1. 분산 데이터베이스의 개요

1990년대에는 데이터베이스를 분산하여 저장하고 그것을 하나의 데이터베이스로 인식하여 사용하는 기술은 아주 난이도가 높은 고급기술로 인식되었다. 2000년도에 클라우드 컴퓨팅, SOA를 인식하듯 분산 데이터베이스를 인식하고 연구·도입하려는 기업이 많았었다. DBMS의 기능이 강해지고 네트워크 속도가 빨라지면서 분산 데이터베이스가 초기에 예상한 만큼 확산되지는 않았지만, 여전히 많은 데이터베이스는 네트워크를 통한 데이터베이스 간의 공유체계를 통해 분산 데이터베이스를 활용하고 있다.
분산데이터베이스의 정의는 다음과 같다.

> 여러 곳으로 분산되어있는 데이터베이스를 하나의 가상 시스템으로 사용할 수 있도록 한 데이터베이스
> 논리적으로 동일한 시스템에 속하지만, 컴퓨터 네트워크를 통해 물리적으로 분산되어 있는 데이터들의 모임. 물리적 Site 분산, 논리적으로 사용자 통합·공유

즉, 분산 데이터베이스는 데이터베이스를 연결하는 빠른 네트워크 환경을 이용하여 데이터베이스를 여러 지역 여러 노드로 위치시켜 사용성/성능 등을 극대화 시킨 데이터베이스라고 정의할 수 있다.

2. 분산 데이터베이스의 투명성(Transparency)

분산데이터베이스가 되기 위해서는 6가지 투명성(Transparency)을 만족해야 한다.

1) 분할 투명성 (단편화) : 하나의 논리적 Relation이 여러 단편으로 분할되어 각 단편의 사본이 여러 site에 저장
2) 위치 투명성 : 사용하려는 데이터의 저장 장소 명시 불필요. 위치정보가 System Catalog에 유지되어야 함
3) 지역사상 투명성 : 지역DBMS와 물리적 DB사이의 Mapping 보장. 각 지역시스템 이름과 무관한 이름 사용 가능
4) 중복 투명성 : DB 객체가 여러 site에 중복 되어 있는지 알 필요가 없는 성질
5) 장애 투명성 : 구성요소(DBMS, Computer)의 장애에 무관한 Transaction의 원자성 유지
6) 병행 투명성 : 다수 Transaction 동시 수행시 결과의 일관성 유지, Time Stamp, 분산 2단계 Locking을 이용 구현

전통적인 분산 데이터베이스 구축과 같이, 분산 환경의 데이터베이스를 위와 같은 특징 모두를 만족하면서 구축하는 사례는 최근에는 드물다. 최근에는 분산 환경의 데이터베이스를 구축하기보다 통합하여 데이터베이스를 구축하는 사례가 더 많이 있다. 그럼에도 불구하고 위의 분산 환경의 데이터베이스를 업무적인 특징 및 지역적인 특징에 따라 적절하게 활용하기만 하면, 다양한 장점을 제공하는 특징을 가지고 있기 때문에 대량 데이터처리의 지역적 처리나 글로벌 처리 등에서는 분산 데이터베이스가 유용하게 활용되고 있다.

3. 분산 데이터베이스의 적용 방법 및 장단점

가. 분산 데이터베이스 적용방법

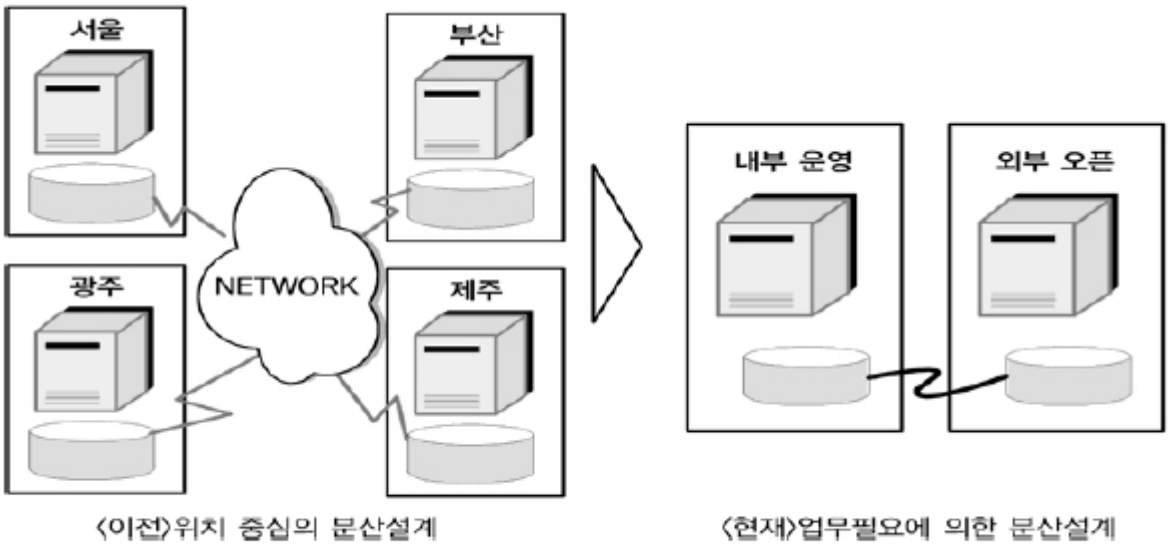
분산 환경의 데이터베이스를 성능이 우수하게 현장에서 가치 있게 사용하는 방법은 업무의 흐름을 보고 업무구성에 따른 아키텍처 특징에 따라 데이터베이스를 구성하는 것이다. 단순히 분산 환경에서 데이터베이스를 구축하는 것이 목적이 아니라, 업무의 특징에 따라 데이터베이스 분산구조를 선택적으로 설계하는 능력이 필요한 것이다. 이러한 측면만을 보았을 때는 데이터베이스 분산설계라는 측면보다는 데이터베이스 구조설계(아키텍처)라는 의미로 이해해도 무방할 것이다.

나. 분산 데이터베이스 장단점

장점	단점
- 지역 자치성, 점증적 시스템 용량 확장 - 신뢰성과 가용성 - 효용성과 융통성 - 빠른 응답 속도와 통신비용 절감 - 데이터의 가용성과 신뢰성 증가 - 시스템 규모의 적절한 조절 - 각 지역 사용자의 요구 수용 증대	- 소프트웨어 개발 비용 - 오류의 잠재성 증대 - 처리 비용의 증대 - 설계, 관리의 복잡성과 비용 - 불규칙한 응답 속도 - 통제의 어려움 - 데이터 무결성에 대한 위협

4. 분산 데이터베이스의 활용 방향성

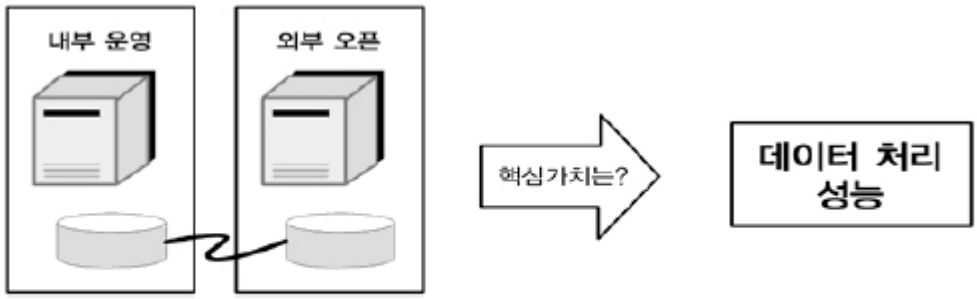
분산 데이터베이스는 업무적인 기능이 다양해지고 데이터의 양이 기하급수적으로 증가하는 최근 데이터베이스 환경에서 적용하는 고급화된 기술이다. 업무적인 특징에 따라 분산 데이터베이스를 활용하는 기술이 필요하다.



[그림 Ⅰ-2-39] 분산설계 방향성

5. 데이터베이스 분산구성의 가치

데이터를 분산 환경으로 구성하였을 때 가장 핵심적인 가치는 바로 통합된 데이터베이스에서 제공할 수 없는 빠른 성능을 제공한다는 점이다. 원거리 또는 다른 서버에 접속하여 처리하므로 인해 발생되는 네트워크 부하 및 트랜잭션 집중에 따른 성능 저하의 원인을 분산된 데이터베이스 환경을 구축하므로 빠른 성능을 제공하는 것이 가능해 진다. 바로 이 점 때문에 분산 환경의 데이터베이스를 구축하게 되는 것이다.



[그림 Ⅰ-2-40] 분산 데이터베이스 핵심가치

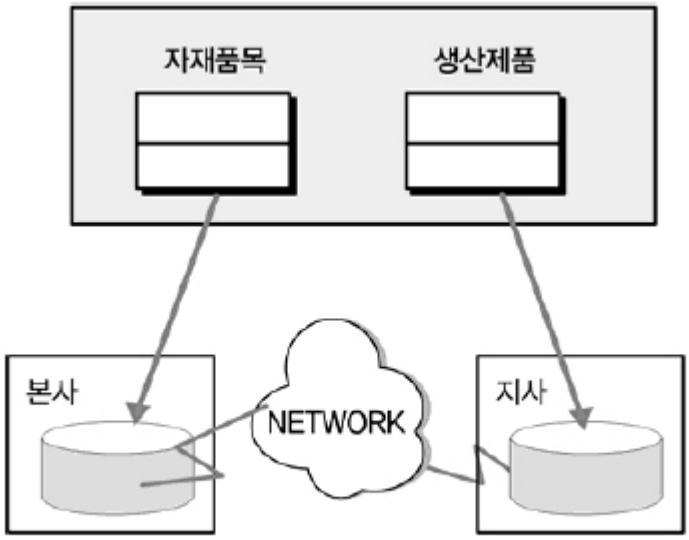
6. 분산 데이터베이스의 적용 기법

데이터베이스의 분산의 종류에는 테이블 위치 분산과 테이블 분할 분산, 테이블 복제 분산, 테이블 요약 분산 전략이 있다. 그 중에서도 가장 많이 사용하는 방식의 테이블의 복제 분할 분산의 방법이고 이 방법은 성능이 저하되는 많은 데이터베이스에서 가장 유용하

게 적용할 수 있는 기술적인 방법이 된다. 분산 환경으로 데이터베이스를 설계하는 방법은 일단 통합 데이터 모델링을 하고 각 테이블별로 업무적인 특징에 따라 지역 또는 서버별로 테이블을 분산 배치나 복제 배치하는 형태로 설계할 수 있다.

가. 테이블 위치 분산

테이블 위치 분산은 테이블의 구조는 변하지 않는다. 또한 테이블이 다른 데이터베이스에 중복되어 생성되지도 않는다. 다만 설계된 테이블의 위치를 각각 다르게 위치시키는 것이다. 예를 들어, 자재품목은 본사에서 구입하여 관리하고 각 지사별로 자재품목을 이용하여 제품을 생산한다고 하면 [그림 Ⅰ-2-41]과 같이 데이터베이스를 본사와 지사단위로 분산시킬 수 있다.



[그림 Ⅰ-2-41] 테이블별 위치 분산

[그림 Ⅰ-2-41]의 분산방법은 설계된 테이블 각각이 지역별로 분산되어 생성되는 경우이다.
각각의 테이블마다 위치가 다르게 지정되어야 한다면 [그림 Ⅰ-2-42]의 표와 같이 각각 테이블마다 위치를 표기하여 테이블을 생성하도록 한다.

테이블 / 위치	자재품목	생산제품	협력회사	사원	부서
본사	•		•		•
지사		•		•	

[그림 Ⅰ-2-42] 테이블별 위치 분산

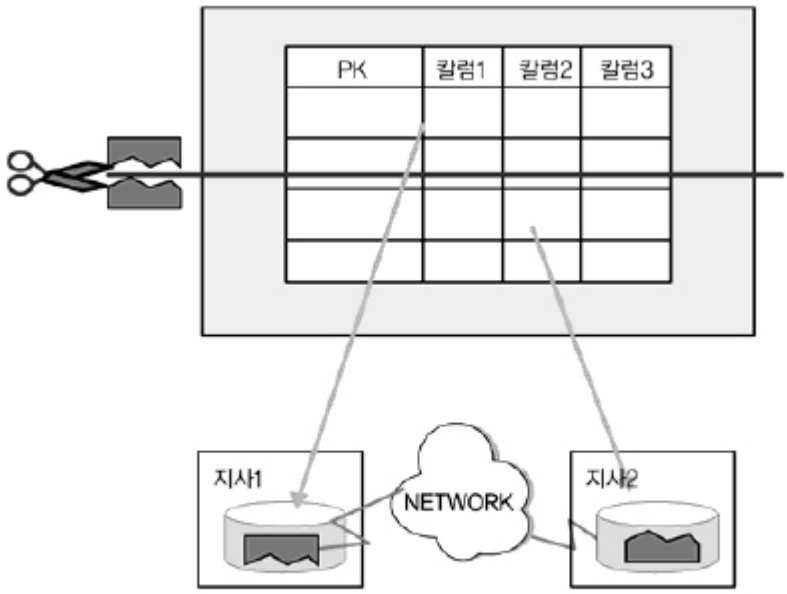
테이블별 위치 분산은 정보를 이용하는 형태가 각 위치별로 차이가 있을 경우에 이용한다. 테이블의 위치가 위치별로 다르므로 테이블의 위치를 파악할 수 있는 도식화된 위치별 데이터베이스 문서가 필요하다.

나. 테이블 분할(Fragmentation) 분산

테이블 분할 분산은 단순히 위치만 다른 곳에 두는 것이 아니라 각각의 테이블을 쪼개어 분산하는 방법이다. 테이블을 분할하여 분산하는 방법은 테이블을 나누는 기준에 따라 두 가지로 구분된다. 첫 번째는 테이블의 로우(Row)단위로 분리하는 수평분할(Horizontal Fragmentation)이 있고 두 번째는 테이블을 칼럼(Column) 단위로 분할하는 수직분할(Vertical Fragmentation)이 있다.

수평분할(Horizontal Fragmentation)

지사(Node)에 따라 테이블을 특정 칼럼의 값을 기준으로 로우(Row)를 분리한다. 칼럼은 분리되지 않는다. 모든 데이터가 각 지사별로 분리되어 있는 형태를 가지고 있다. 각 지사에 있는 데이터와 다??며, 데이터를 한군데 집합시켜 놓아도 Primary Key에 의해 중복이 발생되지 않는다.



[그림 Ⅰ-2-43] 테이블 분할 분산 - 수평분할

이와 같이 수평분할을 이용하는 경우는 각 지사(Node)별로 사용하는 로우(Row)가 다를 때 이용한다. 데이터를 수정할 때는 타 지사에 있는 데이터를 원칙적으로 수정하지 않고 자신의 데이터에 대해서 수정하도록 한다. 각 지사에 존재하는 테이블에 대해서 통합처리를 해야 하는 경우는 조인(JOIN)이 발생하여 성능 저하가 예상되므로 통합처리 프로세스가 많은지를 먼저 검토한 이후에 많지 않은 경우에 수평분할을 해야 한다.

데이터가 지사별로 별도로 존재하므로 중복은 발생하지 않는다. 대신 타 지사에 있는 데이터가 지사구분이 변경되면 단순히 수정이 발생하는 것 이외에 변경된 지사로 데이터를 이송해야 한다. 한 시점에는 한 지사(Node)에서 하나의 데이터만이 존재하므로 데이터의 무결성은 보장되는 형태이다.

지사(Node)별로 데이터베이스를 운영하는 경우는 데이터베이스가 속한 서버가 지사(Node)에 존재하던지 아니면 본사에 통합해서 존재하건 간에 데이터베이스 테이블들은 수평 분할하여 존재한다.

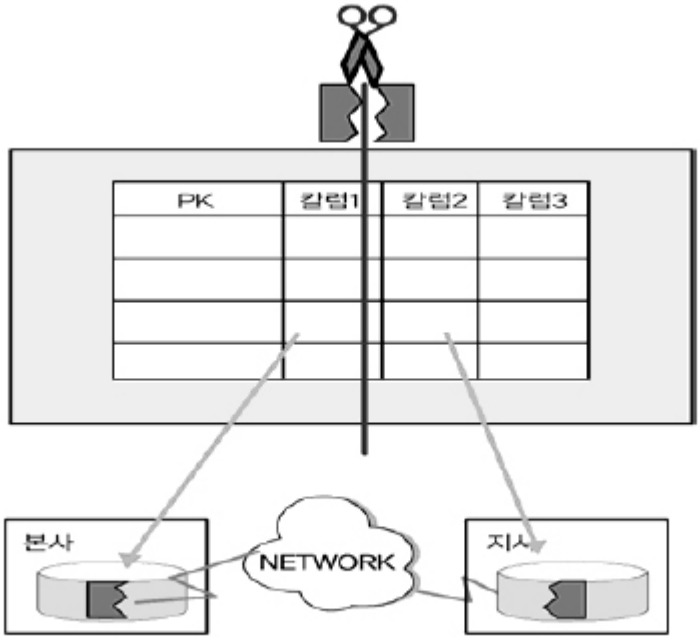
지사별로 운영하는 테이블들의 예를 들어보자. 테이블 고객, 생산제품, 협력회사, 사원, 부서 테이블이 지사1과 지사2에서 일의 시작과 끝이 항상 다르게 발생한다고 하면 각 테이블은 지사별로 수평 분할하여 [그림 Ⅰ-2-44]의 표와 같이 생성되게 된다.

테이블 / 위치	고객	생산제품	협력회사	사원	부서
지사 1	◓	◓	◓	◓	◓
지사 2	◒	◒	◒	◒	◒

[그림 Ⅰ-2-44] 테이블 수평분할

수직분할(Vertical Fragmentation)

지사(Node)에 따라 테이블 칼럼을 기준으로 칼럼(Row)을 분리한다. 로우(Row) 단위로는 분리되지 않는다. 모든 데이터가 각 지사별로 분리되어 있는 형태를 가지고 있다. 칼럼을 기준으로 분할하였기 때문에 각각의 테이블에는 동일한 Primary Key구조와 값을 가지고 있어야 한다. 지사별로 쪼개어진 테이블들을 조합하면 Primary Key가 동일한 데이터의 조합이 가능해야 하며 하나의 완전한 테이블이 구성되어야 한다. 데이터를 한군데 집합시켜 놓아도 동일한 Primary Key는 하나로 표현하면 되므로 데이터 중복은 발생되지 않는다.



[그림 Ⅰ-2-45] 테이블 수직분할

예를 들어 제품의 재고량은 각 지사별로 관리하고 제품에 대한 단가는 본사에서 관리한다고 하면 본사 테이블에는 제품번호, 단가가 존재하고 지사에는 제품번호, 재고량이 존재한다. 이를 본사와 지사단위로 분리된 칼럼의 모습을 도식화 하면 다음과 같이 나타난다.

테이블 / 위치	제품	분할 칼럼
본사	◐	제품번호, 단가
지사	◑	제품번호, 재고량

[그림 Ⅰ-2-46] 테이블 수직분할 모델링

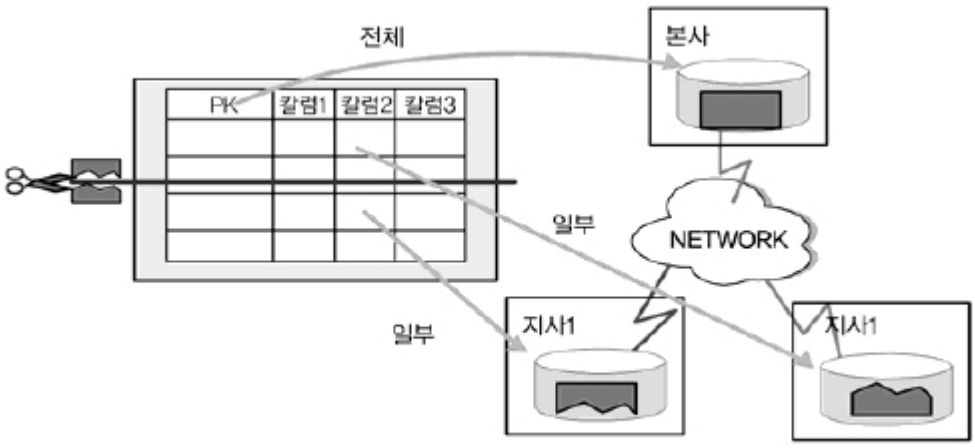
테이블의 전체 칼럼 데이터를 보기 위해서는 각 지사(Node)별로 흩어져 있는 테이블들을 조인(JOIN)하여 가져와야 하므로 가능하면 통합하여 처리하는 프로세스가 많은 경우에는 이용하지 않도록 한다. 일반적으로 실제 프로젝트에서는 이와 같이 칼럼을 쪼개는 테이블의 수직분할 분산 환경을 구성하는 사례는 드물다.

다. 테이블 복제(Replication) 분산

테이블 복제(Replication) 분산은 동일한 테이블을 다른 지역이나 서버에서 동시에 생성하여 관리하는 유형이다.
마스터 데이터베이스에서 테이블의 일부의 내용만 다른 지역이나 서버에 위치시키는 부분복제(Segment Replication)가 있고 마스터 데이터베이스의 테이블의 내용을 각 지역이나 서버에 존재시키는 광역복제(Broadcast Replication)가 있다.

부분복제(Segment Replication)

통합된 테이블을 한군데(본사)에 가지고 있으면서 각 지사별로는 지사에 해당된 로우(Row)를 가지고 있는 형태이다. 지사에 존재하는 데이터는 반드시 본사에 존재하게 된다. 즉 본사의 데이터는 지사데이터의 합이 되는 것이다. 각 지사에서 데이터 처리가 용이할 뿐만 아니라 전체 데이터에 대한 통합처리도 본사에 있는 통합 테이블을 이용하게 되므로 여러 테이블에 조인(JOIN)이 발생하지 않는 빠른 작업 수행이 가능해진다.



[그림 Ⅰ-2-47] 테이블 복제 분산 – 부분복제

[그림 Ⅰ-2-47]을 보면, 본사 데이터베이스에 있는 테이블에는 테이블의 전체 내용이 들어가고 각 지사 데이터베이스에 있는 테이블에는 지사별로 관계된 데이터만 들어가게 된다. 수평분할 분산과 마찬가지로 지사간에는 데이터의 중복이 발생하지 않으나 본사와 지사간에는 데이터의 중복이 항상 발생하게 되는 경우이다.
보통 전국에 있는 고객을 관리할 때 본사에는 전국고객에 대한 정보를 관리하고 지사에는 각 지사와 거래하는 고객정보를 관리한다. 본사의 데이터를 이용하여 통계, 이동 등을 관리하며 지사에 있는 데이터를 이용하여 지사별로 빠른 업무수행을 한다. 보통 지사에 데이터가 먼저 발생하고 본사에 데이터는 지사에 데이터를 이용하여 통합하여 발생된다.

위치 \ 테이블	고객
본사	●
지사 1	◐
지사 2	◑

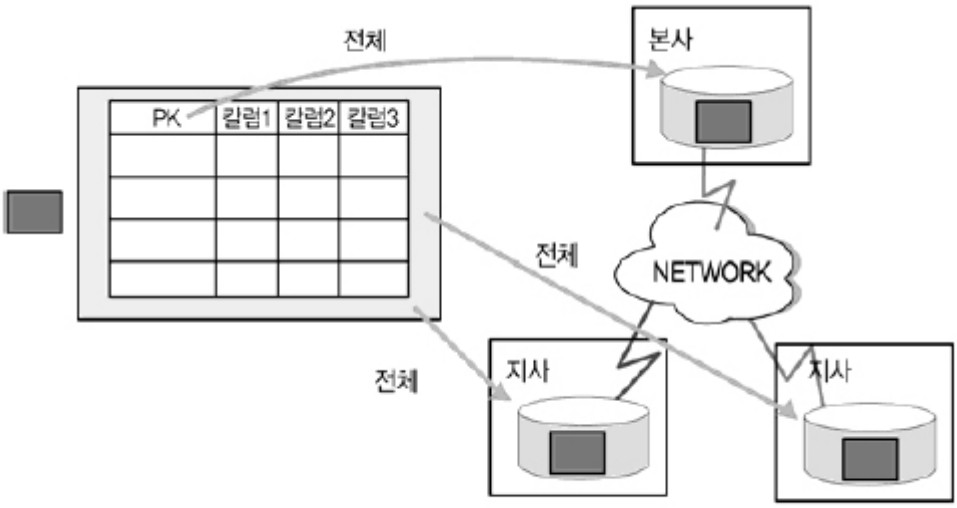
본사에는 전국의 고객정보를 관리하고 지사 1의 고객테이블에서는 지사 1에 속한 고객정보를 지사 2의 고객테이블에서는 지사 2에 속한 고객정보를 관리한다.

[그림 Ⅰ-2-48] 테이블 복제 분산 – 부분복제

실제 프로젝트에서 많이 사용하는 데이터베이스 분산기법에 해당한다. 각 지사별로 업무수행이 용이하고 본사에 있는 데이터를 이용하여 보고서를 출력하거나 통계를 산정하는 등 다양한 업무형태로 이용 가능하다.
다른 지역간의 데이터를 복제(Replication)하는데 많은 시간이 소요되고 데이터베이스와 서버에 부하(Load)가 발생하므로 보통 실시간(On-Line) 처리에 의해 복사하는 것보다는 야간에 배치 작업에 의해 수행되는 경우가 많이 있다.
또한 본사와 지사 양쪽 모두 데이터를 수정하여 전송하는 경우 데이터의 정합성을 일치시키는 것이 어렵기 때문에 가능하면 한쪽(지사)에서 데이터의 수정이 발생하여 본사로 복제(Replication)를 하도록 한다.

광역복제(Broadcast Replication)

통합된 테이블을 한군데(본사)에 가지고 있으면서 각 지사에도 본사와 동일한 데이터를 모두 가지고 있는 형태이다. 지사에 존재하는 데이터는 반드시 본사에 존재하게 된다. 모든 지사에 있는 데이터량과 본사에 있는 데이터량이 다 동일하다. 본사와 지사모두 동일한 정보를 가지고 있으므로 본사나 지사나 데이터처리에 특별한 제약을 받지는 않는다.



[그림 Ⅰ-2-49] 테이블 복제 분산 – 광역복제

예를 들어, 본사에서 코드테이블에 데이터에 대해 입력, 수정, 삭제가 발생하고 각 지사에서는 코드데이터를 이용하는 프로세스가 발생한다. 즉 본사에서는 데이터를 관리하고 지사에서는 이 데이터를 읽어 업무프로세스를 발생시키는 것이다.

위치 \ 테이블	코드
본사	●
지사 1	●
지사 2	●

본사, 지사1, 지사2 모두 동일한 양의 코드 테이블의 데이터를 가지고 있다.

[그림 Ⅰ-2-50] 테이블 복제 분산 – 광역복제

광역복제(Broadcast Replication) 역시 실제 프로젝트에서 많이 사용하는 데이터베이스 분산기법에 해당한다. 부분복제의 경우는 지사에서 데이터에 대한 입력, 수정, 삭제가 발생하여 본사에서 이용하는 방식이 많은 반면 광역복제(Broadcast Replication)의 경우에

는 본사에서 데이터가 입력, 수정, 삭제가 되어 지사에서 이용하는 형태가 차이점이다.

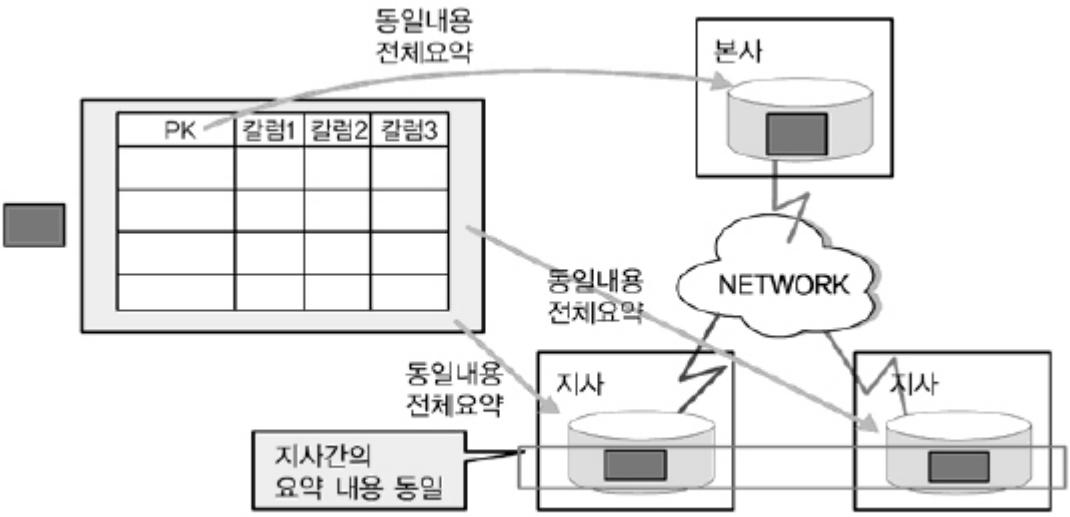
부분복제와 마찬가지로 데이터를 복제(Replication)하는데 많은 시간이 소요되고 데이터베이스와 서버에 부하(Load)가 발생하므로 보통 실시간(On-Line) 처리에 의해 복사하는 것보다는 배치에 의해 복제가 되도록 한다.

라. 테이블 요약(Summarization) 분산

테이블 요약(Summarization) 분산은 지역간에 또는 서버 간에 데이터가 비슷하지만 서로 다른 유형으로 존재하는 경우 있다. 요약의 방식에 따라, 동일한 테이블 구조를 가지고 있으면서 분산되어 있는 동일한 내용의 데이터를 이용하여 통합된 데이터를 산출하는 방식의 분석요약(Rollup Summarization)과 분산되어 있는 다른 내용의 데이터를 이용하여 통합된 데이터를 산출하는 방식의 통합요약(Consolidation Summarization)이 있다.

분석요약(Rollup Replication)

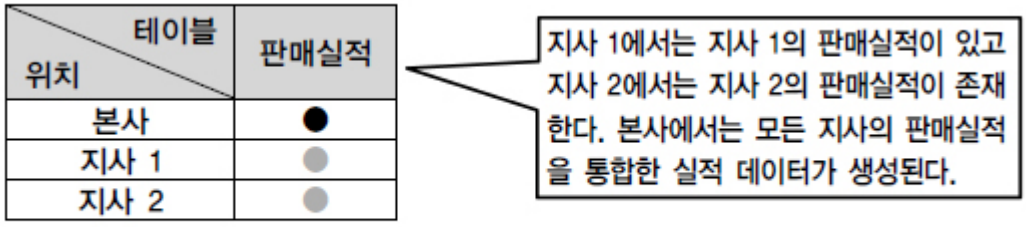
분석요약(Rollup Replication)은 각 지사별로 존재하는 요약정보를 본사에 통합하여 다시 전체에 대해서 요약정보를 산출하는 분산방법이다.



[그림 Ⅰ-2-51] 테이블 요약 분산 - 분석요약

[그림 Ⅰ-2-51]에서 보면, 테이블에 있는 모든 칼럼(Column)과 로우(Row)가 지사에도 동일하게 존재하지만, 각 지사에는 동일한 내용에 대해 지사별로 요약되어 있는 정보를 가지고 있고 본사에는 각 지사의 요약정보를 통합하여 재산출하여 전체에 대한 요약정보를 가지고 있는 것으로 표시되어 있다.

예를 들어, 제품별 판매실적이라는 테이블이 존재한다고 가정하자. 각 지사에서는 취급제품이 동일하다. 지사별로 판매된 제품에 대해서 지사별로 판매실적이 관리된다. 지사1과 지사2에도 동일한 제품이 취급이 되므로 이를 본사에서 판매실적을 집계할 경우에는 통합된 판매실적을 관리할 수 있는 것이다.

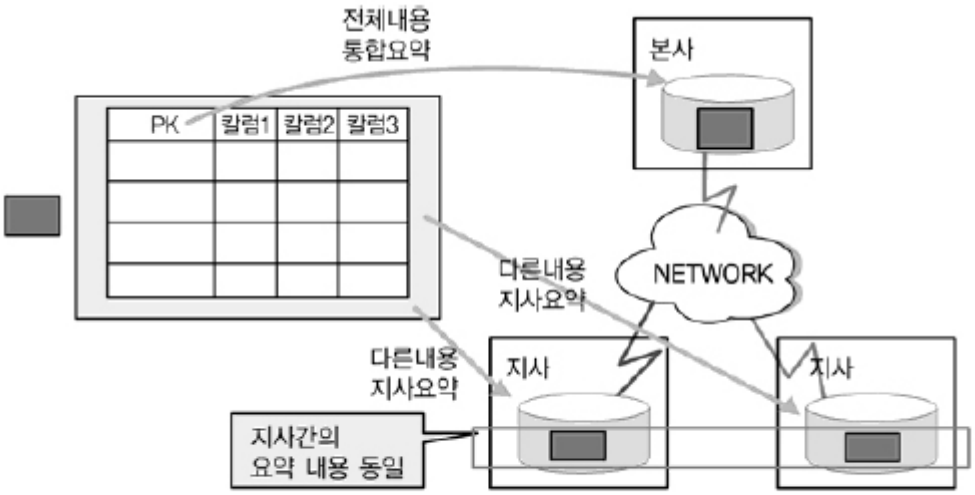


[그림 Ⅰ-2-52] 테이블 요약 분산 - 분석요약

각종 통계데이터를 산정할 경우에, 모든 지사의 데이터를 이용하여 처리하면 성능이 지연되고 각 지사 서버에 부하를 주기 때문에 업무에 장애가 발생할 수 있다. 통합 통계데이터에 대한 정보제공에 용이한 분산방법이다. 본사에 분석 요약된 테이블을 생성하고 데이터는 역시 일반 업무가 종료되는 야간에 수행하여 생성한다.

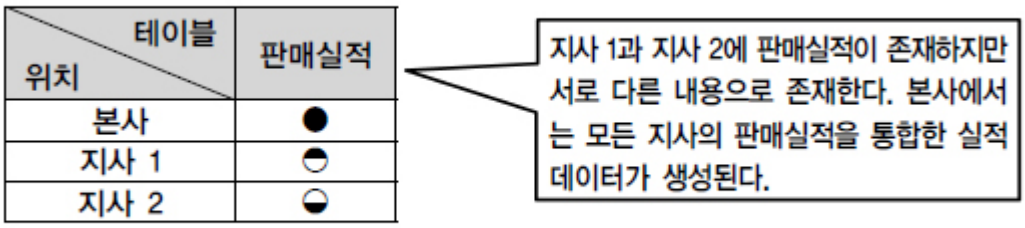
통합요약(Consolidation Replication)

통합요약(Consolidation Replication)은 각 지사별로 존재하는 다른 내용의 정보를 본사에 통합하여 다시 전체에 대해서 요약정보를 산출하는 분산방법이다.



[그림 Ⅰ-2-53] 테이블 요약 분산 - 통합요약

[그림 Ⅰ-2-53]에서 보면, 테이블에 있는 모든 칼럼(Column)과 로우(Row)가 지사에도 동일하게 존재하지만 각 지사에는 타지사와 다른 요약정보를 가지고 있고 본사에는 각 지사의 요약정보를 데이터를 같은 위치에 두는 것으로 통합하여 전체에 대한 요약정보를 가지고 있는 것으로 표시된다.

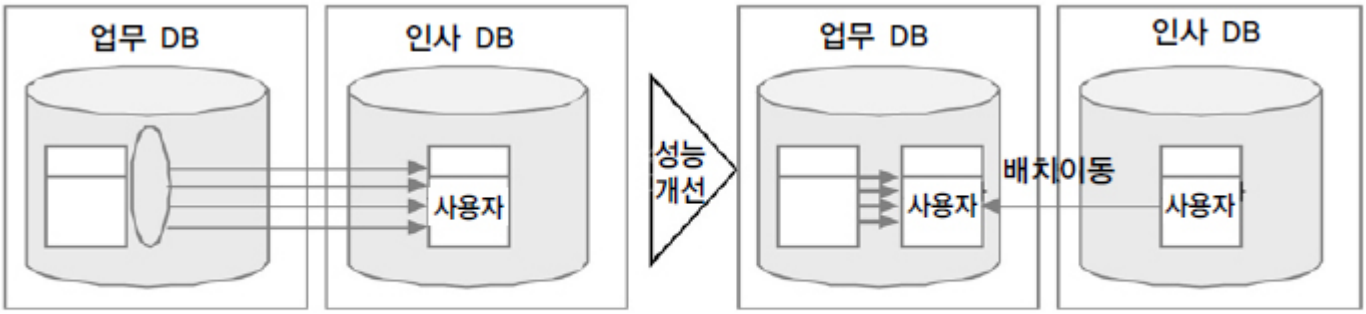


[그림 Ⅰ-2-54] 테이블 요약분산 - 통합요약

본사에 통계데이터를 산정하는 유형은 분석요약과 비슷하나 통합요약은 단지 지사에서 산출한 요약정보를 한군데 취합하여 보여주는 형태이다. 분석요약은 지사에 있는 데이터를 이용하여 본사에서 통합하여 요약 데이터를 산정하였지만 통합요약에서는 지사에서 요약한 정보를 본사에서 취합하여 각 지사별로 데이터를 비교하기 위해 이용되는 것이다.
각종 통계데이터를 산정할 경우에, 모든 지사의 데이터를 조인하여 처리하면 성능이 지연되고 각 지사 서버에 부하(LOAD)를 주기 때문에 업무에 장애가 발생할 수 있다. [그림 Ⅰ-2-54]와 같은 방법은 통합 통계데이터에 대한 정보제공에 용이한 분산방법이다. 본사에 통합 요약된 테이블을 생성하고 데이터는 역시 일반 업무가 종료되는 야간에 수행하여 생성하는 것이 일반적인 적용방법이다.

7. 분산 데이터베이스를 적용하여 성능이 향상된 사례

프로젝트를 수행할 때 단순한 분산 환경의 원리를 이해하지 않고 데이터베이스를 설계하여 성능이 저하되는 경우가 빈번하다. 특히 복제분산의 원리를 간단하게 응용하면 많은 업무적인 특성이 있는 곳에서 그 성능을 향상시켜 설계할 수 있다.
[그림 Ⅰ-2-55]는 개인정보를 관리하는 데이터베이스가 인사 데이터베이스일 때 분산이 안된 경우의 각 서버에 독립적으로 테이블이 있을 때 트랜잭션과 복제분산을 통해 테이블의 정보가 양쪽에 있을 때 트랜잭션 처리의 특성을 보여주는 그림이다. 단순한 개념도 이지만 위의 원리가 복잡한 업무처리에서 효과적으로 성능을 향상시킬 수 있음을 주목해야 한다.



[그림 Ⅰ-2-55] 업무 특성에 따른 분산환경 구성

데이터베이스 분산 설계는 다음과 같은 경우에 적용하면 효과적이다.

성능이 중요한 사이트에 적용해야 한다.
공통코드, 기준정보, 마스터 데이터 등에 대해 분산환경을 구성하면 성능이 좋아진다.
실시간 동기화가 요구되지 않을 때 좋다. 거의 실시간(Near Real Time)의 업무적인 특징을 가지고 있을 때도 분산 환경을 구성할 수 있다
특정 서버에 부하가 집중이 될 때 부하를 분산할 때도 좋다.
백업 사이트(Disaster Recovery Site)를 구성할 때 간단하게 분산기능을 적용하여 구성할 수 있다.