전문가 리포트

이질 데이터베이스 통합

(Integration of Heterogeneous Distributed Databases)

나민영/육군사관학교 전산학교수
Ra, Min-Young/Assistant Professor
Dept. of Mathematics & Computer Science

최근 들어 통신 기술과 데이터베이스 기술의 발달은 사용자의 데이터 처리 환경을 변화시키고 있다. 즉 이미 만들어져 운용되고 있는 서로 다른 데이터베이스들을 논리적으로 합하여 하나의 통합 시스템을 구축함으로서 데이터를 서로 공유할 수 있도록 하고자 하는 요구가 증대되고 있다. 논리적 데이터 통합은 이로 말미암아 사용자로 하여금 하나의 데이터베이스 시스템이라는 착각을 유발하게 하고 서로 다른 DBMS와 여러 액세스 기법에 따른 복잡함으로부터 벗어나게 해 준다.

이는 또한 사용자에게 여러 데이터베이스에 있는 데이터를 동일하게 액세스할 수 있도록 해준다. 이때 이 데이터를 새로운 데이터베이스로 이주시키지 않아도 되고 또한 서로 다른 데이터베이스의 위치나 특성을 알 필요도 없다. 이때 데이터 자원은 서로 연관이 없고 나중에 서로 통합된다는 점을 고려하지 않고 독자적으로 만들어졌다는 점에서 pre-existing이다. 또한 통합 시스템에 참여하는 참여 DBMS는 서로 다른 환경에서 작동하고 서로 다른 데이터 모델, 데이터 정의, 데이터 조작 기능, 트랜잭션 처리 및 동시성 제어 메카니즘, 물리적 데이터 구조를 사용한다는 점에서 이질적이다. 그림 1은 이질 데이터베이스가 통합된 개념적 구조를 보여 준다. 여기서 FDBS (Federated Database Systems) 이란 이질 데이터베이스들이 통합된 시스템을 가리키는 용어로 1에서 설명된다.

본 고에서는 이러한 이질적인 또한 분산 데이터베이스의 통합에 관하여 살펴본다.

1에서는 이질 데이터베이스 통합의 특성과 접근방법을 알아보고, 2에서는 통합시 사용되는 기법중 대표적인 기법인 스키마 변환과 통합에 관하여 살펴본 다음, 3에서는 통합된 시스템에서의 트랜잭션 관리에 관하여 주요 이슈 및 해결방안 등을 살펴본다.

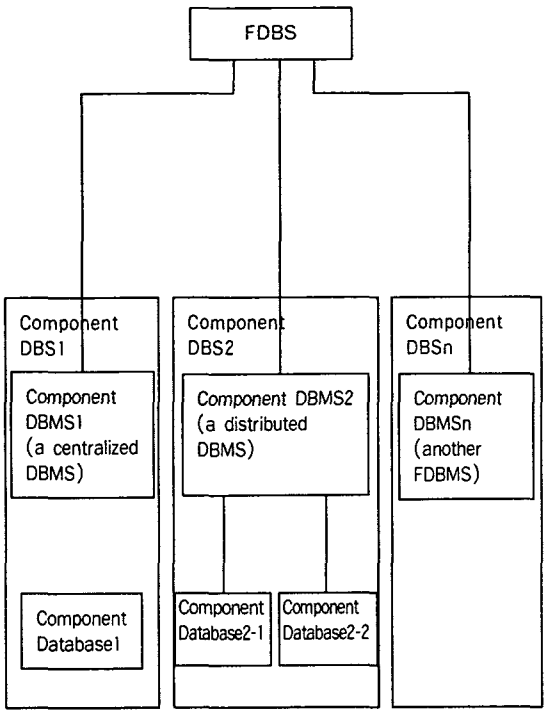

그림 1. 이질 데이터베이스 시스템의 통합

1 이질 데이터베이스 통합

1.1 이질 데이터베이스

데이터베이스 시스템은 DBMS(DataBase Management System)이라는 하나의 소프트웨어와 DBMS가 관리하는 하나 또는 그 이상의 데이터베이스들로 구성된다 [Elma 89]. 현재 운용되고 있는 대부분의 데이터베이스들은 중앙집중형이다. 이는 데이터가 하나의 컴퓨터에 저장되어 있다는 뜻이다. 그러나 점차 여러개의 데이터 베이스들을 다루고자 하는 요구가 증가하고 있다. 왜냐하면 대부분의 조직은 각자의 데이터베이스를 가진 여러대의 컴퓨터를 보유하고 있으며, 사용자 또는 응용프로그램은 이러한 여러 데이터베이스로부터, 각각의 세부적인 구현방법과는 관계없이, 정보를 요청하기를 원하기 때문이다. 따라서 여러 데이터베이스에 대한 액세스를 제공할 수 있는 시스템이 요구되고 있다.

서로 이질적인 여러 데이터베이스들이 우리 주변에 존재하게 되는 이유는 먼저 데이터베이스들이 이미 개발되어 사용되고 있으며 현재 대부분 업무의 기능 및 응용 성격이 분산적이기 때문이다. 이러한 이질적인 데이터베이스들을 통합하게 되면 데이터의 공유를 허용할 수 있고 응용프로그램의 개발을 쉽게 할 수 있으며 또한 관련된 데이터의 일치를 꾀하며 중복을 피할수 있게 되므로 이 분야에 관한 연구는 매우 중요하게 인식되고 있다.

이질 데이터베이스 통합과 관련된 용어는 아직 그 표준이 정해져 있지 않아 학자에 따라 같은 개념으로 비슷한 용어를 사용하고 있는 형편이다. 즉, 이질 분산 데이터베이스 시스템 (Heterogeneous Distributed Database System), 연합 데이터베이스 시스템 (Federated Database System), 멀티 데이터베이스 시스템 (Multidatabase System) 등으로 불린다.

1.2 데이터베이스 시스템 통합 특징

여러개의 DBS으로 구성된 시스템은 보통 분산(distribution), 이질성(heterogeneity), 자치성(autonomy) 의 3차원으로 특징지울 수 있다.

1.2.1 Distribution

데이터는 여러개의 데이터베이스사이에 분산될 수 있다. 이러한 데이터 베이스들은 하나의 컴퓨터 시스템에 저장되거나 또는 여러개의 컴퓨터 시스템에 저장된다.

데이터는 여러개의 데이터베이스에 다양한 방법으로 즉, 관계형 언어로서 표현하면 수평 또는 수직분할로서 분할되어 분산될 수 있다. 여러개의 복제품이 유지되나 이들이 모두 동일한 구조를 가질 필요는 없다. 데이터 분산의 효과는 이미 잘

알려진대로 가용성과 신뢰도뿐만 아니라 액세스 시간을 향상시킨다. 이질 데이터베이스 통합인 경우에는 대부분의 데이터 분산이 이미 존재하는 여러개의 DBS때문에 발생한다.

1.2. 2 Heterogeneity

이질성의 대부분 형태는 기술적 차이에서 비롯된다. 예를 들면 하드웨어,소프트웨어, 그리고 통신 시스템 같은 것들이다. 이러한 기술적 차이를 해결하기 위해 많은 연구가 있어 왔다. 데이터베이스 시스템에 있어서 이질성의 형태는 크게 다음과 같이 DBMS의 차이와 데이터의 시맨틱에 있어서의 차이로 나뉜다.

(1) DBMS의 차이에 따른 이질성

한 조직내에서 예하조직들은 서로 다른 요구사항들을 가지고 서로 다른 DBMS를 선정할 수 있다. 또 오랜 기간에 걸쳐 구입된 DBMS들은 기술의 변화때문에 서로 다를 수 있다. DBMS 차이에 기인한 이질성은 데이터 모델의 차이와 시스템 레벨의 차이로부터 유발된다. 각 DBMS은 데이터 구조와 제약조건을 정의하기 위하여 사용되는 데이터 모델에 기반을 두고 있다. 이 둘 (구조 및 제약조건)뿐만 아니라 사용언어도 이질성을 유발한다. 이들을 설명하면 다음과 같다.

o 구조의 차이 : 서로 다른 데이터 모델은 서로 다른 구조적 프리미티브를 제공한다. 만일 두 표현이 같은 정보의 내용을 다루고 있다면 구조의 차이를 해결하기가 쉬우나 정보의 내용이 다르면 그리 쉽지 않다. 예를 들어 주소가 어느 한 스키마에서는 엔티티로 표현되고 다른 스키마에서는 복합 애트리뷰트로 표현된 경우에는 비교적 다루기 쉽다. 그러나 어느 한 모델에서는 generalization을 지원하고 다른 모델에서는 그렇지 못할 때에는 구조적 차이를 해결하기가 쉽지 않다.

o 제약조건에서의 차이: 두 데이터 모델은 서로 다른 제약조건을 가지고 있을 수 있다. 예를 들어 CODASYL 스키마의 셋 타입은 관계형 스키마의 참조 무결성 제약조건으로서 부분적으로 모델될 수 있다. 그러나 CODASYL은 참조무결성만으로는 해결할 수 없는 insertion, retention 제약조건을 지원하고 있다. 관계형 시스템에서는 이런 경우 trigger를 사용해야만 한다.

o 질의어에서의 차이: 서로 다른 데이터 모델에서 표현된 데이터를 액세스하기 위해 서로 다른 언어들이 사용된다. 두 DBMS들이 같은 데이터 모델을 지원한다 하더라도 그 질의어에는 차이가 있다 (예: QUEL, SQL). 또한 버젼이 달라도 이질성을 야기할 수 있다.

이 외에도 DBMS의 시스템 측면에서의 차이도 이질성을 야기한다.

예를 들면 동시성 제어, 회복등의 트랜잭션 관리 프리미티브에서의 차이, 하드웨어와 시스템 소프트웨어 요구사항에서의 차이, 그리고 통신 능력에서의 차이같은 것들이다.

(2) 시맨틱 이질성

시맨틱 이질성은 같거나 혹은 연관된 데이터의 의미, 해석, 사용에 관한 불일치가 있는 곳에서 발생한다. 이 문제는 그동안 비교적 잘 이해되지 못했고 더우기 정의조차 제대로 내려져 있지 않았다. 시맨틱 이질성의 예를 들면 다음과 같은 것이다. DB1과 DB2는 가상의 릴레이션들이다.

데이터베이스 DB1에 있는 RESTAURANT 릴레이션의 MEAL-COST 라는 애트리뷰트를 생각해보자. 이 애트리뷰트는 서비스 요금과 세금이 없는 레스토랑에서 한 사람당 평균 식사비를 나타낸다. DB2의 BOARDING 릴레이션에 있는 같은 이름을 가진 애트리뷰트가 있다 하자. 이 애트리뷰트는 서비스 요금과 세금이 포함된 평균 식사 비용을 나타낸다. 이 두 애트리뷰트는 구문상으로는 같은 성질을 가지나 시맨틱상으로는 이

질적이다. 이 이질성은 관련된 애트리뷰트의 정의의 차이에 기인한다.

시맨틱 차이를 규명하는 것은 쉬운일이 아니다. 데이터베이스 스키마는 통상 데이터를 일관성 있게 해석하기 위한 충분한 시맨틱을 제공하지 못한다.

1.2.3 Autonomy

서로 다른 DBMS을 관리하는 운영주체는 보통 독자적이다. 즉, DBMS들이 독자적이고 분리된 통제하에 있다는 것이다. 통합 시스템에 참여하는 시스템들의 자치성에 관한 특성과 또 그들을 통합 시스템 상에서 인식해야 함을 이해하는 것은 매우 중요하다. 자치성에는 다음과 같은 4가지 형태가 있다.

Design 자치성: 시스템 구성에 참여한 DBS이 데이터, 데이터 모델, 질의어, 의미해석, 제약조건, 하부구조 구현 등 모든 분야에서 자신의 마음대로 선택할 수 있는 능력을 말한다.

Communication 자치성 : 시스템 구성에 참여한 한 DBMS가 시스템 구성에 참여한 다른 DBMS와 언제 그리고 어떻게 통신하는가를 결정할 수 있는 능력을 말한다.

Execution 자치성 : 통합 시스템에 참여한 한 DBMS가 다른 DBMS에의해 제기된 외부 오퍼레이션(external operation)으로부터 간섭 받음이 없이 지역적 오퍼레이션을 수행할 수 있는 능력을 말한다. 또한 외부 오퍼레이션을 수행시 그 순서 역시 전적으로 참여 DBMS의 결정에 따른다. 따라서 통합시스템은 참여 DBMS의 명령의 수행순서를 강제로 지시할 수 없다. 참여 DBMS는 외부 오퍼레이션도 지역 오퍼레이션과 마찬가지로 처리하게 된다.

Association 자치성: 이는 참여 DBMS가 기능과 자원을 다른 DBMS와 얼마나 공유하는가 하는 능력으로서 이는 참여 DBMS가 통합에 가입하거나 또는 탈퇴할 수 있는 능력도 포함한다.

그러나 실제로 많은 상황에서는 이러한 자치성들이 모두 제공되지도 않을뿐더러 제공되지 않는 것이 오히려 바람직할 경우도 있다.

1.3 이질 데이터베이스 통합 접근방법

이질 데이터베이스를 통합하는 방법에는 크게 tightly coupled approach, loosely coupled approach, 그리고 multimodel multilingual approach의 세가지로 생각해 볼 수 있다.

1.3.1 Tightly coupled approach

이질 데이터베이스 통합을 이루는 한 가지 방법은 tightly coupled approach 라 하여 전역스키마(통합스키마, global schema, universal schema)를 사용하는 것이다. 전역스키마를 사용하는 시스템에서는 스키마 변환 및 스키마 통합을 통해 전역스키마를 구성한다음 이 전역스키마에 해당되는 데이터 조작언어를 이용하여 질의어를 표현한다. 이러한 연구의 대표적인 예로서는 CCA의 MULTIBASE [Katz 81], UNISYS의 Mermaid [Temp 87] 등이 있다. 이 기법을 이용하면 여러 데이터베이스 스키마를 통합하여 단일의 인터페이스를 제공하게 되므로 위치 투명(Location Transparency)를 얻을 수 있으나 스키마를 통합해야하는 어려움이 있다.

<5단계 스키마 구조>

기존 DBS의 3단계 스키마 구조는 중앙집중형에는 적합하나 통합시스템의 구조를 설명하는데는 적합하지 못하다. 따라서 3단계 스키마 구조는 통합시스템의 3차원 --즉, 분산성, 이질성 자치성--을 지원할 수 있도록 확장되어야 한다. 스키

마 구조 확장의 예를 과거의 연구에서 찾아보면 Mermaid [Temp 87]에서의 4단계 구조, DDTS [Devo 82] 및 SERIUS-DELTA [Litw 82]에서의 5단계 구조등이다. 또한 Sheth [Shet 90]등도 이 5단계 구조를 채택 발전시켜 자신들의 5단계 구조를 제안하였다. 여기서는 Sheth의 5단계 구조를 살펴본다 (그림 2 참조). 그림 3은 프로세서와 스키마로 구성된 시스템 구성을 보여준다. 이 5단계 스키마 구조를 설명하면 다음과 같다.

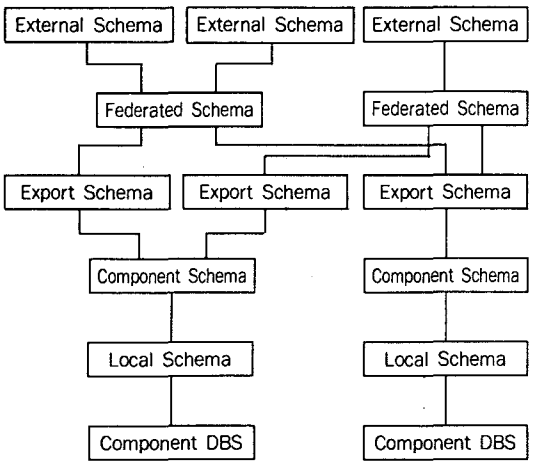

그림 2 5단계 스키마 구조

(1) 지역 스키마 (Local Schema)

: 지역 스키마는 참여 DBS의 개념적 스키마로서 참여 DBMS의 데이터 모델로서 표현된다. 따라서 서로 다른 지역스키마는 서로 다른 모델로서 표현된다.

(2) 구성 스키마 (Component Schema)

: 구성스키마는 지역스키마를 캐노니칼 혹은 공통 데이터모델(CDM: common data model) 이라 불리는 데이터 모델로 변환함으로서 얻어진다. 구성스키마를

CDM으로 정의하는 이유는 첫째, 하나의 표현방법을 사용해서 서로 다른 모든 지역스키마를 나타낼 수 있고, 둘째, 지역스키마에서 잃어버린 의미가 구성스키마에서 되살아날 수 있기 때문이다.

지역스키마로부터 구성스키마로의 스키마 변환 과정은 구성스키마 객체와 지역스키마 객체사이의 매핑으로 이루어진다. 변환 프로세서는 이 매핑을 이용해서 구성스키마상에서의 명령을 지역스키마상에서의 해당되는 명령으로 변환한다. 이러한 변환 프로세서와 구성스키마에 의해 이질적 모습들이 해결된다.

(3) 수출 스키마 (Export Schema)

: 구성 DBS의 모든 데이터가 통합 사용자에게 가용한 것은 아니다. 수출스키마는 구성 스키마중 통합시스템에서 사용 가용한 서브셋을 나타낸다. 예를 들면 특별한 통합 사용자에 의한 사용과 관계되는 액세스 제어 정보 같은 것이다. 수출스키마를 정의하는 목적은 연합자치의 제어 및 관리를 쉽게 하기 위해서이다.

하나의 여과 프로세서 (filtering processor)가 있어 해당되는 구성스키마에 접수된 가능한 오퍼레이션의 셋을 제한함으로서 수출스키마에 명시된 액세스 제어를 제공하는데 사용 수 있다. 이러한 여과 프로세서와 수출스키마는 자치성을 제공한다.

(4) 연합 스키마 (Federated Schema)

: 연합스키마는 여러개의 수출스키마가 통합된 것으로 수출스키마를 통합할때 발생된 데이터 분산에 관한 정보를 포함한다. 프로세서는 연합스키마에 대한 명령을 하나 또는 그 이상의 수출스키마에 대한 명령으로 변환시킨다. 통합시스템에는 여러 개의 연합스키마가 있고 사용자 역시 그 행동에 관련되어 여러 종류가 있다. 예를 들어 모든 관리자는 사용자의 한 클래스가 될 수 있고 모든 고용인과 응용프로그램은 다른 클래스가 될 수 있다.

(5) 외부 스키마 (External Schema)

: 외부스키마는 사용자, 응용프로그램, 또는 사용자 클래스를 위한 스키마이다. 외부스키마를

사용하는 이유는 아래와 같다.

- Customization : 연합스키마는 매우 크고 복잡하며 바꾸기가 쉽지 않으므로 외부스키마는 외부스키마의 사용자와 관련된 정보의 서브셋을 명세하는데 사용된다. 외부스키마는 사용자의 필요에 따라 비교적 쉽게 바꿀 수 있다. 외부스키마를 위한 데이터 모델은 연합스키마에 사용되는 모델보다 더 다양하다.
- 추가적 무결성 제약조건 : 외부스키마를 통하여 무결성 제약조건이 추가적으로 명시된다.
- 액세스 제어 : 외부스키마는 참여 데이터베이스에 의해 관리되는 데이터에 대한 액세스 제어를 제공한다. 뿐만아니라 외부스키마는 통합시스템에 의해 관리되는 데이터에 대한 액세스 제어도 제공한다.

여과 프로세스는 외부스키마에 대한 명령을 연합스키마의 액세스제어와 제약조건에 맞도록 분석한다. 외부스키마가 연합스키마의 모델과 다른 모델로 표현되어 있다면 외부스키마에 대한 명령을 연합스키마에 대한 명령으로 변환하는 변환 프로세서가 요구된다. 대부분의 통합시스템의 프로토타입들은 모든 외부스키마에 대하여 하나의 데이터 모델만 지원한다. 또한 질의어 인터페이스도 Mermaid(SQL과 ARIEL)와 DDTS(SQL과 GORDAS)만 제외하고는 대부분 하나의 인터페이스만 지원한다. 앞으로의 시스템은 멀티모델 외부스키마 및 멀티질의어 인터페이스를 제공할 수 있어야 할 것이다.

1.3.2 Loosely-coupled Approach

이질적인 데이터베이스들을 연합하는 또 하나의 다른 방법은 loosely coupled approach라 하여 전역스키마를 사용하지 않는 대신 강력한 DB 조작 언어를 이용하는 것이다. 이 언어를 Litwin [Litw 88]은 multidatabase language라 불렀다. 대표적인 예로서는 Litwin et al.의 Multidatabase System [Litw 88], 휴스톤 대학의 Omnibase [Rusi 88] 등이 있다. 이러한 언어는 일반 데이터베이스 언어의 능력도 가지고 있어야 함은 물론 관련된 데이터베이스들이 상호작용(interoperable)할 수 있도록 지원해야만 한다.

여기서 상호작용이란 전역스키마의 부재와 자

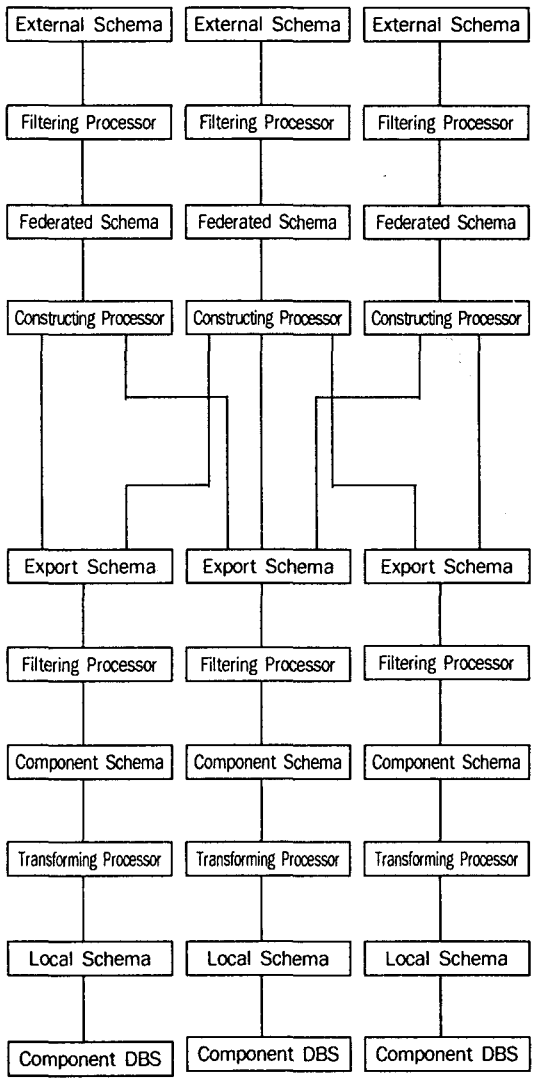

그림 3 시스템 구성도

치성의 제약에도 불구하고 비절차적 오퍼레이션을 통해서 데이터베이스들을 사용할수 있음을 의미한다. 이 기법을 이용하면 스키마 통합을 하지 않아도 되는 편리함이 있는 반면 위치 투명이 제공되지는 않는다.

<참조 구조>

Loosely-coupled 시스템은 준수해야할 구조가 있다. 그림 4는 이러한 구조를 보여주고 있다. 이 그림에서 보는 바와 같이 Loosely-coupled 시스템의 구조는 다음과 같은 여러개의 층으로 구성된다.

- 맨 아래층에는 이미 존재하는 DBS들이 있다
- DBS 층 위에는 통합하고자 하는 데이터베이스들의 개념적 스키마인 멀티데이터베이스 층이 있다. 이 스키마는 실제의 개념적 스키마 일수도 있고 또는 지역적 외부스키마일 수도 있다. 지역적 외부 스키마일때 이를 특히 internal logical 스키마라 한다. 이 개념적 스키마는 특히 서로 다른 데이터 모델을 지원하고 일부 데이터를 은폐한다. 이 계층에서 공통 모델이 요구된다면 이는 각 DBS의 책임이다.
- 멀티데이터베이스 계층은 특히 데이터베이스들의 집합간의 의존도(Dependency)의 정의에 대한 스키마를 포함한다. 의존도는 transitive하거나 혹은 uni or bidirectional하다. 이러한 의존도는 DBA로 하여금 데이터베이스간의 일치, 프라이버시등을 위해 데이터베이스들을 함께 묶는데 사용된다. 전역스키마가 없는 상황에서 의존도는 서로 다른 데이터베이스들간의 데이터 일치를 보장할 수 있는 유일한 툴이다.
- 그 다음 계층에는 외부스키마가 있다. 이 스키마는 데이터베이스의 서브집합을 하나의 통합된 데이터베이스로 나타낸다. 하나의 실제의 데이터베이스는 서로 다른 외부스키마에 동시에 포함될수 있다. 또한 국지적으로도 다루어질 수 있다. 따라서 전역스키마가 있는 것과는 달리 서로 다른 데이터베이스들로부터 데이터가 하나의 데이터베이스로 표현된다 하더라도 데이터베이스간의 적합한 의존도가 선언되지 않으면 데이터의 일치는 보장될 수 없다.

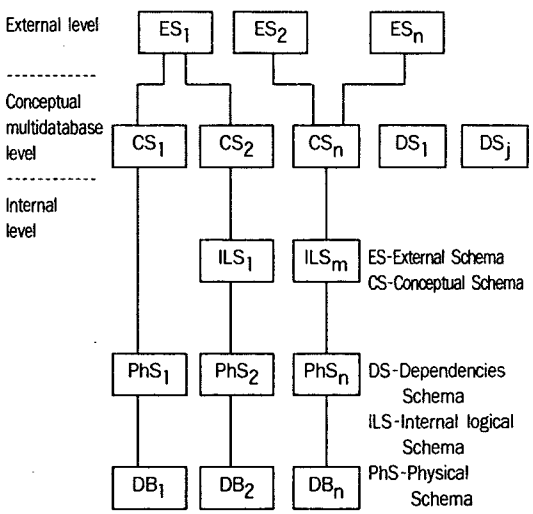

그림 4. 멀티데이터베이스 시스템 참조구조

위의 그림에서 알수 있는 바와 같이 사용자는 다수의 데이터베이스들을 다음과 같은 두 가지 방법으로 액세스한다.

- 멀티데이터베이스 레벨에서 직접 멀티데이터베이스 언어의 기능을 사용해서
- 외부뷰를 통하여 멀티데이터베이스 언어나 데이터베이스 언어를 사용해서

<언어의 기능>

데이터 정의의 자치성을 제공하기 위하여 멀티데이터베이스 언어는 많은 새로운 기능을 제공해야만 한다. 먼저 데이터의 여러 스키마에 들어가기 위한 비절차적 정의--예를 들면 데이터 정의의 export, import 등--를 위한 기능이 요구된다.

또한 데이터가 중복되고 테이블 이름, 구조, 값이 서로 다른 데이터베이스에 위치한 데이터의 비절차적 조작 (검색어, 갱신어)을 위한 기능도 요구된다. 아울러 사용자로 하여금 자치적인 스키마에서의 변화에 무관하도록하는 기능도 요구된다. 기존의 일반 데이터베이스 언어들은 이러한 기능이 결여되어 있다. 멀티데이터베이스 언어의 첫번째 주요 특징은 질의어에 논리적 데이터베이스 이름을 사용하는 가능성에서 나타난다. 이는 이름 충돌을 해결하기 위해 서로 다른 데이터베이스에서 릴레이션을 자격있게 하는 것이다. 일반적인 시스템에 이러한 기능이 없는 이유는 기술적이라기보다는 단지 그러한 것이 구현상 더 쉽기 때문일 뿐이다. 멀티데이터베이스 언어가 갖추어야 할 기능을 살펴보면 다음과 같다.

- 존재하는 데이터베이스들을 멀티데이터베이스로 하기 위한 멀티데이터베이스의 정의와 변경 기능
- 멀티데이터베이스 데이터 정의 : 여러 데이터베이스에 있는 릴레이션창조용의 단순 문장(예 alteration, drop), 데이터 정의의 수입 등- 값 자치성을 위한 데이터

값의 단위와 정확성의 정의- 릴레이션에 대한 정형적인 검색 및 갱신

- 여러 이질 테이블에 대한 오퍼레이션을 수행하는 멀티플 질의어 지원- 데이터 객체의 multiple identification의 가능성
- 이름 자치성을 다루기 위한 데이터 객체간 이질 이름간의dynamic unification의 가능성
- 다이내믹 애트리뷰트 및 다이내믹 애트리뷰트의 갱신 지원- 다양한 built-in 함수들
- 뷰 정의
- 멀티데이터베이스 외부스키마의 정의
- 데이터의 import와 export를 위한 데이터베이스간 질의어- manipulation dependency, equivalence dependency 같은 보조 객체

1.3.3 Multimodel multilingual Approach

이질 데이터베이스 통합의 발전된 형태는 실제로 스카마변환과 언어번역을 통하여 이루어진다 [Hsia 90]. 이 방법은 사용자로 하여금 다른 모델하에서 개발된 데이터베이스라 하더라도 자신의 데이터베이스 스키마와 같은 스키마 환경에서 자신의 데이터베이스 언어로 액세스할 수 있도록 허용하는 것이다. 예를 들어 관계 데이터베이스 사용자가 계층구조 혹은 네트워크 데이터베이스를 액세스하기 위해 SQL을 사용할 수 있다. 그림 5는 이러한 이질 모델간 액세싱 개념을 보여주고 있다.

즉, 그림 5에서 관계형 데이터베이스 사용자는 효과적인 질의어 작성을 위해 스키마 변환기를 이용하여 계층 데이터 베이스의 스키마를 자신의 데이터베이스와 같은 형태인 관계 데이터 베이스 형태로 변환하여 인식한후 관계형 질의어인 SQL로 질의어를 작성한다. 그러면 이 SQL 질의어는 언어 번역기를 통하여 계층 데이터베이스의 질의어인 DL/1으로 바뀌어 계층 데이터베이스를 액세스할 수 있게 된다. DB

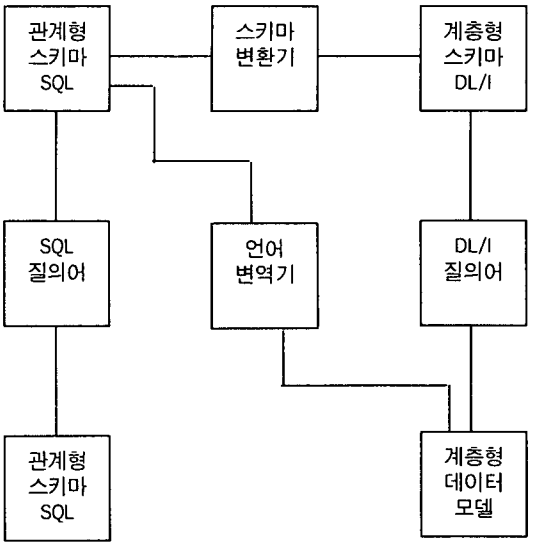

그림 5. 이질 모델간 액세싱