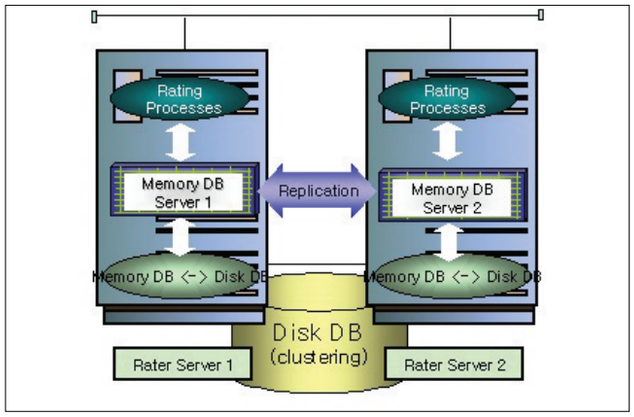
DDBMS와 MMDBMS의 비교 분석

글 / 김상하 으뜸정보기술 대표

DDBMS와 MMDBMS와의 특성을 비교해보면, 여러 가지 기능면에서 MMDBMS가 DDBMS를 대체할 수 있을 정도로 성장



▲ SK텔레콤의 차세대 빌링 시스템 구성도

하기에는 무리한 것으로 여겨진다. 몇몇 MMDBMS를 판매하는 업체들에서 고객을 호도하기 위해 DDBMS보다 MMDBMS가 우수하다고 광고하는데 통신, 증권 업종의 특수 영역, 쇼핑몰 분야, 모바일 분야 등에 국한할 뿐이고 다른 업종에서는 결코 비교 대상이 될 수가 없다. 그 커다란 차이는 대용량 데이터베이스 처리에서 대책이 거의 없고, 같이 사용할 하드웨어나 운영체제, 미들웨어, 개발도구, CASE 도구, 네트워크 프로토콜, DDBMS 등과의 호환성이나 확장성 면에서 고려해야 할 사항이 많기 때문이다. 국내의 우수한 DDBMS들도 결국은 이러한 약점이 있었기에 거의 사장이 됐으므로, 유일하게 MMDBMS 관련해서 시장 점유율이 높은 토종 업체들이 유의할 일이다.

그렇다면 MMDBMS는 기존 DDBMS들이 차지하고 있는 은행, 보험, 증권, 카드업 등 금융권과 제조, 유통 등의 업종으로의 진입이 매우 힘이 든다는 것인가? 그 해답은 '아니오'이다. DDBMS가 이러한 업종에서 터무니없이 대우를 받고 있거나, 또는 고객을 고통스럽게 하는 부분을 MMDBMS가 해결해 줄 수가 있다면 충분히 시장을 넓힐 수가 있다. 그 해법은 DDBMS의 약점을 보완하는 관계로서 위상을 설정하고 연구할 때만이 결실을 볼 수 있을 것이다. 현재처럼 DDBMS를 상대로 힘겨루기를 한다면 DDBMS와 DDBMS를 지원하는 전문가 집단에 의해 공격을 당해 곧 그 위상이 추락해 버리고 말 것이다.

MMDBMS가 DDBMS보다 유리한 오직 한 가지는 메인 메모리상에 데이터를 올려놓고 데이터를 처리하므로, 디스크에서 읽어야 하는 부담을 가지는 DDBMS보다 데이터처리 성능면에서 최소 10배에서 100배 이상까지 큰비용의 지출없이 성능을 올릴 수 있고, 성능관리가 용이하다는 점이다. 따라서 먼저 DDBMS의 성능관리 측면에서 연구를 해보고, MMDBMS가 유리하거나 DDBMS와 호환을 해 DDBMS를 보완할 수 있

〈표 1〉 튜닝 대상 및 제약사항

튜닝 대상		제 약 사 항
SQL	중요 SQL(11본) 덜 중요 SQL(61본)	· SQL 튜닝의 작업 활동이 제한됐음. – 충분하지 못한 컴퓨팅 자원 아래에서 실시간으로 SQL을 튜닝하는데 제한이 많았음. – Rule based 옵티마이저 환경에서 최적의 성능 Path를 찾는 일은 무리였음. Cost based 옵티마이저 환경에서의 다양한 성능 유틸리티를 사용해보지 못함. – DBA 권한의 제약으로 인해 소극적인 튜닝활동이 됨. · 몇몇 SQL 문이 너무 복잡하고, 길이가 긴데 비해 업무해석을 정확하게 받을 수 없었음.(SQL 문의 특성, 변경이력 등) · WAS 서버의 조정 이후 테스트 등은 외부원인으로 인해 분석하지 못함.
클라이언트 애플리케이션(ASP 프로그램)		· 애플리케이션에 대한 시범 적용이 의도대로 이루어지지 않음. 적용) 고객정보변경조회
DBMS 시스템 (메모리, 디스크 I/O, CPU)		· Rule based 옵티마이저, 현행 시스템의 운영상의 민감성 등으로 인해, DBMS를 정상적으로 튜닝해 보지 못함. · DBMS를 관리하는 회사의 상황과 관리자의 책임에 따른 DBMS 정보활용의 극히 적은 부분의 간접적인 이용으로 인한 튜닝결과가 예측정보의 제공에 그침.
데이터 모델/스키마 (결합인덱스, 인덱스 무결성 등)		· 몇몇 중요한 SQL에 대하여는 인덱스, 테이블의 생성 후 튜닝이 이뤄졌으나, 대부분의 경우는 물리적인 스키마의 변경 없이 현상을 개선하는 선에서 튜닝작업이 이뤄짐.

〈표 2〉 튜닝 대상 및 결과, 권고사항

튜닝 대상		현상	튜닝후 결과	권고 사항
SQL	중요 SQL(11본)	· 응답시간 1초이상 : 6본 1초미만 : 5본	· 응답시간 1초이상 : 1본 1초미만 : 10본	· 중요 SQL문의 성능향상 후, 트레이스 파일 검증, 적용권고 · 덜 중요 61본 중 10본 성능향상 후, 트레이스 파일 검증, 적용권고 · 단, Peak Time시 6본 1초이상 소요 → SQL 이외 성능향상 대책권고.
	덜 중요 SQL(61본)	· 응답시간 1초이상 : 6본 1초미만 : 56본	· 응답시간 1초이상 : 3본 1초미만 : 60본	
클라이언트 애플리케이션 (ASP 프로그램)		· 애플리케이션 시범 적용 (SQL 해결 어려움) 불편접수내역조회	개선방법 검토 결과 → TBD	· 현재의 화면컨트롤이 서버에서 이뤄지는데 이에 대해 화면설계방법을 수정하고, 몇몇 애플리케이션에 대해, 서버가 아닌 클라이언트에서 데이터가 처리되게 할 것 권고.
DBMS 시스템 (메모리, 디스크 I/O, CPU)		· 메모리 : 효율적 이용 못함 · 디스크 : 체인현상 존재 · 옵티마이저 모드 : Rule based	현행 문제점 SQL상의 개선방안 제공 결과 → TBD	· Rule based 모드를 Cost based로 변경한 후, 통계를 이용해 성능을 관리할 것. · Rule based 하에서 DBMS를 최적으로 운영하기 위한 현행 문제점 해결책 제공.
데이터 모델/스키마 (결합인덱스, 인덱스 무결성 등)		· 주키 컬럼 순서 : 인덱스의 효율적 이용가능성 저하	· 몇 개의 프로그램 인덱스 사용결과 현저한 성능 향상	· 현행 시스템의 인덱스 구조 분석, 변경사항 권고 · Rule based와 Cost based 환경 대안 권고

는 방법을 찾아보기로 한다.

DDBMS의 성능튜닝 사례연구

다음은 모 은행의 콜센터 관련 데이터베이스 성능튜닝을 한 결과를 보고한 보고서의 일부이다.

'2. 튜닝대상 및 제약사항', '3. 튜닝대상 및 결과, 제약사항', '4. 향후 제언' 의 3가지 내용이 들어가 있는데, 데이터베이스는 오라클이고 튜닝 도구로서는 SQL_Trace를 사용했다.

이 은행에서는 성능을 시급히 향상시켜야 할 SQL 11본(중요 SQL)과 기타 성능에 영향을 줄 다른 요인들을 분석해 시급한 SQL에 대해서는 목표 성능을 달성하고, 나머지 부분들은 대안을 제시해 줄 것을 요청했다.

〈표 1〉은 '튜닝대상 및 제약사항' 에 대한 내용이다. 표에서 좌측의 튜닝 대상을 보면 SQL, 클라이언트 애플리케이션(MS의 ASP 프로그램), DBMS 시스템(메모리, 디스크 I/O, CPU), 데이터 모델/스키마(결합 인덱스, 인덱스, 무결성 등)로 구분한 내용이 있음을 알 수 있을 것이다.

SQL은 SQL_Trace 등의 성능튜닝 유틸리티를 실행시켜 옵티마이저(Optimizer)의 데이터 액세스 실행계획을 살펴 잘못된 부분을 개선해 성능을 높이고, SQL만으로 성능이 개선 안 되는 부분은 클라이언트 애플리케이션을 분석해 개선방안을 찾는다. DBMS 시스템의 메모리 부분과, 디스크 I/O부분, CPU 부분을 튜닝해 성능개선 사항을 찾는데, 이 부분은 주로 메모리의 사용상태나 클라이언트의 요청으로 실행되는 애플리케이션의 인덱스 테이블의 사용 정도와 네트워크 트래픽에 의한 CPU의 점유작업 상태를 분석하게 된다. SQL이나 클라이언트 애플리케이션 및 DBMS 시스템의 성능튜닝을 통해서도 목표한 성능이 나오지 않으면, 근본적인 시스템 개선을 위해 데이터 모델 자체의 추가나 삭제, 분할, 중복 등의 반정규화(De-Normalization) 작업을 수행하게 된다.

〈표 2〉는 '튜닝대상 및 결과, 제약사항' 에 대한 내용이다. 표에서 우측의 튜닝 권고사항을 보면 옵티마이저의 Cost_base 방식과 Rule_base 방식의 장단점 등의 사용방안에 대한 문제점 및 개선방안이 지적되고 있다.

DDBMS의 성능튜닝에서의 가장 큰 문제점은 바로 이 옵티마이저의 데이터 액세스 순서(실행계획, Explain Plan)를 이해하는 것이 급선무이고, DBMS 시스템 내부의 상황을 이해하는 것이 급선무이다. 그런데 이 옵티마이저는 시시각각으로 시스템

의 상태에 따라 개발자들이 작성해 놓은 SQL을 무용지물로 만들어 목표성능 유지를 어렵게 만드는 주범이다. DBMS 시스템의 상태 또한 시시각각으로 변해 예측을 불가능하게 하는 수가 많은데, 이렇게 성능관리에 문제가 생기면 상당히 많은 비용을 성능튜닝 전문가나 DDBMS 업체에 지불하게 되고, 또 DBMS 유지보수 및 DBA 인원에 대한 비용 등 끝없이 발생되는 데이터베이스 성능 관리 및 해결, 유지비용에 대한 문제가 시스템을 운영해 경영에 도움을 받으려는 기관이나 기업에 큰 문제가 되고 있다.

〈표 3〉은 '향후 제언'에 대한 것이다. 표에서 좌측의 튜닝 대상에 대한 우측의 향후 고려할 사항을 적어 놓았다. DDBMS를 사용시에 목표하는 성능을 만족시키기 위해는 데이터 모델링을 이론과 업무특성에 맞춰 잘 설계를 해야 하고, SQL을 사용할 때에는 DDBMS에서 사용하는 옵티마이저의 특성에 맞게 SQL을 작성해야 하며, 관계형 데이터베이스의 특성에 맞게끔 DBMS의 메모리, 디스크 입출력 등을 고려한 복잡하고 여러 가지를 고려한 시스템 설계를 해야 한다. DDBMS는 하드 디스크의 데이터베이스 테이블들을 메모리로 가져오는 과정과, 가져온 데이터를 메모리에서 처리하는 과정 때문에 메모리의 상태에 따라 성능 문제가 발생하기 때문이다.

전술한대로, 이러한 시스템 설계와 운영을 위해서는 다양한 업무 경험과 DDBMS에 정통한 기술을 습득하고 있어야 하고, 기술적으로 자문을 해 줄 업체와 전문가들을 많이 알고 있어야 필요시 도움을 받을 수 있다. 그런데 대규모의 대용량 데이터베이스를 운영하면서 들어가는 많은 비용을 유지하는데 부담이 있거나, 대용량 데이터베이스 중에서 일부분의 데이터만으로 운용이 필요한 경우 DDBMS를 사용하면서 MMDBMS를 사용한다면 효율적으로 시스템을 운영할 수가 있게 된다.

DDBMS에서 데이터를 저장하고 있는 데이터베이스 테이블이 하드디스크와 메모리에 혼재돼서 성능 문제를 일으키는 반면에, MMDBMS는 데이터를 메모리에서만 처리하기 때문에 원천적으로 하드디스크에서 데이터를 가져오는 시간을 아예 고려할 필요가 없다. 메인 메모리 가격이 크게 떨어지고, 처리능력이 크게 향상되는 최근의 메모리 발전 추세에 따라 급격한 비교우위가 생겨날 것이라고 생각한다.

또한 DDBMS에서처럼 하드디스크에서 데이터를 가져오는 특성을 감안한 SQL의 여러 중요한 기능들을 감안하지 않아도 되므로, SQL 문법이나 종류가 매우 단순해 질 수가 있다. 이것은 개발자들을 힘들게 하는 복잡하고 어려운 SQL문 작성의 부담으로부터 해방시켜주는 일이 될 수도 있다. 최근에 나오는 토종 MMDBMS 업체들은 기존 오라클, 인포믹스, 사이베이스,

〈표 3〉 향후 제언

튜닝 대상	향후 고려할 사항
SQL	· 현재 옵티마이저 모드를 Rule based로 해 운영하는 데에 있어서의 장단점 검토와 향후 Cost based로 하기 위한 전체적인 검토가 필요하다. (Rule based로 할 경우 DB 구조를 잘 알고, 인덱스 개념보다 PK 개념으로 사용하는 부분에 알맞은데, 결합인덱스 선정시 유의해야 함.) – 조인시 테이블의 순서가 최적화돼 있지 못할 경우가 발생. – 원하는 인덱스를 활용할 수 없는 경우가 발생 – 향후 Cost based로 갈 경우, 주기적인 애널라이즈 작업 필요. · 트래픽에 따른 성능 저하가 존재하는 SQL문에 대해서, DBMS, 데이터모델, 애플리케이션, 서버 부분에 대한 전체적인 검토가 필요하다. · 중요 테이블(통화이력, 상담이력 등)의 백업&리커버리 전략 수립이 필요하다. · 종합적인 검토에 의한 몇 개의 통계 테이블 생성 활용이 요구된다. (사전에 미리 생성해 놓을 것) · 테이블 전체 스캔에 대한 적절한 인덱스 생성이 필요하다.
클라이언트 애플리케이션 (ASP 프로그램)	· 애플리케이션(ASP)에서 처리할 프로세스를 메모리 공간이 적은 DBMS에서 SQL로 처리하는 부분의 전체적인 검토가 필요하다.
DBMS 시스템 (메모리, 디스크 I/O, CPU)	· 메모리를 효율적으로 사용하지 못하는 현상이 있다(적중률 분석), 데이터 버퍼/SQL 버퍼 부분을 주기적으로 최적화해줘야 한다. · 테이블 스페이스의 조정이 필요하다.(Chained Row 삭제 필요)
데이터 모델/스키마 (결합인덱스, 인덱스 무결성 등)	· 현행 다량의 건수를 가진 테이블의 주키 영역의 컬럼 배열이 일자+시간 순서여서, 특정 애플리케이션에만 유리하고, 다른 컬럼을 검색조건으로 하는 프로그램에는 유리하지 않다. · 인덱스가 많이 사용되는데, 애플리케이션별 최적의 인덱스 개수와 컬럼 순서를 재고할 필요가 있다.

MS-SQL에 뒤지지 않는 SQL 기능이나 함수들을 개발해 개발자들이 쉽게 SQL을 작성하도록 지원하고 있다.

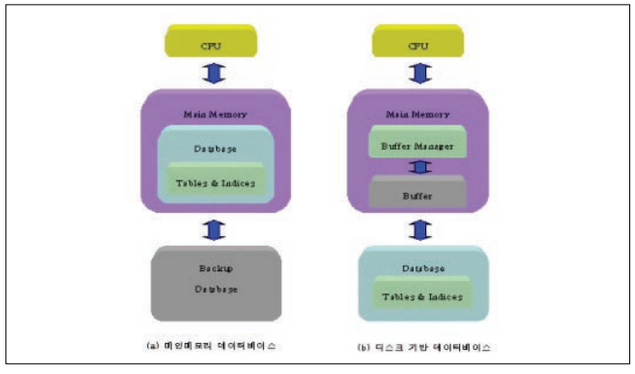
중요한 또 하나의 기능은 기존 DDBMS와 게이트웨이 등을 이용해 데이터를 호환할 수가 있는데, 실시간으로 양방향으로 실행이 가능해 편하게 시스템의 성능목표를 유지할 수가 있다. 특히 DDBMS의 강점을 많이 보완해 데이터를 메모리에서 처리할 때 발생할 수 있는 백업, 보안, 로그처리 등의 기능들을 포함하고 있어서 DDBMS와 병행해 사용해도 전혀 문제될 사항이 없다.

빠른 속도와 분량이 적은 소프트웨어(엔진), 저렴한 가격, 다양한 답변(Replication) 및 이기종 접속 가능 측면을 고려할 때 이동통신, 모바일, 데이터 마트, 콜 센터, 고객관리 솔루션 등에 사용한다면 최적의 투자대비 이익을 거둘 수가 있을 것이다.

DDBMS와 MMDBMS의 성능비교 사례연구

이번에는 구체적으로 DDBMS와 MMDBMS의 성능을 비교하는 테스트 수행결과를 살펴보기로 한다.

결코 DDBMS의 탁월한 기능들과 특성들을 MMDBMS가 능가할 수는 없지만, 몇몇 영역 특히 성능향상 측면에서는 MMDBMS의 활용이 DDBMS를 크게 앞설 수 있음을 사례를 통해 분석해보고자 한다.



▲ 디스크 기반 DBMS와 메모리 DBMS의 데이터 처리 개념도
(출처 : 알티베이스)

테스트 내용

DDBMS인 오라클 데이터베이스내에 테이블을 만들고 자바 프로그램을 이용해 데이터를 입력, 삭제, 수정, 조회하는 과정에서의 소요시간과, 국내 T사의 MMDBMS인 텔코베이스에 동일한 절차로 소요시간을 측정해 본다.

· 제1창(mdb_client 창)

다음은 텔코베이스의 데이터베이스 유틸리티인 mdb_client를 띄우고 작업하는 창에서 수행하는 작업이다. cmd 창을 띄워놓고 아래와 같은 작업을 수행한다. 아래의 디렉토리는 작업자의 컴퓨터 상태에 따라 다를 수 있다.

```
마이크로소프트 윈도 XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.
C:\windows\system32>demo  //demo는 뱃치 파일이다.(demo.bat)
C:\windows\system32>cd c:\Telcobase\demo
C:\Telcobase\demo>ls -al
drwxr-xr-x    3 rlarPcjf Administ     0 May 7 18:48 oracle
drwxr-xr-x    3 rlarPcjf Administ     0 May 7 18:42 telcobase

C:\Telcobase\demo>cd telcobase
C:\Telcobase\demo\telcobase>ls -al
-rw-r--r--    1 rlarPcjf Administ   559226 May 7 18:07
delivery_info.txt
-rw-r--r--    1 rlarPcjf Administ   632028 May 7 18:07 invoice.txt
drwxr-xr-x    2 rlarPcjf Administ     0 May 7 18:33 jdbc
-rwxr-xr-x    1 rlarPcjf Administ    231 May 7 18:07 load.bat

C:\Telcobase\demo\telcobase>
:\Telcobase\demo\telcobase>cat tab.bat
mdb_client -b schema.sql
C:\Telcobase\demo\telcobase>tab  <--- 테이블 생성
C:\Telcobase\demo\telcobase>mdb_client -b schema.sql
Ok, table order_info is created.
Ok, index idx1_order_info is created.
Ok, There are 13 tables
```

TABLE NAME
order_info
orders

:\Telcobase\demo\telcobase>mdb_client <------ 텔코베이스의 SQL입력창 호출.

```
Configuring with  C:\Telcobase\etc\telcobase.conf'
Interactive Query of Telcobase v1.2.5.
Copyright 2001 2002 2003, Telcoware, Ltd.
Type 'help' to see a brief TQL(Telcoware Query Language)
information.

(MDB) show tables;
Ok, There are 13 tables
```

TABLE NAME

order_info ·····〈——— 13개 테이블 중 11개 생략.
orders

(MDB)
(MDB) desc order_info;
Ok, Table order_info has 0 row(s).

Attribute Name	Type
order_no	CHAR(11)
product_no	CHAR(11)
warehouse_no	CHAR(11)
order_qty	INT
unit_discount_rate	DOUBLE

Index list on Table order_info:
Index Name: idx1_order_info Attribute List: order_no, product_no
(unique TTREE 10)
(MDB)

· 제2창(텔코베이스의 테이블에 데이터 값 입력하는 자바프로그램의 컴파일 및 실행 창)

다음은 텔코베이스의 주문정보 테이블에 10,000건의 데이터를 입력, 조회, 수정, 삭제하는 자바 프로그램의 컴파일 및 실행에 관련된 내용이다. 새로운 cmd 창을 띄워놓고 아래와 같은 작업을 수행한다.

```
:\Telcobase\demo\telcobase>cd jdbc
C:\Telcobase\demo\telcobase\jdbc>ls -al
-rw-r--r--  1 rlarPcjf Administ   6471 May 7 18:33 exam1.class
-rw-r--r--  1 rlarPcjf Administ   7502 May 7 18:07 exam1.java
-rw-r--r--  1 rlarPcjf Administ   224 May 7 18:07 makefile

C:\Telcobase\demo\telcobase\jdbc>java exam1 1 1 10000  〈---
Telcobase Insert
Executing benchmark(Insert table), please wait
Elapsed time          : 0.37sec
Transactions Per Sec : 27024
Done.

C:\Telcobase\demo\telcobase\jdbc>java exam1 2 1 10000  〈----
Telcobase select
Executing benchmark(Select table), please wait
Elapsed time          : 0.19sec
Transactions Per Sec : 52626
Done.

C:\Telcobase\demo\telcobase\jdbc>java exam1 3 1 10000 〈----
Telcobase updae
Executing benchmark(Update table), please wait
Elapsed time          : 0.341sec
Transactions Per Sec : 29322
Done.

C:\Telcobase\demo\telcobase\jdbc>java exam1 4 1 10000 〈----
Telcobase delete
Executing benchmark(Delete table), please wait
Elapsed time          : 0.18sec
Transactions Per Sec : 55549
Done.
```

제1창(텔코베이스 mdb_client 창)에서 입력된 데이터 확인하기

다음은 텔코베이스의 주문정보 테이블에 데이터가 입력됐는가를 확인하는 내용이다. 10,000건의 데이터가 입력됐음을 확인할 수 있다.

(MDB) select count(*) from order_info;
Ok, 1 row is selected.

COUNT (*)
10000

· 제3창(sqlplus 창)

다음은 오라클 데이터베이스 유틸리티인 sqlplus를 띄우고 작업하는 창에서 수행하는 작업이다.
cmd 창을 띄워놓고 아래와 같은 작업을 수행한다.

```
C:\Telcobase\demo>ls -al
drwxr-xr-x  3 rlarPcjf Administ     0 May 7 18:48 oracle
drwxr-xr-x  3 rlarPcjf Administ     0 May 7 19:52 telcobase

C:\Telcobase\demo>cd oracle
drwxr-xr-x  2 rlarPcjf Administ     0 May 7 19:32 jdbc
-rwxr-xr-x  1 rlarPcjf Administ   287 May 7 18:44 load.bat
-rw-r--r--  1 rlarPcjf Administ 6246400 May 7 18:44 oracle_int.tar
-rw-r--r--  1 rlarPcjf Administ  639223 May 7 18:44 order_info.txt
-rw-r--r--  1 rlarPcjf Administ    3099 May 7 18:44 schema.sql
-rwxr-xr-x  1 rlarPcjf Administ     32 May 7 18:44 tab.bat

C:\Telcobase\demo\oracle>tab  〈---- 오라클 테이블 생성하는 뱃치파일

C:\Telcobase\demo\oracle>sqlplus scott/tiger@schema.sql
SQL*Plus: Release 8.1.6.0.0 - Production on 금 May 7 19:54:44
2004
(c) Copyright 1999 oracle Corporation. All rights reserved.
Connected to:
oracle8i Enterprise Edition Release 8.1.6.0.0 - Production
With the Partitioning option
JServer Release 8.1.6.0.0 - Production

C:\Telcobase\demo\oracle>sqlplus 〈---- check data from sqlplus

SQL*Plus: Release 8.1.6.0.0 - Production on 금 May 7 19:56:42
```

```
2004
(c) Copyright 1999 oracle Corporation. All rights reserved.
사용자명 입력: scott
암호 입력: (tiger)
Connected to:
oracle8i Enterprise Edition Release 8.1.6.0.0 - Production
With the Partitioning option
JServer Release 8.1.6.0.0 - Production

SQL> select * from tab;
ORDER_INFO              TABLE
SQL> select count(*) from order_info;  <---- 오라클 테이블에 데이터가
저장돼 있는가를 확인.

 COUNT(*)
----------
     0 <------ 데이터가 입력되지 않았음을 알 수 있다.
```

· 제4창(오라클의 테이블에 데이터 값 입력하는 자바프로그램의 컴파일 및 실행 창)

오라클의 주문정보 테이블에 데이터를 1에서 10,000까지 입력한다.

```
C:\Telcobase\demo\oracle>cd jdbc
C:\Telcobase\demo\oracle\jdbc>java exam1 1 1 10000 <---- oracle
insert
Executing benchmark(Insert table), please wait
Elapsed time      : 20.389sec
Transactions Per Sec : 490
Done.

C:\Telcobase\demo\oracle\jdbc>java exam1 2 1 10000 <--- oracle
select
Executing benchmark(Select table), please wait
Elapsed time      : 6.269sec
Transactions Per Sec : 1594
Done.

C:\Telcobase\demo\oracle\jdbc>java exam1 3 1 10000 <--- oracle
update
Executing benchmark(Update table), please wait
Elapsed time      : 18.136sec
Transactions Per Sec : 551
Done.

C:\Telcobase\demo\oracle\jdbc>java exam1 4 1 10000 <--- oracle
delete
Executing benchmark(Delete table), please wait
Elapsed time      : 19.378sec
Transactions Per Sec : 515
Done.
```

테스트 환경

테스트를 간편하게 하기 위해 윈도 XP 환경에서 JDK를 설치하고, 오라클 8.1.6 윈도용을 설치했으며, 텔코베이스 1.2.5 윈도용과 유틸리티(gateway, jdk, env, 오라클 Driver 등)을 설치한다. 편의상 설치 과정은 생략하기로 한다.

테스트 과정

cmd 창(DOS 창)을 5개를 띄워 상호 비교가 가능하게 한다. 하나는 텔코베이스 SQL 입력창(mdb_client), 텔코베이스가 가지고 있는 테이블에 데이터를 입력하는 자바 프로그램이 위치한 창(자바 프로그램의 컴파일 및 실행), 오라클 데이터베이스의 SQL 입력창(sqlplus), 오라클 데이터베이스가 가진 테이블에 데이터를 입력하는 자바 프로그램이 위치한 창(자바 프로그램의 컴파일 및 실행), 자바 프로그램의 소스를 편집할 수 있는 창의 5가지로 나누어 띄운다. 여기서는 편의상 진행절차에서 나타난 결과들을 생략하면서 필요한 결과만 보여주도록 했다.

상기한 예를 정리해보면 DDBMS인 오라클과 MMDBMS인 텔코베이스의 동일한 테이블에 10,000 건의 데이터를 입력, 삭제, 수정, 조회하는 데 걸리는 시간(초)은 다음과 같다. 아래의 표에서 보는 대로, MMDBMS는 DDBMS보다 33배~108배 정도 빠르게 데이터를 처리하고 있는데 데이터가 많아질 수록 그 격차는 매우 커졌다.

〈표 4〉

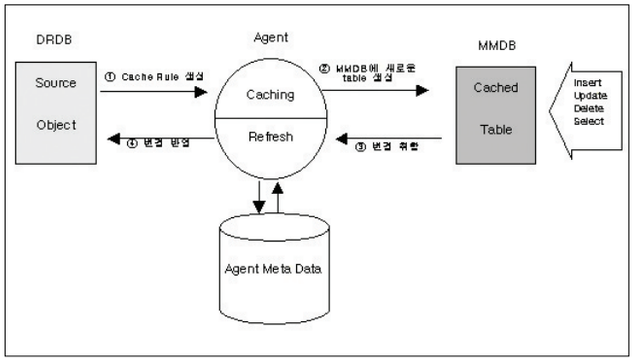
구분	입력시(초)	조회시(초)	수정시(초)	삭제시(초)
오라클	20.389	6.269	18.136	19.378
텔코베이스	0.37	0.19	0.341	0.18

DDBMS와 MMDBMS의 데이터호환 사례연구

이번에는 DDBMS인 오라클과 MMDBMS인 텔코베이스간에 동적으로 데이터가 호환되는 과정을 살펴보기로 한다. 텔코베이스의 게이트웨이를 이용해 텔코베이스와 오라클간의 데이터가 동기적으로 양방향으로 상호 교환된다.

· 제3창(oracle sqlplus 창)에서 테이블내의 로우(레코드) 건수 체크.

다음은 오라클 데이터베이스의 sqlplus 창에서 주문정보 테이블의 로우 건수를 체크한 것이다.



▲ 디스크 기반 DBMS와 메모리 DBMS의 데이터 처리 개념도
(출처 : 알티베이스)

```
SQL> select count(*) from order_info;
 COUNT(*)
----------
        0
```

· 제1창(텔코베이스 mdb_client 창)에서 데이터베이스 내의 모든 테이블의 제거 및 오라클에서 특정 테이블을 가져오기.

다음은 텔코베이스 데이터베이스의 모든 테이블을 제거하고 오라클 데이터베이스에서 주문정보 테이블만을 텔코베이스 내로 가져오는 과정이다.

```
(MDB) show tables;
Ok, There are 13 tables
```

TABLE NAME
delivery_invoice ……. order_info

```
(MDB) reinit; <--- 텔코베이스의 모든 테이블의 제거
Do you really want to reinitialize MMDB (y/n) ? y
Ok, DB is reinitialized
(MDB)
(MDB) show tables;
Ok, There is 0 table
(MDB)
(MDB) start daemon all; <--- gateway 구동 : 오라클에서 주문정보 테이블만 텔코베이스로 가져오기
Ok, starting daemon MGATEW'
  (command = [c:\Telcobase\bin\gateway -M MGATEW -m "oracle gateway" -L 4 ora1 -S -T oracle.jdbc.driver.oracleDriver jdbc:oracle:thin:@10.10.2.74:ora8 scott tiger "order_info" 2>&1 | C:\Telcobase\bin\tee2 -xp c:\Telcobase\trace\MGATEW])
(MDB)
(MDB) check daemon all;
Ok, 1 row is selected.
```

DaemonName	PID	Remark
MGATEW	3656	oracle gateway

```
(MDB)
(MDB) show tables; <--- 오라클로부터 가져온 테이블의 내역보기.
Ok, There is 1 table
```

TABLE NAME
order_info

```
(MDB)
(MDB) select count(*) from order_info; <--- 입력작업을 하기 전에 테이블의 로우 확인하기.
Ok, 0 row is selected.
```

· 제2창(Telcobase jdbc 창)에서 exam1.java 프로그램 실행시키기.

다음은 오라클에서 가져온 텔코베이스 주문정보 테이블에 100건의 데이터를 입력하는 내용이다.

```
C:\Telcobase\demo\telcobase\jdbc>java exam1 1 1 100
```

〈--- Telcobase insert
Executing benchmark(Insert table), please wait
Elapsed time : 0.01sec
Transactions Per Sec : 9900
Done.

· 제3창(oracle sqlplus 창)에서 확인하기

다음은 오라클의 주문정보 테이블에 데이터가 입력됐는가를 확인하는 내용이다. 텔코베이스의 주문정보 테이블에 데이터가 입력됨과 동시에 오라클의 주문정보 테이블에 데이터가 입력됐음을 알 수 있다.

SQL〉 select count(*) from order_info;
COUNT(*)

100

· 제4창(oracle jdbc 창)에서 데이터 입력하기

다음은 오라클의 주문정보 테이블에 데이터를 입력하는 내용이다. 오라클에 데이터가 입력되면 동시에 텔코베이스에도 데이터가 입력되게 된다.

C:\Telcobase\demo\oracle\jdbc〉java exam1 1 101 200
〈--- oracle insert
Executing benchmark(Insert table), please wait
Elapsed time : 0.341sec
Transactions Per Sec : 290
Done.

· 제1창(텔코베이스 mdb_client 창)에서 입력된 데이터 확인하기

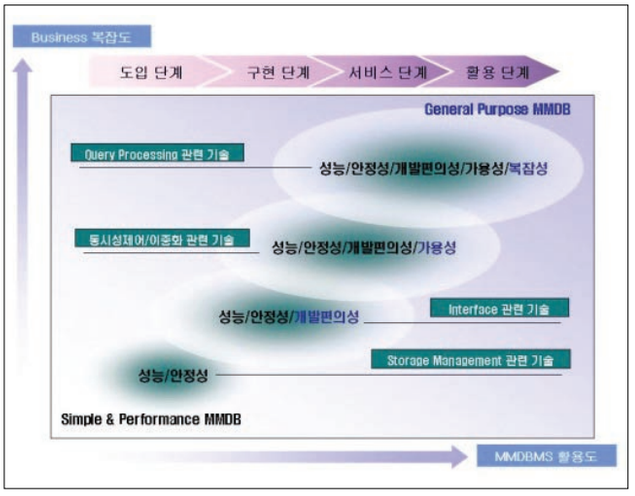
다음은 오라클의 주문정보 테이블에 데이터를 입력하고 동시에 텔코베이스의 주문정보 테이블에 데이터가 동일하게 입력됐는가를 확인하는 내용이다.

(MDB) select count(*) from order_info;
Ok, 1 row is selected.

COUNT (*)
200

MMDBMS인 텔코베이스에서 입력한 데이터는 동기화된 DDBMS인 오라클로 데이터가 자동으로 입력이 됐고, 마찬가지로 오라클에서 입력한 데이터는 동기화된 텔코베이스로 자동으로 데이터가 입력이 됐다.

오라클의 데이터베이스에서 일부 테이블 그리고 대용량의 데이터를 가진 특정 테이블에서 일부분의 데이터를 텔코베이스 테이블로 가져오게 할 수도 있다. 이렇게 함으로써 대용량의 DDBMS에서 개발자가 많은 노력과 최고의 기법을 사용해 SQL문을 작성해 성능향상에 도움이 되게 하는 작업과 비용을 경감할 수가 있게 된다.



이처럼 대용량의 DDBMS가 가지는 성능향상과 유지보수의 고비용을, 저렴한 가격의 MMDBMS로서 지원할 수가 있다면 시스템적으로 안정적이고 미래지향적인 기술이 적용된 DDBMS에서 없어서는 안 될 파트너가 돼 데이터베이스 응용프로그램 개발자들을 도와주는 것은 물론, 대용량의 고가이고 목표성능 유지에 비용이 많이 드는 데이터베이스인 DDBMS를 운영하는 현업 담당자들이나 경영진에게 시간과 노력을 경감시켜 줄 수 있는 좋은 친구가 될 것이다.