객체-관계형 데이터 모델을 토대로 한 다중 데이터베이스 시스템의 스키마 통합에 관한 연구

o
이상태, 주경수
순천향 대학교 전산학과

A Study on the Schema Integration of Multidatabase System based on the Object-Relational Data Model

o
Sang-Tae Lee, Gyung-Soo Joo
Dept. of Computer Science, SoonChunHyang Univ.

요 약

다중 데이터베이스 시스템은 분산되어 있고 이질적이며 자치적인 데이터베이스 시스템들의 연합이다. 다중 데이터베이스의 환경에서는 공간적으로 데이터들이 분산되어 있고, 자료의 특성에 따라 서로 다른 데이터 모델에 의해 데이터베이스가 구축되어 있으며, 독자적으로 각각의 데이터베이스 시스템에 의해 관리되고 있다. 이러한 환경하에서는 정보를 효율적이고 종합적으로 관리하고 공유할 필요가 있으며, 이에 따라 사용자 관점의 논리적인 통합이 이루어 질 수 있을 것이다. 이러한 논리적인 통합을 위해 분산된 그리고 자치적이며 이질의 데이터베이스를 통합하기 위한 통합 데이터 모델이 요구되며, 이러한 통합 데이터 모델에 의해 완전하고 일치된 정보 모델이 제공될 것이다.

본 연구에서는 객체-관계형 데이터 모델을 토대로 하여 분산·이질·자치적인 데이터베이스들을 효율적이고 종합적으로 통합하기 위한 효과적인 방안에 대하여 연구하였다.

1. 서론

데이터베이스 시스템은 데이터의 집합으로서, 데이터 모델에 의해 조직되는 데이터베이스와 이를 관리하는 소프트웨어인 데이터베이스 관리 시스템(DBMS)으로 구성된다. 시스템의 실제 데이터 구조와 구성은 스키마로 정의되며, 사용자는 DBMS에 의해 지원되는 질의 언어를 이용하여 데이터에 접근한다.

네트워크 기술이 발달함에 따라 데이터베이스시스템은 분산 환경을 취급하기 위한 분산 데이터베이스 시스템으로 발전하고 있으며, 분산 환경은 이질적인 특성을 가지게 되는데, 그 이질성은 각 구성 사이트(site)에서의 하드웨어 구성 및 운용 시스템 그리고 네트워크 프로토콜 등이 상이하기 때문에 발생한다. 또한, 데이터베이스 측면에서의 이질성은 각각의 로컬(local) 사이트에서 사용된 데이터 모델, 질의 언어, 그리고 스키마 등과 같은 데이터베이스 수준에서의 구성 요소가 상이하기 때문에 발생한다[5,7,10].

분산 환경하의 이질성을 극복하여 통합 데이터베

이스를 구축하기 위해서는 이질의 데이터 모델을 통합시키는 것이 무엇보다 중요하며, 이를 위하여 로컬 데이터 모델을 사상(mapping)시키는 통합 데이터 모델이 요구된다. 기존의 연구에서는 이미 구축된 로컬 사이트가 대부분 관계형에 의해 구축되었으므로, 통합 데이터 모델은 관계형이 주종을 이루었고, 최근에는 객체지향 방법으로 모델링하고 이를 JOIN 등을 이용하여 관계형 데이터베이스에 구현하는 것을 많이 볼 수 있다.

통합 데이터베이스의 환경에서는 표준화된 정보를 효율적이고 통합적인 방법으로 공유할 수 있도록 하는 것으로 통합 데이터베이스 시스템 구현의 핵심 요소라 할 수 있다. 여기에서 통합이란 의미는 물리적으로 하나의 컴퓨터나 데이터베이스를 의미하는 것이 아니라, 사용자 관점의 논리적인 통합을 말하는 것으로 다양한 형태의 정보를 언제 어디에서나 투명하게 실시간에 접근할 수 있도록 한다는 의미이다.

본 연구에서는 객체-관계형 데이터 모델을 토대로 하여 통합 데이터베이스의 환경, 즉 분산·이질·자치적 데이터베이스들을 통합하기 위한 효과적인 방안에 대하여 연구하였다.

본 논문의 구성은 제2장에서는 관련 연구로서 객체-관계형 데이터 모델에 대하여 살펴보고, 제3장에서는 다중 데이터베이스 시스템에 대하여 서술하며, 제4장에서는 스키마 통합을 설명하고, 제5장에서는 이 논문의 마지막으로 결론을 기술한다.

2. 객체-관계형 데이터 모델

2.1 기본 타입 확장

SQL-92는 정수, 실수, 고정 또는 가변 길이의 문자열, 날짜, 시간, 시간간격(time interval), 수치와 십진수로 테이블의 항목을 제한한다. 그리고 각 데이터 타입에 연관된 정확한 함수의 집합과 연산자를 정의하고, 정수에 대해서 표준 산술 및 비교 연산자가 제공된다. 그러나 SSQL-92에서 지원하는 데이터 타입과 연산들은 제한적이며, 많은 실세계 문제들은 프로그램으로 작성하기에 매우 어렵다.

(1) 확장 데이터 타입

객체-관계 DBMS에서 제공되는 확장 데이터 타입은 효율성 문제를 일으키는 부자연스러운 타입 시뮬레이션 문제를 근본적으로 제거한다. 그러기 위해서는 사용자가 자신의 데이터 타입을 생성할 수 있도록 되어있다.

(2) 사용자 정의 함수 및 연산자

기존의 SQL 시스템은 수치 데이터 타입에 대해 산술과 비교 연산자를 제공한다. 사용자 정의 타입의 경우는 그 타입에 관련된 연산은 사용자가 추가할 수 있도록 되어있다.

2.2 완전한 객체-관계형 타입 확장

(1) 동적 링크

대규모 사용자 정의 함수 라이브러리를 정적으로 링크함으로써 DBMS의 크기가 5배에서 10배로 증가한다면 대단한 일이다. 이와 같은 경우는 패턴 인식이나 이미지 분석 기능들에서 실제로 가능한 일이다. 대부분의 운영체제에서는 이러한 발자국이 증가할수록 추가적인 운영체제의 부담과 페이징(paging) 활동이 따르게 된다. 그리고 이 발자국이 너무 클 경우에는 운영체제가 중단되는 경우도 있게 된다. 그러므로 운영체제의 성능면에서부터 사용자의 편리성에 이르는 여러 가지 이유로 인하여 객체-관계 DBMS는 사용자 정의 함수를 동적으로 링크한다.

(2) 클라이언트 또는 서버측에서의 구동

서버측이나 클라이언트측 구동은 특정한 응용 분야에 따라 의미를 가지게 된다. 객체-관계 DBMS는 두가지를 모두 제공한다. 그리고 응용이 사용되는 기간동안 변화되는 기계의 속도, 부하 그리고 네트워크 연결상태의 변경 등에 따라 두가지 기능을 스위치하여 사용한다.

(3) 보안

서버측 구동 방식에서는 사용자가 실수 또는 고의적으로 보안 규칙을 위반하여 데이터베이스의 데이터를 검색하거나 기록하는 사용자 정의 함수를 정의할 위험이 있게 된다. 그러므로 서버측 구동을 지원하는 시스템에서는 이와 같은 위험한 작업을 제거할 수 있도록 되어있다.

(4) 콜백

객체-관계 DBMS에서는 함수가 자신의 매개변수와 데이터베이스를 비교하는 방법으로 결과를 처리한다. 이와 같은 기능을 제공하기 위해서 사용자 정의 함수는 "콜백(callback)" 기능을 가지고 있다. 즉 하나의 함수내에서 다른 질의문을 수행할 수 있다.

(5) 임의 길이 타입

사용자가 정의 데이터 타입에는 고정 길이나 짧은 길이의 데이터 타입도 사용이 되지만 이미지 타입과 같이 고정된 길이지만 길이가 긴 데이터 타입도 있다. 그리고 압축된 이미지 표현과 같은 가변 길이이며 매우 긴 데이터 타입도 사용된다. 객체-관계 DBMS에서는 {길다,짧다}와, {고정,가변} 길이의 임의의 조합을 가지는 데이터 타입을 지원한다.

2.3 복합 객체

(1) 타입 생성자

객체지향 데이터베이스 복합 객체는 여러 개의 기본 또는 사용자 정의 타입으로 구성된 객체이다. 합성, 집합, 참조는 객체-관계 DBMS에서 복합 타입을 생성하는 기본적인 빌딩 블록들이다. 합성 타입을 생성하려면 타입의 이름과 함께 구성 요소의 이름과 데이터 타입을 명시하여야 하며, 합성 타입의 요소에 다른 합성 타입을 사용할 수 있다.

(2) 기본 타입과 복합 객체

객체-관계 DBMS에서는 시스템을 새로운 기본 데이터 타입과 함수의 정의 기능, 그리고 집합, 참조, 합성의 타입 생성자를 사용한 새로운 복합 객체와 함수의 정의 기능을 사용하여 확장 할 수 있게 한다. 그것은 자연스러움을 제공하고 캡슐화를 가능하게 하며 OID를 사용하여 공간 사용에 따른 부담을 줄인다[11,12].

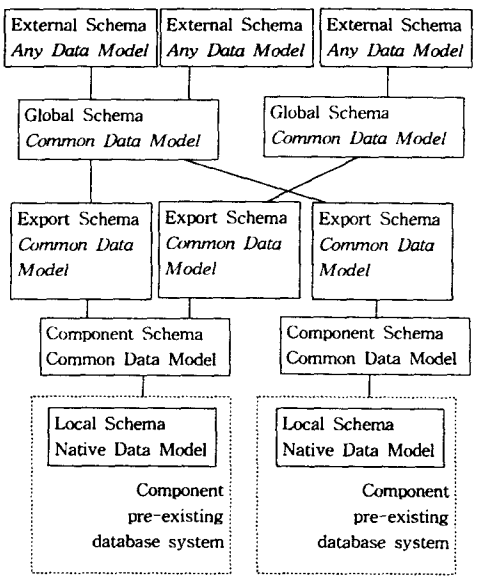
3. 다중 데이터베이스 시스템

3.1 스키마 구조

공통 데이터 모델은 다중 데이터베이스 시스템을 설계하고 구현함에 있어 유연한 구조를 제공한다[8]. 중앙집중 데이터베이스 시스템의 스키마 구조를 기술키 위한 전통적인 3단계 스키마는 다중 데이터베이스 시스템의 구조를 기술하기 위하여 확장되었는데, 이는 5단계 스키마 구조로 나타내지며 각 컴포넌트 데이터베이스(component database)의 개념적 스키마를 로컬 스키마(local schema)라 부른다[2,9]. 로컬 스키마는 컴포넌트 데이터베이스 자체의 데이터 모델에 의해서 표현될 수 있다. <그림1>은 다중 데이터베이스 시스템의 5단계 스키마 구조를 보여준다[6].

시스템에 대한 접근을 위하여 로컬 스키마를 공통 데이터 모델로 변환한다. 이 변환으로 만들어진 스키마를 컴포넌트 스키마(component schma)라고 부른다. 글로벌 스키마(global schma)는 복수개의 엑스포트 스키마(export schma)들을 집약함으로써 만들어진다.

스키마 변환은 한 개의 데이터 모델에서 표현된 스키마가 다른 데이터 모델의 동등한 스키마에 사상될 때 발생되며 이에 따라 다중 데이터베이스 시스템의 스키마 변환은 로컬 스키마를 컴포넌트 스키마로 변환 시 나타나고, 아울러 글로벌 스키마를 외부 스키마로 변환할 때 발생된다[1,15,16,17].



<그림1> 5단계 스키마 구조

3.2 통합

자율적인 데이터베이스들의 통합에 대한 필요성은 오늘날 모든 큰 조직에서 절실히 느껴지고 있다[3,4,13,14,19].

스키마 통합은 분산되어 있고 이질적인 데이터베이스 시스템들을 통합하여야 한다는데 어려움이 있다. 즉 스키마 속성이 수백 또는 수천이 되는 것은 일반적이므로 많은 데이터베이스들을 통합하는 것은 힘드는 일이다. 그러나 매우 필요한 일이며, 스키마 통합은 많은 수의 비교를 수동적으로 처리하는 경우와 같이 복합적이고 오류가 나타날 수 있는 진행 과정이다. 이것은 소프트웨어 툴이 이러한 연산을 해야 하는 것을 의미한다.

(1) 스키마 통합 접근

스키마 통합은 다음과 같은 객체들에 의해서 이루어진다.

1. 사용자의 요구가 고려되어야 한다.
2. 현존하는 데이터와 스키마가 변경되어서는 안된다.
3. 사용자는 글로벌 의미 모델을 볼 수 있어야 한다. 즉 사용자는 새로운 변형을 배울 필요가 없으며, 중요한 스키마와 스키마 통합 실행을 이해하기 위하여 모델을 배울 필요가 없어야 한다.
4. 증가하는 스키마 통합이 가능해야 한다. 새로운 스키마가 통합될 때, 또는 현존하는 스키마가 변형될 때 사용자는 모든 스키마를 다시 통합할 필요가 없어야 한다.
5. 형태나 불명확한 정보에 대한 기술이라기 보다는 객체 전달자를 정의하는데 사용자를 돕기 위해 행해져야 한다.
6. 글로벌 스키마와 컴포넌트 스키마 사이에 사상은 자동적으로 실행되어져야 한다.

● 스키마 통합은 다음의 사항들로 나누어진다.

1. 전달 확인을 일반화하기 위한 스키마의 비교를 위해 기술을 증명하는 응용
2. 사용자가 시스템 일반화를 확인하거나 또는 새로운 단정을 열거하는 것
3. 새로운 단정에 대한 자동 생성 또는 사용자가 정의한 것에 토대를 둔 삭제
4. 사용자가 확인한 것의 일관성에 대한 점검과 확인
5. 특수한 확인과 옵션에 따라서 객체를 병합
6. 글로발 스키마와 컴포넌트 스키마 사이에 사상의 일반화

(2) 데이터 모델

사용한 데이터 모델은 객체지향 개념을 가진 확장된 ER 모델로서 다음과 같은 특성들을 갖고 있다.

1. 클래스 구조가 개체타입으로 구성되어 있다.
2. 속성 연결을 통하여 이중 관계성을 나타낼 수 있다.
3. 관계성 구조는 관계 구조라고 불린다
4. 복합 객체는 집단이라 불린다.
5. 개체 타입은 구성 객체 타입이라 불린다.
6. 다중 상속은 슈퍼 클래스에서 하위클래스로 속성 전달이 일어난다.
7. 메소드는 제한된 속에서 진행되는데 속성도 마찬가지이다. 그들은 독자적인 함수를 실행하지 않으나 단순값을 되돌려준다.

3.3 통합의 문제

스키마 통합은 컴포넌트 스키마들 사이에 공통부분을 확인하기 위하여 서로간에 스키마 객체들을 확인하는 것을 말하며, 또한 그들 사이에 전달을 통하여 병합하는 것을 말한다.

전달에 있어서 많은 서로 다른 타입들은 컴포넌트 스키마의 객체들 사이에서 일어난다. 컴포넌트 스키마들로부터 두개의 타입은 서로간에 대등, 포함 또는 상위클래스, 이중겹침, 비공유, 무관계 등이다.

전달은 개체 타입의 의미에서 형성되거나 그들의 인스턴스 집합에서 형성된다. 그것은 전달의 무관계, 비결합 타입들 사이에서 만들어진다.

(1) 관계 타입들 간의 전달

서로 다른 두가지 관계 타입은 대등관계를 통하여 이루어진다. 대등관계는 아래의 두 조건을 만족하여야 한다. 즉 (1) 관계 타입의 도메인은 같아야 한다. (2) 관계타입은 양쪽에서 같은 역할을 해야한다.

(2) 속성 타입들 간의 전달

전달의 두 가지 종류는 속성 타입들 사이에서 확인될 수 있다. 그들은 속성의 도메인들이 어떻게 관계되는가를 열거하고 또 부분적인 인스턴스를 위해 전달되는 속성값들이 어떻게 관계되는가를 열거한다. 도메인은 어떻게 주체적인 개체 타입이 관계되며 그래서 개체 타입의 전달을 확정하기 위해 유효하게 묘사하는가의 의미정보의 이동을 전달한다. 그리고 우리는 도메인에 바탕을 둔 두 속성들 사이에 전달의 포함이나 중첩을 확인할 수 있다.

전달은 개개의 속성 값이 어떻게 하부의 데이터베이스로부터 공통모양으로 유도되는가에 대한 정보를 가져온다. 이러한 전달은 질의 과정을 위해 데이터 값의 전달 필요성에 대한 정보를 포함한다.

(3) 서로 다른 객체 타입들 간의 전달

비슷한 개념이 서로 다른 모델 구조의 사용을 보여주기 때문에 서로 다른 객체들 사이에서 전달이 일어난다. 개체 타입과 속성 타입 사이에 그리고 개

체 타입과 관계 타입 사이에, 관계 타입과 속성 타입 사이에 전달은 동일하다. 포함과 중첩 관계는 서로 다른 타입들 간에 있을 수 있다. 단순 클래스를 구성하기 위한 서로 다른 스키마들로부터 적절한 새로운 클래스에 대한 계층화를 사용한다.

4. 스키마 통합

객체-관계형 데이터베이스 시스템에서는 현존하는 데이터와 스키마를 변경하지 않고 어플리케이션을 개발 할 수 있도록 되어 있다.

지리적으로 분리되어 있는 기존의 상용데이터베이스들을 하나로 통합, 마치 하나의 데이터베이스처럼 사용할 수 있게 한다. 기존의 데이터베이스와 어플리케이션들을 그대로 활용하고 여기에 새로운 어플리케이션을 추가할 수 있다. 그리고 분산 네트웍내의 트랜잭션을 관리한다.

긴 데이터를 처리할 수 있는 기능과 복잡한 데이터 타입을 처리할 수 있는 기능을 가지고 있다. 즉 임의의 클래스를 데이터 타입으로 정의할 수 있으며, 테이블의 필드에는 복수개의 데이터를 포함시킬 수 있다. 또한 계승 개념으로 데이터베이스에 유연성을 부여한다.

복수의 이기종 관계형 및 객체형 데이터베이스들 상에서 단일 데이터베이스 언어와 단일 글로벌 뷰를 이용한 어플리케이션의 개발이 가능하도록 되어있다.

사용자들은 동시에 질의 및 갱신을 할 수 있다. 사용자들은 글로벌 데이터베이스에 대하여 단일 트랜잭션으로 질의문과 갱신문의 집합을 제출할 수 있다. 그리고 다른 사용자들에게 그 데이터베이스의 일부분에 대하여 권한을 부여하거나 부여된 권한을 회수할 수도 있다.

복수의 이기종 관계형 및 객체형 데이터베이스들 상에서 단일 데이터베이스 언어와 단일 글로벌 뷰를 이용한 어플리케이션의 개발을 가능하도록 하는 멀티데이터베이스 시스템이다.

MDBS는 하나의 '글로벌' 데이터베이스 스키마를 유지하며, 사용자들은 이 글로벌 스키마에 대해 질의 및 갱신을 할 수 있다. MDBS는 단지 글로벌 스키마만을 유지하고, 실제로 모든 사용자 데이터베이스는 외부 데이터베이스 시스템이 유지하고 있다. 글로벌 스키마는 외부 데이터베이스 스키마들을 통합함으로써 구축되어진다.

어떠한 관계형 데이터베이스는 물론 어떠한 객체형 데이터베이스들과도 연결할 수 있다. 오직 한 관계형 데이터베이스와 연결되어 있다면, 관계형 데이터베이스 엔진위에 오브젝트 관리 계층으로 이용할 수 있고, 또한 하나의 객체형 데이터베이스와 연결되어 있다면, ANSI SQL 호환 질의 언어를 제공하지 못하는 객체형 데이터베이스 엔진위에 객체형 SQL 질의 계층으로 사용할 수가 있다.

5. 결론

다중 데이터베이스 시스템은 분산되어 있고 이질적이며 자치적인 데이터베이스 시스템들의 연합이다. 본 연구에서는 객체-관계형 데이터 모델을 토대로 하여 분산·이질·자치적인 데이터베이스들을 효율적이고 종합적으로 통합하고, 의미적 정보 사용을 용이하게 해주는 점에 대하여 고찰 하였다.

다중 데이터베이스 시스템이 객체-관계형 데이터 모델을 바탕으로 구축되면 논리적인 정보 공유가 가능하므로써 다중 데이터베이스 환경하에서 정확한 정보를 효과적으로 관리 및 제공할 수 있게 된다.

참 고 문 헌

(1) A. Savasere, A. Sheth, S. Gala, S. Navathe and H. Marcus. "On Applying Classification to Schema Integration." In Proceedings of the First Intenational Workshop in Interoperability in Multidatabase Systems, Kyoto, Japan, 1991.

(2) A. SHETH, and J. LARSON. "Federated database systems." ACM Comput. Surv. 22, 183-236. 1990.

(3) A. Sheth, S. Gala, and S. Navathe. "On Automatic Reasoning for Schema Integration." Interational Journal of Intelligent Co-operartive Information Systems, 2:1, 1993.

(4) C. BATINI, M. LENZERINI, and S. B. NAVATHE. "Comparison of methodologies for database schema integration" ACM Comput. Surv. 18, 4 (Dec.), 323-364. 1986.

(5) E. PITOULA. "Extending an object-oriented programming language to support the integration of database systems." In 28th

Annual Hawaii International Conference on System Sciences (HICSS-28) (Maui, Hawaii, Jan.), 707-716. 1995.

(6) E. PITOULA, O. BUKHRES, and A. ELMAGARMID. "Object Orientation in Multidatabase Systems." ACM Comput. Surv. 27, 2 (June), 141-195. 1995.

(7) F. MANOLA, and S. HIELER. "An Approach to interoperable object models." In Proceedings of the International Workshop on Dstributed Object Management (Edmonton Canada, Aug.), 326-330. 1992.

(8) F. MANOLA, and S. HEILER, D. GEORGAKOPOULOS, M. HORNICK, and M. BRODIE. "Dstributed Object Management." Int. J. Intell. Cooperative Info. Syst. 1, 1 (June). 1992.

(9) J. LARSON, S. NAVATHE, and R. ELMARSI. "A thory of attribute equalance in databases with applications schema integration." IEEE Trans. Softw. Eng. 15, 4 (April), 449-463. 1989.

(10) O. A. BUKHRES, W. CHEN, J., DU, A. K. ELMAGA RMID, AND R. PEZZOLI. "InterBase: An execution environment for heterogeneous software systems." IEEE COMPUTE R, (Aug.), 57-69. 1993.

(11) Q. LI, and D. MCLEOD. "An object-oriented approach to federated databases." In Proceedings of the First International Workshop on Interoperability in multidatabase systems (April), 64-70. 1991.

(12) R. M. SOLEY. "Using object technology to integrate distributed applications." In Enterprise Integration Modeling, Proceedings of the First International Conference, MIT Press, Cambridge, Mass., 446-454. 1992.

(13) S. Navathe and S. Gadgil. "A Methodology for View Integration in Logical Database Design." In Proceedings of the Eighth International Conference on Very Large Data bases, VLDB Endowmant, Mexico City, Mexico, pp. 142-164. 1982.

(14) S. Navathe, R. elmasri and J. Larson. "Integrating User Views in Database Design." IEEE Computer, 19:1, pp.50-62. 1986.

(15) U. DAYAL, and H. HWANG. "View definition and generalization for database integration in a multidatabase system." IEEE Trans. Softw. Eng. 10, 6, 628-645. 1984.

(16) W. KIM, and J. SED. "Classifying schematic and data heterogeneity in multidatabase systems." IEEE Computer 24, 12 (Dec.), 12-17. 1991.

(17) W. KIM, I. S. CHOI, and M. SCHEEVEL. "On resolving schematic heterogeneity in multidatabase systems." Int. J. Parallel Distrib. Databases 1, 251-279.IEEE Computer 24, 12 (Dec.), 12-17. 1993.

(19) W. Whang S. Chakravarthy, and S. Nacathe. "Heterogeneous Databases : Toward Merging and Quering Component Schemas." Computing Systems, 6:3, 1993.