# 인포믹스와 XML

## - 가져오기, 내보내기 및 변환하기 -

**데이터베이스,. DBMS, XML, 데이터 상호교환**

**폴 브라운(Paul Brown)**
**Chief Plumber**

인포믹스의 CTO(Chief Technology Office)에서 근무하는 폴 브라운(Paul Brown)은 인포믹스의 'Chief Plumber'이다. 그는 초기 객체 관계형 DBMS 프로토타입인 Postgres 를 구축한 UC 버클리(UC Berkeley) 팀의 일원이었으며 ORDBMS 기술을 최초로 상용화한 캘리포니아의 신생 업체인 일러스트라(Illustra)에 근무한 바 있다. 브라운은 "Object-Relational DBMS: Tracking the Next Great Wave" 및 "Developing Object-Relational Database Application"을 공동으로 집필하는 등 여러 연구 보고서를 집필한 바 있다.

## 개요

***데이터베이스 시스템, 인터넷***

웹은 모든 것을 변화시켰다. 웹은 정보 액세스, 정보 공유 능력 등을 확장시킴으로써 개인 생산성은 물론 전체 조직의 효율성까지 향상시킨다. 이러한 변화는 차츰 시들해지고 있으며 닷컴 열풍에 다소 가리워지기는 했지만, 우리의 삶 그리고 다음 세대의 삶에 인터넷이 미치게 되는 영향을 과소 평가할 수는 없다.

한 때 IT 전문가들은 정보 시스템을 섬처럼 구축했다. 이들 데이터 섬 간을 이동하는 것은 상당히 어려웠으며 이들이 교환되는 경우는 더욱 드물었다. 따라서 여타의 과제 즉 언어 격차, 표준 등은 전혀 관심을 끌지 못했다. 그러나 웹은 이들 데이터 섬 간을 보다 쉽게 이동할 수 있도록 했다. 따라서 XML 은 정보 교환에 있어 의미론적인 장벽을 초월할 수 있는 강력한 방법이기 때문에 XML 의 중요성이 더욱 높아지게 된 것이다.

따라서 인포믹스와 같은 기술 벤더들은 변화하고 있다. 우리는 XML 을 자체 제품 및 솔루션의 필수 요소로 간주하고 있다. 전통적으로 인포믹스는 DBMS 소프트웨어의 주도적인 공급업체였다. 보다 많은 사람이 보다 많은 정보를 공유할수록 인포믹스의 확장성이 뛰어난 트랜잭션 데이터 관리 제품에 대한 요구가 어느 때 보다 크다. 또한 웹 소프트웨어의 기본

요구조건 중 하나는 유연성이다. 웹 사이트 급속도로 발전해 룩 앤 필, 컨텐트 및 제공하는 서비스의 유형 등을 변경시키고 있다. 따라서 웹 애플리케이션이 특정한 목적에 맞는 질의를 제시하고 답을 얻을 수 있도록 하는 선언적 질의 중심 인터페이스가 매우 유용하다.

## 익스텐서블 또는 객체 관계형 DBMS

***객체 관계형 데이터베이스, Java, XML***

다행스럽게도 최근 DBMS 기술의 변화는 이러한 통합을 보다 손쉽게 만들었다. 오늘날, 최고의 DBMS 엔진은 확장성이라는 중요한 특성을 가지고 있다. 즉, 이들은 개발자가 DBMS 엔진내에 프로시저 코드 모듈을 내장하고 이들 모듈을 추출된 논리적 데이터 모델 내에서 이용할 수 있도록 한다. 다시 말해, INTEGER, VARCHAR, DECIMAL, FLOAT 및 BLOB 데이터 유형을 저장하고, 이러한 데이터를 정보로 전환하는데 미들웨어 또는 클라이언트 측 로직을 이용하는 것이 아니라 객체 관계형 데이터베이스의 테이블 내에 Java 빈즈, 온도 기록 {120F, 41C, 304K}, 물리적 수량{85Kg, 180LB}, 지리적 지점 및 다각형, 지문 등 다형적이고 다양한 원자(atomic) 객체의 인스턴스를 보유하고 있다.

또한 ORDBMS 를 이용해 개발자는 질의 언어로 이들 객체에 대해 추리할 수 있다. 예를 들어, 소매 구매자가 쉽게 부패할 수 있는 음식물을 찾아내고, 냉장 화물차의 여유 공간을 확인하며, 재고를 구매할 입찰에 대한 메시지를 전송하는 한편 화물차의 여유 공간을 예약 등을 할 수 있는 B2B 전자상거래 익스체인지를 생각해 보자. ORDBMS 내에서 이러한 스키마 및 질의는 다음과 같다.

```
CREATE TABLE Perishable Food(                    CREATE TABLE FreightSpace (
What        Food_Type        NOT NULL,           From_To   Geo_Path NOT NULL,
Where       Geo_Point        NOT NULL,           When      Period     NOT NULL,
Available   Period           NOT NULL,           Capacity  Mass       NOT NULL,
);                                               Space     Vollume    NOT NULL,
                                                 Goods     SET(Packages NOT NULL)
                                                 );
SELECT F.From_To, F.When
FROM      Freight_Spaces F, Perishable_Food P
WHERE     Geo_Within       (Circle(P.Where, '10 Miles'), From(F.From_To) )
AND       Time_Within      (F.Available, Start(P.When) )
AND       Has_Space_For    (P.What, F.Capacity, F.Space, F.Goods);
```

**<그림 1> 객체 관계형 스키마 및 질의 예제. 테이블 내에 익스텐서블 객체를 저장하는 객체 관계형 스키마와 이들 객체를 이용하는 선언적 질의 식에 대한 예제를 제시하고 있다.**

이 그림은 오늘날 데이터 관리 시스템의 정교함을 보여주고 있다. 뿐만 아니라 이러한 애플리케이션에서 XML 의 필요성을 제시하고 있다. 이들 테이블 내의 값은 어디에서 오는 것인

가? 서로 연관되지만 다양하게 표현되어진 데이터(각 도매 공급업체 및 운송 업체는 데이터 저장을 위해 고유의 포맷 및 구조를 지닌 고유의 기존 관리 정보 시스템을 보유하고 있을 것이다) 감안할 경우 이 모든 것을 어떻게 통합할 수 있는가? 그 해결책은 XML 이다.

희망적인 것은 DBMS 의 확장성은 XML 의 실현시키는데 필요한 대부분의 구성 요소 및 기반 기술을 DBMS 내에 직접 포함시킬 수 있다는 것을 의미한다는 것이다. (물론 이는 XML 이 ORDBMS 와 대화할 수 있는 유일한 방법 또는 ORDBMS 는 XML 을 위한 유일한 활용 방안이라는 것을 의미하는 것은 아니다.) 본 보고서의 뒷 부분에서 이것이 어떻게 가능한지를 살펴보도록 하겠다.

## XML 가져오기

***XML 파서***

이러한 시스템을 구축하는데 있어 문제는 연관되어 있지만 서로 다른 형태로 각각 존재하는 데이터의 수와 유형이다. XML 은 이러한 문제를 극복하는데 있어 효과적인 해결책이다. DBMS 와 독립적으로 우리의 B2B 사이트 개발자는 정보를 커뮤니케이션 하는 방법에 대해 설명하는 DTD 스펙 세트를 구축할 수 있다. 우리가 제시한 예제 애플리케이션에서 메시지 샘플은 아래와 같다.

```
<goods>
  <food_item>
    <fodd_type>
     <name>Apples</name>
    <mass unit= “Kg”>12</mass>
    <space unit= “M”>1x1x1</space>
    <store unit= “C”>12</store>
    </food_type>
   <loc> (2.371, 48.937) </loc>
    <available>
     <from> 05/02/2000</from>
     <to> 10/02/2000 </to>
    </available>
   </food_item>
  <food_item>
          etc
    </food_item>
   </goods>
```

```
<freight_spaces>
  <transport>
   <trip>
   (2.371, 48.937. 4.01, 49.24)
    </trip>
    </when>
     <from>05/02/2000</from>
     <to>06/02/2000</to>
    </when>
   <cap units= “T”> 1.5 </cap>
   <vol units= “M”> 3x3x2 </vol>
    <goods>
     <item>Wine
     <mass units= “Kg”> 175</mass>
    <space unit= “M”> .5x.5x.5</space>
    <store unit= “C”> 20 </store>
    </item>
    <item>
      etc
    </item>
    </transport>
    </freight_spaces>
```

**<그림 2> 비즈니스 정보에 대한 XML 교환의 예제**

이는 XML 이 어떻게 복잡한 비즈니스 데이터를 나타내는 표준 방법으로 사용될 수 있는가

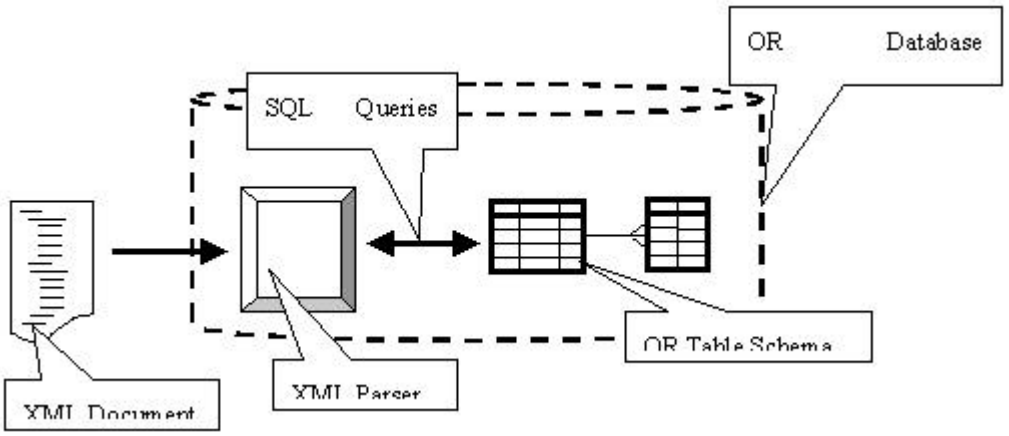
를 설명한 것이다. 이 예제의 XML 데이터는 적절한 표준화된 DTD 또는 스타일 시트(Style Sheet)를 따르게 될 것이다.

이러한 방법은 그림 2 에서 제시된 데이트 유형(이는 여러 다양한 소스에서 가져온 것일 수 있다) 및 그림 1 에서 볼 수 있는 구조 유형(엔드 유저가 자신들의 질문에 대답하는) 간의 격차를 메우게 된다.

XML 의 강점 중 하나는 표준 ASCII 텍스트를 활용한다는 점이다. XML 은 간단한 구조를 가지고 있기 때문에 비록 XML 문서 내에서 데이터에 액세스하기 위해서는 문서를 먼저 처리해야 하지만 이를 위한 쓰기 파서는 비교적 간단한 프로그래밍 작업 만을 필요로 한다. 이에 따라 다양한 상용 품질 파서가 무료로 웹 상에 다양한 소스에서 제공되고 있다. 이들 파서의 대부분은 Java 로 작성됐다.

익스텐서블 DBMS 는 Java 코드를 가져와 이를 DBMS 내에서 그대로 운영할 수 있다. 따라서 우리는 여러 무료 Java XML 파서를 자체 서버 프레임웍 내에 직접 임베드할 수 있다. 아래 그림 3 은 일반적인 아키텍처를 제시하고 있다.

대용량 정보 교환이 이루어지는 대형 문서 및 시스템의 경우 이러한 접근법은 외부 프로그램 및 DBMS 간에 질의 및 데이터 이동하는 오버헤드를 피할 수 있기 때문에 성능 상의 이점을 제공한다. 또한 내장된 코드는 보다 전형적인 프로그램을 이용할 때 처럼 ORDBMS 내에 링크되지 않기 때문에 지속적인 운영 및 유지보수 측면에서 효과적이다. 익스텐서블 DBMS 는 동적 링크 및 호출 기술을 활용하기 때문에 비어있는 데이터베이스 테이블을 드롭하는 것 만큼 쉽게 이들 모듈을 교체할 수 있다.




**<그림 3> XML 파서를 ORDBMS 에 내장시키기 위한 아키텍처**

본 예제는 ORDBMS 에 XML 파서를 내장하는 방법을 보여주고 있다. 파서는 XML 문서를 구별해내고 발견하는 것에 따라 데이터베이스 스키마의 상태를 수정한다. 파서는 파싱되지 않은 전체 문서를 저장하도록 선택할 수 있다. 파서는 XSL 스펙을 이용해 XML 문서의 데이터를 스키마 내 위치에 할당할 수 있다.

본 프로세스는 다음을 포함할 경우 더욱 쉽게 구현되어 질 수 있다.

- ✍✍ DTD 및 스타일 시트를 해당 관계형 스키마에 매핑하는 툴
- ✍✍ 혼합(멀티 파트) 유형, 모음(세트) 및 텍스트 색인화와 같은 반 구조화된(semi-structured) 데이터를 저장하기 위한 기능 등을 비롯한 ORDBMS 의 첨단 데이터 모델 기능

## XML 내보내기

***SGML 마크업***

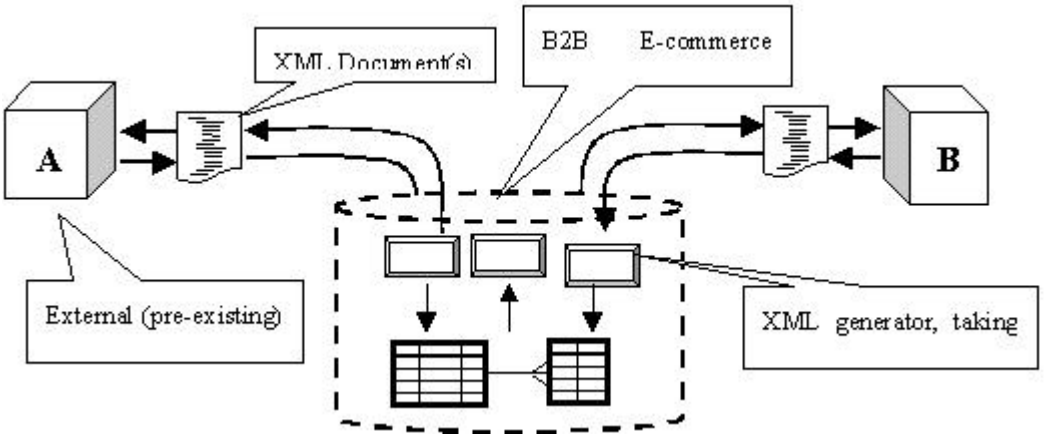
XML은 SGML의 파생물이며 HTML도 마찬가지이다. 그동안 DBMS 제공 업체는 SQL 질의의 결과를 이용하고 HTML 태그로 마크 업해 반송하는 툴을 제공해왔다. 이들 툴이 XML을 처리할 수 있도록 재작성하는데는 상당한 엔지니어링 측면의 작업이 필요했다.

이러한 웹 개발 툴은 질의 결과를 지정된 컬럼 세트로 구성하는 방법을 이용한다. ORDBMS 열 내의 데이터는 혼합 형식(다중 요소를 포함한 단일 컬럼) 또는 COLLECTION(값 세트로 구성된 단일 행/컬럼 데이터 객체)으로 구성될 수 있다. 다행히도 이들 첨단 기능 모두 XML 데이터 모델에 손쉽게 결합될 수 있다.

데이터 분산과 관련해 XML을 데이터베이스에서 내보내는 것은 예제에서와 같이 중앙 서버의 기능을 캡슐화할 수 있도록 한다. 정보를 교환하려는 다른 시스템은 자체 정보를 XML 형태로 전송하고 XML 로 응답을 수신한다. 이러한 방식을 채택한 전체 아키텍처는 아래 그림 4처럼 나타나게 된다.

이 그림에서 여러 이기종 간에 분산되어 있는 데이터를 보다 뛰어난 개별 비즈니스 효율성을 실현하기 위해 공유하도록 하는 방법을 보여주고 있다. 또한 이 그림에서 운송 업체 및 식품 도매업체에 의해 개발된 시스템에 있는 정보 하위 세트는 중앙 B2B 서비스(XML 이용)를 이용해 교환된다. 이 서비스를 이용해 다른 기업들은 쉽게 부패하는 제품의 배정을 위해 입찰하고 제공된 상품의 품질은 물론 지리적 위치 및 배송 여부에 따라 자체 견적 결정을 내릴 수 있다.

본 예제에서는 데이터를 중앙 저장소에 가져오고, 중앙 저장소에서 정보를 내보낸 다음 각 외부 정보 시스템으로 반송하는데 XML 이 이용되는 것을 볼 수 있다.




**<그림 4> B2B 인프라 아키텍처**

보다 큰 그림은 본 문서에서 설명된 XML/ORDBMS 전략이 전체 B2B 인프라에 수용되는 방법을 나타낸 것이다.

## 변환하기

***XSL, XML, 문서 객체 모델***

대부분의 데이터 관리 업체 및 많은 웹 애플리케이션은 이러한 유형의 모델을 채택할 것이다. 그러나 이는 모든 유형의 XML 에 적합하지 않다. XML 의 또 다른 이용 방안은 문서 교환이다. 이 문제 도메인에서 XML 데이터는 일반적으로 앞서 우리가 예측한 시나리오에서 보다 취약한 구조를 제시한다. 그럼에도 불구하고 XML 데이터를 트랜잭션 시스템에 저장하고 외부 사용자가 이와 상호 작용해 질의하고 읽는 등을 실행할 수 있도록 하는 것이 보다 바람직하다. 즉, 그 모든 XML 은 가져오고 내보내는 것은 물론 변환을 처리해야 한다.

결과적으로 XML 문서는 완전히 비구조적지만 마크 업될 수 있다. 이러한 문서 유형에서 키워드나 어구는 라벨을 붙여 의미론적 정보를 제공해야 한다. 때로 이들 태그는 사용자 인터페이스 프로그램을 위한 표시이지만 때로 사용자는 이러한 문서의 컨텐트에 대해 질문하기를 원할 수 있다. 예를 들어, 사용자는 "파리라는 단어가 목적지로 태그된 리파지토리에 저장된 문서를 보여주세요."라고 말하고 싶을 수 있다.

이러한 문서 데이터를 저장하기 적합한 방법은 문서 색인, 문서 컨텐트별 질의 등과 같은 데이터 관리 기법을 이용하는 것이다. 객체 관계형 DBMS 는 이러한 기능 유형으로 확장될

수 있다. 데이터를 저장하기 위해 DBMS 를 결과적으로 이용하든 이용하지 않든 ORDBMS 는 색인 및 확장 가능한 기능을 통해 데이터 처리에 대해 중요한 역할을 수행할 수 있다.

대체적으로 스타일 시트 또는 DTD 가 없을 경우 XML 문서를 공유하는 것이 바람직하다. 덮어쓰기(Shredding)는 그 요소의 값을 테이블 내 해당 행에 지정하는 대신 XML 의 파싱을 필요로 한다. 이러한 전략과 관련된 분명한 과제는 객체 관계형 모델 내에 XML 문서의 원래 구조를 유지하는 방법이다.

## 요약 및 결론

본 백서에서 XML 및 익스텐서블 또는 객체 관계형 DBMS 기술이 상호 보완하는 방법에 대해 살펴봤다. 요약하면 웹 애플리케이션을 개발하는데 있어 XML 의 중요성 및 유용성은 데이터 상호 교환 포맷으로서 분산되어 있는 데이터 간의 정보 교환을 실행하는 것이다. 그러나 XML 을 효과적으로 이용하기 위해서는 데이터베이스 관리 시스템이 개발되는 방법 상의 변화를 필요로 한다. 또한 효과적인 웹 애플리케이션을 개발하려는 개발자는 과거, 관계형 DBMS 를 이용하는 방법과는 다른 방법으로 객체 관계형 DBMS 를 이용해야 할 것이다.

본 문서의 핵심 사항은 다음과 같다.

1. 객체 관계형 DBMS 의 코어 확장성은 벤더 및 고객이 XML 데이터를 파싱하기 위해 로직을 DBMS 내에 내장하고 이후 문서 컨텐트를 토대로 데이터베이스를 변경할 수 있도록 한다. 마찬가지로 최고의 DBMS 기술은 개발자들이 로직을 내장해 SQL 질의 결과를 XML 로 변환할 수 있도록 한다.
2. 객체 관계형 데이터 모델은 혼합(다중 요소) 데이터 구조 및 COLLECTIONS(세트), XML 및 ORDBMS 간의 매핑 작업 등과 같은 기능을 포함하기 때문에 SQL 은 XML 과 SQL 의 이전 버전 간의 매핑 작업처럼 복잡하지 않다. 또한 관련 데이터의 복잡성 때문에 XML 을 이용하는 것은 이러한 시스템을 진정으로 지원하는데 있어 필수적이다.

요약해서 XML 을 지원하는데 있어 필수적인 3 가지는 XML 데이터를 데이터베이스로 가져오고 외부 시스템이 이를 필요로 할 경우 XML 을 내보내고 이를 저장해야 할 경우 XML 을 변환해야 한다.