디지털산업정보학회 논문지 제5권 제4호-2009년 12월

메모리 상주 DBMS에서의 응용 트랜잭션 성능평가에 관한 연구

김 희 완* · 이 혜 경**

A Study on the Performance Evaluation of Application Transaction in the Main Memory DBMS

Kim, Hee Wan · Rhee, Hae Kyung

〈Abstract〉

Recently, the Main Memory DBMS is gradually being expanded by the appearance of a large capacity of a Main Memory System, the increase in business area where it requires a real time process, and the rise of the users' required level. The Main Memory DBMS, which is able to go through a large capacity data process of the disk-based DBMS and guarantees a high efficiency, has domestically developed and has been put to a practical use. This paper presents an examination of the applied technologies and the limits of Altibase system, which is Main Memory DBMS. Moreover, it evaluated and performed a comparative analysis on the performance level of the Main Memory DBMS and the disk-based DBMS based on the same application. After five trials of the experiment based on the operating application, it was confirmed that the performance level of the Main Memory DBMS is enhanced and is higher by 4.13 to 7.89 times than the disk-based DBMS.

Key Words : Main Memory DBMS, Disk-based DBMS, Performance Level

Ⅰ. 서론

MMDBMS(Main Memory DataBase Management System)는 데이터베이스를 메모리에 상주 시켜 운영하는 DBMS로, 실시간 처리를 요구하는 응용분야가 많아짐에 따라 점점 그 수요가 커지고 있다. 또한 대용량의 메모리 탑재 시스템의 등장, 실시간 요구의 증대, 사용자 요구수준의 향상 등이 MMDBMS의 실용화를 앞당겨 왔다.

일반 디스크 기반 DBMS와 대비되는 MMDBMS의 가장 큰 특징이자 차이점은 운영 데이터의 존재 위치가 다르다는 점이다. 디스크 DBMS는 보통 버퍼 매니저를 통하여 메모리상에 일정 크기만큼의 버퍼를 잡고서 알고리즘에 의하여 디스크에 존재하는 데이터를 메모리 버퍼에 올려서 접근(Access)한다. 버퍼의 크기가 유한하기 때문에 일정시간이 경과하거나 최근에 접근하지 않은 데이터는 다시 디스크로 내려서 데이터를 관리하게 되는데 DBMS 입장에서는 이로 인한 버퍼 매니저 관리 비용이 상당히 많은 단점을 가지고 있다. 한편 MMDBMS는 구

* 삼육대학교 컴퓨터학부 부교수
** 용인송담대학 컴퓨터게임정보과 부교수(교신저자)

동과 동시에 디스크에 존재하는 테이블의 내용이 메모리에 모두 올라가는 구조로 운영된다. 디스크 데이터와 메모리 데이터 관리를 위한 버퍼 매니저가 필요 없으며 버퍼 매니저 관리를 위한 비용을 절감하여 리소스를 다르게 활용할 수 있다[1]. 따라서 디스크 기반 DBMS에서 충분히 큰 메모리를 갖고 있을 경우에도 버퍼 매니저의 관리 비용으로 메모리 상주형 DBMS에서의 성능 보다는 우수하지 못함을 다른 연구에서도 보였다[2-3].

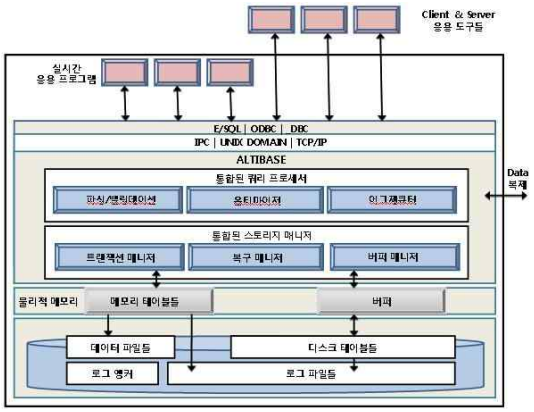
CPU가 메모리에 접근하는 속도는 디스크의 I/O 처리를 감안하면 수천 배의 빠름에도 불구하고 성능에서는 수십 배 정도의 차이가 나는 것은 MMDMBS도 디스크 파일시스템을 기본적으로 사용하여 데이터를 저장하고 만약의 경우를 대비하여 트랜잭션 로그파일을 파일 시스템에 유지하기 때문이다. 기존 연구에서는 MMDBMS와 디스크 DBMS에서의 성능을 SQL의 SELECT, INSERT, UPDATE, DELETE문에서 성능을 비교하였다. 데이터 변경 연산(Insert, Update, Delete문)은 약 50배 정도의 성능 차이를 보였으며, 데이터 조회 작업은 디스크 DBMS 역시 버퍼 캐쉬에 존재하는 데이터를 대상으로 하였기 때문에 약 3~4배 정도의 차이가 발생함을 보였다[1].

본 논문에서는 조회, 삽입, 삭제, 갱신 연산이 통합된 응용 트랜잭션들에 대한 성능을 측정하고자 한다. 이를 위해 국내에서 개발되어 보급되고 있는 메모리 상주 DBMS인 ALTIBASE 시스템과 디스크 기반 DBMS인 SYBASE 시스템간의 성능을 응용 트랜잭션을 근간으로 평가하고, MMDBMS의 성능에서의 우수성을 평가하고자 한다.

Ⅱ. ALTIBASE 구조 및 적용 기술

2.1 전체적인 시스템 구조

'ALTIBASE 4' 의 전체적인 시스템 구조는 <그림 1>과 같다.



<그림 1> ALTIBASE 구조

'ALTIBASE 4' 는 기존의 범용의 MMDBMS를 확장하여 디스크 DBMS 기능을 추가하였다. 기존 MMDBMS의 기능과 특성 부분은 그대로 구현하였고, 디스크 DBMS에서 필요한 기능들을 추가하였다. 메모리 테이블을 처리하기 위해 기존의 테이블을 구조화한 데이터 페이지 개념을 그대로 가져가고 디스크 테이블을 처리하기 위해서 버퍼 매니저와 저장 공간의 관리를 위한 테이블 스페이스 개념을 도입하였다. 이를 통해 MMDBMS의 단점인 대용량 DB의 처리 문제를 해결하였고 메모리 테이블의 처리속도는 기존 MMDBMS에서 처리하던 속도를 그대로 유지할 수 있다. SQL해석기는 DBMS 내부에서 테이블 특성(메모리, 디스크)에 따라서 SQL문이 따로 해석되며 최적화 되어 수행된다. 예를 들면 디스크 테이블의 경우 일정량을 초과하는 인덱스를 검색하면 전체 테이블을 검색하는 것보다 디스크 I/O 비용이 더 많이 들기 때문에 SQL 해석기가 판단하여 검색한다. 반면에 메모리 테이블의 경우에는 대부분의 데이터가 메모리에 존재하기 때문에 인덱스 검색을 하는 것이 더 유리하다. 이와 같은 상황을 해석기가 자동적으로 판단하여 SQL문을 해석하게 된다. 또한 디스크 테이블과 메모리 테이블간의 SQL 수행이 가능하다. 기존의 시스템 구조로는 MMDBMS와 디스크 DBMS를 같이 사용하더라도 이질적인 DBMS이기 때문에 SQL문을 이용한 DBMS간 연결이 쉽지가 않

았다. 하지만 이 기능이 가능해 짐으로써 애플리케이션의 구현이 쉬워지고 단일 DBMS만을 이용할 수 있다. MMDBMS의 최적화를 위해 메모리용과 디스크용이 통합된 저장 관리자를 제공한다. DBMS에서 저장관리자는 락(Lock) 관리, 인덱스 관리, 데이터 저장소 관리, 로그파일 관리 등을 수행한다. 로그파일 관리는 고성능 트랜잭션 처리를 위한 개선된 다중 병행수행 제어 방법을 제공한다. 데이터의 로킹 단위(granularity)를 세분화하여, 즉 테이블과 레코드 단위로 구분하여 다중버젼 병행수행 제어 방법[4-5]를 사용한다. 버젼 기법을 복구를 위해 사용하는 로그파일 측면에서는 메모리 테이블과 디스크 테이블에서 발생하는 트랜잭션을 순차적으로 일관성 있게 관리하기 위해서 동일한 로그파일을 이용 가능하도록 통합했으며 데이터 저장소는 메모리 테이블을 위한 메모리 테이블 스페이스 개념을 도입하여 디스크 테이블의 일반 테이블 스페이스와 구분하였다. 이와 같이 때로는 구조를 유연하게 합쳐서 일관성을 유지하였고 때로는 구조를 분리하여 구현함으로써 최대한의 성능을 향상시킬 수 있도록 설계되었다. 마지막으로, 테이블의 위치(메모리, 디스크)에 상관없이 테이블 접근은 사용자에게 투명하도록 설계되어 있다. 테이블의 위치가 테이블 생성 시에 지정되면 테이블 이름만을 통하여 접근할 수 있다. 그러므로 개발자는 테이블의 위치에 상관없이 애플리케이션을 만들 수가 있고 위치에 상관없이 투명하게 데이터를 이용할 수 있다.

2.2 ALTIBASE의 응용에 대한 이중화 기술

MMDBMS는 빠른 성능을 제공하는 것 이외에도 시스템 구성 측면에서도 탁월한 유연성을 제공하는데, 이는 '이중화(Replication)'라는 핵심 기술 때문이다[1].

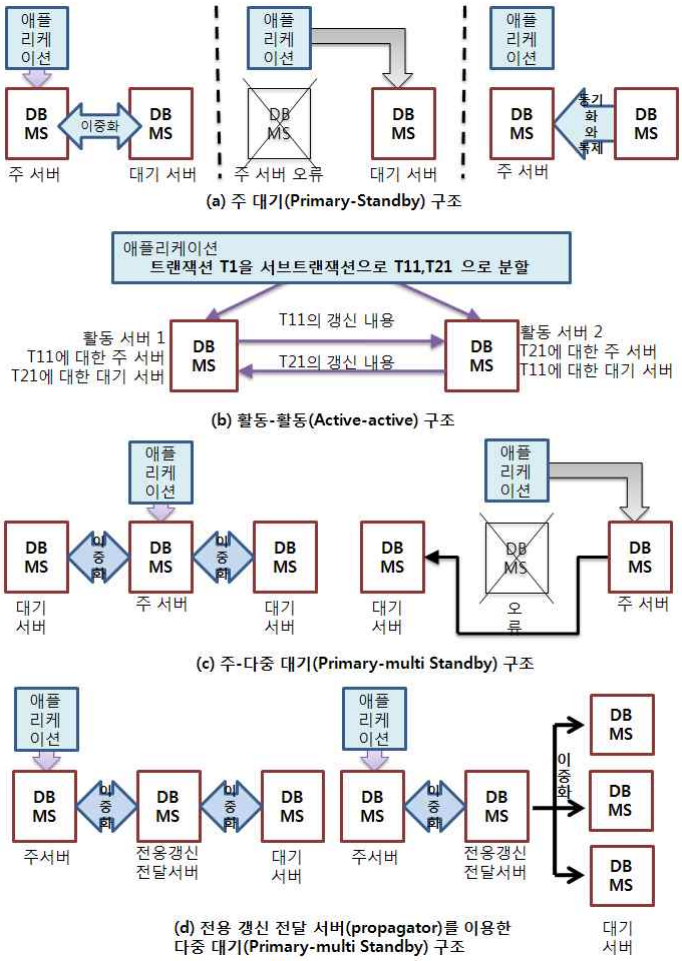
이중화는 물리적으로 떨어져 있는 여러 개의 데이터베이스에 대해 로컬 데이터베이스의 변경 내용을 원격 데이터베이스에 복제하고 관리하는 기술이다. 사용자는 하나의 데이터베이스에 대해서만 작업을 수행해도 데이터베이스 이중화 시스템에 연결되어 있는 다른 데이터베이스에서도 작업 내용이 동일하게 적용돼 여러 개의 데이터베이스를 동시에 관리할 수 있다. 이런 데이터베이스의 이중화는 데이터베이스의 무정지 서비스를 가능하게 한다.

따라서 디스크 공유 방식의 시스템 페일오버(Failover) 형태의 서비스를 사용하는 디스크 기반 DBMS는 하드웨어 장애 발생 시 백업 미디어를 통한 복구를 수행해야 하고, 어떤 복구 방식을 채택한다고 하더라도 서비스 중단 사태를 방지할 수 없다. 하지만 MMDBMS는 TCP방식의 이중화 구성을 이용해 각 서버는 최소한의 간섭으로 독립적으로 운용될 수 있기 때문에 가용성(Availability)과 확장성(Scalability)을 높일 수 있다. 즉 주 시스템에 장애가 발생하더라도 n-way로 구성된 서비스에는 영향이 끼치지 않기 때문에 무정지 시스템 구성이 가능하다. 데이터베이스 이중화 기법으로는 주-대기(Active-Standby) 위상과 활동-활동(Active-Active) 위상이 있는데, 최근에는 활동-활동 위상에 대한 요구가 크게 증가하고 있다. 이에 따라 동시성 제어와 트랜잭션 관리 기술이 빠르게 발전하고 있다. 서버 구성에 따라 데이터베이스 이중화를 위한 통신 모델은 <그림 2>와 같이 4가지로 구분된다.

주-대기 서버 이중화 모델은 데이터베이스 이중화뿐만 아니라 분산 시스템의 전형적인 이중화 모델이다. 이 모델은 정상적인 운영 모드에서 주 서버는 일반적인 데이터베이스 서버를 제공하는 것처럼 보이지만, 실제로 자신의 변경 내용을 대기 서버로 전달하고 있으며, 대기 서버는 데이터베이스 서비스는 하지 않은 채, 전달받은 내용을 자신의 데이터베이스에 반영한다.

주 서버에 오류가 발생하면, 주 서버의 서비스를 받던 애플리케이션들을 대기 서버로 옮겨 대기 서버에서 새로운 서비스를 시작한다. 주 서버의 오류가 복구되면 그 동안의 대기 서버의 변경 트랜잭션을 주 서버에 보내 이중화 작업을 계속 수행한다. 그 후, 양 서버의 역할을 바꿔 계속 수행하거나, 주 서버와 대기 서버의 역할을 바꿔 수행하기도 한다.

활동-활동 서버 이중화 모델은 애플리케이션업무를



<그림 2> ALTIBASE 시스템의 이중화 통신모델

분리해 처리하거나 부하 분산을 위해 주로 사용된다. 변경된 데이터 갱신 내용을 서로 상대편 서버에 이중화하는 작업을 수행하고, 응용해서 발생하는 트랜잭션을 두 그룹으로 분리해 각각 서버에서 수행하게 한다. 이후 변경 트랜잭션을 교환, 양 서버의 데이터 변경 내용을 일치시켜 상호 이중화하는 방식이다.

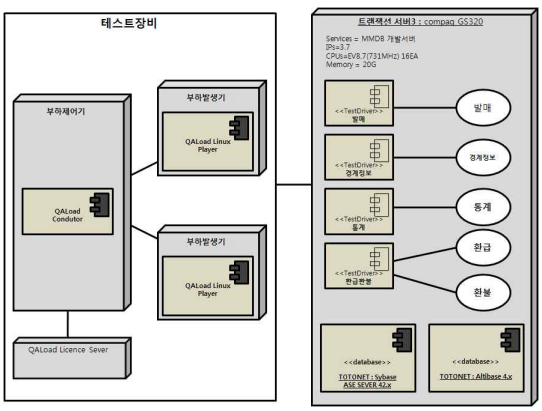
즉 애플리케이션 수준에서 트랜잭션 T1을 T11과 T21로 분할 한 후, 각 서브 트랜잭션을 <그림 2>의 (나) 모델을 적용하여, T11을 활동 서버1에게, T21을 활동 서버2에게 전송시켜 각각의 서버에서 서브 트랜잭션을 수행시킨 후, 결과를 조합하여 최종 결과를 얻을 수 있게 된다. 이 때, 트랜잭션 T11에 대해서는 활동 서버1이 주서버, 활동 서버2가 대기 서버 역할을 하게 되며, 반대로 트랜잭션 T21에 대해서는 활동 서버2가 주서버, 활동 서버1이 대기 서버 역할을 하게 된다. 데이터베이스 이중화 기능을 위해서는 활동 서버1에서 수행된 T11의 변경 내용은 활동 서버2로 전송되고, 마찬가지로 활동 서버2에서 수행된 T21의 변경 내용은 활동 서버1로 전송되어, 상호 이중화 기능을 갖도록 수행하는 한편, 두 서버를 이용한 부하 분산 기능을 갖도록 한다.

주-다중(Active-Multi Standby) 대기 이중화 모델은 성능에 민감하고 높은 가용성을 요구하는 애플리케이션에 적합한 모델로서, 하나의 주 서버와 두 개 이상의 대기 서버로 구성된다. 주-대기 서버와 기본적으로 동일한 구성을 갖지만, 대기 서버 수가 많다는 점에서 시스템의 가용성을 더욱 높일 수 있다. 이 구조는 높은 고가용성을 얻을 수 있어, 크리티컬한 애플리케이션에 적합한 모델이지만 비용이 많이 든다는 단점이 있다.

다중 대기 서버에 변경 트랜잭션을 전달하기 위해 소요되는 주 서버의 오버헤드를 절감시키기 위해 <그림 2>의 (라)와 같은 전용 변경 전달서버(propagator)를 운영해 주-다중 대기 서버 모델을 운영할 수 있다.

Ⅲ. 메모리 상주 DBMS와 디스크 기반 DBMS의 응용 성능 평가

3.1 평가 시스템



<그림 3> 평가 시스템 구성

본 절에서는 DBMS 자체의 정확한 성능 검증을 위하여 윈도우즈 기반의 성능테스트 도구가 갖게 되는 네트워크단의 부하를 최소화하기 위하여 S사에서 제공한 테스트 Driver형태의 테스트 Scripts를 사용하여 시험하였다. 본 절에서 수행한 시험은 <그림 3>과 같이 731MHz CPU 16개와 20GB의 메모리를 보유한 compaq GS320 서버에서 수행하였다. 테스트 Driver는 기존 관계형 DBMS인 Sybase 응용에서 사용해 오던 공통 라이브러리(Library)를 사용하면서 게임 프로세서를 시뮬레이션하기 위한 프로세스를 생성하여 데이터베이스 입출력(I/O)을 수행하도록 구성하였다.

3.2 평가 대상 데이터

두 시스템의 성능 평가를 위한 데이터 크기는 동일한 데이터를 사용하였으며, Test 시간은 Altibase와 Sybase DB를 동시에 로드한 상태에서 테스트 하였으며, 그 내용은 다음 <표 1>과 같다. 또한 시험 트랜잭션은 ALTIBASE와 SYBASE 시스템에서 제공하는 프로시저를 사용하였으며, 트랜잭션의 데이터베이스 요구는 모두 질의 처리기를 거쳐 최적 수행 계획에 따라 수행되었다. 전체 대상 프로그램은 810본이었으며, 3, -582회의 이벤트를 발생시켜서 실험을 수행하였다. 애플리케이션 최적화 차원에서 애플리케이션 단에서 시스템 장애시 데이터 복구, 데이터 오류 시 디버깅을 위해 남기고 있는 다양한 로그는 남기지 않도록 하고 테스트를 수행하였다.

<표 1> 평가 Data Size

table-name	rows
PROGRAM	810
EVENT	3,582
WINNING	1,964
BMF_PRICEHOURSTAT	48,000
BMF_PRICEDAYSTAT	64,000
BMF_PRICEMONSTAT	15,000
BMF_WINNER_1_1281	500,000

3.3 트랜잭션별 성능 결과

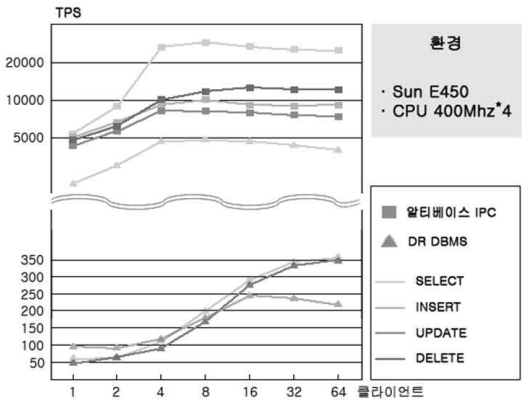
단일 사용자 환경에서의 시험 결과는 아래 <표 2>와 같이 4종류(INSERT, UPDATE, DELETE, SELECT)의 트

랜잭션들에 대한 TPS 결과는 다른 연구[6]에서 보였다.

<표 2> 단일 사용자 환경에서의 실험 결과

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

단위 : TPS(Transactions Per Second)



<그림 4> 연산별 성능 비교

<그림 4>는 MMDBMS와 디스크 DBMS에서의 성능의 차이를 수치화하여 그래프로 표시한 것이다. 데이터 변경(Insert, Update, Delete문) 연산은 약 50배 정도의 성능 차이가 났으며, 데이터 조회 작업은 디스크 DBMS 역시 버퍼 캐쉬에 존재하는 데이터를 대상으로 하기 때문에 약 3-4배의 차이가 발생했다[7].

본 논문에서는 S사[8]에서 운영 중인 응용 트랜잭션을 근간으로 MMDBMS인 Altibase 시스템과 디스크 기반 DBMS인 Sybase에서 TPS(Transactions Per Second)를 기준으로 시험하였다. 5회에 걸친 시험결과 <표 3>에서와 같이 Altibase가 Sybase에 비하여 4.13 ~ 7.89배의 성능 향상이 있었음을 확인하였다.

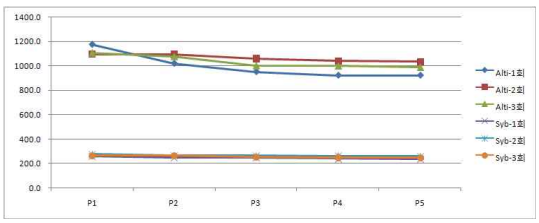
<표 3> 응용업무별 성능시험 결과

업무 종류	Altibase TPS	Sybase TPS	성능비
발매	1243.75	216.72	5.74
환급/환불	1029.71	249.34	4.13
통계	886.26	112.37	7.89
게임정보조회	3192.55	429.9	7.43

3.4 트랜잭션별 동시사용자수 비교

3.4.1 환급/환불 트랜잭션에서의 TPS 비교

환급/환불 트랜잭션에서의 프로세스별 TPS는 <그림 5>와 같다. 3회의 테스트에서 Altibase의 경우에는 평균 1029.71의 TPS를 적용할 때 트랜잭션당 평균 응답시간이 0.971ms가 되고, Think Time을 10ms라 가정할 때에 동시사용자수는 11,297명을 처리할 수 있다. Sybase의 경우에는 평균 249.34의 TPS을 적용할 때 트랜잭션당 평균 응답시간이 4.01ms가 되어 10ms라 가정할 때에 동시사용자수는 3,493명을 처리할 수 있다.

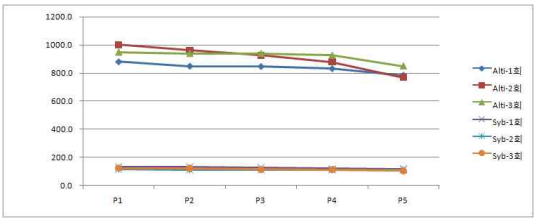


<그림 5> 환급/환불 업무에서의 성능 비교

3.4.2 통계 트랜잭션에서의 프로세스별 TPS 비교

통계 트랜잭션에서의 프로세스별 TPS는 아래 <그림 6>과 같다. 3회의 테스트에서 Altibase의 경우에는 평균 886.26의 TPS를 적용할 때 트랜잭션당 평균 응답시간이 1.238ms이므로 Think Time을 10ms라 가정할 때 동시사용자수는 9,862명이 된다. 같은 조건으로 Sybase의 경우

에는 평균 11, 2.37의 TPS를 트랜잭션당 평균 응답시간이 8.899ms가 되어 같은 Think Time을 가정할 때 동시사용자수는 2,123명이 된다.



<그림 6> 통계 업무에서의 성능 비교

따라서, 응용업무 종류별로 메모리상주 DBMS와 디스크기반 DBMS의 성능비가 좋은 통계조회의 경우 동시사용자수는 9,862명 : 2,123명이 되어 Altibase가 4.64배의 사용자를 동시에 처리할 수 있으며, 환급/환불의 경우에도 3.23배의 사용자를 동시에 처리할 수 있다.

그러나, ALTIBASE와 SYBASE의 성능의 차이는 SQL에서 구문별로 평가한 성능과는 상당한 차이가 있음을 볼 수 있었다. 이는 MMDBMS는 운영 데이터를 메모리에 모두 상주시켜서 구동하여야 하므로 데이터의 크기가 일정량 이상의 큰 업무에는 적용하기에 한계가 있음을 보여준다. 또한 메모리에 관리할 수 있는 데이터의 양이 제한적이므로, 테이블 레코드의 양이 계속 증가한다면 별도의 조치를 취하여야 한다. 통계성 작업이나 배치 작업의 경우에는 별도의 이력 데이터는 따로 관리하여 성능을 고려하여야 한다. 즉 트랜잭션을 위한 중요한 데이터는 MMDBMS에서 관리하고 일정시간이 지난 통계성 업무에만 사용하는 데이터는 디스크 기반 DBMS가 관리하게 할 수 있을 것이다.

Ⅳ. 결론

본 논문에서는 주기억 상주 DBMS인 Altibase 시스템에서 구현된 기술들에 대하여 살펴보았으며, 디스크 기반 DBMS인 Sybase 시스템과의 성능분석을 응용업무별로 비교하였다. 데이터 변경연산에서는 이론적으로 약 50배의 성능차이가 나타나고, 데이터 조회작업은 디스크 DBMS가 버퍼 캐쉬에 존재하는 데이터를 대상으로 할 경우에 3-4배의 성능차이가 나는 것으로 조사되어 있으나, 실제적으로 조회 및 변경 연산이 통합된 일반 응용 트랜잭션에 적용한 결과 4.13~7.89배의 성능 향상을 볼 수 있었다. 또한 트랜잭션별 동시 사용자 수를 비교해 본 결과 환급/환불 업무에서는 디스크 기반 DBMS에서는 3,493명을 처리할 수 있었으나, MMDBMS에서는 11,297명의 사용자를 처리할 수 있었다. 본 연구에서는 삽입, 변경, 삭제 및 검색 처리가 혼합된 일반 업무 트랜잭션에서 MMDBMS의 성능과 디스크 기반 DBMS의 성능을 비교 평가하여 그 성능의 차이를 확인할 수 있었다.

참고문헌

[1] 임세환, “구축사례로 살펴본 MMDBMS의 활용방안,” on the Net, July, 2006, pp. 168-172.

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

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

[4] D. Agrawal and V. Krishnaswamy, “Using Multiversion Data for Non-Interfering Excution of

Write-Only Transactions," Proc. of the ACM SIGMOD International Conference on Management of Data, 1991.

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

[6] 이규웅, "주기억장치 DBMS의 트랜잭션 성능평가," 한국컴퓨터산업교육학회 논문지, 2005. 6 Vol 6., No 2. pp. 559-565.

[7] 신범수, "MMDBMS의 등장배경," on the Net, June, 2006, pp. 154-159.

[8] 스포츠토토(주), "STMMDB: SportsTOTO MMDB 구축 프로젝트 - DB_Performance_Test_Report," March, 2007.

■ 저자소개 ■

김 희 완
Kim, Hee Wan

2001년 3월~현재 삼육대학교 컴퓨터학부 부교수
2002년 2월 성균관대학교 전기전자 및 컴퓨터공학부(공학박사)
1995년 8월 성균관대학교 정보공학과(공학석사)
1987년 2월 광운대학교 전자계산학과(이학사)
1988년 한국전력공사 정보처리처 정보관리 기술사, 정보시스템 수석감리원

관심분야 : 분산 DB, 보안 데이터베이스, 정보시스템 감리
E-mail : hwkim@syu.ac.kr

이 혜 경
Rhee, Hae Kyung

2001년 8월~현재 용인송담대학 컴퓨터게임정보과 부교수
2000년 2월 성균관대학교 전기전자 및 컴퓨터공학부 (공학박사)
1985년 4월 University of Illinois (Urbana-Champaign) 전산학과 (공학석사)
1979년 2월 숭실대학교 전자계산학과 (공학사)
1992년 3월~2001년 8월 경인여자대학 멀티미디어정보전산학부 조교수

관심분야 : 데이터베이스, 데이터 모델링, Privacy, 정보보호
E-mail : leehk@ysc.ac.kr

논문접수일 : 2009년 10월 15일
수 정 일 : 2009년 11월 13일
게재확정일 : 2009년 11월 19일