대한산업공학회/한국경영과학회 2000 춘계공동학술대회 논문집. 2000년4월21-22일. 경남대학교
Session PB9.5

다중 웹 데이터베이스에서 SQL 질의 처리를 위한 가상 처리장치 아키텍처 설계

Designing The Architecture of A Virtual Processor for Processing SQL Queries Using Multiple Web Databases

성동훈, 최덕원
성균관대학교 산업공학과

Abstract

인터넷의 발빠른 성장으로 모든 비즈니스가 웹에 집중되고 있다. 현재 웹에서 접할 수 있는 사이트들의 대부분은 하나의 DB에서 모든 작업을 처리하고 있는 실정이다. 데이터베이스들이 각각의 다른 지역 DBMS에 흩어져 있고 상이한 DBMS 테이블 사이에 하나 이상의 relation이 존재할 경우에는 한번의 질의로 서로 다른 DBMS로부터 트랜잭션을 처리하는데는 많은 문제가 발생할 수 있다.

일반적으로, 데이터베이스는 한 지역에서 하나의 DBMS에 의해 관리되지만, 효율적인 관리를 위해서 다른 지역으로 분산되어 존재하기도 한다. 더 나아가 독립된 데이터베이스를 서로 다른 지역에서 다수의 DBMS로 관리하는 다중 데이터베이스 시스템을 이용하기도 한다. 다중 데이터베이스 시스템을 이용하면 각 지역적으로 독립적인 관리 전략을 세울 수 있다는 장점을 가지고 있지만, 트랜잭션을 처리하기 위해서 더 복잡한 질의 처리 시스템을 필요로 하게 된다. 이것은 데이터베이스의 무결성과 동시처리 성능으로 직결되는 중요한 문제이다.

본 논문에서는 기존의 C/S 시스템을 이용한 다중 데이터베이스 시스템 설계를 웹으로 확장시켜 무결성 제약사항을 유지하고, 동시처리를 가능하도록 가상 처리장치(virtual processor) 아키텍처를 설계하는 데 그 목적이 있다.

1. 서 론

데이터의 저장소인 데이터베이스에 관한 연구는 웹의 발전과는 상관없이 많은 관심속에 활발히 진행되어 왔다. 이러한 데이터베이스는 단일 DBMS 안에서 하나의 데이터베이스로 존재하거나 여러 데이터베이스로 분산되어 존재하고 있는 실정이며, 시간이 흐를수록 더욱 많은 데이터를 보존하고 관리할 목적으로 다중 DBMS에 분산하여 처리하고 있다. 이렇게 데이터를 한 곳에 저장하지 않고 여러 곳으로 분산시켜 저장할수록 많은 제약사항을 고려해야 하며 더 복잡한 트랜잭션 처리를 수행하기 위한 방법들이 필요하게 되었다.

인터넷이라는 거대 네트워크는 모든 사람들의 관심을 한 몸에 받은지 오래되었고, 생활의 모든 부분이 여기에 집중되고 있다. 이러한 이유로 개인이나 기업의 업무환경이 시·공간의 제약을 받지 않고 웹을 통해서 원하는 작업을 수행할 수 있도록 설계되어야만 경쟁에서 살아남을 수 있는 세상으로 변하게 되었다.

이제 데이터베이스의 트랜잭션 처리는 제한된 공간에서 제한된 사용자에 의해 수행되고 관리되는 지역적인 개념에서 웹을 통해 모든 사람들에 의해서 수행되고 관리되는 전역적인 개념으로 확장되고 있다. 이와 같이 웹을 통해서 트랜잭션을 처리하는 방법은 웹 데이터베이스를 얼마나 효율적으로 이용할 수 있는지를 결정하는 중요한 작업으로 인식되고 있다.

본 논문에서는 웹 상에서 대량의 트랜잭션 처리를 하기 위해서 다중 웹 데이터베이스를 이용하며, 무결성 제약사항을 유지하면서 SQL 질의를 처리하기 위해 가상 처리장치(virtual processor : VP로 약칭) 아키텍처를 설계하는 데 그 목적이 있다. VP는 웹 상에서 요청되는 트랜잭션이 다중 웹 데이터베이스로 넘어가기 전에 데이터베이스의 무결성 여부를 검사하고 동시처리를 위해 잠금 방법을 통한 제어를 담당하게 된다.

본 연구에 앞서 기존의 C/S(client/server) 시스템에 기반한 데이터베이스 시스템과 웹 데이터베이스 시스템간의 비교와 VP를 구현할 웹 프로그래밍 언어인 ASP에 대해서 자세히 알아보기로 한다.

2. 관련 연구

웹과 데이터베이스를 연동시켜 트랜잭션을 처리하기 위해서 사용될 수 있는 언어들에는 CGI(common gateway interface), PHP(hypertext preprocessor), JAVA, ASP(active server page) 등의 프로그래밍(스크립트) 언어와 분산 컴퓨팅 기능을 제공하는 DCOM(Distributed Component Object Model), 그리고 COM 개념에 근거를 둔 ActiveX 기술 등이 있다.

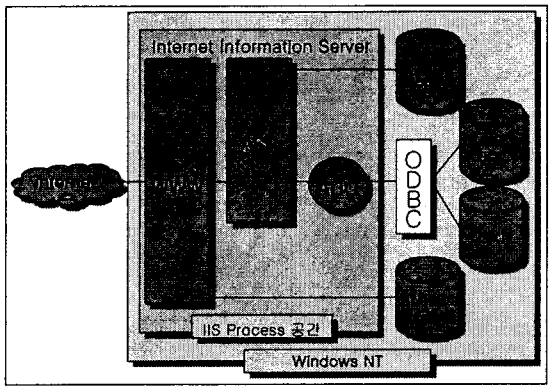
CGI 프로그램은 클라이언트에서의 요청이 발생하면 작업을 위한 프로세서가 생성이 되는데, 이것은 요청하는 클라이언트의 수만큼 발생하므로 서버의 부하를 증가시킨다. 따라서 서버의 성능저하와 직결되는 단점을 가지고 있다.

분산 환경을 제공하는 DCOM은 COM에서 확장되었고, Windows NT 4.0에 와서야 제대로 구현되었다. 그리고 Microsoft의 액티브 플랫폼의 좀 더

하부에서 소프트웨어 객체간의 인터페이스를 제공하며 연관 관리(software plumbing), 즉 연결 고리 역할을 하는 ActiveX에 의해서 지원된다. ActiveX 기술은 많은 사용자 인터페이스를 제공하는 반면, 초기 개발작업이 어려우며, 클라이언트가 서버의 데이터베이스와 작업할 때 COM 객체와의 메시지 교환을 위해 많은 API 함수를 호출하기 때문에 트랜잭션 양이 증가하면 네트워크 트래픽이 많이 발생할 수 있다.

ASP와 PHP는 기존에 클라이언트에서 실행되던 스크립트를 서버측에서 실행되도록 하는 기능을 제공하며 소스 코드를 클라이언트가 볼 수 없는 장점이 있다. PHP는 CGI와 같이 프로그래밍에 사용되는 언어의 이식성이 다른 언어들보다 뛰어나며, 주로 C 언어로 작성되어 대부분의 웹서버에서 실행된다.

ASP의 장점은 같은 작업에 대해서 클라이언트의 요청이 여러번 행해지더라도 하나의 프로세서를 생성시켜 서버쪽의 부하를 줄일 수 있으며, COM을 준수하는 외부 언어에 의해 만들어진 컴포넌트들과의 인터페이스를 제공하는 장점을 가지고 있다. 단적인 예로는 ODBC를 이용한 DBMS와의 연결에 ADO(activeX data object)를 사용할 수 있도록 인터페이스를 제공한다.



<그림 1> IIS에서의 ASP 작동 원리

<그림 1>은 ASP가 ODBC와 연결하기 위해 ADO를 사용하고 있음을 보여주고 있다. 이러한 작동 환경은 C/S 시스템 환경과 약간의 차이점을 보이는데, 이것에 관해서는 다음 절에서 살펴보기로 한다.

이상에서 살펴 본 바와 같이 본 논문에서는 VP 아키텍처 설계를 위해 NT platform에서 운영(UNIX나 LINUX 플랫폼에서 운영할 수 있는 프로그램도 있다)되는 ASP를 사용하며, ASP를 운영하기 위한 웹 서버는 IIS (Internet Information Server) 5.0을 사용한다.

3. VP와 C/S 시스템의 작업환경

기존의 C/S 시스템 작업환경이 Web 환경으로 옮겨가면서 애플리케이션의 제작에 많은 변화를 가져왔다. 기존의 C/S 시스템에서의 애플리케이션들은 클라이언트 측 프로그래밍에 많은 노력을 요구하였고, 클라이언트가 많은 작업을 하도록 설계되었다. 하지만, 웹 상에서의 시스템 구축은 C/S 시스템을 구축하는 것에 비해 적은 비용과 노력으로 쉽게 구현할 수 있으며 클라이언트는 별도의 접속 S/W를 필요로 하지 않는다.

구체적으로 트랜잭션 처리를 위해 웹 VP를 이용 할 때와 C/S 시스템을 이용할 때의 장·단점을 살펴보면 <표 1>과 같다.

구분 / 비고	VP	C/S
장 점	-작업을 위한 시·공간의 제약을 받지 않는다. -고객 지향적인 서비스 제공이 쉽다. -프로그래밍이 쉽다. -클라이언트 애플리케이션이 특별히 필요하지 않다.	-일반 사용자들이 이용할 수 없으므로 시스템 보안 및 관리가 용이하다. -클라이언트/서버간의 작업 분산이 용이하여 서버의 부하를 줄일 수 있다.
단 점	- 사용자 접근이 빈번하므로 데이터베이스의 관리가 어렵다. - 보안상의 문제가 발생할 수 있다. - 서버 부하가 증가할 수 있다.	- 작업을 위한 시·공간적인 제약이 따른다. - 클라이언트는 작업을 위해 애플리케이션이 필요하다. - 애플리케이션 개발이 어렵다.

<표 1> 웹 VP vs C/S 의 장·단점 비교

<표 1>에서 알 수 있듯이 Web상의 VP를 이용할 경우에는 시·공간의 제약을 전혀 받지 않고 사용자는 단지 웹 브라우저를 통해서 서버와 접속해서 작업을 할 수 있으므로 작업의 효율성 측면에서 C/S 시스템 보다 훨씬 우수하다.

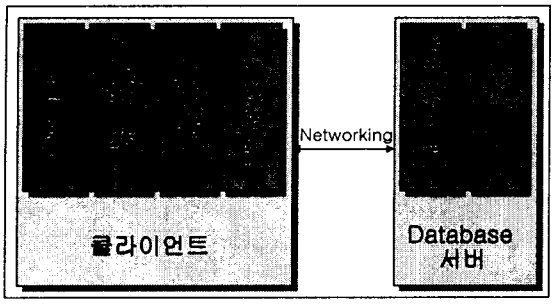
VP를 이용하는데 있어서 한가지 중대한 문제점은 웹 데이터베이스가 많은 사용자들에게 개방되어 있기 때문에 데이터베이스의 설계와 관리 그리고 보안 측면에서 더 많은 연구가 필요하다. 이러한 결점을 해결하기 위해서 VP 아키텍처 설계에 대해 자세히 검토할 것이다.

VP를 이용할 경우와 C/S 시스템을 이용할 경우에 클라이언트와 서버가 담당하게 될 작업이 어떻게 달라지는지에 대해서 살펴보자.

VP를 이용할 경우에 클라이언트는 단지 웹 브라우저(이것이 클라이언트 애플리케이션이 된다)를 통해서 웹 서버에 접속한 후 작업을 처리한다. 그러나, C/S 시스템에서는 클라이언트가 서버에 접속하기 위해서 전용 애플리케이션이 별도로 필요하게 되므로 보다 많은 작업을 떠맡게 된다. 이 두가지 방법에 관한 자세한 사항은 <그림 2>와 <그림 3>을 통해 살펴보기로 한다.

<그림 2>의 C/S 시스템을 살펴보면, ODBC는 MS에서 제공하는 미들웨어(middleware)로 Oracle, Cybase 등의 서로 다른 DBMS 제품에 같은 방법으로 접속할 수 있도록 인터페이스를 제공하는 API(application programming interface)이다. <그림 2>와 같이 C/S 시스템에서는 클라이언트 애플리케이션이 ODBC Driver Manager, ODBC Driver 그리고 ODBC API에 근거한 애플리케이션을 별도로 가지고 있어야 한다. 클라이언트의 Application 층은 API 함수를 호출하고, ODBC Driver

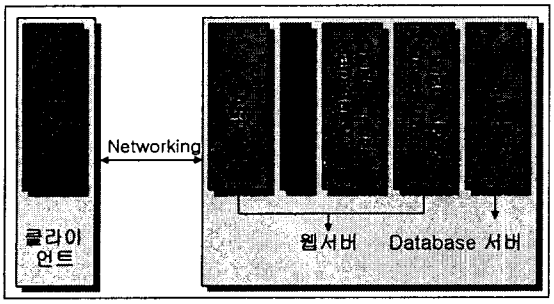
Manager 층은 ODBC Driver를 로드(load)하게 된다. 그리고, ODBC DBMS Driver 층은 함수를 처리하고 특정 DBMS에게 SQL 문을 넘기게 된다. 서버의 DBMS 층은 ODBC Driver로부터 보내온 SQL 문장을 처리하고 그 결과를 Driver에게 다시 반환한다. C/S에 각각 존재하는 S/W 층은 DBMS 연결을 위한 통신절차에 해당한다.



<그림 2> C/S 시스템의 작업분할

이상에서 C/S 시스템에서는 클라이언트가 서버의 DBMS에 접근하기 위한 API 지원 애플리케이션을 별도로 가지도록 프로그래밍 되고 클라이언트에서 많은 작업이 수행된다는 것을 알 수 있다.

C/S 시스템과 대조적으로 웹 VP를 이용할 경우에는 <그림 3>과 같이 작업 분산이 이루어진다.



<그림 3> VP 이용시의 작업분할

클라이언트는 트랜잭션 처리시에 데이터베이스 서버에 접속하기 위해 별도의 애플리케이션을 필요로 하지 않으며, 웹 브라우저만으로 쉽게 웹 데이터베이스 서버에 접속할 수 있다. 이런 장점을 이용하면 시·공간상의 제약을 받지 않고 트랜잭션을 처리할 수 있으며, 사용자들이 직접 웹 데이터베이스에 접속할 수 있는 환경이 만들어지기 때문에 원하는 정보를 쉽게 얻을 수 있는 고객 지향적인 서비스 환경이 가능하게 된다.

웹 브라우저를 통해서 사용자가 웹 서버에 접속하게 되면 요청에 따라 ASP 스크립트가 웹 서버 상에서 실행된다. 이때 DBMS와의 연결을 필요로 할 경우에는 ADO를 이용하게 된다. OLE(object linking & embedding) DB의 인터페이스에 근간한 ADO는 ODBC를 통해서 관계형 데이터베이스(RDB)와 연결되도록 설계되었으며, ODBC가 지원되는 데이터 원본에 대해서는 그것이 어떤 형태의 데이터베이스이든 자유로운 액세스를 제공하는 연결 메커니즘이다. 한가지 주목할 점은 ADO의 기반 구조가 ASP의 융통성과 결합되어 인터넷상에서 작동되기 때문에 특정 웹 브라우저에 국한되지 않는 C/S 시스템을 구현할 수 있도록 해준다.

웹 VP를 이용할 경우에는 C/S 시스템에서 클라이언트가 수행하던 ODBC 서비스 부분이 웹서버에서 행해진다. 다시 말해서, 클라이언트가 아닌 웹 서버의 ODBC 설정으로 DBMS로의 연결이 가능하게 된다. 이때 DBMS는 웹 서버에서 운영될 수도 있으며, 독립적인 데이터베이스 서버에서 운영될 수도 있다.

4. 다중 웹 데이터베이스 시스템 모델

다중 데이터베이스 시스템은 분산 데이터베이스 시스템과 비교할 때 데이터베이스의 분산 방법과 관리 및 제어 등의 몇 가지 점에서 구별된다.

분산 데이터베이스 시스템은 각 지역으로 분산된 DBMS의 형태가 접근 방법, 최적화 전략, 동시성 제어 전략 그리고 데이터 모델 등에서 동질성을 가지는 것으로 특징지을 수 있다. 그리고 중앙 데이터베이스에 있는 테이블 조각들은 몇 가지 기준(수직, 수평 혹은 혼합적인 방법)에 의해 각 지역 데이터베이스로 분산되어진다.

이에 반해 다중 데이터베이스 시스템에서는 각 지역 데이터베이스가 중앙 데이터베이스의 테이블 조각을 가지는 것이 아니라 별도의 테이블들이 서로 다른 특성(relational DB, hierarchical DB 등)으로 분산되어 운영되기 때문에 지역만의 고유한 접근 방법 및 동시성 제어 전략 등이 행해질 수 있다. 이러한 이유로 다중 데이터베이스 시스템에서는 각 지역 데이터베이스들이 서로 이질적이며, 서로 다른 DBMS에 의해 관리되므로 그들만의 고유한 지역 자치성(autonomy)을 보장받을 수 있다. 다중 데이터베이스 시스템은 이러한 장점이 있지만, 동시에 각각의 지역 데이터베이스를 관리하기 위해서 분산 데이터베이스 시스템보다 훨씬 복잡한 관리 알고리즘을 필요로 한다.

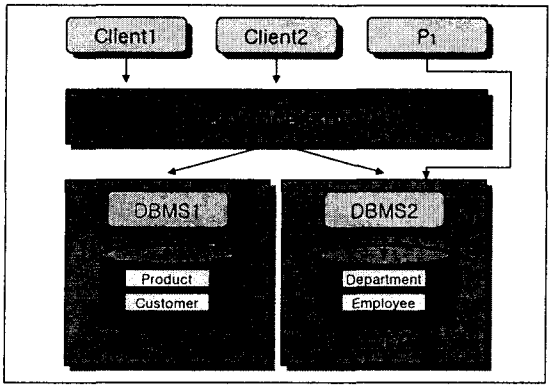
다중 데이터베이스 시스템은 각각의 지역 DBMS로 일률적으로 트랜잭션 처리를 하기 위해서 보통 전역 처리장치를 사용한다. 이 전역 처리 장치들은 지역 자치성을 보장하고, 각 지역 데이터베이스의 무결성 제약조건을 유지시키면서 적절하게 트랜잭션을 처리할 수 있도록 설계되어야 한다.

본 논문에서는 이 전역 처리장치를 기존의 C/S 시스템에서 Web으로 확장시켜 지역 데이터베이스 시스템의 지역 자치성 보장을 전제로 무결성 및 동시처리(concurrency) 제어를 위한 가상 처리 장치 아키텍처를 설계하는 것에 관해서 연구하였다. 각 지역 데이터베이스에 흩어진 테이블들은 하나 이상의 관계(relation)를 가지고 있음을 전제로 한다.

다중 웹 데이터베이스 시스템은 웹 상의 많은 사용자들에게 동시에 개방되어 사용자(고객) 지향의 서비스를 제공할 수 있다는 큰 장점을 가지고 있다. 하지만, 데이터베이스를 관리하는 관점에서는 C/S 시스템으로 구현되는 것에 비해 훨씬 많은 어려움이 존재한다. 이러한 문제점은 크게 트랜잭션 처리시의 무결성 제약사항 유지와 동시처리 제어로 나누어 생각할 수 있다.

<그림 4>는 다중 웹 데이터베이스 시스템의 트랜잭션 처리 과정을 보여주고 있다. 이 그림에서 Client는 웹 브라우저를 통해서 웹 서버에서 작동하는 VP에 접근할 수 있는데, 요구된 트랜잭션은 VP에 의해서 각각의 지역 DBMS로 분리되어 처리

된 후 결과를 다시 Client에게 돌려주게 된다. Client와 VP는 인터넷을 통해 연결되며, VP와 각각의 DBMS는 LAN/WAN으로 연결될 수 있다.



<그림 4> 다중 웹 데이터베이스 시스템 모델

다음의 간단한 예제를 통해 트랜잭션 처리과정에서 발생할 수 있는 문제점을 살펴보자.

Product(PRO_NO, PRO_NAME, DEP_NAME)
Customer(CUS_NO, CUS_NAME, EMP_NO)
Department(DEP_NAME, EMP_NO, PRO_NO)
Employee(EMP_NO, EMP_NAME, DEP_NAME)

Client1은 VP를 통해 테이블 Product의 레코드 P_h를 판독하고 테이블 Department의 레코드 D_i를 갱신하며(트랜잭션 G_1), Client2는 VP를 통해 테이블 Department의 레코드 D_i와 테이블 Employee의 레코드 E_j를 판독한다(트랜잭션 G_2). 그리고, 지역 L2에서는 VP를 거치지 않고 자체적으로 테이블 Department의 레코드 D_i를 갱신하고 테이블 Employee의 레코드 E_j를 판독하는 트랜잭션 P_1이 이루어진다.

각 지역별로 행해지는 트랜잭션의 스케줄은 다음과 같이 나타낼 수 있다.

지역 L1의 스케줄 : $G_1[R(P_h)]$
지역 L2의 스케줄 : $G_1[W(D_i)]$, $G_2[R(D_i,E_j)]$,
$P_1[W(D_i),R(E_j)]$

위의 스케줄에서 트랜잭션 G_1과 P_1의 실행에 있어서 지역 L2에서 행해지는 트랜잭션 P_1 중 테이블 Department의 레코드 D_i를 갱신하는 과정에서 relation이 되어 있는 지역 L1의 테이블 Product의 속성 PRO_NO에 기존의 레코드에 없는 새로운 데이터가 추가된다면 지역 L1의 테이블 Product의 레코드에도 같은 레코드가 추가되어야 한다. 그러나, 다중 데이터베이스 시스템에서 지역 자치성 및 DBMS의 기능상 하나의 지역 데이터베이스에서 행해지는 작업이 다른 지역 데이터베이스에 영향을 미치지 못하므로 이것이 불가능하다. 따라서 무결성 제약사항을 위반하게 된다. 이것을 해결하기 위해서는 각 지역에서 별도로 행해지는 트랜잭션들도 반드시 웹 VP를 통해서 수행되도록 제한할 필요가 있다.

또한, 트랜잭션 G_1에 의해서 테이블 Department의 레코드인 D_i가 판독된 후 갱신되기 전에 트랜잭션 G_2에 의해 같은 레코드인 D_i가 판독된다면 Client2는 G_1에 의해서 갱신되어야 할 올바른 레코드 정보를 가지지 못하는 결과를 초래한다. 이러한 동시처리 제어를 해결하기 위해서 잠금(lock) 기법을 이용할 수 있다.

이상에서와 같이 분산 데이터베이스 시스템이 아닌 다중 데이터베이스 시스템에서는 한 지역 데이터베이스에서 행해진 작업의 영향이 다른 지역 데이터베이스에까지 반영될 수 있도록 DBMS가 자체적으로 전역적인 관리 차원의 서비스를 지원하지 못하며 기본적인 질의만을 제공하는 실정이다.

다음 절에서는 전역 트랜잭션을 처리하기 위해 무결성 제약조건을 유지하는 동시에 동시처리 제어를 가능하게 하는 VP 아키텍처 설계에 관해서 설명한다.

5. VP 아키텍처 설계

다중 데이터베이스 시스템을 사용하는 이유중의 하나는 잠금 메커니즘이나 타임 스탬프 전략 등을 행할 수 있는 자치성을 지역적으로 가질 수 있기 때문이다. 하지만, 지역 자치성을 너무 강화하게 되면 전역 트랜잭션을 처리할 때 동시처리 성능을 저하시키는 등의 많은 제약 사항이 따르게 된다.

본 논문에서는 지역 자치성을 적절히 보장하고 무결성 제약사항 및 동시처리 제어를 가능하도록 하기 위해서 전역 또는 지역 트랜잭션 처리는 반드시 웹 상의 VP를 거쳐야 하는 것으로 가정한다. 예를 들어, 웹 VP를 통하지 않고 행해지는 지역 DBMS에서의 작업에 대해서는 다른 지역 데이터베이스와의 무결성 검증이 불가능하므로, 웹 VP를 통하지 않는 어느 한 쪽 지역 DBMS만의 갱신 작업은 이루어지지 않는다고 전제하는 것이다.

무결성 제약사항을 유지하기 위해서 VP는 한 지역의 레코드를 갱신할 때 레코드의 속성과 relation이 되어있는 다른 지역 테이블을 검색하여 존재하지 않는 새로운 데이터가 추가될 경우에는 갱신 작업을 할 수 없도록 해야한다. 이렇게 트랜잭션이 지역 데이터베이스의 갱신작업으로 처리될 때는 반드시 갱신될 레코드의 속성 중 하나가 다른 지역의 테이블과 relation이 되어 있는지를 검사한다. 만일, 갱신하려는 데이터가 그 지역의 레코드에 존재할 경우에는 트랜잭션을 처리하고 그렇지 않으면 트랜잭션을 거부하게 된다.

그리고, 동시처리 제어를 해결하기 위해서 각 지역 데이터베이스에는 데이터를 담고 있는 테이블 이외에 테이블의 상태 정보, 즉 레코드의 잠금 여부와 레코드 갱신 작업중인 트랜잭션이 시작된 시간정보를 포함하고 있어야 한다.

6. 결론

웹 VP를 이용할 경우에는 모든 작업이 VP를 거쳐서 행해지므로 한 지역에 국한된 관리 알고리즘은 전역 트랜잭션에 영향을 미치지 않도록 설계되어야 하기 때문에 완벽한 지역 자치성을 보장하지 못한다. 이러한 점을 해결하기 위해서 지역 자체적인 관리 알고리즘도 웹 VP에서 처리될 수 있도록 설계될 필요가 있다.

참 고 문 헌

참고문헌은 저자에게 e-mail로 연락바랍니다.
e-mail address : bluesdh@iesys.skku.ac.kr