제18회 한국정보처리학회 추계학술발표대회 논문집 제9권 제2호 (2002.11)

XML Database를 위한 3단계 스키마 설계 방법론

최문영* 주경수**,
순천향대학교 공과대학 컴퓨터학부

griffin@hyejeon.ac.kr* gsoojoo@sch.ac.kr**,

3 Phases Schema Design Methodology for XML Database

Choi Mun-Young*, Joo Kyung-Soo**
Dept. of Computer Science and Engineering, College of Engineering
Soonchunhyang Univ. Asan 336-745. Korea

요 약

XML을 이용하면 이기종 컴퓨팅 환경으로 구성되어 있는 웹 상에서 정보를 공유할 수 있었고, 이제 XML은 정보가 아닌 프로세스를 공유할 수 있는 아주 단순하면서도 유연한 방법을 제공해 주고 있다. 이러한 XML 기술을 기반으로 하는 웹 서비스와 ebXML을 이용하면 어떤 어플리케이션은 물론 어떤 비즈니스 프로세스 역시 웹 상에서 통합될 수 있다.

기업용 애플리케이션에 XML을 이용하는 일이 점점 늘어남에 따라 많은 조직들이 XML 문서를 저장하고 관리하는 문제에 직면하고 있다. 문제는 이미 많은 기업들이 이들 XML 문서를 저장하는데 기존에 사용하던 관계형 데이터베이스를 계속 사용해도 되는 것으로 생각하고 있다는 것이다. 이것은 XML 데이터를 다루는데 많은 문제를 야기한다. 관계형 데이터베이스는 XML 같은 확장성 데이터를 다루도록 설계되지 않았다는 태성적 한계가 있기 때문이다. 그러므로 본 논문에서는 XML Database 스키마 설계방법론을 이용하여 이러한 문제점을 해결하려한다.

1. 서론

새로운 XML database의 개발에 있어서 어떻게 DBMS에서 데이터를 획득해야 되는지 이해하는 것은 중요하다. 사용자와 DBMS사이에 가장 많이 영향을 미치는 상호작용은 데이터베이스에 대한 스키마를 디자인하는 것에 있을 것이다. 스키마는 데이터베이스의 자료가 보이는 것처럼 기술된다. 스키마는 데이터베이스에 대한 청사진이며 데이터베이스의 내용을 기술한다.

스키마 디자인은 데이터베이스에 접근하는 애플리케이션을 개발하는 것을 이해하는 또 다른 이유이다. 데이터베이스 스키마를 위한 훌륭한 디자인도 데이터베이스를 사용하는데 필요한 설명이다.

이 논문에서는 데이터베이스를 디자인하는 방법을 표현한다. 데이터베이스 디자인 처리는 개념적 디자인, 논리적 디자인, 물리적 디자인의 세 가지 단계로 이루어져 있다.

2. 데이터베이스 디자인

데이터베이스 디자인은 멀티단계 처리이다 그리고 각 단계에서는 서로 다른 영역을 요구한다. 어떠한 영역처럼 기술, 구조, 직물 디자인, 그래픽 아트, 또는 요구사항을 이해하고 계획하는 것을 요구한다.

데이터베이스는 기능적이지 않은 요구사항, 규칙, 정치적인 상태, 트리, 큰 바위, 언덕, 실재적인 디자인에 제출되어야 하는 색채 배합 설계, 제공자 상태, 그리고 기타 등등을 나타낸다. 그러나, 데이터베이스 개발에서 가장 중요한 상태는 도메인이다.

어떠한 디자인에서 선택된 모델링 언어는 중요한 형식과 디자인의 미묘한 차이의 정도에 따라 모델링 언어의 능력에 크게 영향을 미칠 수 있다. 엔티티-관계형 데이터 모델링 언어는 관계형 데이터베이스에 필요한 만족을 위해 사무적인 응용에서 자주 선택된다. 그러나 그것은 개념적 모델을 지나치게 제한할 수 있고 구현에 대하여 적합하지 않다.

논리적 설계 처리에서 데이터 모델 중의 수학적으로 정밀한 논리는 프레임워크로 선별되고 암착된다. 비록 도메인의 몇몇 미묘한 차이의 정도가 자주 상실되더라도, 결과적으로 볼 때 데이터베이스 모델링은 능률적이고 유용하다.

물리적 디자인은 수학적으로 문제가 없는 데이터 모델을 제작하는 것이다. OS와 DBMS 대다수 교환, 항목, 그리고 최적화는 이 단계에서 해결된다. 물리적인 디자인은 전체 설계에 영향을 미치지 않고 논리 데이터 모델에 포함된다. 그러나, 이 항목은 데이터베이스를 작용하는데 중요하다. 개념적 설계는 새로운 건물을 설계하기 위하여 흥미로운 대화를 여는 것과 유사하다. 논리적 설계는 계획 약간 청사진을 개발하고 다이어그램을 설계하는 것과 유사하다, 그리고 물리적 설계 단계는 돌담 위로 돌을 얻는 방법과 유사하다.

디자인 3 단계의 언어로 XML을 사용한다.

① 개념적 모델링 언어로 XML을 사용하고 관계형 또는 객체지향 데이터베이스를 생성할 수 있다.
② eXcelon과 같은 XML DBMS에 사용되는 논리 데이터 모델링 언어로 XML를 사용한다.
③ 데이터베이스를 생성하기 위하여 XML 스키마를 사용하는 것과 같이 물리적 데이터 모델링 언어의 스키마를 지정할 때 XML을 사용한다.

3. 개념적 모델링

개념적 모델링은 어떠한 DBMS와도 독립적으로 기술되는 처리이다. 개념적 모델링의 목표는 도메인이 구현된 구조를 그리는 방법에 대하여 도메인의 중요한 표시 양상을 캡쳐하는 것이다.

논리 모델링을 위해 사용되는 개념적 모델링을 위해 더욱 쉬운 모델링 언어는 유용하다. XML을 위해 개념적 모델링 언어는 XML의 계층보다 구조적인 탄력성을 많이 가지고 있어야 하고 트리의 컬렉션으로 XML 문서의 수학적 추상화를 보존해야 한다. 개념적 언어는 논리 XML 데이터 모델에 대한 언어와 다르지 않다.

그래프 개념적 모델링은 개념과 도메인의 관계성을 캡쳐하는 것으로 사용된다. 개념적 모델링 처리의 목표는 도메인 전문가로부터 도메인 정보를 추출하는 데이터베이스 개발을 위해 있다. 개념과 관계성은 그래프와 반복을 세분하여 캡쳐 할 때까지, 도메인의 본질을 충분히 캡쳐한다.

그림 1은 개념과 도메인의 특성을 기술하기 위하여 라벨이 붙

인다.

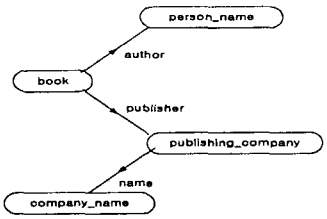

그림 1 간단한 개념적 스키마

개념과 관계성은 그래프의 노드로 캡처되고 문자로 라벨이 붙는다. 구체적인 객체, 잘 알려진 개념, 그리고 분명히 정의된 관계는 그래프의 노드로 모두 캡처된다. 개념은 그래프의 모서리로 특성을 기술하는 것에 의해서 나눠진다. 예를 들면, 이름, 크기, 색, 형, 버전, 상태, 객체의 구조는 모서리로 기술된 특성이다.

그래프 개념적 모델은 개념과 도메인의 관계성이 그래프라는 형식적인 시스템이다. 도메인을 그래프로 표현하는 방법은 다음과 같다.

① 그래프는 개념, 레벨 모서리, 그리고 카디널리티의 컬렉션에 있다.
② 그래프와 모델의 단순한 개념과 도메인의 관계의 노드는 개념에 있다.
③ 두 개의 개념과 모델, 개념의 특성 그리고 도메인의 2진 관계는 라벨 모서리에 연결한다.
④ 카디널리티 한 쌍은 완전한 정수 모서리로 연상된다. 카디널리티는 정수대신에 조건 "Many(다수)"로 표현한다. 다수 카디널리티는 0 또는 더 많은 출현을 언급하는 것이다. 그러나 그 카디널리티 "one-or-more"는 "+"에 의해서 나타내어진다. 만약 카디널리티가 없으면, 그것은 "1"로 가정한다.

Rolodex에 저장된 정보의 예제는 그림 2에 보여진다. 사람이름, 전자우편, 전화번호, 회사, 그리고 주소를 가지고 있다. 그림 2의 주소 개념은 다중 주소선, 도시, 국가, 그리고 우편번호를 가지고 있다. 다중 주소 선은 "*" 다음에 라인 모서리의 라인 개념이 나타난다. 주소 라인의 다수가 2 라인에 제한되도, 라인은 "2"에 의해서 "*"로 표시될 것이다.

그림 3는 네 가지 공통 카디널리티에 대한 사람과 이름사이의 카디널리티 상태를 나타낸다.

a. 1-to-1 카디널리티 - 각각의 사람은 하나의 이름을 가지고 있다.
b. 1-to-many 카디널리티 - 각각의 사람은 많은 이름을 가지고 있을 수 있다.
c. many-to-1 카디널리티 - 많은 사람은 하나의 이름을 공유할 수 있다.
d. many-to-many 카디널리티 - 많은 사람은 많은 이름들을 공유할 수 있다.

"1" 카디널리티는 다이어그램에서 표시를 하지 않는다. 부서에 종업원을 기술한 그래프 개념 스키마는 그림 4에서 보여진다. 관리자는 또한 많은 종업원 중의 종업원이다. 종업원 개념이 관리자 특성을 기억하도록 시킨다. 그래서 화살표는 종업원에서 관리자까지 관리된다. 많은 종업원은 부서에 있고 부서는 수, 이름, 그리고 로케이션을 가지고 있다.

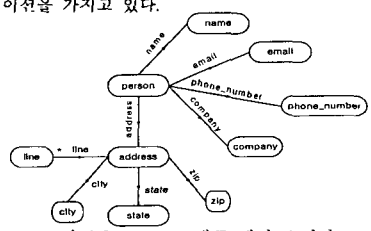

그림 2 Rolodex 그래프 개념 스키마

개념적 모델링의 장점은 그래프를 사용하는 것이다. 그래프는 시각, 비선형, 탄력적인 작동에 관하여 융통성이 있다. 조직화, 개념추가, 특성추가, 개념삭제, 그리고 특성을 삭제한다. 그래프는 스키마를 개발하는데 변경하기에 간단하고 논리 스키마로 변환하기에 간단하다.

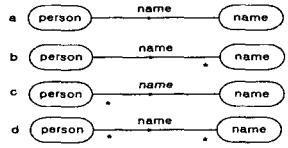

그림 3 그래프 카디널리티 예제

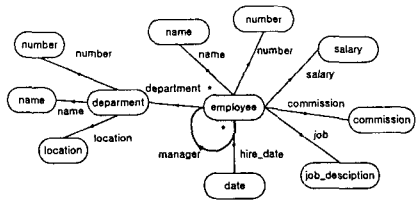

그림 4 Employee 그래프 개념 스키마

그래프 개념적 모델의 복잡한 구성을 쉽게 다루기 위하여 XML의 구조를 도메인 전문가가 이해할 수 있게 하고, 데이터베이스 개발에 사용할 수 있게 하고, 그리고 완전하게 XML 데이터베이스에 대한 자료를 캡처하는 것으로 사용한다. 도메인 전문가는 비전문가가 추정하는 것보다 도메인에 정보를 캡처하기 위하여 약간 많이 표현되는 언어를 필요로 한다.

3.1 XML 개념적 모델

이 장의 첫 번째 점이 XML 데이터베이스의 스키마를 개발하기 위하여 개념적 모델링을 사용하는 방법을 기술해도, 어떻게 XML이 단순한 구조로 데이터베이스를 개발하기 위한 개념적 모델링 언어로 사용될 수 있는지를 조사하는 것이다. XML을 개념적 모델링 언어로 사용하는 것은 이 장에 기술된다.

XML은 상속 없이 계층을 구조화된 도메인의 집합한 정보를 캡처하기 위한 개념적 모델링 언어로 사용될 수 있다. 예제 1 또는 예제 2는 유용한 모델로 다음과 같이 적당한 규정으로 개발될 수 있다 :

- XML 개념 스키마는 XML 엘리먼트의 컬렉션으로 이루어져 있다.
- 빈 속성 값은 제한이 없는 속성 값을 표현한다.

Employee 관계성을 기술한 XML 개념 스키마는 예제 3에서 보여진다. 그 XML 개념 스키마는 다음과 같이 관계형 스키마로 번역될 수 있다.

예제 1 : XML에 표현된 생물학 분류법의 단순한 개념 스키마

```
<kingdom name=" ">
  <phylum name=" ">
    <class name=" ">
      <order name=" ">
        <family name=" ">
          <genus name=" ">
            <species name=" ">
            </species>
          </genus>
        </family>
      </order>
    </class>
  </phylum>
</kingdom>
```

예제 2 : XML에 표현된 책 부품의 단순한 개념 스키마

```
<book title=" " author=" ">
  <tableofContents/>
  <chapter title=" ">
    <section>
      <subsection/>
    </section>
  </chapter>
  <index/>
</book>
```

a. 각각의 엘리먼트는 테이블이 된다(관계).
b. 각각의 XML 속성은 칼럼이 된다(관계 속성).
c. 엘리먼트 각각의 보조 엘리먼트는 보조 엘리먼트에 의해서 지정된 테이블에 외래 키의 중요한 상태를 가진 테이블의 칼럼으로 된다.
d. 모두 같은 이름의 엘리먼트는 같은 관계를 나타낸다.

예제 3 : XML에 표현된 employee 개념 스키마

```
<employee number="" name=" " job=" " hiredate=" " commission=" "
salary=" ">
  <department number=" " name=" " location=" "/>
</employee>
```

예제 3은 다음 관계로 번역될 수 있다.

Employee(*number*, *name*, jobm hiredate, commission, salary, department)
Department(number, name, location)

4. 논리적 모델링

논리적 설계는 DBMS의 데이터 모델에 대한 스키마에 개념 스키마를 그리는 처리이다. 논리적 설계는 데이터베이스에 일반적으로 강조된 최대량 표시 양상인 것으로 처리를 계획한다. 도메

인이 데이터베이스 디자이너에 의해서 이해될 때, DBMS의 데이터 모델은 유일한 모델링 언어로 자주 사용된다. 그러나, 어떤 개념적 모델이 사용될 때, 논리적 설계 단계 부분의 방법이 논리 스키마에 개념 스키마를 번역하는 것을 기본으로 도메인을 정확하게 캡쳐하고 논리 데이터 모델의 구조물을 효과적으로 사용한다. 논리 데이터 모델은 개념적인 모델보다 기능성과 이상에 더 많이 제한적이다.

XML 논리 스키마 형식의 작성을 그래프 개념 스키마로 표현하기 전에, 엔티티-관계 다이어그램, 관계 스키마, 그리고 객체 모델에 그래프 개념 스키마의 번역을 기술한다.

4.4 XML 논리 스키마

일반적인 스키마 언어는 논리적 설계 단계에서 XML을 모델링할 수 없다. Document Type Definition(DTD)과 XML Schema Definition Language(XSDL)를 위해 언어는 물리적 설계에서 더욱 적합하다. 그러나, 문서형식 또는 문서 스키마 정의가 다음과 같이 설명된 처리로부터 생성될 수 있다. 논리 XML 스키마의 생성은 이전 장의 처리 모델을 만드는 객체로부터 다이어그램을 사용하기를 원할 수 있다. 개념적 다이어그램, 카디널리티 정보, 도메인 객체 다이어그램, 그리고 집단화 트리 등 XML 논리 스키마를 표현하는 것의 세 가지 선택은 나중에 소개할 것이다.

XML 논리 스키마는 그래프 개념 스키마에서 다음 단계를 통해 생성된다.

- 각각의 방식은 같은 이름으로 엘리먼트 형이 된다.
- 모서리 없는 개념을 특정하게 나타낼 때, 속성 또는 보조 엘리먼트 타입이다. 만약 특성이 도메인 존재에 대한 개념을 결정하고 카디널리티 1:1 또는 Many:1을 가지고 있으면 속성이 된고 그렇지 않으면 보조 엘리먼트가 된다. 예를 들면 Rolodex 도메인(그림 2), "Name", "Email", 그리고 " Phone Number" 속성이 되고 Company는 보조 엘리먼트가 된다.

예제 4 : 빈 XML로 표현된 혼성체 필터 예제

```
<experiment id=" " creation_date=" " comment=" ">
  <investigator>
    <person name=" " priviege=" " user_number=" ">
      <laboratory/>
    </person>
  </investigator>
  <probe id=" ">
  <filter id=" " creation_date=" ">
    <spot id=" ">
      <clone id=" "/>
    </spot>
  </filter>
  <positive_hybridization>
    <hybridization>
      <probe/>
      <target>
        <clone/>
      </target>
    </hybridization>
  </positive_hybridization>
</experiment>
```

예제 5 : XML 엘리먼트를 사용하는 표현의 혼성체 필터 예제

```
<?xml version="1.0"?>
<schema>
  <element name="experiment">
    <element name="investigator">
      <element name="person">
        <element name="laboratory"/>
        <attribute name="id"/>
        <attribute name="creation_date" type="date"/>
        <attribute name="comment" minOccurs="0"/>
      </element>
    </element>
    <element name="probe">
      <attribute name="id" type=string"/>
    </element>
    <element name="filter">
      <element name=""spot: minOccurs="1" maxOccurs="96">
        <element name="clone">
          <attribute name="id"/>
        </element>
        <attribute name="id"/>
      </element>
      <attribute name="id"/>
      <attribute name="creation_date" type="date"/>
    </element>
    <element name="positive_hybridization">
      <element name=""hybridization" maxOccurs="unbounded">
        <element name=""probe"/>
        <element name="target">
          <element name="clone"/>
        </element>
      </element>
    </element>
  </element>
</schema>
```

필터 혼성체 예제는 예제 4에 보여진 것처럼 XML을 사용한 스키마 정의로 번역될 수 있다. 빈 엘리먼트와 속성은 제한이 없는 값을 표시한다. 이 스키마가 형식을 포함한 문서의 컬렉션을 기술하기 위하여 "empty" XML 문서를 사용하는 것을 기억한다. 많은 정보를 제공하는 스키마 예제 5는 XSDL의 문법에 기초를 두었다. 예제 6은 XSL이 템플릿에 처리 명령을 많이 추가하지만 형 정보와 빈 문서를 증가시키기 위하여 XML Namespaces를 사용한 두 가지 예제의 혼합 접근이다.

물리적 모델링에서 속성과 보조 엘리먼트에 대한 선택의 일부는 변할지도 모른다. 예를 들면, 값이 아니라 문자를 담고 있는 속성은 유효하다. 속성 값이 보조 엘리먼트로 될 수 있고 CDATA 섹션에 요약 돼있기 때문에 문자를 탈퇴하는 것으로 에러를 피한다. 부가적으로, 보조 엘리먼트는 독립적으로 존재하는 것을 요구하지 않거나 간단한 것은 속성이 된다.

예제 6 : 빈 XML을 표현한 필터 혼성체 예제

```
<?xml version="1.0"?>
<schema xmlns:xs="http://www.xweave.com/xmlns/xmldb/xsl">
  <experiment>
    <investigator>
      <person>
        <xs:attribute name="id"/>
        <xs:attribute name="creation_date" type="date"/>
        <xs:attribute name="comment" minOccurs="0"/>
        <laboratory/>
      </person>
    <probe>
      <xs:attribute name="id"/>
    </probe>
    <filter>
      <xs:attribute name="id"/>
      <xs:attribute name="creation_date" type="date"/>
      <spot xs:minOccurs="1" xs:maxOccurs="96">
        <xs:attribute name="id"/>
        <clone>
          <xs:attribute name="id"/>
        </clone>
      </spot>
    </filter>
    <positive_hybridization>
      <hybridization xs:maxOccurs="unbounded">
        <probe/>
        <target>
          <clone/>
        </target>
      </hybridization>
    </positive_hybridization>
  </experiment>
</schema>
```

5. 물리적 모델링

물리적 설계는 사용되는 DBMS로 연상되는 구현 언어의 구조물로 일반적인 논리 데이터 모델의 스키마를 능률적으로 변환시키는 처리이다. 예를 들면, 관계 스키마는 속성 NAME(문자열) 그리고 EMPLOYEE_NUMBER와(수) EMPLOYEE 관계를 표시한다. 대응하는 물리적인 스키마는 EMPLOYEE 테이블이 2 칼럼을 가지고 있는 것을 표현한다.

물리적 모델링은 디자인의 실용적인 표시 양상을 보존한다. 다양한 인수는 자료의 크기, 기밀 보호, 그릴듯한 접근 패턴, 자화율의 변화, 가능성의 재사용 그리고 필수 접근 속도를 포함하여 물리적 설계에게 영향을 미친다. 부가적인 물리적 모델링 데이터베이스에 발생되는 문제에 추가하여 데이터베이스 그리고 데이터 교환을 위해서 사용되는 XML을 위해 물리적인 스키마 생성이 추가적인 문제로 발생한다.

5.1 XML 물리적 스키마

물리적 설계를 만드는데 필요한 하나의 선택을 사용하기 위하여 XML 문서의 부분과 어떤 메커니즘으로서 스키마를 명백히 규정하는지를 결정한다. XML DBMS에서 스키마를 규정할 때, 그 결정은 선택된 DBMS에 의해서 결정된다.

DTD 표기법은 XML 1.0 표준이고 스키마를 규정하는 것으로 사용된다. DTD는 데이터베이스에 대하여 일반적인 형 상태의 단계를 제공할 수 없다. 이 이유 때문에, XML Schema는 개발되었다.

XML Schema는 W3C 권장이다. 문서에서 일어나는 구조와 데이터 값을 지정하기 위하여 평균을 제공한다. 공교롭게도, XML Schema는 가정으로 개발되었다. 이처럼, 스키마 정의 언어는 (XSDL) 특정한 스키마에 소프트웨어를 적당히 제한하기에 유용하다. 그러나 스키마 정의 언어는 스키마를 디자인하지 않는다.

스키마를 디자인하기 위하여 또 다른 메커니즘을 사용하여야 한다. 스키마를 디자인한 후에, 데이터베이스에 대한 XSDL에 물리적인 스키마 정의를 생성할 수 있다. DTD를 사용하여 예제 7에 표현하고 XSDL 스키마는 예제 8에서 표현했다. 스키마의 첫 번째 사용은 문서의 내용을 확인하는 것이다. 하나의 철학적 원리는 힘든 시스템 설계와 타당성 검사가 시스템 컴포넌트 사이에 수행되는 것을 요구한다.

예제 7 : 혼합체 필터 XML DTD 표현 예제

```
<!ELEMENT experiment (investigator, probe, filter, positive_hybridization*)>
<!ATTLIST experiment
   id CDATA #REQUIRED
   creation_date CDATA #REQUIRED
   comment CDATA #IMPLIED>
<!ELEMENT investigator (person)>
<!ELEMENT person (laboratory)>
<!ATTLIST person
   name CDATA #REQUIRED
   privilege CDATA #REQUIRED
   user_number CDATA #REQUIRED>
<!ELEMENT laboratory ANY>
<!ELEMENT probe EMPTY>
<!ATTLIST probe
   id CDATA #REQUIRED><!ELEMENT filter (spot*)>
<!ATTLIST filter
   id CDATA #REQUIRED
   creation_date CDATA #REQUIRED>
```

```
<!ELEMENT spot (clone)>
<!ATTLIST spot
   id CDATA #REQUIRED>
<!ELEMENT clone EMPTY>
<!ATTLIST clone
   id CDATA #REQUIRED>
<!ELEMENT positive_hybridization (hybridization)>
<!ELEMENT hybridization (probe, target)>
<!ELEMENT target (clone)>
```

예제 8 : 혼합체 필터 XML Schema 표현 예제

```
<schema>
<complexType name="cloneType">
   <attribute name="id" type="string"/>
</complexType>
<complexType name="probeType">
   <attribute name="id" type="string"/>
</complexType>
<element name="experiment">
<complexType>
   <sequence>
      <element name="investigator">
         <complexType>
            <sequence>
            <element name="person">
               <complexType>
                  <sequence>
                  <element name="laboratory">
                     <complexType>
                        <any/>
                     </complexType>
                  </element>
                  </sequence>
                  <attribute name="id" type="string"/>
<attribute name="creation_date" type="date"/>
      <attribute name="comment" type="string" minOccurs="0"/>
               </complexType>
            </element>
            </sequence>
         </complexType>
      </element>
      <element name="probe" type="probeType"/>
      <element name="filter">
         <complexType>
            <sequence>
            <element name="spot" minOccurs="1" maxOccurs="96">
               <complexType>
                  <sequence>
                     <element name="clone" type="cloneType">
                     <attribute name="id" type="string"/>
                  </sequence>
               </complexType>
            </element>
            </sequence>
            <attribute name="id" type="string"/>
            <attribute name="creation_date" type="date"/>
         </complexType>
      </element>
      <element name="positive_hybridization">
         <complexType>
            <sequence>
            <element name="hybridization" maxOccurs="unbounded">
               <complexType>
                  <sequence>
                  <element name="probe" type="probeType">
                  <element name="target">
                     <complexType>
                        <element name="clone" type="cloneType">
                     </complexType>
                  </element>
                  </sequence>
               </complexType>
            </element>
            </sequence>
         </complexType>
      </element>
   </sequence>
</complexType>
</element>
</schema>
```

6. 결론

인터넷은 새로운 정보 처리 방식을 요구한다. 기업과 기업 간의 비즈니스 네트워크의 관문을 개방하여 정보는 이제 날씨같은 것이 되어 가고 있다. 네트워크 안에서 이들은 스스로 표현하고, 취소하고, 변경하고, 제거한다. 컨텍스트와 데이터가 동시에 존재하고 이들을 실시간으로 변경하고 확장할 수 있는 XML의 출현은 정보를 보다 풍부하게 그리고 보다 동적으로 만들었다. 이런 인터넷의 새로운 데이터들을 처리하기 위해서 이 논문에서는 XML Database를 설계하는 개념적, 논리적, 물리적 방법을 제안했다.

XML 데이터베이스는 비즈니스 정보 시스템에의 적용능력을 가능하게 한다. 이들 XML 데이터베이스 시스템들은 비지니스 프로세스와 데이터베이스 계층을 모두 확장하는 XML의 핵심 요소를 지원한다. 선두 시스템들은 기업이 보다 쉽고 저렴한 비용으로 변화되는 비즈니스 요구를 수용할 수 있도록 하여 준다

참고문헌

[1] Extensible Markup Language(XML)1.0, http://www.w3.org/TR/1998/REC-xml-19980210
[2] [2] Liam Quin, Open Source XML Database Toolkit, WILEY, 2000
[3] 'XML and Databases', Ronald Bourret, Technical University of Darmstadt, September, 1999, HTTP://www.informatik.tudarmstadt.de/DVS1/staff/bourret/xml/XMLAndDatabases.htm'