고성능 고가용성을 위한 ALTIBASE™ 자료저장 관리기의 설계

정 광 철† · 이 규 웅†† · 배 해 영†††

요 약

기존 디스크 기반 데이터베이스 관리 시스템은 디스크를 주저장 장치로 활용하는 특징적 환경 때문에 주기억 상주 데이터베이스 관리 시스템의 메모리 관리, 인덱스 관리, 자료저장 관리 기능 등에 대한 설계 및 구현 기술이 기본적으로 다르다. 본 논문에서는 현재 상용 시스템으로 사용되고 있는 ALTIBASE™ 주기억 상주 DBMS의 설계 및 구현 내용을 기술한다. 특히 자료저장 관리기의 세부 구성요소에 대한 구조적 특징과 주요 기능인 트랜잭션 처리 기법과 회복 관리 기법에 대하여 기존 DBMS의 기능과 비교 설명하며, 고가용성을 위한 데이터베이스 이중화 기능 및 이중화 작업 시 발생하는 트랜잭션 충돌 해결 방법에 대하여 설명한다. 또한 ALTIBASE™ 시스템의 성능을 측정하기 위하여 다양한 환경에서 실험된 TPS 결과를 보인다.

Design of ALTIBASE™ Storage Manager for High Performance and High Availability

Kwang Chul Jung† · Kyu Woong Lee†† · Hae Young Bae†††

ABSTRACT

Main memory database systems use the different implementation techniques to structure and organize the user data and system catalogs, since traditional database systems are optimized for the characteristics of disk storage environment. We present, in this paper, the design considerations for our main memory database system ALTIBASE™ that is currently applied to the time-critical applications. We focus on the design issues of storage manager in ALTIBASE™. The major components are introduced, and features and characteristics of transaction management and recovery method are described. We also present the database replication mechanism and its conflict resolution mechanism for high availability and performance. In order to evaluate our transaction performance, we show various experimental reports as measured by the TPS.

키워드 : 주기억 상주 데이터베이스(Main Memory Database(MMDB)), 자료저장 관리기(Storage Manager), 트랜잭션(Transaction), 회복(Recovery), 로그(Log), 이중화(Replication)

1. 서 론

통신산업 분야의 라우팅, 스위칭 응용분야나 실시간 산업 분야에서는 고성능 데이터 접근 및 효율적 데이터 관리를 요구하는 응용들이 증가하고 있다. 시간 제약적인 요구사항과 데이터베이스에 대한 요구를 동시에 갖는 이러한 응용들은 개개의 트랜잭션에 대한 빠른 응답 시간 뿐만 아니라 트랜잭션의 영속성(durability)과 가용성(availability)을 요구하고 있다.

대부분 디스크 기반(Disk Resident) 데이터베이스 관리 시스템(DR-DBMS)들은 개개의 트랜잭션에 대한 빠른 응답 시간을 보장하기 보다 전체적인 트랜잭션 처리량의 향상을 위해 설계되었다. 즉, 하나의 트랜잭션에 대한 시간제약사항을 만족하기보다는 트랜잭션의 처리율을 높여 전체적인 성능 향상을 위한 방법으로 설계되었다. 따라서 이러한 디스크 기반의 데이터베이스 시스템은 실시간 응용 분야에 적합하지 못하다.

이러한 수요를 만족하기 위해 기존 디스크 기반 데이터베이스 시스템들은 풍부한 메모리 공간을 활용하여 모든 데이터를 메모리에 적재하여 사용할 수 있는 대체방안을 제시하였다. 그러나 이러한 방법은 순수한 주기억 상주 데이터베이스 관리시스템(MM-DBMS)과 그 기능 및 성능 면에서 다르다. 예를 들어, 충분한 주기억 공간을 갖는 디스크 기반 데이터베이스 시스템의 경우, 데이터 접근을 위해서는 실제 물리적 디스크 주소가 결정되어야 하고, 이 데이

† 정 회 원 : (주)알티베이스 이사/연구소장
†† 정 회 원 : 상지대학교 컴퓨터정보공학부 교수
††† 종신회원 : 인하대학교 전자계사학과 교수
논문접수 : 2003년 5월 21일, 심사완료 : 2003년 7월 25일

터 블록이 버퍼에 위치하는지 검사한 후, 해당 데이터 블록이 프로세스에게 전달된다. 반면에 순수한 주기억 상주 데이터베이스 시스템에서는 모든 데이터가 주기억장치에 있음이 보장되므로, 모든 데이터의 요구는 단순히 가상 기억장치의 주소로 직접 변환하여 사용하면 된다. 즉, 모든 저장장치가 계층적 구조에서 평면구조로 변환되어 버퍼 관리기를 통할 필요가 없으므로, 그 만큼 성능상에서 유리하다. 디스크 공간을 주 저장장치로 설계된 시스템에서는 충분한 주기억장치 공간을 갖는다 하여도 순수 주기억 상주 데이터베이스 시스템에서의 성능 보다 우수하지 못함을 이미 다른 연구에서도 보였다[4, 5].

주기억 상주 데이터베이스 시스템은 기존의 데이터베이스 시스템에서 지원되는 병행수행 제어, 인덱스 관리, 로그 및 회복 기능 등 모든 기능을 지원해야 하고 아울러 기본 저장장소로서 주기억장치를 사용한다. 그러나, 기존 시스템에서 사용하던 기법들은 기본 저장장소가 기억장치로 변환되면서 직접적으로 적용하기 어려운 점이 많다[3]. 본 ALTIBASE^{TM} 시스템은 이러한 문제점들을 해결하여 실시간 응용 및 고성능 트랜잭션 처리율을 요구하는 산업 분야에 실제 적용될 수 있는 주기억 상주 DBMS로 개발되었다.

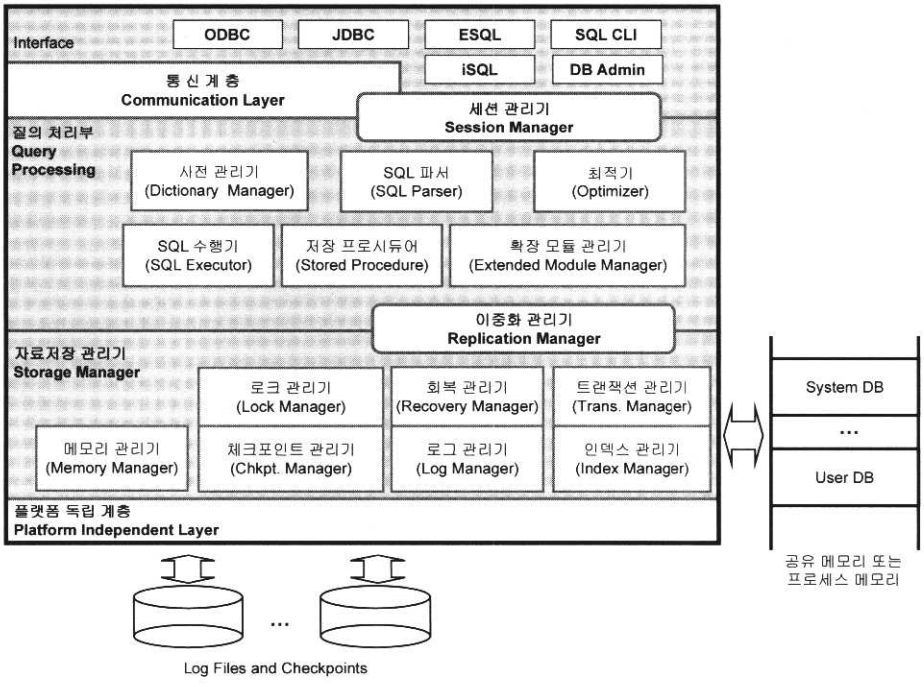
본 논문에서는 ALTIBASE^{TM}의 자료저장 관리자가 고성능 고가용성을 위해 사용한 여러 기법들을 소개한다. 본 논문의 구성은 다음과 같다. 2장에서 ALTIBASE^{TM} 시스템의 트랜잭션 및 회복 기능, 인덱스 관리 기능, 이중화 기법 등을 포함한 특징적 주요 기능을 소개하고, 전체적인 시스템 구조를 설명한다. 3장에서는 ALTIBASE^{TM} 시스템 구조 중에서 자료저장 관리기가 제공하는 기법, 즉, 트랜잭션 관리 기법, 로깅 메커니즘 및 검사점(checkpoint)을 포함한 회복 및 로깅 기법 등을 지원하기 위한 설계 고려 사항을 다루며, 4장에서 고가용성 및 고성능 트랜잭션 처리를 위한 데이터베이스 이중화 기법 및 이 때 발생하는 트랜잭션 충돌 상황에 대한 해결방법을 설명한다. 또한 5장에서 본 ALTIBASE^{TM} 시스템을 통한 단위 트랜잭션 처리율을 시스템 부하의 변화에 따라 측정함으로써 본 시스템이 실시간 응용 분야나 시간제약을 갖는 응용 분야에 적합한 성능을 제공함을 보인다. 끝으로 6장에서 결론을 맺는다.

2. ALTIBASE^{TM}의 주요 기능 및 시스템 구조

2.1 구조적 특징 및 ALTIBASE^{TM} 시스템 구조도

2.2.1 ALTIBASE^{TM}의 구조적 특징

ALTIBASE^{TM}는 주기억장치를 사용하는 관계형 데이터베이스 시스템으로서, 범용적 응용 뿐만 아니라 특정 실시간 응용에도 적합한 클라이언트-서버 구조와 응용 내장형 구조, 멀티 쓰레드 구조, 그리고 서버 연결 풀 구조를 제공한다. 클라이언트-서버 구조를 기반으로 응용 프로그램은 다양한 통신 방법을 사용하여 서버에 접속할 수 있으며 또한 응용 프로그램을 데이터베이스 서버에 내장함으로써 응용 프로그램과 서버간의 통신 비용을 제거하는 응용 내장형 구조를 지원한다. 멀티 쓰레드 구조를 기반으로 프로세스간 문맥 교환 시간을 제거하였으며, 사용자의 증가에 따른 시스템 자원을 감소할 수 있는 구조를 지원한다. 또한



(그림 1) ALTIBASE^{TM} 시스템 구조

서버 기능을 갖는 다수의 시스템 쓰레드를 서비스 쓰레드 풀로 구성하여 서버 연결 풀과 연동되게 구성하여 한정된 개수의 쓰레드를 통하여 다수의 동시 사용자에게 서비스를 극대화 할 수 있는 서버 연결 풀 구조를 갖는다.

ALTIBASE™ 시스템의 구조는 (그림 1)과 같이 인터페이스 부분, 통신 계층, 질의 처리부, 그리고 자료저장 관리기로 분류할 수 있다.

그 중 데이터베이스 시스템의 가장 핵심이 되는 서비스를 관리하고 있는 자료저장 관리기는 다시 로크 관리기, 회복 관리기, 트랜잭션 관리기, 메모리 관리기, 체크포인트 관리기, 로그 관리기 및 인덱스 관리기로 구성된다.

2.2 ALTIBASE™ 시스템의 주요 기능

2.2.1 트랜잭션 관리 기능

주기억 상주 데이터베이스 시스템은 전통적인 디스크 기반 시스템보다 수 십 배 이상의 트랜잭션 처리율을 보이므로 병행수행 제어에 소요되는 시간적 비용이 적고 전체적인 성능을 향상시킬 수 있어야 한다. 그러나 로크 기반 병행수행 제어 기법은 완료되지 않은 데이터의 읽기 금지와 갱신 연산에 따른 예측 불가능한 대기 시간, 로크 상승(escalation)의 처리에 소요되는 시간적 비용 때문에 실시간 응용에서 요구하는 성능을 만족하기 어렵다.

ALTIBASE™ 시스템에서 트랜잭션을 처리하기 위한 기본적인 병행수행 제어 방법으로 다중 버전(multi-version) 기법을 이용한 병행수행 제어 방법을 사용한다. 따라서, 다중 버전 기법의 기본 전략에 따라 판독 전용(read-only) 트랜잭션이 많고, 갱신 트랜잭션의 비율이 적은 응용 분야에서 우수한 트랜잭션 처리율을 보인다[1, 2]. 또한 본 시스템에서 사용하는 병행수행 제어 방법에서는 데이터의 획득 단위(granularity)를 여러 단위로 설정하여, 보다 큰 단위의 데이터(예, 테이블 단위)에 대해서는 로크 기반 병행수행 제어를 적용하고, 보다 낮은 단위의 데이터(예, 레코드 단위) 접근에 대해서는 기본적인 다중 버전 병행수행 제어 기법을 적용하도록 하였다. 따라서 데이터 접근 단위에 따라 분리된 병행수행 제어 기법이 되므로 로크 기반의 예측가능하지 못한 데이터의 기다림을 방지할 수 있어 보다 높은 병행성을 지원할 수 있으며, 빠른 성능을 목적으로 하는 응용분야의 요구에 부합할 수 있도록 설계하였다. 또한 트랜잭션의 고립화 수준을 데이터베이스 일관성에 따라 약화시킬 수 있어, 특정 응용 분야에 적합하게 더욱 높은 성능을 제공할 수 있으며, ALTIBASE™의 구조적 특징인 서버 쓰레드 풀 구조에 의하여 동시 사용자 수의 증가에 따라 접속자 수와 서비스 쓰레드를 1 : 1 구조에서 M : M 구조로 자동 변환하게 하여 시스템 확장성(system scalability)과 서비스 확장성(service scalability)을 보장한다.

또한, 다중 버전 기법의 단점인 여러 복사본에 의한 사용되지 않는 데이터 버전의 잔재를 제거하기 위하여 불필요한 오래된 버전의 데이터를 파기할 수 있도록 "*garbage collection thread*"를 구현하였다. 디스크 기반 시스템에서 보다 더 효율적으로 저장장치를 관리해야 한다는 점에서 불필요한 데이터 버전의 수거는 반드시 필요한 기능이다.

2.2.2 회복 관리 기능

주기억 상주 데이터베이스 시스템은 주 저장소의 휘발성 특징을 고려할 때, 디스크 기반 시스템보다 더욱 안정성을 요구한다. ALTIBASE™의 회복기능은 트랜잭션의 ACID 특성을 완벽하게 지원하므로, 그 중 트랜잭션의 영속성(durability)을 지원하기 위해 디스크와의 동기화 과정을 필수적으로 사용해야 한다. 그러나 이와 같은 디스크 동기화 과정은 디스크 접근에 따르는 성능 저하를 유발하게 된다. 따라서, 트랜잭션의 로그 레코드를 관리하기 위한 기법으로 메모리 맵드 파일 또는 메모리 버퍼를 로그 버퍼로서 선택적으로 사용할 수 있게 하였으며, "*log flush thread*"를 구현하여 현재 수행 중인 트랜잭션에 간섭없이 로그 레코드들을 디스크에 존재하는 로그 파일로 동기화 할 수 있도록 구현하였다.

또한 응용 시스템의 특징에 따라 트랜잭션의 영속성보다 성능을 중요시하는 경우를 위하여 세 가지의 로깅 레벨을 제공하며, 이에 따라 관리해야 하는 로그 레코드를 제어할 수 있으며 디스크 동기화 수준 또한 제어할 수 있다.

2.2.3 메모리 관리 기능

주기억 상주 데이터베이스의 주 저장장치가 메모리이므로 메모리의 효율적인 관리는 시스템의 전체 성능에 직접적인 영향을 주는 최대 주요 요인이 된다. ALTIBASE™ 시스템에서는 메모리를 크게 영구공간(persistent space)과 임시공간(temporary space)으로 구분한다. 영구공간은 실제 사용자 데이터와 메타 정보에 대한 내용을 저장하기 위하여 사용되며 인덱스 데이터나 질의 처리시 필요한 공간은 임시 공간을 사용한다. 이러한 공간 할당 및 해제를 위한 메모리 연산 역시 시스템 성능에 큰 영향을 주기 때문에 ALTIBASE™는 메모리 풀을 이용한 메모리 관리 모듈을 구현하여 메모리 관련 시스템 호출연산에 의한 성능저하를 제거하였다.

또한 모든 사용자 데이터 및 메타 정보가 메모리에 적재되어 있는 상태이지만, 실제로 서비스 운영이 사용되는 사용자 데이터는 일부분에 지나지 않는 경우가 많다. 따라서, 사용되지 않는 데이터를 선택적으로 로딩하게 하여 메모리 공간을 효율적으로 사용할 수 있는 선택적 로딩 기법을 지원한다. 대량의 데이터에 대한 변경 및 삽입, 삭제 연산이 수행되다 보면 실제 공간 보다 더 많은 메모리를 요구하는 경우가 발생할 수 있다. 이러한 경우 ALTIBASE™은 테이블 압축(table compaction) 기능을 제공하여 불필요한 메모

리 공간을 시스템에 반환할 수 있도록 설계하였다.

2.2.4 인덱스 관리 기능

기존 데이터베이스 시스템에서 제공하는 인덱스 기법의 주목적은 디스크 접근 회수를 최소화하는 것인 반면, 주기억 상주 데이터베이스 시스템에서는 최소한의 메모리 공간을 활용하면서 인덱스를 처리하는데 소요되는 시간 비용을 최소화하는 것이다. 대부분의 CPU 사용시간은 반 이상이 메모리 접근에 소요되는 대기시간으로 캐쉬 메모리의 적중률이 낮아 발생한다. 따라서 ALTIBASE™ 시스템은 캐쉬 계층의 최적화 기법과 캐쉬를 고려한 인덱스의 병행수행 제어 방법을 적용한 개선된 R^*-Tree, T-Tree 및 B^+-Tree를 제공하여 캐쉬 적중률을 높였으며, 전체 트랜잭션의 수행 성능을 향상시켰다.

2.2.5 이중화 기능

고가용성(high availability) 지원을 위해 물리적으로 분리되어 있는 여러 데이터베이스에 지역 데이터베이스의 변경 내용을 원격지로 복제할 수 있는 이중화(replication) 기능을 지원한다. ALTIBASE™ 시스템에서는 데이터베이스의 변경 로그를 기반으로 하는 점대점(point-to-point) 이중화 기법을 적용한다. 즉, 지역 서버에서는 데이터베이스의 변경로그를 XLOG라는 실행계획으로 원격서버에 전송하고, 원격서버에서는 이 로그를 마치 회복하는 방법처럼 이중화를 수행한다. 또한 N-way 이중화를 제공함으로써 네트워크 구조의 이중화 위상을 제공할 수 있다.

2.2.6 질의 처리 기능

기존에 구현 개발된 주기억 상주 데이터베이스 시스템들은 질의 처리 언어(SQL)를 제공하지 않거나 SQL의 일부기능을 지원할 수 있는 API만을 제공하여 응용 프로그램 작성을 어렵게 한다. ALTIBASE™ 시스템은 표준 SQL 2를 완벽히 지원하며, 이의 처리를 위해 비용 및 규칙(rule) 기반 최적기, 중첩 반복 조인, 해쉬 및 정렬을 이용한 조인, 정렬-합병(sort-merge) 조인 연산처리 기능 등을 지원하는 질의 처리기를 개발하여 적용하고 있다.

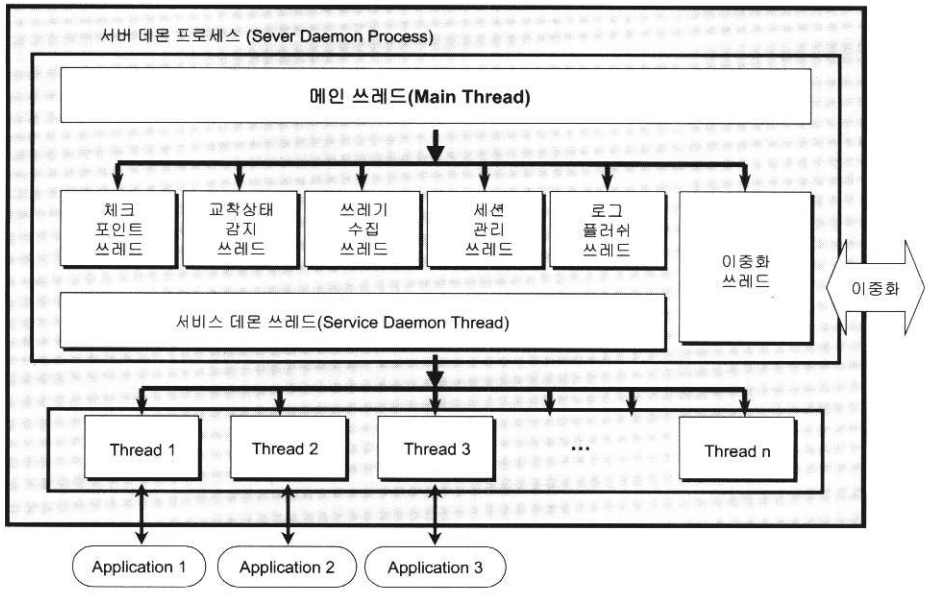
2.2.7 응용 프로그래밍 인터페이스

상용 주기억 상주 데이터베이스 시스템 분야는 개발 역사가 오래되지 않아 프로그래밍 인터페이스의 기능을 충분히 제공하지 못하고 있다. ALTIBASE™ 시스템은 다양한 산업 표준 응용 프로그래밍 인터페이스를 제공하고 있다. X/OPEN CLI 표준사양을 따르는 SQL CLI 인터페이스, ODBC 및 JDBC 인터페이스, 내재된 SQL 프로그래밍을 지원하는 ESQL 인터페이스, 서버내장형 SQL 기반 인터페이스, 관리자 인터페이스 및 대화형 SQL 인터페이스 등을 제공하여 디스크 기반의 상용 데이터베이스 시스템에서 제공하는 대부분의 프로그래밍 인터페이스를 지원한다.

3. 고성능 자료저장 관리기의 구조 및 설계

3.1 서버 프로세스의 구성도

2장의 (그림 1)에 나타난 바와 같이 ALTIBASE™ 시스템의 자료저장 관리기는 여러 세부 관리기들에 의해 구성되며, 이 들 구조는 멀티 쓰레드 구조와 서버 연결 풀 구조에 의하여 시스템 자원을 최소화 하고, 다수의 동시 사용자를 지원하여 고성능 트랜잭션 처리를 가능하게 한다. 또한 각 세부 모듈별 독립성을 고려하여 계층형 구조로 설계하



(그림 2) ALTIBASE™ 시스템의 서버 프로세스 구성도

였으며, 각 모듈의 단순화를 통해 전체적인 성능 향상을 이루었다. ALTIBASE™ 시스템의 핵심 기능을 제공하고 있는 자료저장 관리기의 서버 프로세스의 구성도는 (그림 2)와 같다.

또한 플랫폼 독립 계층에 의해 다양한 플랫폼에 이식이 가능하며 현재 모든 유닉스 계열 시스템, 윈도우 NT 시스템, 그리고 RTOS(QNX 등)와 같은 실시간 운영체제에서 운영 가능하다.

3.2 트랜잭션 관리기의 설계

ALTIBASE™ 시스템의 트랜잭션 관리기는 트랜잭션 테이블(Transaction Table)과 트랜잭션 프리 리스트(Transaction Free List)를 관리하는 책임을 지고 있다. 트랜잭션 테이블의 각 엔트리는 하나의 트랜잭션이 생성되는 경우 할당되며, 각각의 트랜잭션에 대한 정보를 보관한다. 트랜잭션이 완료 또는 철회되는 경우 테이블 엔트리를 반환한다. 이 때, 트랜잭션 엔트리를 할당 받고자 하는 각 트랜잭션이 트랜잭션 테이블을 동시에 접근하게 되어 트랜잭션의 병행 수행에 큰 지장을 주게 되므로, 트랜잭션 프리 리스트를 사용하여 트랜잭션 엔트리를 연결리스트(linked list)로 구성하여 관리한다.

트랜잭션 테이블의 정보는 크게 트랜잭션에 관한 정보, 로크에 관한 정보, 회복에 관한 정보로 구분 되어진다. 트랜잭션에 관련된 정보로는 트랜잭션 ID, 트랜잭션의 현재 상태, 트랜잭션의 고립화 수준, 완료 일련 번호, 변경 객체에 대한 OID 리스트 등을 관리하게 되며, 회복에 관련된 정보로는 갱신 연산 수행 여부를 나타내는 상태 값, 트랜잭션의 최초 로그 일련 번호(LSN)와 최종 LSN, 로그 버퍼를 위한 할당된 메모리 공간의 포인터 등을 관리한다.

트랜잭션이 시작되는 경우 트랜잭션 프리 리스트에서 하나의 엔트리를 할당 받고, 트랜잭션이 관리할 OID 리스트 및 저장점(savepoints)의 메모리 사용공간을 초기화 하고, 현재 트랜잭션의 상태를 "begin" 상태로 설정하게 된다. 트랜잭션이 완료하는 경우에는 갱신 연산을 수행한 트랜잭션에 대하여 완료 로그를 생성하고, 트랜잭션이 관리하던 OID 리스트를 버전 관리기로 넘겨 완료 일련 번호(Commit_SCN)를 할당 받게 된다. 버전 관리기는 갱신된 OID에 대하여 접근할 수 없는 버전인 경우 공간을 해제하여 사용할 수 있는 메모리 공간으로 반환한다. 또한 트랜잭션의 상태를 "*commit*" 상태로 설정한 후, 트랜잭션이 획득한 모든 로크를 해제하고, 트랜잭션 엔트리를 반환한다. 그 후 트랜잭션의 상태는 "*end*" 상태가 된다. 트랜잭션이 철회하는 경우에는 그 트랜잭션이 갱신한 모든 레코드들에 대해 취소 연산을 수행한 후 철회 로그를 생성하며, 트랜잭션이 관리하던 갱신 객체에 대한 목록 OID 리스트를 버전 관리기에게 넘겨주고 완료 일련 번호를 할당 받으며 트랜잭션의 상태는 "*rollback*" 상태가 된다. 그 후, 트랜잭션이 소유한 모든 자원을 해제 한 후 테이블 엔트리를 반환한다.

3.2.1 트랜잭션 고립화 수준

ALTIBASE™ 시스템의 트랜잭션 관리기는 "*consistent*", "*repeatable*", "*no_phantom*"과 같은 세 가지 고립화 수준을 제공한다.

- Consistent : 트랜잭션이 현재 읽은 데이터는 완료된 트랜잭션에 의한 것임을 보장한다. 이 수준은 참고문헌 [2]의 갱신 일관성(update consistency)을 보장한다. 즉, 판독 전용 트랜잭션을 하나 포함하는 직렬화 그래프 사이클(serialization graph cycle)이 존재할 수 도 있다. 그러나 갱신 트랜잭션 입장에서 일관된 뷰를 갖는다.
- Repeatable : 트랜잭션이 한 데이터에 대해 여러 번 읽기 연산을 수행하여도 언제나 같은 값을 판독할 수 있음을 보장하는 고립화 수준이다. 이 수준은 약한 일관성(weak consistency)을 보장한다[2]. 즉, 두 개 이상의 판독 전용 트랜잭션을 포함하는 직렬화 그래프의 사이클을 허용한다. 각각의 판독 트랜잭션 입장에서 일관된 뷰를 보장한다.
- No_phantom : 일반 데이터베이스 시스템에서 트랜잭션 수행의 기준으로 사용하는 직렬화 가능성(serializability) [9]을 보장한다.

이와 같은 트랜잭션의 고립화 수준은 트랜잭션이 판독하는 데이터의 일관성 보다 성능을 중요시하는 특정 응용 시스템의 목적에 따라 다르게 설정하여 트랜잭션의 성능을 최대화 할 수 있도록 한다. 각 고립화 수준의 설정에 따라 로크 관리에서 획득하는 로크의 종류는 <표 1>과 같다.

〈표 1〉 고립화 수준에 따른 로킹 연산

연산 \ 고립화 수준	Consistent	Repeatable	No-Phantom
읽기 연산	IS	S	S
쓰기 연산	IX	IX	X

IS : Intention Shared Lock , IX : Intention Exclusive Lock,
S : Shared Lock, X : Exclusive Lock

3.2.2 트랜잭션 저장점(Transaction Savepoints)

ALTIBASE™ 시스템은 부분 철회(partial rollback)를 지원하기 위하여 두 종류의 저장점을 제공한다. 명시적 저장점(explicit savepoint)은 사용자의 요구에 의해 설정되며 암시적 저장점(implicit savepoint)은 시스템 운영 목적에 따라 트랜잭션 관리기에 의해 설정된다. 저장점 설정은 디스크의 백업 데이터베이스 파일로 동기화를 유발하므로 이 때 현재 수행중인 트랜잭션과의 병행성 제어 문제 때문에 심각한 성능 저하를 유발할 수 있다. 따라서 ALTIBASE™ 시스템에서는 두 가지 저장점을 제공한다. 동기화 시점을 시스템과 사용자가 설정할 수 있도록 하여 현재 수행중인 트랜잭

션에 부하를 주지 않은 채 동기화 과정을 수행할 수 있도록 하였다. 두 가지의 저장점의 비교는 <표 2>와 같다.

〈표 2〉 ALTIBASE™ 시스템의 저장점

	명시적 저장점	암시적 저장점
저장점 이름	사용자에 의해 지정	NULL
설정 및 저장점으로의 철회	사용자에 의해 설정	트랜잭션 관리기에 의해 수행
설정 개수	제한 없음	트랜잭션당 하나
로그 생성 여부	로그 생성	로그 없음

3.3 회복 관리기의 설계

ALTIBASE™ 시스템의 회복 관리기는 각종 고장에 대한 회복을 위해 시스템의 정상 수행중의 모든 데이터베이스 변경 연산에 대하여 로그를 기록하며, 이를 활용하여 올바른 데이터베이스 상태로 복구하는 기능을 제공한다. 또한 로깅시 ARIES 시스템[7, 8]의 WAL 프로토콜에 따라 다양한 경우의 트랜잭션을 회복 시점에 정확히 반영한다. 따라서 이러한 로그를 관리하기 위하여 회복 관리기 내에 로그 관리기를 구현하여 데이터베이스 갱신 연산에 대한 모든 로그들을 관리할 수 있도록 한다.

로그 관리자의 중요한 자료 구조인 로그 앵커(log anchor)는 로그의 현재 상태와 관련된 주요 정보를 저장하는 메모리 객체로서, 서버가 재시작하는 경우 서버의 종료 시점의 올바른 데이터베이스 상태로 반영하기 위한 최소 정보들을 저장한다. 로그 앵커는 회복에 관련된 주요 정보를 담고 있으므로 데이터베이스의 일관성 유지를 위해 디스크와 같은 장치에 저장하게 되는데 이를 로그 앵커 파일이라 한다. 여러 트랜잭션이 수행중인 환경에서 이와 같은 로그 앵커를 동시 접근하게 되므로 이에 대한 병행수행 제어를 담당하는 것이 로그 앵커 관리기이다. 로그 앵커 관리기는 로그 앵커에 대한 생성 및 삭제 연산, 초기화 연산, 변경 및 디스크 파일로의 플러쉬, 로그 앵커의 온라인 백업 등을 지원한다.

3.3.1 로그 파일 관리

주기억 상주 데이터베이스 시스템은 모든 데이터가 메모리에 상주하므로 트랜잭션이 생성한 로그를 기록할 파일 및 회복에 사용될 로그 파일 전체가 메모리 내에 존재한다. 그러나 모든 로그 파일이 메모리에 적재되는 것이 아니라, 현재 로그 기록에 사용되는 파일과 트랜잭션 철회에 필요한 로그 파일만 즉, 현재 접근 요구가 있는 로그 파일들만 메모리 내에 존재하게 된다. 디스크 기반 데이터베이스 시스템에서는 별도의 로그 버퍼를 운영하여 로그 레코드들을 디스크로 저장하지만 ALTIBASE™ 시스템에서는 로그 파일이 적재된 메모리 포인터만을 이용하여 로그 관련 I/O 작업을 수행하게 된다. 또한 다중 로그 파일을 지원하여 사용중인 로그 파일의 여분이 없는 경우, 다른 로그 파일을 사용할 수 있도록 한다. 로그 파일이 메모리에 적재되는 시기는 시스템 가동시와 로그 파일에 대한 첫번째 접근 요구가 있는 경우이다. 즉, 다른 로그 파일의 사용을 요구하는 경우와 회복시 필요한 로그를 다른 로그 파일로부터 읽어오게 되는 경우이다. 메모리에 적재된 로그 파일은 해당 파일을 닫게 되는 경우는 로그 파일의 EOF로 인해 더 이상 기록할 수 없는 경우와 회복시 필요한 로그 레코드를 해당 로그 파일로부터 모두 읽은 경우이다.

3.3.2 *Log Sync Thread*의 설계

주기억 상주 데이터베이스 시스템은 로그 파일이 메모리에 위치하므로 모든 레코드들을 휘발성이 강한 저장장치에 보관할 수 밖에 없다. 예기치 못한 시스템 고장시 회복이 불가능한 상황을 유발할 수 있으므로, 비록 주기억 상주 데이터베이스 시스템이라 할지라도 메모리내의 로그화일을 주기적으로 영속성을 지닌 디스크에 반영시켜야만 한다[6, 10]. 이러한 디스크 기록 작업은 현재 수행중인 트랜잭션에 치명적인 성능 저하를 유발할 수 있으므로 별도의 쓰레드로 동작하게 구현하였다. "*Log Sync Thread*"는 시스템 가동시 대기상태로 존재하며 주기적으로 동작하여 메모리에 적재된 로그 파일을 디스크에 반영하고, 참조되지 않는 로그 파일들을 메모리에서 제거하는 작업을 수행한다. 이 쓰레드는 "*SyncLogFileList*" 자료구조를 통해 로그 파일 리스트를 관리하며 이 리스트상에 존재하는 로그 파일들에 대해 주기적인 디스크 기록 작업을 수행한다. 리스트상의 로그 파일들은 로그 파일의 기록 용량을 모두 채운 "*Full Log File*"과 현재 작업중인 로그 파일, 그리고 사용되지 않은 "*Empty Log File*"로 구별된다. "*Full Log File*"은 로그 파일 전체를 디스크에 동기화 한 후, "*SyncLogFileList*"에서 제거하고 더 이상 참조되지 않는 경우 메모리 내에서 완전히 삭제한다. 현재 사용중인 로그 파일에 대해서는 기록된 로그가 포함된 페이지까지 디스크 동기화를 수행하고, "*Empty Log File*"에 대해서는 디스크 동기화 작업을 수행하지 않는다. 또한 이 쓰레드는 "*Checkpoint Thread*"에 의해서도 호출되어 수행될 수 있으며, 이 경우 병행수행 제어를 위해 *MUTEX*를 획득한 후 수행하도록 설계되었다.

3.3.3 검사점(checkpoint) 및 회복 기법의 구현

검사점은 오류 회복시 소요되는 시간과 비용을 줄이기 위해 시스템의 정상수행 과정 중에 시스템 차원에서 데이터베이스의 일관된 상태를 만드는 작업이다. 검사점 수행 시 변경된 모든 데이터 페이지가 백업용 디스크 데이터베이스로 기록되므로, 현재 수행중인 트랜잭션의 수행에 영향을 미칠 수 있다. ALTIBASE™ 시스템은 퍼지 검사점(fuzzy checkpoint)과 더불어 핑퐁 검사점(ping-pong checkpoint)를 함께 제공하여 두 가지 백업 디스크를 교체적으로 번갈

아 사용하여 현재 수행중인 트랜잭션에 부하를 주지 않고 수행할 수 있도록 설계하였다.

또한 검사점 수행방법은 검사점 수행 시점에서의 변경 페이지 목록과 현재 수행 중인 트랜잭션들의 리스트를 구성하여 회복시 필요한 정보를 제공하는 것이다. 기존 ARIES 방법[7, 8]에서는 검사점 수행시 현재 수행중인 활동 트랜잭션 정보를 구성하여 "UNDO" 대상 트랜잭션으로 활용하지만, 트랜잭션 테이블을 모두 조사하여 활동 트랜잭션 정보를 구성하는 작업은 비용과 시간이 많이 드는 작업이므로, ALTIBASE™ 시스템에서는 검사점 수행시 활동 트랜잭션 목록을 구성하지 않고, 다만 최소 로그 일련 번호만을 저장하여, 회복시 최소 로그 일련 번호에 위배되는 트랜잭션들에 대한 "UNDO" 작업을 수행할 수 있도록 설계하였다. 또한 퍼지 검사점과 핑퐁 검사점을 적용하여 현재 수행중인 트랜잭션의 수행에 영향을 미치지 않으면서, 검사점 이전의 모든 데이터 변경은 디스크에 반영할 수 있도록 구현하였다.

ARIES 시스템의 회복 기법은 분석 단계(analysis phase), 재수행 단계(redo phase), 취소 단계(undo phase)로 구성되어 있으며, 분석 단계에서는 재수행 및 취소 단계에서 필요한 변경 페이지 목록과 활동 트랜잭션 목록을 만들고 재수행 단계의 시작점을 결정한다[7, 8]. 재수행 단계에서는 로그 목록상의 이후 사본(after image)을 이용하여 재수행 하며, 취소 단계에서는 분석 단계에서 제공하는 트랜잭션 목록에 대해 "UNDO" 작업을 수행한다. ALTIBASE™ 시스템의 회복 기법에서는 분석 단계에서 갖는 시간과 비용을 절감하기 위하여 재수행 단계와 취소 단계 만으로 회복 작업을 수행한다. 검사점 시작 로그에 기록된 최소 로그 일련 번호를 기준으로 기존의 분석 단계에서 제공하는 활동 트랜잭션 목록을 쉽게 찾아 낼 수 있으므로 재수행 단계와 취소 단계를 바로 수행할 수 있다.

3.3.4 트랜잭션 영속성 수준(durability level)과 로깅 수준(logging level)의 설계

앞에서 언급하였듯이 ALTIBASE™ 시스템은 로그 버퍼를 주저장소인 메모리 내에 적재하여 사용하지만, 로그 버퍼의 내용은 회복시 필수적인 내용이므로 디스크의 동기화를 필수적으로 수행하여야 한다. 따라서 ALTIBASE™ 기본적인 로그 버퍼로서 메모리 맵드 파일(memory mapped file)을 제공한다. 메모리 맵드 파일은 로그 버퍼로 사용하는 경우 디스크의 입출력이 느리거나 운영체제의 부하가 많은 경우 ALTIBASE™ 시스템 전체의 성능 저하를 유발할 수 있게 된다. 따라서 트랜잭션의 영속성 수준에 따라 메모리 버퍼와 메모리 맵드 파일 두 종류의 로그 버퍼를 선택적으로 사용할 수 있게 하여, 데이터베이스 응용 시스템을 관리자의 제어에 따라 조절할 수 있게 구현되었다. 트랜잭션의 영속성은 모두 5가지 수준으로 제공되며 각 수준에 따른 로그 버퍼 및 기능은 <표 3>과 같다.

〈표 3〉 ALTIBASE™ 시스템의 트랜잭션 영속성 수준

트랜잭션 영속성 수준	로그 버퍼 및 디스크 동기화	기능 설명
Level 1	메모리 버퍼	로그는 메모리 버퍼에만 반영되며 변경 내용을 디스크에 수행하지 않는다.
Level 2	메모리 버퍼 디스크 로그 파일 Log Sync Thread 동작	로그는 메모리 버퍼에 반영되고 Log Sync Thread에 의해 로그 파일에도 주기적으로 반영된다. 디스크의 로그 파일에 로그가 반영된 것을 보장하기 전에 트랜잭션의 완료가 선언되므로 완료 트랜잭션의 영속성을 보장하지 않는다.
Level 3	메모리 맵드 파일	모든 로그는 디스크에 반영된다. 운영체제의 파일 버퍼가 적용되므로 트랜잭션의 영속성을 보장한다.
Level 4	메모리 버퍼 메모리 맵드 파일 Log Sync Thread 동작	로그는 메모리 버퍼에 반영되고 Log Sync Thread에 의해 메모리 맵드 파일에도 주기적으로 반영된다. 메모리 맵드 파일에 로그가 기록된 것을 보장하기 전에 트랜잭션의 완료가 선언되므로 완료 트랜잭션의 영속성을 보장하지 않는다
Level 5	메모리 버퍼 디스크 로그 파일 Log Sync Thread 동작	로그는 메모리 버퍼에 반영되고 Log Sync Thread에 의해 로그 파일에도 주기적으로 반영된다. 트랜잭션의 완료는 완료 로그를 포함한 모든 로그가 디스크 로그 파일에 기록된 후에 선언된다.

또한 트랜잭션의 변경 연산에 따른 다양한 로그들을 기록하며, 오류 회복시 재수행 작업과 취소 작업의 양을 줄이기 위해 검사점을 주기적으로 수행한다. 따라서 오류에 대비하기 위하여 불필요한 로그의 기록 때문에 전체적인 성능 저하를 유도할 수 있다. ALTIBASE™ 시스템에서는 이러한 상황을 효과적으로 적용할 수 있도록 로그의 수준을 모두 세 가지로 분류하여 트랜잭션의 영속성에 따라 적절히 적용할 수 있도록 구현하였다. 즉 트랜잭션의 영속성 수준이 0인 경우, 모든 로그를 기록하는 것은 불필요한 작업이 되므로 기록하는 로그의 수준에 따라 <표 4>와 같은 로깅 수준을 제공한다.

〈표 4〉 ALTIBASE™ 시스템의 로깅 수준

로깅 수준	로그 내용	기능 설명
Level 0	로그를 기록하지 않음	변경을 유도 하는 트랜잭션에 대한 회복은 불가능하다. 트랜잭션의 영속성은 마지막 검사점까지만 보장된다. 이후에 완료된 트랜잭션은 데이터베이스에 반영되지 않는다.
Level 1	DML(Insert, Update, Delete) 연산에 대한 로그	변경연산에 대한 취소 연산을 수행하기 위한 로그를 기록한다. 검사점 이후에 완료한 트랜잭션은 데이터베이스에 반영되지 않는다.
Level 2	모든 로그 기록	모든 로그를 기록하며 모든 오류 상황에 대해서 회복이 가능하다

4. 고가용성을 위한 데이터베이스 이중화의 설계

4.1 데이터베이스 이중화 구조 및 기능

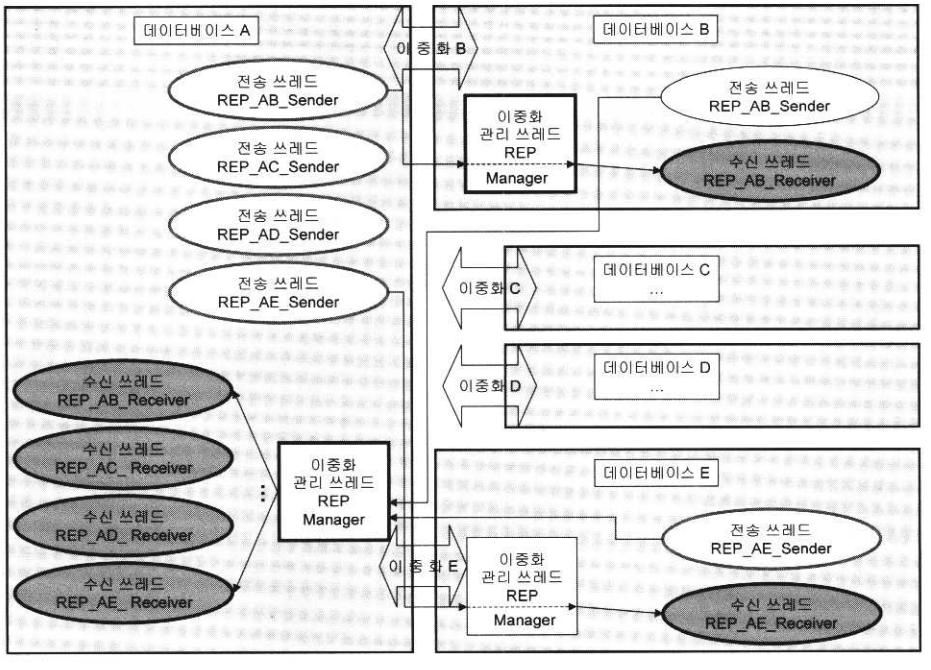
실시간 응용 분야에서 적용되는 데이터베이스 시스템에서는 시스템의 예기치 못한 고장이나 미디어의 파손으로 인해 치명적인 경제적 손실을 야기할 수 있으므로 시스템의 고가용성(high availability)과 안정성(stability)은 필수적인 기능이라 할 수 있다[11, 12]. ALTIBASE™ 시스템은 고가용성과 안정성 제공을 위해 데이터베이스 이중화(replication) 기능을 지원한다. ALTIBASE™ 이중화 기능은 다음과 같은 특징을 지닌다.

- 최소 비용 : 데이터 복제는 로그 레코드를 기반으로 수행되므로 추가적인 데이터 관리가 필요 없어 지역 서버의 작업을 방해 하지 않아 성능상의 오버헤드를 줄일 수 있다.
- 신뢰성 : 지역 서버와 원격 서버간의 네트워크 오류나 원격 서버에 오류가 발생하면, 오류 회복 후에 전송하지 못한 로그 레코드를 재전송하여 복제 데이터베이스의 일치성을 유지할 수 있다.
- 독립성 : 상대 서버에 서로 종속적인 관계가 없기 때문에 상대 서버의 오류 또는 네트워크 장애가 발생하여도 독자적인 기능을 수행할 수 있다.
- 다중 복제 : 한 지역 서버가 여러 원격 서버에 데이터베이스 복제 기능을 수행할 수 있다.

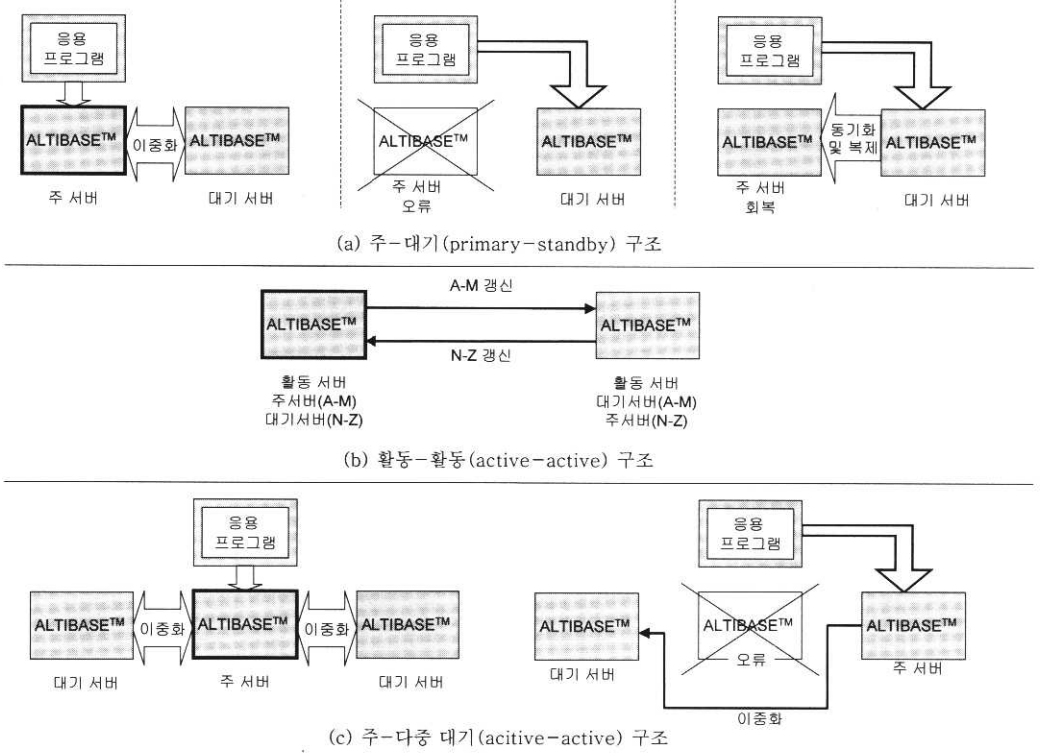
ALTIBASE™ 시스템의 이중화 기능은 테이블 단위로 이루어지며 한 서버에 16개의 복제 객체를 정의할 수 있으며, 한 객체에는 여러 개의 테이블을 포함할 수 있다. ALTIBASE™ 시스템의 이중화 기능을 위한 쓰레드 구조는 복제를 구성하는 서버들간의 통신을 담당하는 관리 쓰레드(Rep_Manager)와 복제 데이터를 원격 서버에 보내는 전송 쓰레드(Rep_XY_Sender), 원격 서버의 복제 데이터를 전달 받아 자신의 데이터베이스에 반영하는 수신 쓰레드(Rep_XY_Receiver)로 구성되며, "XY"는 각 지역 서버의 고유 식별자를 나타낸다. 이와 같은 쓰레드 구조는 (그림 3)과 같이 도식화 할 수 있다.

ALTIBASE™ 시스템의 이중화 관리기는 한 지역 서버가 최대 16 개의 원격서버와 데이터 복제를 수행할 수 있으며, 지역 서버에는 각각의 복제에 해당하는 전송 쓰레드(REP_A*_SENDER)가 생성되고, 원격 서버에는 이에 대응하는 수신 쓰레드(REP_A*_RECEIVER)가 생성되며, 원격지 데이터베이스 전송 쓰레드와의 서버간 동기화는 이중화 관리기에 의해서 수행된다. 동기화가 이루어지면 실제 데이터 송수신은 이중화 관리기를 통하지 않고 전송 쓰레드와 수신 쓰레드가 직접 담당한다. 이중화 관리기는 각 데이터베이스 마다 하나씩 존재하며 서버 구동시 자동으로 생성된다. 또한 표준 SQL 문장에 기반한 인터페이스를 제공하여 사용자 및 관리자가 이중화 관리 명령을 수행할 수 있다.

ALTIBASE™ 시스템이 제공하는 이중화 기능을 이용하여 각 서버를 여러 유형으로 설정할 수 있다. ALTIBASE™ 시스템에서 지원하는 서버 유형은 주(primary) 서버, 대기(standby) 서버, 활동(active) 서버로 나눌 수 있으며, 이러한 서버 유형에 따라 (그림 4)와 같이 주-대기(primary-standby) 구조, 활동-활동(active-active) 구조, 주-다중 대기(active-multi standby) 구조로 구성하여 사용할 수 있다.



(그림 3) ALTIBASE™ 시스템의 이중화 쓰레드 구조



(그림 4) ALTIBASE™ 시스템의 이중화 구조 유형

주-대기 구조는 가장 일반적인 이중화 시나리오로서 주 서버의 변경 내용을 대기 서버에게 전달하는 방식이며, 대기 서버는 주 서버의 전달내용을 자신의 데이터베이스에게 반영하기만 하며 서비스는 하지 않는다. 주 서버에 오류가 발생하는 경우 응용들의 연결 서버를 대기 서버로 대치하여 서비스를 제공하며, 오류가 복구된 후 그 동안의 변경내용을 다시 주 서버에 전달한다. 양 측 서버는 원래의 역할로 돌아가거나 주-대기의 역할을 바꾸어 수행할 수도 있다. 활동-활동 구조는 응용의 업무를 분리하여 처리하거나 부하 분산(load balancing)을 위해서 사용한다. 변경된 내용을 상대편 서버에 복제하여 만약의 시스템 오류에 대비할 수 있게 하며, 응용에서 발생하는 트랜잭션을 두 그룹으로 분리하여 서로 다른 서버에서 처리되도록 함과 동시에 변경된 데이터를 상호 이중화 하도록 하는 것이다. 성능이 중요시되고 높은 가용성을 요구하는 경우 하나의 주 서버와 여러 대기 서버를 구성하는 주-다중 대기 서버로 구성할 수 있다. 이 구조는 비용이 많이 드는 반면 더욱 높은 고가용성을 제공할 수 있다.

4.2 네트워크 이중화 및 이중화 서버간의 트랜잭션 충돌 해결

물리적인 두 개 이상의 네트워크 장치를 가진 두 서버 간의 이중화를 위해서 다수의 인터넷 주소(IP address)를 사용할 수 있도록 IP 이중화 기능을 제공한다. 무정지 서비스를 요구하는 고가용성 데이터베이스 응용의 경우 어떠한 경우에도 서비스가 정지되어서는 안된다. 따라서 데이터베이스 이중화 작업 도중 네트워크 단절로 인한 데이터의 이중화 기능이 중단되면 고가용성을 제공할 수 없게 된다. ALTIBASE™ 시스템은 다중 네트워크 장치를 지원하는 서버들에 대해서 다중 네트워크 이중화 기능을 지원한다. 예를 들어 앞 절에서 언급한 주-대기 이중화 구조에서 데이터베이스 이중화 작업을 위해 주 서버가 대기 서버로 이중화 로그를 전달하는 경우 첫번째 전송에서 실패하게 되면, 기본 접속 통로를 해제하고 등록된 보조 주소들을 이용하여 재 연결을 시도하여 네트워크 이중화를 사용한다. ALTIBASE™은 여러 개의 IP 주소가 등록 가능하므로 네트워크 오류에도 지속적인 이중화 기능을 지원할 수 있다.

또한 이중화 서버 구조 중에서 활동-활동 구조인 경우, 양측의 서버가 동등하게 데이터베이스 서비스를 실시간으로 제공하게 되므로, 동일한 키 값에 대한 변경작업을 시도하게 되는 경우 두 서버간의 레코드 내용이 일시적으로 다른 상태가 될 수 있다[12]. 이러한 경우 두 서버를 주 서버와 종속 서버로 분리하여 설정한 후 트랜잭션 처리 시 충돌이 발생하게 되는 경우, 주 서버의 작업을 우선적으로 처리한 후, 종속 서버에서 각 트랜잭션의 변경 특성에 따라 반영하도록 하는 방법을 사용한다. 따라서, ALTIBASE™ 시스템이 지원하는 시스템 카탈로그 중 SYS_REPLICATIONS 테이블에 이중화 기능 지원을 위한 속성 이외에 충돌 해결을 위한 서버 설정을 할 수 있도록 구현하였다.

5. 성능 평가

본 절에서는 ALTIBASE™ 시스템의 트랜잭션 수행에 대한 성능평가를 보인다. 특히 여러 종류의 시스템 부하에 따른 TPS(Transaction per Second) 값을 집중적으로 평가하여 시간제약적인 응용 분야에 적합한 시스템임을 보인다. 본 절에서 수행한 모든 실험은 400MHz CPU 4개와 4G 바이트 메모리를 보유한 "*Sun Enterprise* 3500" 플랫폼과 "*Solaris* 2.5.8" 운영체제 하에서 수행하였다. 또한 실험 트랜잭션은 ALTIBASE™ 시스템에서 제공하는 저장 프로시듀어 인터페이스를 사용하여 구현하였다. 따라서 트랜잭션의 데이터베이스 요구는 모두 질의 처리기를 거쳐 최적 수행 계획에 따라 수행되며, 네트워크 지연에 따른 간섭은 실험에 평가되지 않는다. 실험에 사용된 트랜잭션은 모두 4종류, 검색, 삽입, 변경 그리고 삭제 트랜잭션이며, 대상 테이블은 "*number*", "*real*", "*varchar*" 등의 여러 가지 속성들로 구성되는 총 20개의 속성을 갖는 단일 테이블이다. 동시 사용자 수는 실험에 따라 단일 사용자에서 50명의 사용자로 변화를 주었으며, 레코드의 개수는 실험에 따라 총 10,000개에서 500,000개의 레코드로 구성하였다. 검색 트랜잭션의 경우 모든 속성을 검색하도록 수행되었으며, 삽입 및 변경 트랜잭션들도 모든 속성들에 대해 삽입 및 변경 작업을 수행하도록 구성하였다. 모든 트랜잭션의 조건식은 인덱스를 갖는 속성에 대해 조건식을 작성하였다. 실험에 사용된 트랜잭션 영속성은 4, 로깅 레벨은 2로서 가장 일반적인 응용 환경과 동일한 조건에서 수행되었다. 대표적인 실험 결과는 <표 5>에 보인 바와 같이 단일 사용자 환경에서 수행한 4종류의 트랜잭션들에 대한 TPS 결과이다.

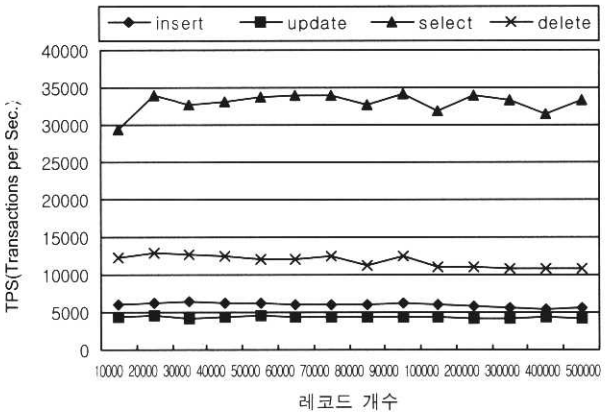
〈표 5〉 단일 사용자 환경에서의 TPS 측정값

단위 : TPS(Transactions Per Second)

삽입 트랜잭션	변경 트랜잭션	검색 트랜잭션	삭제 트랜잭션
6,134.97	4,405.29	29,411.76	12,345.68

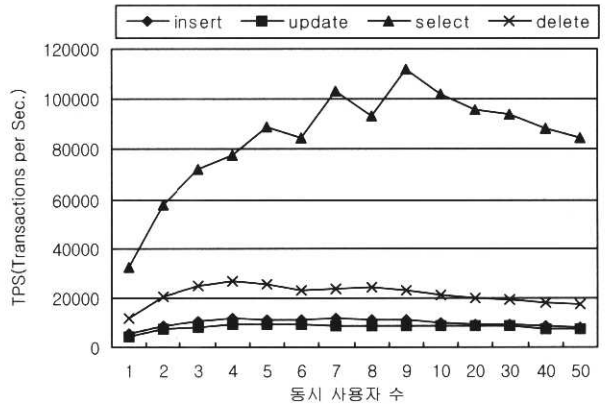
검색 트랜잭션의 경우 다른 트랜잭션의 수행 결과보다 우수한 성능을 보이며, 삭제 트랜잭션 역시 메모리 관리기에서 관리하는 메모리 객체 자료구조만 변경하면 되므로 변경, 삽입 트랜잭션의 경우보다 우수한 성능을 보임을 알 수 있다. 본 <표 5>에서 측정한 결과로 다른 상용 주기억 상주 데이터베이스 시스템의 측정 결과보다 우수함을 알 수 있다.

(그림 5)는 테이블 레코드 수의 변화에 따른 TPS 변화를 측정한 결과이다. 레코드의 개수는 10,000에서부터 500,000개의 레코드가지 변화를 주어 실험하였다. 이 실험에서 레코드 수의 변화는 TPS에 민감한 변화를 주지 않음을 알 수 있다. 그 이유는 ALTIBASE™ 시스템의 자료저장 관리기에서 제공하는 효율적인 메모리 관리 기능과 인덱스 관리 기능 때문이며, 또한 CPU 이용도가 균등하기 때문에 초당 트랜잭션의 수행 개수의 변화에 직접적인 영향을 미치지 못하기 때문이다.



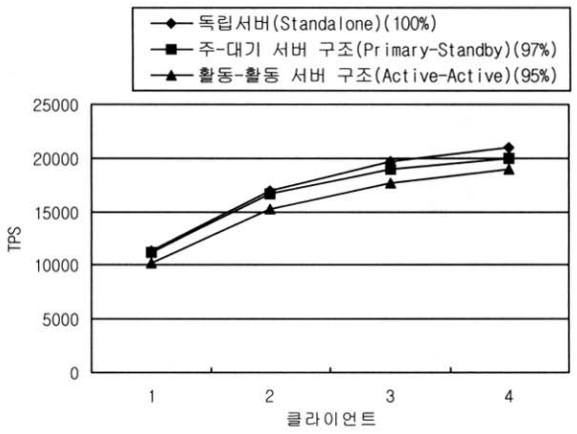
(그림 5) 레코드 개수의 변화에 따른 TPS

반면에 동시 사용자 수의 변화는 TPS의 결과에 어느정도 영향을 미치고 있음을 (그림 6)의 결과를 통해 알 수 있다. 동시 사용자 수가 증가 할수록, CPU의 처리능력이 충분한 지점까지는 초당 처리할 수 있는 트랜잭션의 수가 점점 증가하게 됨을 알 수 있으나 동시 사용자 수가 10명 이상이 되는 지점에서부터 CPU 처리능력이 부족하여 TPS 수치가 점차 약간 감소하거나 일정 수준을 유지함을 알 수 있다. 이 결과를 통해 ALTIBASE™의 성능 확장성과 시스템 가용성을 보장함을 알 수 있다.



(그림 6) 동시 사용자 수의 변화에 따른 TPS

(그림 7)은 ALTIBASE™ 시스템의 이중화 기능에 대한 트랜잭션 처리율을 평가한 것이다. 즉 데이터베이스 이중화 기능을 사용하지 않는 경우에 대비하여 활동-대기 이중화 구조를 사용하는 경우에 트랜잭션 처리율이 다소 감소하지만, 이중화 기능을 사용하는 장점에 비해 무시할 만한 차이를 보인다. 두 서버의 모두 활동 서버로 서비스하는 이중화 구조에서도 다른 구조에서도 다소 감소된 트랜잭션 처리율을 보이지만, 독립 서버가 제공하는 트랜잭션 처리율의 95% 이상의 성능을 보임을 알 수 있다.



(그림 7) 데이터베이스 이중화 기능에 따른 TPS

결국, 본 실험을 통해 측정된 결과는 다른 상용 주기억 상주 DBMS 들과 유사하거나 다소 우세한 실험 결과임을 알 수 있으며, 데이터베이스 이중화 기능 또한 부가적인 오버헤드 없이 지원가능 함을 알 수 있다. 따라서 본 성능 평가를 통하여 ALTIBASE™ 시스템이 시간제약 사항을 지닌 실시간 응용 분야나 통신 산업 분야의 응용 업무에 적용 가능함을 알 수 있다.

6. 결 론

본 논문에서는 주기억 상주 데이터베이스 시스템인 ALTIBASE™ 시스템에 대한 설계에 대한 고려사항을 기술하였으며, 특히 자료저장 관리기의 세부 구성요소와 구조적 특징과 기능에 대하여 설명하였다. 본 ALTIBASE™ 시스템은 높은 트랜잭션 성능을 제공하기 위해 국내 최초로 주기억 상주 데이터베이스 시스템에 다중 버전 병행수행 제어 기법을 적용하였으며 아울러 로크 기반 기법을 적용하여 각 기법의 장점을 모두 이용하는 트랜잭션 처리 기법을 적용하였다. 또한 다중 로그 파일, 퍼지 및 핑퐁 검사점의 구현, 다양한 트랜잭션의 영속성 및 로깅 수준 제공을 통해 회복 관리기를 구현하였으며, 주기억 상주 데이터베이스의 안정성 문제를 보장하였다. 현재 ALTIBASE™ 3.0이 상용 시스템으로 발표되었으며, 이 시스템은 성능 및 안정성 그리고 고가용성을 요구하는 여러 응용 분야에 범용적으로 활용되고 있다. 다만 주기억 상주 데이터베이스 시스템의 단점이라 할 수 있는 대용량 데이터베이스관리의 한계를 극복하는 것이 과제로 남아있다. 이를 해결하기 위하여 ALTIBASE™의 향후 연구 및 개발 방향은 디스크 기반 데이터베이스 시스템과 주기억 상주 데이터베이스 시스템을 혼합하여 사용할 수 있는 다중 저장 장치 데이터베이스 시스템을 연구하는 것이다.

참 고 문 헌

[1] D. Agrawal and V. Krishnaswamy, "Using Multiversion Data for Non-Interfering Execution of Wirte-Only Transactions," Proc. of the ACM SIGMOD International Conference on Management of Data, 1991.

[2] P. M. Bober and M. J. Carey, "Multiversion Query Locking," Proc. of the 18th Conference on Very Large Database, 1992.

[3] P. Bohannon, D. F. Lieuwen, R. Rastogi, A. Silberschatz, S. Seshadri, and S. Sudarshan, "The Architecture of the Dali Main-Memory Storage Manager," Multimedia Tools and Applications, 4(2), 1997.

[4] P. Bohannon, J. Parker, R. Rastogi, S. Seshadri, A. Silberschatz, and S. Sudarshan, "Distributed Multi-Level Recovery in Main-Memory Databases," Proc. of the International Conference on Parallel and Distributed Information Systems, 1996.

[5] H. Garcia-Molina and K. Salem, "Main Memory Database Systems : An Overview," IEEE Transactions on Knowledge and Data Engineering, 4(6), 1993.

[6] H. V. Jagadish, A. Silberschatz and S. Sudarshan, "Recovering Main Memory Lapses," Proc. of the 19th Conference on Very Large Databases, 1993.

[7] C. Mohan, "Repeating History Beyond AREIS," Proc. of the 25th International Conference on Very Large Databases, 1999.

[8] C. Mohan, D. Haderle, B. Lindsay, H. Pirahesh and P. Schwarz, "ARIES : A Transaction Recovery Method Supporting Fine-Granularity Locking and Partiall Rollbacks Using Write-Ahead Logging," ACM Transactions on Database Systems, 17(1), 1992.

[9] K. Ramamritham and P. K. Chrysanthis, "A Taxonomy of Correctness Criteria in Database Applications," VLDB Journal, 5(1), 1996.

[10] R. Rastogi, S. Seshadri, P. Bohannon, D. W. Leinbaugh, A. Silberschatz and S. Sudarshan, "Logical and Physical Versioning in Main Memory Databases," Proc. of the 23rd International Conference on Very Large Databases, 1997.

[11] Bettina Kemme and Gustavo Alonso, "A New Approach to Developing and Implementing Eager Database Replication Protocols," ACM Transactions on Database Systems, 25(3), Sep., 2000.

[12] Jim Gray, Pat Helland, Patrick O'Neil, and Dennis Shasha, "The Dangers of Replication and a Solution," Proc. of the AMC SIGMOD International Conference on Mangement of Data, 1996.

정 광 철

e-mail : jungkc@altibase.com

1987년 인하대학교 전자계산학과(학사)

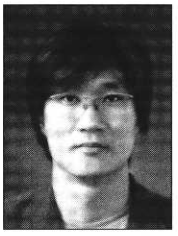
1989년 인하대학교 대학원 전자계산학과 (이학석사)

1992년 LG 소프트웨어 연구원

1999년 한국전자통신연구소 선임연구원

1999년~현재 (주)알티베이스 이사/연구소장

관심분야 : 실시간 DBMS, 미들웨어, 분산시스템, 멀티미디어 서비스



이 규 웅

e-mail : leekw@sangji.ac.kr

1990년 한국외국어대학교 전자계산학과 (이학사)

1992년 서강대학교 대학원 전자계산학과 (공학석사)

1998년 서강대학교 대학원 전자계산학과 (공학박사)

2000년 한국전자통신연구원 인터넷서비스 연구부 선임 연구원

2000년~현재 상지대학교 컴퓨터정보공학부 조교수

관심분야 : 트랜잭션 처리, SAN 기반 자료저장 시스템, 분산 및 실시간 DB

배 해 영

e-mail : hybae@inha.ac.kr

1974년 인하대학교 응용물리학과(공학사)

1978년 연세대학교 대학원 전자계산학과 (공학석사)

1989년 숭실대학교 대학원 전자계산학과 (공학박사)

1985년 Univ. of Houston 객원 교수

1994년 인하대학교 전자계산소 소장

1982년~현재 인하대학교 전자계산공학과 교수

1999년~현재 지능형 GIS 연구센터 소장

2000년~현재 중국 중경우전대학교 대학원 명예교수

관심분야 : 분산 데이터베이스, 공간 데이터베이스, 지리정보 시스템, 멀티미디어 데이터베이스 등