이질데이터베이스통합 (Ⅲ)

(Integration of Heterogeneous Distributed Databases)

나 민 영
육군사관학교 전산학 교수
Ra, Min-Young. Ph.D.
Assistant Professor Dept. of
Mathematics & Computer Science
Korea Military Academy

차 례

3 이질 분산 트랜잭션 관리

3.1 이질 분산 데이터베이스 트랜잭션 모델

3.1.1 데이터베이스 트랜잭션

통합된 이질 데이터베이스 시스템은 서로 다른 여러 지역 데이터베이스들을 관리하는 여러개의 지역 DBMS들은 각 장소에서 수행되는 트랜잭션이 다음과 같은 ACID 성질을 만족한다고 가정한다.

Atomicity : 트랜잭션의 모든 연산들은 데이터베이스에 적절하게 반영되거나 아니면 하나도 반영되지 않는다.

Consistency : 독자적인 트랜잭션의 실행은 데이터베이스의 일관성을 유지해야 한다.

Isolation : 각 트랜잭션은 시스템내에서 독자적으로 수행되고 지역 DBMS은 트랜잭션의 중간 결과가 동시에 수행되는 다른 트랜잭션과는 무관하다고 가정한다.

Durability : 트랜잭션에 의해 변경된 값들은 트랜잭션의 성공적 수행후 지속적으로 유지되어야 한다.

이질 데이터베이스를 위한 트랜잭션은 이러한 ACID성질을 만족하는 기본 트랜잭션이든가 아니면 이질 데이터베이스 환경의 특별한

요구 사항을 다루기 위한 확장된 트랜잭션이다. 트랜잭션 시맨틱의 확장은 새로운 트랜잭션이 갖추어야 할 다음과 같은 추가적인 특징을 첨가함으로서 이루어진다.

Function replication : 한 트랜잭션에 의해 주어진 목표는 기능적으로 동등한 트랜잭션에 의해 지역 시스템에서 달성될 수 있다.

Mixed 트랜잭션 : 어떤 트랜잭션들은 commit된 후에 보상 서브 트랜잭션에 의해 시맨틱 상으로 'undone' 될 수도 있다. 이는 전역 트랜잭션이 commit되기 전에 일부 서브 트랜잭션이 commit될 수 있음을 의미한다. 이와 같이 보상/무보상 서브 트랜잭션의 결합된 트랜잭션을 mixed트랜잭션이라 한다. 이는 트랜잭션의 isolationt성질을 완화시킨 것이다.

시간 트랜잭션 : 트랜잭션 수행을 위한 완료시간을 명시해준다. 이 정보는 전역 트랜잭션 수행 스케줄을 정하기에 유용하다.

본 고에서는 기본 트랜잭션 모델에 근거하여 설명하기로 한다. 통합된 이질 데이터베이스 시스템은 각 지역 DBMS를 다른 DBMS나 통합시스템에 관한 지식이 없이 자치적으로 작동되는 블랙박스로 간주한다. 지역 가치는 통합된 이질 데이터베이스 시스템을 재래식 분산데이터베이스시스템으로부터 구분해주는 주요 특징이다. 자치에는 다음의 세가지 유형이 있다.

설계 자치(Design Autonomy) : 통합시스템에 부합하기 위해 지역 DBMS 소프트웨어에 수정을 가하지 않는다.
기존의 소프트웨어의 변화는 비용이 매우 들고 성능면에서 하락을 가져올뿐더러 더 나아가 이미 개발되어 있는 응용프로그램들을 못쓰게 만들 수도 있다.

실행 자치(Execution Autonomy) : 각 지역 DBMS는 자기 장소에서 수행되는 트랜잭션에 대해 온전한 통제를 할 수 있어야 한다. 이는 즉 지역 DBMS는 트

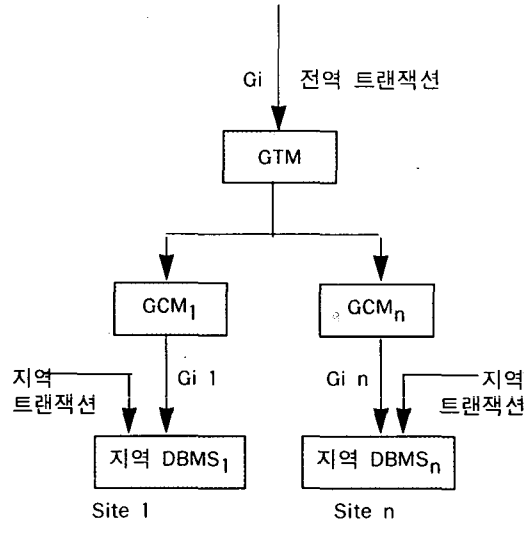


그림 1. 통합된 이질 데이터베이스 트랜잭션 모델

랜잭션 수행 도중 언제고 이를 중지시킬 수 있어야 함을 의미한다.

통신 자치(Communication Autonomy) : 통합시스템에 의해 통합된 지역 DBMS는 여러 장소에서 수행중인 전역 트랜잭션의 행위와 협력할 수 없다.

이 제약 조건은 지역 DBMS는 자신의 제어 정보를 다른 지역 DBMS나 통합시스템과 공유 하지 않음을 의미한다.

시스템에 참여하는 DBMS들은 각기 다른 자치 수준을 가질 수 있다.

예를 들어 어떤 사이트는 전역 트랜잭션의 협동에 참여하는(낮은 통신 자치)반면 다른 사이트에서는 그렇지 않을 수도 있다(높은 통신 자치).

3.1.2 통합된 이질 데이터베이스 트랜잭션 모델

이질 데이터베이스 통합시스템은 사이트 S1, S2, …, Sm에 위치한 이미 존재하는 자치적인 여러 지역 DBMS들로 구성된다. 트랜잭션 Gi는 commit나 abort 연산에 의해 종료되는 read와 write연산의 연속이다.

각 지역 DBSDMS 지역 트랜잭션이나 전역 트랜잭션의 서브트랜잭션을 받아 수행시킨다.

다음 그림 1은 이와 같은 통합된 이질 데이터베이스의 트랜잭션 모델을 보여준다.

통합된 이질 데이터베이스 환경은 다음 두 형태의 트랜잭션을 지원한다.

지역 트랜잭션(local transaction) : 단일 DBMS에 의해 관리되는 데이터를 액세스하는 트랜잭션

전역 트랜잭션(global transaction) : 통합시스템 통제하에 수행되는 트랜잭션. 전역 트랜잭션은 여러 트랜잭션으로 구성되는데 각각의 서브 트랜잭션은 그 서브 트랜잭션이 수행되는 DBMS의 입장에서 보면 보통의 지역 트랜잭션이 된다.

사이트 sk에서 지역 스케쥴 Sk는 사이트 sk에서 수행되는 얻어지는 지역 트랜잭션과 전역 트랜잭션의 연산의 연속이다. 트랜잭션 Gi는 만일 Sk가 ci(또는 ai)연산을 포함하고 있으면 Sk에서 commit(또는 abort)되었다고 한다. 트랜잭션 Gi는 Sk에서 commite되거나 abort되지 않았을때 active하다고 한다.

통합시스템 소프트웨어는 그림 1에서 보는 바와 같이 기존재하는 지역 DBMS상에서 수행되는 것으로 전역 트랜잭션 관리기 (Global Transaction Manager:GTM)과 각 지역 DBMS와 연관되는 여러 전역 통신 관리기(Global Communication Manager:GCM)로 구성된다. 각 전역 트랜잭션은 GTM에 연산을 보낸다. 보내진 각 연산에 대해 GTM은 수행될 장소로 보낼 것인지, 중지시킬 것인지, 또는 지연시킬 것인지를 결정한다. 연산이 보내지면 GTM은 그 연산이 수행되어질 장소를 결정한다. 각 보내진 ri(x)wi(x)연산에 대해 GTM은 전역 데이터 요소인 x의 복제품을 갖고 있는 모든 사이트를 결정한다. 전역 ri(x)에 대해 GTM은 read연산이 수행될 사이트로서 하나를 선정하고 ri(x)를 지역 read연산으로 전

환한다. 전역 Wi(x)연산에 대해서는 모든 사이트가 전역 write연산의 수행중 사용되어야 한다. 일반적으로 각 전역 데이터 요소는 각 지역사이트에서 오직 하나의 지역 데이터 요소와 상응된다고 가정한다. GTM이 전역 트랜잭션 연산을 지역 DBMS로 보내는 작업은 GCM을 통하여 이루어진다.

이 통신 관리기는 GTM과 지역 DBMS간의 연락관 역할을 수행한다. 전역 서브 트랜잭션에 속한 연산은 GCM에 의해 단일 트랜잭션으로서 지역 DBMS에게 보내진다. 각 지역 DBMS는 GCM을 통해 응답 회신을 보낸다고 가정한다.

3.2 이질 분산 데이터베이스 트랜잭션 관리상의 이슈들

GTM은 GTM이 모르는 지역 트랜잭션이 있는 경우에 있어서도 전역 트랜잭션의 ACID 특성을 보장해야 한다. 또한 GTM은 전역 트랜잭션의 deadlock free 수행을 보장해야 하고 어떠한 형태의 시스템 장애시에도 회복시킬 수 있는 수단을 제공해야만 한다. GTM이 당면한 문제는 다음과 같다

(1) Global Serializability 문제

여러 지역 DBMS들은 서로 다른 동시성 제어 프로토콜을 사용할 수 있다. 예를들면 2PL, timestamp, 또는 serializa-tion graph ordering 등이다. Global serializibility를 보장해 주기 위한 기존 해결책은 각 사이트는 같은 동시성 제어 스킴을 사용하고 제어 정보를 공유함을 가정한다. 따라서 이러한 방법들은 통합된 데이터베이스 시스템 환경에서는 사용할 수 없다.

지역 트랜잭션이 GTM의 통제 밖에서 수행되므로 GTM은 오직 전역 트랜잭션의 실행 순서의 통제를 통하여 global serializability를 보장할 수 있다. 그러나 이러한 환경하에서는 전역 트랜잭션의 serial 수행이라 하더라도 global serializability를 보장해 주지 못한다.

(2) Global Atomicity and Recovery 문제

Global atomicity 요구는 트랜잭션의 서브 트랜잭션이 모두 commit되거나 모두 abort됨을 의미한다. 동질형의 분산 데이터베이스 시스템에서는 트랜잭션의 atomicity가 'atomic commit 프로토콜'에 의해서 보장된다.

이 프로토콜은 각 참여 지역 사이트가 각 서브 트랜잭션에 대하여 prepared state를 제공할 것을 요청한다. 서브 트랜잭션은 coordi-nator가 그 트랜잭션을 commit또는 abort 할때까지 prepared state상태로 유지되어야 한다.

만일 참여 지역 DBMS 각각의 실행 자치가 유지되기 원한다면 지역 DBMS들은 트랜잭션의 prepared state를 수출하지 않을 것이라고 가정해야 한다.

그렇다면 이러한 환경 하에서는 DBMS는 서브트랜잭션을 그 commit에 어느 때고 일방적으로 abort시킬 수 있다. 이 때문에 전역 트랜잭션이 atomic이 되지 않을뿐만 아니라 정확하지 않은 전역 스케줄을 야기할 수도 있다.

(3) Global Deadlock 문제

각 지역 DBMS가 지역 serializabilty를 보장

하기 위하여 록킹 메카니즘을 사용하는 통합된 이질 데이커베이스 시스템을 생각해 보자.

각 지역 DBMS지역 deadlock을 발견하고 회복하기 위한 메카니즘을 가지고 있다고 가정한다.

그러나 그러한 시스템에서는 GTM에 의해 발견될 수 없는 전역 deadlock이 발생할 가능성이 있다.

3.3 문제 해결방안

3.3.1 Global serializability 문제

Global serializability는 전역 트랜잭션 처리의 정확성을 결정짓는 핵심적인 척도이다.

이에 대한 해결책은 쉬운 문제가 아니며 그 연구가 지금도 이루어지고 있다. 여기서는 그 중 두 가지를 소개한다.

<Stub process approach>

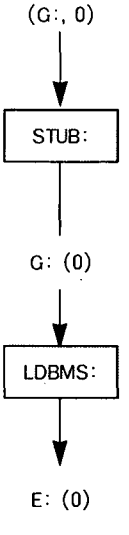


그림 2. Stub프로세서의 이용

이 approach에서는 전역 DBMS는 전역 트랜잭션을 각 지역 DBMS로 보내기 전에 전역 트랜잭션의 serialization order를 결정한다. 지역 DBMS입장에서 보면 전역 순서를 지키는 문제는 기정의된 순서를 지닌 동시성 제어 문제로 볼 수 있다.

이 전역 serialization orderds는 각 지역 사이트에서 서로 다른 기법에 의해 지켜진다.

이 방법의 기본 아이디어는 주어진 순서 요구가 단순히 전역 서브 트랜잭션을 지역 DBMS로 보내는 것을 통제함으로 지켜질 수 있다는 것이다.

여기서는 stub 프로세스를 사용해서 지역 레벨에서 전역 서브 트랜잭션의 보냄을 통제하는 프로토콜을 다룬다. 이 개념은 다음 그림 2와 같이 표현된다.

여기서 0는 통합 시스템에 의해 결정된 순서를 나타내고, Gi(0)는 전역 트랜잭션 Gi의 서브 트랜잭션의 셋을 나타내는데 이때 이 셋은 특정 순서를 갖추고 지역 DBMS에 보내진다.

Stub 프로세스는 실제적으로 통합시스템과 지역 DBMS 사이의 인터페이스로서 작용한다. 통합시스템은 전역 서브 트랜잭션을 stub 프로세스로 보낸다. 전역 서브 트랜잭션을 모아서 지역 DBMS로 다시 보내는 작업은 stub프로세스의 역할이다. 반면에 지역 DBMS는 전역 서브 트랜잭션의 수행에 관해 stub프로세스와 교통한다.

개념적 레벨에서 볼때 stub프로세스는 전역 서브 트랜잭션과 계획과 순서를 입력으로 받아 보는데 정책을 출력으로 내보내는 하나의 함수로 생각될 수 있다.

이와 같은 Stub 프로세스를 이용하면 global serializability를 보장할 수 있다. 즉, stub 프로세스가 서브 트랜잭션을 보낼때 이전의 모든 서브 트랜잭션이 serialization event에 도달했을때에만 보내면 된다. 이때 serialization event는 지역 DBMS에서 사용되는 동시성 제어 기법에 따라 약간씩 차이가 있다.

<Optimistic Ticket Method (OTM)>

이 방법은 서브 트랜잭션의 상대적 serialization order를 결정하기 위해 티켓을 사용하는 기법이다. 티켓은 논리적 timestamp로서 그 값은 보통의 데이터 요소로서 각 지역 DBMS에 저장된다.

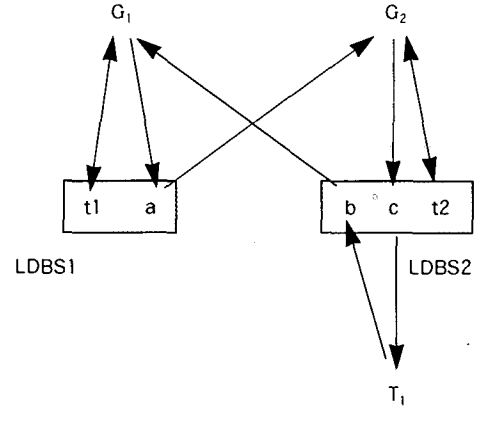
OTM은 지역 DBMS 자치를 위배하지 않고 만일 참여 DBMS 가 local serializability를 지킨다면 global serializability를 보장해준다. 이 방법은 다음의 두 물음에 대한 해답을 제공할 수 있다.

(1) 어떻게 통합 시스템이 전역 트랜잭션의 서브 트랜잭션의 상대적인 serialization order에 관한 정보를 얻을 수 있는가?
(2) 어떻게 통합 시스템이 전역 트랜잭션의 서브 트랜잭션이 모두 참여 DBMS에서 같은 상대적 serialization order를 가짐을 보장할 수 있는가?

OTM 기법에서는 전역 트랜잭션의 서브 트랜잭션은 Take-a-Ticket 연산을 취하도록 요구되어 진다. Take-a-Ticket 연산은 지역 자치를 위배하지 않는다. 왜냐하면 지역 시스템에 어떠한 변화로 요구하지 않기 때문이다. 오직 전역 트랜잭션의 서브 트랜잭션만 티켓을 갖는다. 지역 트랜잭션은 아무런 영향이 없다.

다음 그림은 티켓 사용의 예를 보여 준다. 지역 DBMS 1과 2에서 티켓 데이터 아이템은 각각 t_1과 t_2로 표현된다. LDBS1에서 G_1과 G_2의 서브트랜잭션에 의해 얻어진 t_i값들은 상대적인 serialization order를 반영한다. 이 스케쥴은 LDBS1에서 지역 동시성 제어기에 의해 허용될 것이다. LDBS2에서는 지역 트랜잭션 T_1이 $G_2 \rightarrow T_1 \rightarrow G_1$과 같은 간접 conflict를 유발한다. 그러나 서브트랜잭션으로 하여금 티켓을 취하도록 요구함으로써 $G_1 \rightarrow G_2$와 같은 부수적인 conflict를 요구하게 된다. 이 부수적인 티켓 conflict가 LDBS2에서의 수행이 지역적으로 non-serializable하도록 한다. 그러므로 LDBS2에서의 스케쥴은 지역 동시성 제어에 의해 허용될 수가 없다. 다음 그림 3은 이러한 티켓 사용의 효과를 보여 주고 있다.

위 그림에서 G_1 과 G_2는 전역 트랜잭션을 가리키고 T_1은 지역 트랜잭션을 가리킨다. 데이터 아이템 a를 읽음을 나타내고 반대 방향 화살표는 쓰기를 나타낸다.



LDBS1 : $RG_1(t_1)WG_1(t_1+1)$
$WG_1(a)RG_2(t_1)WG_2(t_1+1)$
$RG_2(a)$, i. e., $G_1 \rightarrow G_2$

$LDBS_2$: $WT_1(b)RG_1(t_2)WG_1(t_2+1)RG_1(b)RG_2(t_2)$
$WG_2(t_2+1)WG_2(c)RT_1(C)$, i.e.,
$G_2 \rightarrow T_2 \rightarrow G_1$

그림 3. 티켓 방법의 효과

이와 같이 티켓을 이용하면 상대적인 serialization order를 결정할 수가 있으므로 global

serializability 문제를 해결할 수 있다.

3.3.2 Global atomicity 문제

통합된 이질 데이터베이스 시스템에서는 동질형 시스템에서와 마찬가지로 실패는 트랜잭션의 중지, 시스템 failure, GTM의 failure, 통신 failure등으로부터 유발된다. 또한 통합된 이질 데이터베이스 시스템에서는 각 지역에서의 전역 트랜잭션은 보통의 DBMS연산(예를 들면 지역 deadlock)의 결과로서 통합 시스템에 의해 중지될 수 있고 같은 트랜잭션이 다른 지역에서 commit될 수 있다.

통합된 이질 데이터베이스 시스템에서의 회복 프로시쥬어는 GTM이 abort나 failure로부터 모두 회복 가능해야 함을 의미한다. 각 지역 DBMS에서 회복 프로시쥬어는 지역 트랜잭션과 전역 트랜잭션의 atomicity와 durability를 보장하므로 분산된 시스템에서 트랜잭션의 atomicity와 durability를 보장하기 위한 작업은 각 전역 트랜잭션이 모든 사이트에서 commit하거나 또는 사이트에서 abort하는 작업으로 압축될 수 있다.

GTM 회복 프로시쥬어의 설계에 영향을 주는 요소는 지역 DBMS에 의해 제공되는 인터페이스이다. 최근에 지역 DBMS들이 prepare-to-commit 연산을 제공하는 가는 계속 논란의 대상이 되고 있다. 만일 각 지역 DBMS들이 그러한 연산을 제공한다는 atomicity를 유지하는 작업은 상대적으로 쉬워진다. 왜냐하면 2PC 프로토콜 같은 atomic 프로토콜이 사용될 수 있기 때문이다. 반면에 지역 DBMS가 prepare-to-commit연산을 지원하지 않으면 한 전역 트랜잭션이 한 사이트에서는 commit되고 다른 사이트에서는 abort되는 일이 발생한다. 이러한 경우 전역 트랜잭션의 atomicity를 제공하기 위해 제안된 다음의 세가지 메카니즘을 살펴본다.

redo : failed 서브트랜잭션의 쓰기는 서브 트랜잭션에 의해 수행되는 모든 쓰기 연산으로 구성되는 redo 연산을 수행시킴으로서 install 된다.

retry : 쓰기 연산뿐만 아니라 중지된 서브 트랜잭션 자체가 다시 수행된다.

compenasate : 전역 트랜잭션의 서브트랜잭션이 commit한 각 사이트에서 commit된 서브 트랜잭션의 효과를 undo하기 위해 보상의 서브 트랜잭션이 수행된다.

이 세가지중 redo와 retry기법은 트랜잭션의 표준 atomicity를 보장해준다.

그러나 compensate의 경우에는 중지된 전역 트랜잭션의 효과가 다른 트랜잭션으로 구체화될 수 있으므로 좀 약한 의미와 atomicity가 사용된다.

4 요약및결론

이미 만들어져 운용되고 있는 서로 다른 데이터베이스들은 논리적으로 합하여 하나의 통합 시스템을 구축함으로서 데이터를 서로 공유하는 것은 차세대 데이터베이스 응용중 매우 중요한 분야이다. 이때 이 데이터 자원들은 서로 연관이 없고 나중에 서로 통합된다는 점을 고려하지 않고 독자적으로 만들어졌다는 점에서 pre-existing이다. 본 고에서는 이러한 이질적이고 분산된 또한 이미 운용중에 있는 데이터베이스의 통합에 관하여 통합구조 방법

론, 통합 핵심기법, 그리고 통합된 시스템에서의 트랜잭션 관리 등을 중심으로 살펴본다.

스키마 변환기법과 통합기법은 통합시스템을 구축하기 위한 기초 기법으로서 효과적인 스키마 변환 및 통합을 위하여는 통합의 순서, 통합절차, 그리고 충돌분석 및 해결등이 고려되어야 한다. 또한 이질 분산 통합시스템이 구축되어 이상없이 사용되려면 트랜잭션 관리상의 문제들, 즉 global serializability 문제, global atomicity문제 등이 반드시 해결되어야 한다. 이러한 이질 데이터베이스 통합에 관한 연구는 앞으로 실생활에 유용하게 적용될 결과가 나올때까지 계속 수행되어야 할 중요 과제라 생각된다.

참 고 문 헌

[Bars 92] Barsalou, T., and Gangopadhyay, D., "M(DM) : An Open Framework for Integration of Multimodel Multidatabase Systems, " Proc. of International Conf. on Data Engineering, 1992.

[Bati 86] Batini, C., Lenzerini, M., and Nabathe, S. B., "A Comprehensive Analysis of Methodologies for Database Schema Integration," ACM Computing Surbeys, Vol. 18, No. 4, December 1989.

[Bell 92] Bell, D., and Grimson, J., Distributed Database Systems, Addison Wesley, 1992.

[Brei 92] Breitbart, Y., Garcia-Molina, H., and Silberschatz, A., "Overview of Multidatabase Tranaction Management, " Technical Report HP-82-273, Honeywell Compuetr Sciences Center, Camden, MN, 1982.

[ElDu 90] Elmagarmid, A. K., Litwin, L. W., and Rusinkiewicz, M., "A Multidatabase Tranaction Model for InterBase, " Proceedings of the 16th VLDB Conference, 1990.

[Elma 84] Elmasri, R., and Nabathe, S. B., "Object Integration in Logical Database Design, " Proc. of the First International Conf. on Data Engineering, April 1984.

[Elma 89] Elmasri, R., and Nabathe, S. B., Fundamentals of Database Systems, Benjamin/Cummings Publishing, 1989.

[Geor 91] Georgakopoulos, D., and Rusinkiewicz, M., "On Serializability of Multidatabase Transactions Through Forced Local Conflicts," Data Engineering. 1991.

[Herz 92] Herzig, R., and Gogolla, M., "Transforming Conceptual Data Models into an Object Model," The 11th ER Conference, 1992.

[Hsia 90] Hsiaom D. K., Kamel, M. N., and Wu, C. T., "The Federated Databases and System : A new Generation of Advanced Database Systems," Database and Expert Systems Application, 1990.

[Katz 81] Katz, R., Goodman, N., Landers, T., Smith, J. M., and Yedwab, L., "MULTIBASE - A System for Integrating Heterogeneous, Distributed databases, " Computer Corporation of America, Technical Report 81-06, May 1981.

[Lim 94] Lim, E., Srivastava, J., and

Shekhar, S., " Resolving Attribute Incompatibility in Database Integraion, " Proc. of International Conf. on Data Engineering, 1994.

[Litw 82] Litwin, W., et al., "SIRIUS Systems for Distributed Data Management," In Distributed Databases, H.J. Schneider, Ed. North-Holland, The Netherlands, 1982.

[Litw 88] Litwin, W., Mark, L., and Roussopoulos, N., "Interoperability of Multiple Autonomous Databases," Technical Report, 1988.

[Meng 93] Meng, W., et al., "Construction of a Relational Front-end for Object-Oriented Database Systems," Proc. of International Conf. on data Engineering, 1993.

[Ozsu 91] Ozsu, M. T., and Valduriez, P., Principles of Distributed Database Systems, Prentice Hall, 1991.

[Perr 91] Perrizo, W., "HYDRO : A Heterogeneous Distributed Database System," SIGMOD, 1991.

[Rusi 88] Rusinkiewicz, M., Czejdo, B., Elmasri, R., Georakopoulos, D., Jamoussi, A., Karabatis, G., Li, Y., Loa, K., Gilbert, J., and Musgrove, R., "Query Processing in OMNIBASE-A Loosely Coupled Multidatabase System, " Technical Report UH-CS-88-05, University of Houston, February 1988.

[Shet 88] Sheth, A. P., Larson, J. A., Cornelio, A., and Nabathe, S. B., "A Tool for Integrating Conceptual Schemas and User Biews, Proc. of International Conf. on Data Engineering. 1988.

[Shet 90] Sheth, A., and Larson, J., " Federated Database Systems for Managing Distributed, Heterogeneous, and Autonomous Databases, " ACN Computing Surverys, 22(3), September 1990.

[Shet 92] Sheth, A., and Marcus, M., "Schema Integration : Methodology, Techniques, and Prototype Toolkit," Bellcore Special Report SR-STS-002211, February 1992.

[Song 93] Song, I., and Godsey, H. M., "A Knowledge Based System Converting ER Model into an Object-Oriented Database Schema, " Proc. of the 3rd International Symposium on Database Systems for Advanced Applications, 1993.

[Temp 87] Templeton, M., Brill, D., Dao, S. K., Lund, E., Ward, P., Chen, A. L. P., and Macgregor, R., "Mermaid - A Front-end to Distributed Heterogeneous Databases, " Proceedings of the IEEE, Vol. 75, No. 5, May 1987.

[나민영 94] 나민영, "이질 데이터베이스의 상호운용을 위한 구조," '94동계 데이터베이스 학술대회 논문집, 1994.