ELEMENT H_m (E_1)*>

<!ELEMENT E_1 (E_2)*>

. . .

<!ELEMENT E_x (E_y)*>

$Q = /H_1/ .../H_m$, $P_{xi}(1 \leq i \leq n)$는 집합 $\{E_1,...,E_x\}$에 있는 엘리먼트와 그것의 임의의 애트리뷰트들로 이루어져 있고 P_y는 E_y와 임의의 애트리뷰트들로 이루어져 있을 경우에, 만약 1, 2, 3번의 조건들을 만족한다면 XML FD인 $F= (Q, [P_{x1}, P_{x2}, ..., P_{xn} \rightarrow P_y])$는 잘 구조화 되었다고 부른다.

1. 단일 RHS 엔티티 타입이다.
2. Q에 있는 순서화된 XML 엘리먼트인 LHS 엔티티 타입과 RHS 엔티티 타입($P_{x1}, P_{x2}, ..., P_{xn}, P_y$)은 정렬(lineage)을 형성 한다; (집합에 있는 모든 노드들이 N의 조상인 집합 L에 노드 N이 존재하고, L에 있는 모든 노드 M에 대하여 만약 M의 조상들이 L에 있고 M의 부모 역시 L에 있다면, 트리에서 노드들의 집합 L은 정렬이다.)
3. LHS 엔티티 타입은 어떤 중복된 LHS 엔티티 타입도 포함하고 있지 않다.

정의 3. DTD를 고려한다 :

<!ELEMENT H_1 (H_2)*>

. . .

<!ELEMENT H_{m-1} (H_m)*>

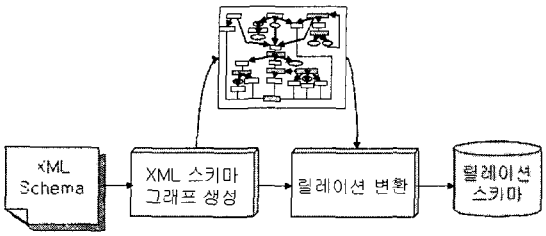
<!ELEMENT H_m (E_1,...,E_x,E_y)>

$Q = /H_1/ .../H_m$, $P_{xi}(1 \leq i \leq n)$는 집합 $\{E_1,...,E_x\}$에 있는 엘리먼트와 임의의 애트리뷰트들로 이루어져 있고, P_y는 E_y와 임의의 애트리뷰트들로 이루어져 있다고 할 경우에 XML FD인 $F = (Q, [P_{x1}, P_{x2}, ..., P_{xn} \rightarrow P_y])$는 평평하다고 부른다.

잘 구조화된 FD 와 평평한 FD는 릴레이션에서 애매모호하고 막연한 것들을 제거하기 때문에 FD의 이러한 두개의 형식을 고려한다.

3. XML 스키마를 릴레이션으로 매핑

XML 스키마를 릴레이션으로 매핑하는 변환 절차는 그림 1과 같으며, XML 스키마에서 릴레이션 스키마로 변환하는 전체 과정을 나타낸 것이다.



(그림 1) 릴레이션 스키마 변환 흐름도

관계 데이터에서처럼, XML 데이터의 의미론은 XML FD에 의해 표현된다. 여기서는 FD를 반영하여 XML 스키마에서 릴레이션 스키마로 매핑을 하기위한 절차를 그림 1에 따라 상세하게 기술한다.

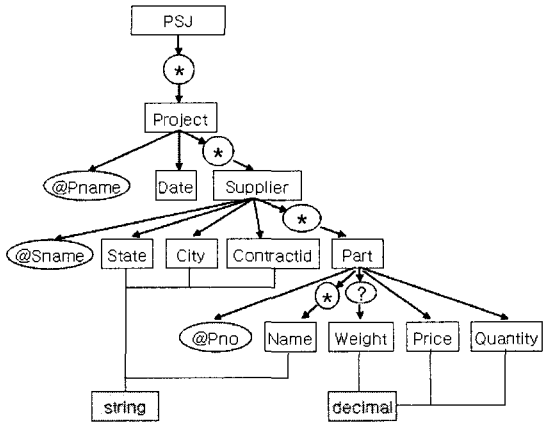
(1) 논문 [3]에 있는 기법을 사용하여 XML 스키마를 가지고 "XML 스키마 그래프"를 그린다.
(2) 각 (Q, [P_{x1}, P_{x2}, ..., P_{xn} → P_y])에 대하여, P_{x1}, P_{x2}, ..., P_{xn}, P_y로부터 축출된 애트리뷰트들의 릴레이션을 생성하고, 키로서 P_{x1}, P_{x2}, ..., P_{xn}에 대응되는 애트리뷰트들을 상술한다. 그다음에 루트로부터 P_y까지의 경로 중에 마지막 에지를 마크한다.
(3) 진입 차수(in-degree)가 0인 엘리먼트 노드는 개별적인 릴레이션을 생성한다.
(4) 순환(recursion) 형태인 사이클이 형성되어 있는 경우에는 개별적인 릴레이션을 생성한다.
(5) 주어진 데이터 형식에 포함되어 있는 엘리먼트의 출현 지시자인 maxOccurs의 값이 2 이상일 때 독립적인 릴레이션을 생성한다.
(6) 진입 차수가 2 이상인 엘리먼트 노드는 독립적인 릴레이션을 생성하고, 진입차수가 1인 노드들은 인라인한다.
(7) 복잡 형식들의 유도 관계에서 확장에 의한 유도인 경우에 상속으로 연결된 최종 노드인 조상 노드는 개별적인 릴레이션을 생성한다.
(8) 자식 릴레이션에서 각각의 부모-자식 관계가 부모 id에 의해 기록된 것을 보증한다.

릴레이션 생성 기준에 따라 (1)에 의해 생성된 XML 스키마 그래프를 순회하면서 (2)번부터 (8)번까지의 방법으로 릴레이션 스키마를 생성한다. [3]에서 제시한 방법과 구별된 이 기법은 (2)에서 제약조건들을 유지한다. 제시한 기법은 각각의 FD에 대한 릴레이션 작성을 통하여 종속보존 유지를 하며, XML 문서에 있는 애트리뷰트들이나 텍스트 문자 노드들로부터 값을 추출하는데 몇몇 릴레이션에 있는 애트리뷰트는 늘 존재하며, XML 문서의 내용은 유지된다. 그리고 (2)에서는 죠인 후에 얻은 큰 릴레이션과, 키와 외래키를 사용하여 이 단계에서 생성한 릴레이션 사이에서의 관계성을 기록한다. 즉 모든 문서는 손실 없게 조인 연산문을 사용하여 재구축 할 수가 있다. (3)번의 경우는 루트 엘리먼트에 해당하는 경우로 어떤 다른 노드로부터도 도달하지 않기 때문이다. (7)번의 경우에 XML 스키마에서 데이터 형식의 상속은 유도 관계로서 표현한다. 특히 확장에 의한 유도에는 기존의 내용 모델인 텍스트 내용, 엘리먼트 내용, 혼합 내용에 단순 형식, 복잡 형식, 자식 엘리먼트, 속성등이 추가되어 새로운 복잡 형식을 만든다. 이러한 경우에는 관계형 스키마 생성 시에 부모 복잡 형식에 릴레이션이나 칼럼이 추가된 상태이므로, 상속받은 곳에서 다시 추가하게 되면 중복이 발생한다. 그러므로, 유도 관계로 연결된 최종 단말 노드인 조상 노드는 새로운 릴레이션을 생성한다. (8)에서 XML의 구조는 #id 와 ParentID에 의해 저장된 것을 보증한다.

```
<xs:element name="PSJ">
<xs:complexType>
<xs:sequence>
<xs:elementref="Project" minOccurs="0"
maxOccurs="unbounded" />
</xs:sequence>
</xs:complexType>
</xs:element>
<xs:element name="Project">
<xs:complexType>
```

```
<xs:sequence>
<xs:elementref="Date" />
<xs:elementref="Supplier" minOccurs="0"
maxOccurs="unbounded" />
</xs:sequence>
<xs:attributename="Pname" type="xs:string"
use="required" />
</xs:complexType>
</xs:element>
<xs:element name="Supplier">
<xs:complexType>
<xs:sequence>
<xs:elementref="State" />
<xs:elementref="City" />
<xs:elementref="Contractid" />
<xs:elementref="Part" minOccurs="0"
maxOccurs="unbounded" />
</xs:sequence>
<xs:attributename="Sname" type="xs:string"
use="required" />
</xs:complexType>
</xs:element>
</xs:schema>
<xs:element name="Part">
<xs:complexType>
<xs:sequence>
<xs:elementref="Name" minOccurs="0"
maxOccurs="unbounded" />
<xs:elementref="Weight" minOccurs="0" />
</xs:sequence>
<xs:attributename="Pno" type="xs:string"
use="required" />
</xs:complexType>
</xs:element>
<xs:element name="Date" type="date"/>
<xs:element name="State" type="string"/>
<xs:element name="City" type="string"/>
<xs:element name="Contractid" type="string"/>
<xs:element name="Weight" type="decimal"/>
<xs:element name="Price" type="decimal"/>
<xs:element name="Quantity" type="decimal"/>
```

(그림 2) PSJ XML 스키마



(그림 3) XML 스키마 그래프

그림 2는 PSJ XML 스키마이다. 그림 3은 그림 2의 PSJ XML 스키마로 [3]에서 제안한 기법을 이용하여 표현한 XML 스키마 그래프이다. 그림 3의 애트리뷰트들의 의미로부터 다음과 같은 FD들이 존재한다.

FD1 = (/PSJ, [Project.Pname,Supplier.Sname -> Contractid])

FD2 = (/PSJ/Project, [Supplier.Same, Part.Pno -> Price])

FD3 = (/PSJ, [Project.Pname, Supplier.Sname, Part.Pno -> Quantity])

FD4 = (/PSJ/Project/Supplier, [City -> State])

관계형 데이터의 정의에 의해서 상기의 FD들은 실제로 각각 다음과 같이 기술할 수가 있다.

(FD1') PSJ.Project.Pname,PSJ.Project.Supplier.Sname
-> PSJ.Project.Supplier.Contractid

(FD2') PSJ.Project.Supplier.Sname,PSJ.Project.Supplier.Part.Pno
-> PSJ.Project.Supplier.Part.Price

(FD3') PSJ.Project.Pname,PSJ.Project.Supplier.Sname,PJS.Project.Supplier.Part.Pno
-> PSJ.Project.Supplier.Part.Qquantity

(FD4') PSJ.Project.Supplier.City -> PSJ.Project.Supplier.State

그러므로, 만약 XML FD의 집합이 Σ에 의해 표시되면, 관계형 데이터에 있는 FD의 대응하는 집합은 Γ로 표시될 수 있다. 또한 정의 2로부터 FD1, FD2, FD3, FD4는 잘 구조화 되었다고 볼 수가 있다.

추가로, 키는 FD에 의해 기술될 수 있다. 예를 들어, 각각의 ID는 몇몇 노드의 키로서 주시해볼 필요가 있다. (/PSJ/Project, [Supplier.#id-> Contractid]), (/PSJ/Project, [Supplier.#id->City]) 등과 같이 ID의 애트리뷰트를 #id로 표시하는 FD를 정의할 수가 있다.

제안한 기법에서 두개의 결론을 내릴 수 있다.

명제 1.

매핑은 σ, XML 스키마 S를 따르는 XML 문서는 T, S에 대한 XML FD의 집합을 Σ로 주어지고, 관계형 데이터에 있는 FD의 대응하는 집합은 Γ로 주어지면, $\sigma(T) \models \Gamma$ iff $T \models \Sigma$ 이다.

(증명)

$\forall$ XML FD $\beta = (Q, [P_{x1}, P_{x2}, ..., P_{xn} \rightarrow P_y]) \in \Sigma$,

Y에 있는 각 애트리뷰트들은 $P_{x1}, P_{x2}, ..., P_{xn}$로부터 축출하고, m은 P_y로부터 축출하는, Γ에 있는 그것의 대응인 $\beta = Y \rightarrow m$이 존재한다.

만약 XML 문서 T가 β를 만족하면, 그때 m 에 대해서도 동의해야만 한다. 즉, $\sigma(T) \models \beta$이다. σ(T)에서, 만약 릴레이션에 있는 두개의 투플이 Y에 대하여 동의하는 단계 (2)에 따라서 같은 시간에 Y 와 m을 포함한 오직 하나의 릴레이션이 존재한다.

그러므로 만약에 $T \models \Sigma$이면 $\sigma(T) \models \Gamma$이다. 이에 반하여, 관계 스키마의 인스턴스 σ(T)를 주면, XML 문서는 σ의 이(reverse)에 의해 구축될 수 있다. 그러므로, 또한 같다.

명제 2.

매핑은 σ, XML 스키마 S를 따르는 XML 문서는 T, S에 대한 XML FD의 집합을 Σ로 주어지고, 관계형 데이터에 있는 FD의 대응하는 집합은 Γ로 주어지면, 만약 $T \models \Sigma$ 이면 σ(T)에 있는 각각의 릴레이션은 3NF이다.

(증명)

1. σ(T)에 있는 모든 릴레이션의 애트리뷰트들이 문서에 있는 텍스트 노드들이나 애트리뷰트들로부터 축출되므로, 모든 릴레이션의 애트리뷰트들은 원자(atomic)이다. 그러므로, 모든 릴레이션은 1NF이다.
2. XML FD들은 집합 Σ에 모두 있기 때문에 의미들은 Σ에 있다. 관계형 데이터에 있는 모든 FD는 Σ의 대응 Γ이다. 또한 각각의 릴레이션에서 키가 아닌 애트리뷰트가 릴레이션의 기본키를 함수적으로 종속하는 것을 추정할 수 있다. 즉, 모든 릴레이션은 2NF이다.

   매핑의 절차에 따라서 릴레이션은 각각의 FD에 대하여 생성된다. 그러므로, 단계 (2)에서 생성된 릴레이션들은 3NF이다.
3. 다른 단계에서 생성된 릴레이션들은 기본키가 몇 개의 애트리뷰트인 폼과 기본 키에 종속되지 않은 애트리뷰트들을 포함하고 있는 FD들을 가지고 있다. 즉, 이러한 릴레이션들은 3NF이다. 그러므로, σ(T)에 있는 모든 릴레이션은 3NF이다.

4. 사례 연구

XML 문서는 그림 2에서 주어진 XML 스키마에 따르는 것을 가정한다. 따라서 그림 2에서 정의한 PSJ XML 스키마를 다음과 같이 매핑한다.

(1) 그림 3에서 보인 XML 스키마 그래프를 그린다.

(2) FD1에 대하여, 릴레이션 FD1(PSJ.Project.Pname,PSJ.Supplier.Sname,PSJ.Project.Supplier.Contractid)가 생성된다. FD2에 대하여, 릴레이션 FD2(PSJ.Project.Supplier.Sname,PSJ.Project.Supplier.Part.Pno, PSJ.Project.Supplier. Part.Price)가 생성된다. FD3에 대하여, 릴레이션 FD3 (PSJ.Project.Pname,PSJ.Project.Supplier.Sname,PSJ.Project.Supplier.Part.Pno, PSJ.Project.Supplier.Part.Quantity), FD4에 대하여, 릴레이션 FD4(PSJ.Project.Supplier.City, PSJ.Project.Supplier.State)가 생성된다. 그러면 Supplier로부터 Contractid까지의 에지, Part로부터 Price까지의 에지, Part로부터 Quantity까지의 에지, Supplier로부터 State까지의 에지가 표시된다.

(3) root를 위한 릴레이션 PSJ (#id)를 생성한다. 그림 3의 XML 스키마 그래프에서 PSJ가 이 경우에 해당하여 새로운 릴레이션을 생성하게 된다.

(4) XML 스키마 그래프에 순환이 없기 때문에 단계 4에서는 아무것도 하지 않는다.

(5) PSJ로부터 Project까지의 에지에 대하여, 릴레이션 Project (#id)를 생성한다; Project로부터 Supplier까지의 에지에 대하여, 릴레이션 Supplier (#id)를 생성한다; Supplier로부터 Part까지의 에지에 대하여, 릴레이션 Part (#id)를 생성한다; Part로부터 Name까지의 에지에 대하여, 릴레이션 Name(#id, Name)을 생성한다; 그러면 4개의 에지가 표시된다.

(6) 릴레이션 Project 안에서 Project.Pname과 Project.Date를 인라인 한다. 릴레이션 Supplier 안에서 Supplier.Sname과 Supplier.City를 인라인 한다. 릴레이션 Part 안에서 Part.Pno, Part.Weight를 인라인 한다. 따라서 릴레이션들은 Project(#id, Project.Pname, Project.Date), Supplier(#id, Supplier.Sname, Supplier.City), Part(#id, Part.Pno, Part.Weight) 이다. 이때 관계된 에지는 표시된다.

(7) 확장에 의한 유도인 경우에 상속으로 연결된 최종 노드인 조상 노드가 없기 때문에 단계 7에서는 아무것도 하지 않는다.

(8) 부모-자식 관계성을 보증한다. 릴레이션 Project, Supplier, Part, Name 안에서 ParentID를 추가한다. 그리고 부모 코드는 여기서 필요하지 않다.

최종적으로 완성된 릴레이션들은 그림 4와 같다. 그림 4에 있는 릴레이션들은 내용과 구조의 손실이 없는 FD를 만족시킨다. 또한 제3정규형(3NF)도 만족시킨다.

PSJ(#id);

Project(#id, ParentID, Project.Pname, Project.Date);

Supplier(#id, ParentID, Supplier.Sname, Supplier.City);

Part(#id, ParentID, Part.Pno, Part.Weight);

Name(#id, ParentID, Nmae);

FD1(PSJ.Project.Pname,PSJ.Project.Supplier.Sname, PSJ.Project.Supplier.Contractid);

FD2(PSJ.Project.Supplier.Sname,PSJ.Project.Supplier.Part.Pno, PSJ.Project.Supplier.Part.Price);

FD3(PSJ.Project.Pname,PSJ.Project.Supplier.Sname,PSJ.Project.Supplier.Part.Pno, PSJ.Project.Supplier.Part.Qquantity);

FD4(PSJ.Project.Supplier.City, PSJ.Project.Supplier.State)

(그림 4) 최종적인 릴레이션

5. 결론

함수적 종속성 제약조건은 전통적인 데이터베이스 이론에서 절대 빠트릴 수 없는 부분을 차지하고 있기 때문에, 어떻게 이 개념을 XML에 적

용할 수 있는지가 중요하다. XML 키에서의 대부분의 연구들은 XML 구조 내에서 키 종속성을 정의하였다. 그러나 XML 구조 사이에서의 관계성은 정의하지 않았다. [11]에서는 XML 데이터베이스에 있는 함수적 종속성을 위한 의미들과 표현을 제안함으로서 이러한 간격을 채웠다.

XML FD는 XML을 마치 관계형 데이터와 같이 정의한다. 관계형 데이터베이스에서 FD의 임계작업(critical task)은 중복을 감소시킬 뿐만 아니라 갱신 이상을 막는다. 계층적 데이터에서 중복은 피할 수 없는 현상이다. XML FD 정의에 의해 중복 정보는 상술할 수가 있으며, 생성된 릴레이션은 가능한 이러한 문제를 피하는 것이 필요하다.

이 논문에서 매핑을 처리하는 데 하이브레인 인라이닝 기법[15]에 기초한 방법인 [3]에서 제시한 기법을 제공한다. 내용과 구조와 마찬가지로, 제약조건을 제공하는 릴레이션에 XML을 매핑한다. 앞으로의 연구과제는 XML 스키마[2]의 특징인 복잡한 유도 관계와 다형성을 고려한 XML FD가 필요하다. 이러한 FD가 주어지면 매핑 기법은 더욱 유연하고 적합하게 적용될 수 있을 것이다.

참 고 문 헌

[1] D. Fallside, "XML Schema Part 0:Primer," http://www.w3.org/TR/xmlschema-0/, 2000.

[2] Thompson H, et al., "XML schema," Recommendation, http://www.w3.org/TR/xmlschema1, Oct. 2004.

[3] 조정길, "XML 문서의 관계형 스키마 변환 기법", 충북대학교 박사학위 논문, 2003.

[4] 김정섭, "XML Schema로부터 관계형 스키마의 자동 생성", 한국과학기술원 석사학위 논문, 2002.

[5] E. F. Codd, "A Relational Model of Data for Large Shared Data Banks," j-CACM, 13(6): 377-387, June 1970.

[6] Jennifer Widom, "Data Management for XML:Research Directions," IEEE Data Engineering Bulletin, 22(3):44-52, 1999.

[7] Peter Bunemana, Susan Davidson, Wenfei Fan, CarMem Hara, and Wang-Chiew Tan, "Keys for XML," In Proceedings of the WWW'10, Hong Kong, China, 2001.

[8] Buneman P, Fan W, Simeon J, Weinstein S, "Constraints for semistructured data and XML," SIGMOD Record(ACM Special Interest Group on Management of Data), 30(1):47~54, 2001.

[9] Fan W, Schwenzer P, Wu K, "Keys with upward wildcards for XML," Mayr HC, et al., eds. Database and Expert Systems Applications, 12th International Conference. Lecture Notess in Computer Science 2113, 657~667, Munich, Germany, Springer-Verlag, 2001.

[10] Alin Deutsch, Val Tannen, "XML queries and constraints, containment and reformulation", Theoretical Science 336(2005) 57~87, 2005.

[11] Lee ML, Ling TW, Low WL, "Designing functional dependencies for XML," Jensen CS, et al., eds. Advences in Database Technology--EDBT 2002, 8th International Conference on Extending Database Technology, Lecture Notes in Computer Science 2287, 124~141, Prague, Czech Republic:Springer-Verlag, 2002.

[12] Chen Y, Davidson S, Zheng Y, "Constraint preserving XML storage in relations," Technical Report, MS-CIS-02-04, University of Pennsylvania, 2002.

[13] Davidson S, Fan W, Hara C, "Propagating XML keys to relations," Technical Report, MS-CIS-01-33, University of Pennsylvania, 2001.

[14] Lee D, Chu WW, "Constraints-Preserving

transformatin from XML document type definition to relational schema," Laender AHF, et al., eds. Conceptual Modeling--ER 2000, 19th International Conference on Conceptual Modeling. Lecture Notes in Computer Science 1920, 323~338, Salt Lake City, Utah:Springer-Verlag, 2000.

[15] J. Shanmugasundaram, H. Gang, K. Tufte, C. Zhang, D. DeWitt, and J. Naughton, "Relational Databases for Querying XML Documents: Limitations and Opportunities," Proceedings of the 25th VLDB Conference, pp. 302-314, Edinburgh, Scotland, 1999.

[16] Clark J, DeRose S, "XML path language(XPath)," W3C Working Draft, http://www.w3.org/TR/xpath. 2001.

◐ 저 자 소 개 ◑

조 정 길(Jung-Gil Cho)

1986년 숭실대학교 전자계산학과 졸업(공학사)
1993년 숭실대학교 정보과학대학원 졸업(이학석사)
2003년 충북대학교 대학원 전자계산학과 졸업(이학박사)
1987 ~ 1992 DHL코리아 정보시스템실 근무
1992 ~ 1993 에스콰이어 정보시스템부 근무
1996 ~ 2004 남서울대학교 컴퓨터학과 겸임교수
2004 ~ 현재 성결대학교 공과대학 컴퓨터공학부 교수
관심분야 : XML 문서관리, 정보검색, 시맨틱 웹, 객체지향 재사용/테스트
E-mail : jkcho@sungkyul.edu