

Real Alternative DBMS **ALTIBASE**

Altibase User Memory Tablespace

알티베이스 기술 백서
2008년 6월



목 차

사용자 메모리 테이블스페이스의 필요성	3
테이블스페이스 종류	3
사용자 메모리 테이블스페이스 특징.....	4
메모리 테이블스페이스 구조.....	4
메모리 테이블스페이스 속성.....	5
메모리 테이블스페이스의 상태	5
메모리 테이블스페이스 활용.....	6
메모리 객체 분산 관리	6
백업 및 완전/불완전 복구.....	7
메모리 리소스 한계 상황 대처	8
체크포인트 시 디스크 부하 분산.....	9
일부 데이터를 이용한 별도 시스템 구축.....	9
메모리 테이블스페이스 관리	10
결론.....	11

Copyright © 2008 ALTIBASE Corporation. All Rights Reserved.

이 문서는 정보 제공을 목적으로 제공되며, 사전에 예고 없이 변경될 수 있습니다. 이 문서는 오류가 있을 수 있으며, 상업적 또는 특정 목적에 부합하는 명시적, 묵시적인 책임이 일체 없습니다.

이 문서에 포함된 알티베이스 제품의 특징이나 기능의 개발, 발표 등의 시기는 알티베이스 재량입니다. 알티베이스는 이 문서에 대하여 관련된 특허권, 상표권, 저작권 또는 기타 지적 재산을 보유할 수 있습니다.

Altibase 사용자 메모리 테이블스페이스

효율적인 대용량 메모리 데이터 관리 솔루션

초기 버전의 Altibase 메인 메모리(Main Memory) 상주형 DBMS는 시스템의 제한적인 메모리 용량 때문에 적은 양의 데이터를 고성능으로 처리하는 것에 만족되었습니다.

그러나 기술의 발달로 점차 시스템에서 사용 가능한 메모리 용량 제한이 사라지면서 많은 기업들이 대용량 데이터까지도 메인 메모리 DBMS를 적용하여 서비스할 수 있게 되었습니다.

알티베이스는 이러한 변화에 맞춰 DB 관리자가 대용량 메모리 데이터를 보다 효율적이고 안정적으로 관리할 수 있도록 사용자 메모리 테이블스페이스(User Memory Tablespace)를 제공합니다.

사용자 메모리 테이블스페이스의 필요성

Altibase 5 이전의 Hybrid DBMS는 기존의 메모리 DBMS 형태에 추가적으로 디스크 데이터를 지원하기 위해 논리적으로 하나의 메모리 테이블스페이스와 다수의 디스크 테이블스페이스를 지원하는 형태로 제공되었습니다. 그러나 Hybrid DBMS가 점차 대용량의 메모리 데이터 지원과 보다 많은 디스크 데이터를 함께 관리하도록 도전 받으면서, DB 관리자는 보다 다양한 상황에서 데이터베이스를 관리해야 할 필요성이 증가했습니다.

알티베이스는 기업이 유연하고 효율적으로 대용량의 데이터베이스를 관리할 수 있도록 사용자 메모리 테이블스페이스의 개념을 도입했습니다. 데이터베이스 관리자는 자사에 Altibase를 적용해 대량의 메모리 데이터를 여러 개의 사용자 메모리 테이블스페이스로 구성함으로써 보다 다양한 방법으로 백업 계획을 수립할 수 있도록 했습니다. 또한 장애가 발생하여도 적은 비용(시간)으로 복구가 가능해져 데이터베이스 관리를 위한 여러 작업들을 보다 쉽게 수행합니다.

테이블스페이스 종류

Altibase Hybrid DBMS는 데이터가 저장되는 위치에 따라 다음의 3가지 테이블스페이스를 지원합니다.

- 디스크 테이블스페이스
- 메모리 테이블스페이스
- 휘발성(Volatile) 테이블스페이스

디스크 테이블스페이스에 생성된 객체의 데이터는 파일 시스템에 기록이 되며 필요한 경우 버퍼 풀에 데이터를 적재하여 접근합니다.

메모리 테이블스페이스에 생성된 객체의 데이터는 DBMS 구동 시 메모리에 적재되며 이후 해당 데이터 접근 시 디스크 입출력을 유발하지 않고 메모리에서 직접 접근이 가능합니다.

휘발성 테이블스페이스는 데이터가 메모리 상에서 보관되는 점은 메모리 테이블스페이스와 동일합니다. 그러나 메모리 테이블스페이스가 데

데이터의 백업 이미지와 변경 이력을 디스크에 모두 저장하여 DBMS를 재구동시 데이터의 무결성을 보장하는 것과 달리 휘발성 테이블스페이스는 데이터의 변경 이력을 포함한 어떠한 정보도 디스크에 기록하지 않습니다. 이러한 특징으로 휘발성 테이블스페이스는 DBMS 구동을 종료할 때 저장된 객체들이 모두 사라집니다.

사용자 메모리 테이블스페이스 특징

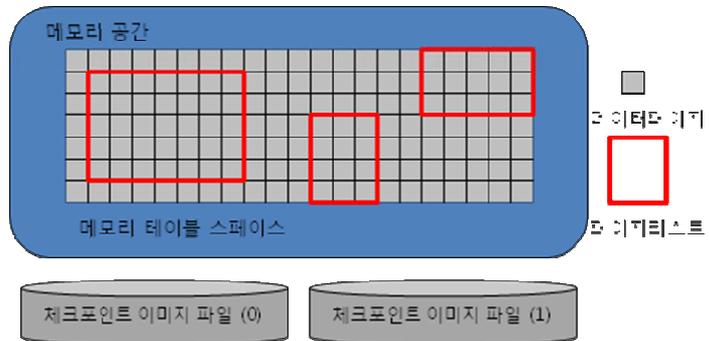
사용자 메모리 테이블스페이스란 Altibase Hybrid DBMS에서 메모리 객체를 저장할 수 있는 사용자 정의 테이블스페이스입니다.

사용자 메모리 테이블스페이스는 디스크 테이블스페이스와의 독립성이 보장되며 기존 디스크 테이블스페이스와 동일한 인터페이스를 제공합니다. 사용자는 메모리 테이블스페이스를 생성, 삭제 및 변경을 할 수 있으며 메모리 테이블스페이스 별로 백업 및 복구가 가능합니다. 또한 특정 테이블스페이스를 오프라인 상태로 변경할 수 있습니다.

메모리 테이블스페이스 구조

Altibase 메모리 테이블스페이스는 시스템 메모리 상의 메모리 데이터 페이지들의 리스트와 디스크 상의 체크포인트 이미지로 구성됩니다.

메모리 데이터 페이지들은 논리적으로 시스템 메모리 공간 상의 메모리 테이블스페이스에 저장되며, 체크포인트를 할 때 각 메모리 테이블스페이스의 체크포인트 이미지 파일에 그 변경 내용이 기록됩니다.



<그림1> 메모리테이블스페이스 구조

체크포인트 이미지 파일은 메모리 테이블스페이스 내의 객체들을 백업하기 위한 메모리 테이블스페이스 전용 파일로 DBMS 재 구동 시 저장된 객체들을 메모리 공간으로 올리고 백업 및 복구 시간을 단축하는 용도로 사용됩니다. Altibase 메모리 테이블스페이스는 핑퐁 체크포인트 방식을 사용하기 때문에 각 메모리 테이블스페이스는 두 개의 이미지 파일과 연결됩니다.

메모리 데이터에 대한 인덱스 객체는 기본적으로 운영 중 변경 내용이 기록되지 않습니다. 메모리 데이터의 인덱스 객체는 메모리 테이블스페이스가 아닌 별도의 메모리 공간에 저장되며, DBMS가 재 구동시 해당 인덱스들을 다시 생성합니다.

그러나 Altibase는 메모리 데이터의 인덱스 객체에 대해 영구 (Persistence)적인 속성으로 변경이 가능합니다. 이 속성을 사용하면 정상적으로 DBMS를 종료하거나 메모리 테이블스페이스를 오프라인 상태

로 변경할 때 디스크에 인덱스의 내용을 저장할 수 있어, DBMS를 재구
동시 저장된 이미지를 메모리로 빠르게 적재합니다.

메모리 테이블스페이스 속성

처음 데이터베이스를 생성하면 시스템 테이블스페이스를 제외하고 사용
자의 데이터 저장을 위하여 SYS_TBS_MEM_DATA라는 이름의 기본
메모리 테이블스페이스와 SYS_TBS_DISK_DATA라는 이름의 기본 디
스크 테이블스페이스가 자동으로 생성됩니다. 사용자는 이 두 개의 기
본 테이블스페이스 이외에 추가적으로 메모리 테이블스페이스나 디스크
테이블스페이스를 생성하여 사용할 수 있습니다.

메모리 테이블스페이스는 여러 개의 데이터 파일로 구성되는 디스크 테
이블스페이스에서 데이터 파일 별로 속성이 적용되는 것과 달리 모든
객체들이 하나의 선형적인 메모리 공간에 저장됩니다. 또한 체크포인트
이미지 파일만을 가지고 있어 속성이 테이블스페이스 전체에 적용됩니
다. 메모리 테이블스페이스는 생성할 때 다음의 몇 가지 속성들을 지정
할 수 있습니다.

- 초기 테이블스페이스의 크기
- 자동 확장 여부 및 확장 크기
- 테이블스페이스의 최대 크기
- 체크포인트 이미지 파일 저장 경로 및 파일 분할 여부

생성된 메모리 테이블스페이스에 대해 초기 크기와 체크포인트 이미지
파일의 분할 여부를 제외하고는 속성 변경이 가능합니다.

메모리 테이블스페이스의 상태

사용자 메모리 테이블스페이스는 디스크 테이블스페이스와 마찬가지로
다음의 3가지 상태를 가질 수 있습니다.

- 온라인 상태
- 오프라인 상태
- 폐기(Discard) 상태

메모리 테이블스페이스는 운영 상태에서 온라인/오프라인 상태로 전환
이 가능하며, DBMS 컨트롤(Control) 단계에서 폐기상태로 전환할 수 있
습니다.

온라인 상태는 테이블스페이스 내에 저장된 객체들에 정상적으로 접근
할 수 있는 상태로 모든 테이블스페이스는 일반적으로 이 상태에서 운
영됩니다. 온라인 상태의 메모리 테이블스페이스는 모든 객체 및 데이
터가 메모리에 적재되어 있습니다.

오프라인 상태는 해당 테이블스페이스가 메모리에 적재되지 않은 상태
이며 해당 테이블스페이스에 속한 객체들에 접근하지 못합니다. 메모리
테이블스페이스를 오프라인 상태로 전환하면 그때까지 변경된 페이지
내용을 연결된 체크포인트 이미지 파일에 기록하고 해당 테이블스페이
스 메모리 공간을 해제하고 시스템에 반납합니다.

폐기 상태는 테이블스페이스를 삭제하기 위한 상태로 한번 폐기 상태가
되면 다른 상태로 전환이 불가능합니다. 예를 들어 체크포인트 이미지
파일이 손상된 경우 DBMS가 정상적으로 구동되지 않을 때, 이를 해결

하기 위해 DBMS 컨트롤 단계에서 문제의 테이블스페이스를 폐기 상태로 전환합니다. 이후 서버를 다시 정상적으로 구동하여 나머지 데이터를 복구하고, 손상된 테이블스페이스를 삭제합니다.

메모리 테이블스페이스 활용

Altibase의 메모리 테이블스페이스는 디스크 테이블스페이스와는 구조적인 차이가 있습니다. 그러나 사용자가 메모리 테이블스페이스와 디스크 테이블스페이스를 구분하지 않고 동일한 인터페이스로 동일한 작업을 할 수 있도록 설계하고 구현했습니다.

따라서 사용자가 메모리 테이블스페이스를 생성하면 이후 사용은 디스크 테이블스페이스와 크게 다르지 않습니다. 다만 특정 객체를 메모리 테이블스페이스에 생성하면 해당 객체는 서버 구동 시 메모리 공간으로 적재되며 이후 해당 객체에 접근 시에 디스크 입출력 없이 빠르고 일정한 응답속도를 보장합니다.

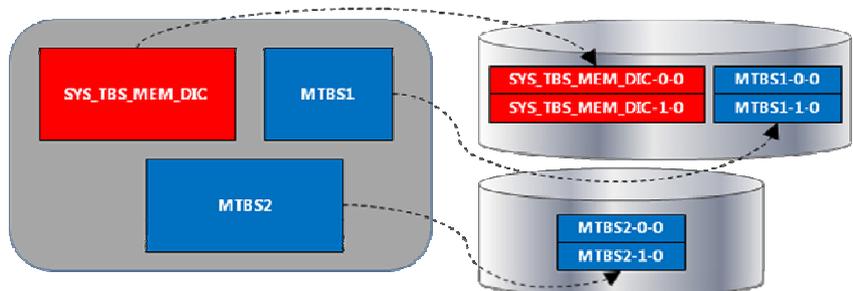
데이터베이스 관리자는 사용자 메모리 테이블스페이스를 사용하여 다음과 같이 보다 안정적이고 효율적으로 데이터베이스를 관리할 수 있습니다.

- 메모리 객체 분산 관리
- 메모리 테이블스페이스 별 백업 및 완전/불완전 복구
- 메모리 리소스 한계 상황에서 데이터베이스 관리자의 적극적이고 유연한 대처
- 체크포인트 이미지 경로 분산을 통한 디스크 부하 분산
- 일부 메모리 객체만을 이용하여 별도의 시스템 구축

메모리 객체 분산 관리

사용자 메모리 테이블스페이스를 지원하지 않을 경우 모든 메모리 객체는 하나의 체크포인트 이미지 파일로 관리되기 때문에 해당 체크포인트 이미지 파일이 손상되면 전체 메모리 객체를 사용할 수 없게 됩니다.

사용자는 메모리 객체의 중요도나 접근 빈도에 따라 각기 다른 테이블스페이스로 구성하여 데이터베이스를 운영할 수 있습니다. 이와 같이 메모리 객체를 특정 기준에 따라 분산하여 관리함으로써 예측하지 못한 장애 발생 시 그 피해를 최소화할 수 있습니다.



<그림2> 메모리 테이블스페이스 분산으로 데이터 보호

분산된 테이블스페이스 중 일부에 문제가 발생한 경우 문제가 된 테이블스페이스를 제외하고 나머지 테이블스페이스만을 이용하여 구동하는 것이 가능합니다. 이러한 방법으로 전체 데이터베이스의 데이터 중 일

부를 복구할 수 있습니다.

어떤 경우 불필요한 데이터의 증가로 인해 전체 데이터베이스가 영향을 받는 문제가 발생할 수 있습니다. 업무 중요도에 따라 객체들을 구분하고 각각 다른 테이블스페이스를 이용하여 데이터베이스를 분산하여 구축한 경우 상대적으로 중요도가 낮은 객체들을 저장하고 있는 테이블스페이스에 크기 제한을 설정함으로써 이와 같은 문제를 사전에 예방할 수 있습니다. 불필요한 데이터가 증가하면 해당 테이블스페이스는 크기 제한에 걸려 더 이상 데이터가 쌓이지 않으며 해당 객체를 접근하는 응용 애플리케이션 이외에 나머지의 응용 애플리케이션들은 중요한 업무에 대해 정상적으로 서비스를 계속 수행할 수 있습니다.

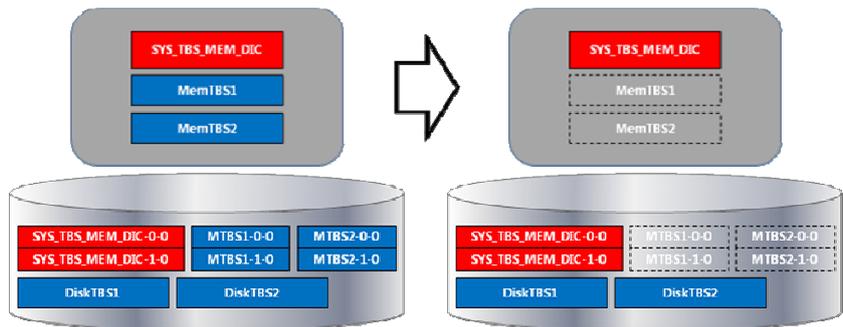
이 외에도 전체 메모리 테이블스페이스에 대한 크기를 제한함으로써 문제가 더 크게 악화되는 것을 방지할 수 있습니다. 분산되어 구축된 모든 메모리 테이블스페이스 크기의 합은 Altibase 데이터베이스의 속성인 MEM_MAX_DB_SIZE 속성의 설정 값 이상으로 확장될 수 없습니다.

Altibase는 사용자 메모리 테이블스페이스를 지원하면서, 데이터베이스 운영 중에 필요한 메타 데이터는 시스템 테이블스페이스 중 하나인 디셔너리(Dictionary) 테이블스페이스에 별도로 저장하여 보호합니다.

백업 및 완전/불완전 복구

Altibase에서는 사용자 메모리 테이블스페이스를 지원하면서 기존의 디스크 테이블스페이스와 동일한 백업 방법 및 복구 방법을 지원합니다. 데이터베이스 관리자는 디스크와 메모리의 테이블스페이스 구분 없이 원하는 일부의 데이터를 테이블스페이스 단위로 백업하는 것이 가능합니다. 그리고 백업 받은 이미지들을 이용하여 완전복구 또는 불완전 복구를 수행할 수 있습니다.

사용자 메모리 테이블스페이스가 지원되기 전에는 Altibase 운영 시 필요한 메타 정보가 논리적인 하나의 메모리 테이블스페이스에 저장되었습니다. 그렇기 때문에 일부 디스크 테이블스페이스만 백업을 받고 이후 해당 백업 이미지로 데이터베이스를 복구하는 것이 불가능했습니다.



<그림3> 메타 테이블스페이스만으로 테이블스페이스 복구 가능

따라서 복구 시 반드시 전체 메모리 테이블스페이스가 필요하였으며 점점 메모리 데이터가 대용량화되면서 데이터베이스 관리자는 많은 어려움을 겪어야 했습니다. Altibase는 사용자 메모리 테이블스페이스를 지원하면서 시스템 디셔너리 테이블스페이스를 별도로 분리함으로써 사용자 메모리 데이터가 증가하는 것과 별도로 다수의 백업 및 복구 정책을 수행할 수 있도록 지원합니다. 이제 DB 관리자는 모든 메모리 객체를 합

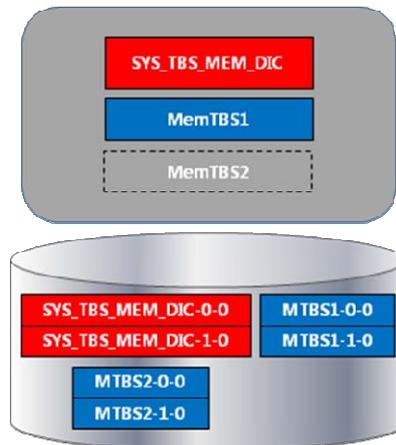
계 백업할 필요 없이 시스템 디렉터리 테이블스페이스만을 백업하여 일부 데이터의 복구 계획을 수립할 수 있습니다.

메모리 테이블스페이스에 대해서도 DBMS를 아카이브(Archive) 방식으로 운영한다면 완전 복구뿐만 아니라 아카이브 트랜잭션 로그들을 이용하여 특정 트랜잭션 로그의 위치 혹은 특정 시점으로 불완전 복구하는 것이 가능합니다. 데이터베이스 관리자는 이제 메모리 객체에 대해서도 시스템 고장으로 인한 사고 뿐만 아니라 사용자 실수에 의한 사고에 대해서도 백업된 데이터를 이용하여 복구하는 것이 가능합니다.

메모리 리소스 한계 상황 대처

모든 메모리 객체는 DBMS 구동 시 시스템 메모리 상으로 적재되어야 하며 운영 시에도 항상 메모리 상에 존재하여야 하기 때문에 시스템의 메모리 공간이 부족해지면 서비스 장애 상황이 발생할 수 있습니다. 불필요한 데이터가 쌓임으로 인해 시스템 메모리가 부족해진다면 불필요한 데이터를 삭제하고 DBMS를 재 구동하든지 전체 데이터베이스를 재 구성하여 메모리 공간을 확보해야 합니다. 그러나 대용량 데이터베이스의 경우 전체 데이터베이스를 재 구성하는데 소요되는 시간이 상당히 많이 필요하기 때문에 작업 자체가 불가능할 수 있습니다. 또한 작업 도중 전체 데이터베이스에 대한 접근이 불가능하기 때문에 서비스가 계속 중지된 상태로 작업을 수행해야 하는 어려움이 있습니다.

Altibase에서 제공하는 사용자 메모리 테이블스페이스를 이용하여 데이터베이스를 구축할 경우, 시스템의 메모리 부족이 예상된다면 특정 테이블스페이스의 상태를 오프라인 상태로 변경하였다가 다시 온라인 상태로 만들어 불필요한 메모리 공간을 줄일 수 있습니다. 이 때 기업은 서비스를 중단 없이 할 수 있습니다.



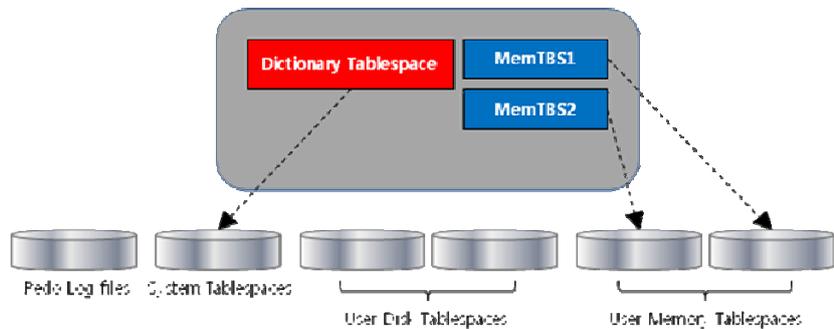
<그림4> 오프라인으로 전환하여 부족한 메모리 공간 확보

또한 메모리 공간이 부족하여 DBMS가 비정상적으로 종료된 경우 이미 커져버린 메모리 데이터 때문에 시스템에 메모리 증설 없이는 구동을 못하는 경우가 발생할 수 있습니다. 이러한 경우 삭제 가능한 테이블스페이스를 폐기 상태로 변경하는 방법으로 시스템 메모리 증설 없이 DBMS를 정상적으로 구동할 수 있습니다. 이와 같이 일부 테이블스페이스만을 이용하여 구동시킨 후 필요한 작업을 수행하여 불필요한 메모리 자원을 해제하고 DBMS를 정상적인 상태로 만들 수 있습니다.

체크포인트 시 디스크 부하 분산

시스템에 사용 가능한 물리적인 디스크가 여러 개 존재한다면 메모리 테이블스페이스의 체크포인트 이미지 경로를 나누어 지정하여 메모리 테이블스페이스 체크포인트 시 발생하는 디스크 입출력 부하를 분산할 수 있습니다.

메모리 테이블스페이스의 체크포인트의 경우 퍼지(Fuzzy) 체크포인트 기법과 핑퐁(Ping-pong) 체크포인트 기법을 적용하여 구현하였기 때문에 특별히 트랜잭션 수행에 영향을 미치지 않습니다. 하지만 데이터 변경에 대한 트랜잭션 로그가 기록되는 디스크 또는 디스크 테이블스페이스의 저장 공간과 혼용하여 사용하는 경우 디스크 입출력 경합이 발생하여 전체 DBMS 성능에 영향을 줄 수 있으므로 데이터베이스 구축 시 이러한 점을 반드시 고려해야 합니다. 가능한 한 트랜잭션 로그 저장용 디스크와 시스템 테이블스페이스, 사용자 디스크 테이블스페이스, 사용자 메모리 테이블스페이스 저장 공간은 분리하여 구축하는 것을 권장합니다.



<그림 5> 적절한 분산으로 구축된 데이터베이스 구성 예

하나의 메모리 테이블스페이스를 사용하는 경우에도 체크포인트 이미지 경로를 여러 개 지정하여 물리적으로 디스크를 분산 시키는 것이 가능합니다. 이 경우 해당 메모리테이블스페이스의 파일 분할 속성에 따라 새로 분할되는 체크포인트 이미지가 주어진 체크포인트 이미지 경로들에 차례로 생성됩니다.

일부 데이터를 이용한 별도 시스템 구축

중중 운영중인 시스템과 별도로 백업 시스템을 구축하든지 테스트 시스템을 구축하기 위해 사용중인 데이터베이스로부터 데이터를 이관할 필요가 있습니다. 만일 대용량의 메모리 데이터를 사용하고 별도로 구축해야 하는 시스템이 충분한 메모리 리소스를 가지지 못한 경우라면 일부 메모리 데이터만을 사용하여 이러한 시스템을 구축해야 합니다.

그러나 Altibase 사용자 메모리 테이블스페이스를 이용하여 메모리 객체들을 서로 다른 사용자 메모리 테이블스페이스에 분산시켜 놓았다면 데이터베이스 관리자는 쉽게 일부 데이터만을 이관하여 시스템을 구축할 수 있습니다.

데이터베이스 관리자는 이관 대상 메모리 객체들을 확인하고 해당 객체들이 포함된 메모리 테이블스페이스의 체크포인트 이미지를 복사합니다. 그리고 DBMS의 컨트롤 단계에서 나머지 메모리 테이블스페이스들에

대해 폐기 상태로 전환하여 구동한 다음 옮겨지지 않은 테이블스페이스들을 삭제하는 것으로 주어진 목적을 달성할 수 있습니다.

마찬가지로 메모리 객체를 제외한 디스크 객체들만 필요한 경우에도 전체 메모리 객체를 모두 이관할 필요 없이 필요한 디스크 테이블스페이스와 디셔너리 테이블스페이스만을 이관하여 구축하는 것이 가능합니다.

메모리 테이블스페이스 관리

Altibase에서 제공되는 성능 뷰(Performance View)를 통하여 테이블스페이스의 정보를 확인하는 것이 가능합니다.

테이블스페이스 관련 되어 다음과 같은 성능 뷰가 제공됩니다.

- V\$TABLESPACES
- V\$MEM_TABLESPACES
- V\$MEM_TABLESPACE_CHECKPOINT_PATHS
- V\$MEM_TABLESPACE_STATUS_DESC

데이터베이스 관리자는 DBMS를 컨트롤 단계로 구동한 후 V\$TABLESPACES 성능 뷰를 이용하여 각 테이블스페이스들의 종류와 상태를 확인할 수 있습니다. 또한 특정 테이블스페이스의 상태를 폐기 상태로 변경할 수 있습니다.

```

iSQL(sysdba) > select name,
      DECODE(type,
        0, 'MEMORY_SYSTEM DICTIONARY',
        1, 'MEMORY_SYSTEM DATA',
        2, 'MEMORY_USER DATA',
        3, 'DISK_SYSTEM DATA',
        4, 'DISK_USER DATA',
        5, 'DISK_SYSTEM TEMP',
        6, 'DISK_USER TEMP',
        7, 'DISK_SYSTEM UNDO',
        8, 'VOLATILE_USER DATA') TTYPE,
      DECODE(state, 1, 'OFFLINE',
        2, 'ONLINE',
        4, 'BACKUP',
        128, 'DROPPED',
        1024, 'DISCARDED') TSTATE
from v$tablespaces ;
-----
NAME                                TTYPE                                TSTATE
-----
SYS_TBS_MEM_DIC                     MEMORY_SYSTEM_DICTIONARY           ONLINE
SYS_TBS_MEM_DATA                    MEMORY_SYSTEM_DATA                 ONLINE
USR_TBS_MEM_DATA                    MEMORY_USER_DATA                   ONLINE
SYS_TBS_DISK_DATA                   DISK_SYSTEM_DATA                   ONLINE
USR_TBS_DISK_IDX                    DISK_USER_DATA                     ONLINE
USR_TBS_DISK_DATA                   DISK_USER_DATA                     ONLINE
SYS_TBS_DISK_TEMP                   DISK_SYSTEM_TEMP                   ONLINE
SYS_TBS_DISK_UNDO                   DISK_SYSTEM_UNDO                   ONLINE
8 rows selected.
iSQL(sysdba) >

```

<그림 6> V\$TABLESPACES 성능 뷰 조회 쿼리와 결과 값

DBMS가 구동되어 서비스 중이라면 V\$MEM_TABLESPACES 성능 뷰를 이용하여 메모리 테이블스페이스에 대한 할당받은 크기와 사용량 등의 여러 속성 정보를 확인 할 수 있습니다.

또한 V\$MEM_TABLESPACE_STATUS_DESC 성능 뷰와의 조인(Join)으로 메모리 테이블스페이스들의 구체적인 상태를 확인하는 것이 가능합니다.

```

iSQL> select space_name,
        b.status_desc desc,
        autoextend_maxsize/1024/1024||' M' MAX,
        current_size/1024/1024||' M' CURR,
        (alloc_page_count*32)/1024||' M' ALLOC,
        (free_page_count*32)/1024||' M' FREE,
        (alloc_page_count-free_page_count)*32/1024||' M' USED
from v$mem_tablespaces a, v$mem_tablespace_status_desc b
where a.space_status = b.status

SPACE_NAME          DESC          MAX          CURR          ALLOC          FREE          USED
-----
-
USR_TBS_MEM_DATA    ONLINE        134217727.96876 M  2500 M  2500.03125 M  25.84375 M  2474.1875 M
SYS_TBS_MEM_DATA    ONLINE        134217727.96876 M   8 M    8.03125 M    0 M    8.03125 M
SYS_TBS_MEM_DIC     ONLINE        134217727.96876 M   4 M    4.03125 M    0 M    4.03125 M
3 rows selected.
iSQL>

```

<그림 7> 메모리 테이블스페이스 상태 및 크기 조회

V\$MEM_TABLESPACE_CHECKPOINT_PATHS 성능 뷰는 메모리 테이블스페이스들의 지정된 체크포인트 이미지 경로를 확인할 수 있습니다. 메모리 테이블의 정보를 알려주는 V\$MEM_TBL_INFO 성능 뷰와 메모리 테이블에 생성되어 있는 인덱스 정보를 알려주는 V\$ MEM_BTREE_HEADER, V\$ MEM_BTREE_NODEPOOL 성능 뷰를 조합하여 특정 메모리 테이블스페이스에 포함된 메모리 객체들과 각 객체들의 크기를 확인할 수 있습니다.

```

iSQL> select b.table_name,
        ((FIXED_ALLOC_MEM+VAR_ALLOC_MEM)/1024/1024||' M' ALLOC,
        ((FIXED_USED_MEM+VAR_USED_MEM)/1024/1024||' M' USED,
        ((FIXED_ALLOC_MEM+VAR_ALLOC_MEM)-(FIXED_USED_MEM+VAR_USED_MEM))/1024/1024||' M' FREE
from v$memtbl_info a, system.tables b, v$mem_tablespaces c
where a.table_oid = b.table_oid
and a.tablespace_id = c.space_id
and c.space_name = 'USR_TBS_MEM_DATA';

TABLE_NAME ALLOC          USED          FREE
-----
T1          5.03125 M    5 M          0.03125 M
T2          2473.28125 M 1864.35248565674 M 608.928764343262 M
2 rows selected.
iSQL>

```

<그림 8> 특정 메모리 테이블스페이스에 저장된 테이블 객체 정보 및 크기 조회

```

iSQL> select index_name,
        (used_node_count+prepare_node_count)*8*128/1024/1024||' M' ALLOC
from v$mem_btree_header a, v$mem_tablespaces b
where a.index_tbs_id = b.space_id
and b.space_name = 'USR_TBS_MEM_DATA';

INDEX_NAME          ALLOC
-----
SYS_IDX_ID_102      2.0615234375 M
SYS_IDX_ID_123      25.158203125 M
T2_IDX              25.158203125 M
3 rows selected.
iSQL>

```

<그림 9> 특정 메모리 테이블스페이스에 소속된 인덱스 객체 정보 및 크기 조회

결론

대부분의 기업들은 고객들로부터 더욱 빠르고 정확한 서비스를 요구 받고 있으며 그에 따라 처리해야 하는 데이터의 양은 점차 증가하고 있습니다. 기업들은 다양한 고객의 요구에 맞춰 좀 더 빠른 서비스를 제공 하면서 데이터를 안전하게 관리하기 위해 메인 메모리 데이터베이스에 관심을 갖게 되었습니다.

초기의 메인 메모리 DBMS가 적은 데이터를 사용하는 일부 서비스에만 적용하는 수준에 만족했으나, 점차 그 활용 범위가 넓어져 이제는 대용

량의 메모리 데이터를 관리하고 서비스하는 것도 보편화되었습니다.

알티베이스는 기존 디스크 기반 DBMS에서 데이터 관리에 활용하던 테이블스페이스의 개념을 메인 메모리 DB에도 동일하게 확장 적용하여 대용량 메모리 데이터를 관리하는데 어려움을 겪는 기업들을 효과적으로 관리하도록 도와줍니다. 기업들은 대용량의 데이터를 메인 메모리 DBMS로 처리할 수 있어 빠른 응답시간을 보장하면서도 안정적이고 효율적으로 데이터를 관리할 수 있습니다.

각 기업의 데이터베이스 관리자는 Altibase에서 제공하는 사용자 메모리 테이블스페이스를 이용하여 메모리 데이터를 보다 효율적으로 구성하고 운영할 수 있으며 여러 예기치 못한 장애에 보다 효과적으로 대응할 수 있습니다.

ALTIBASE[®]

㈜알티베이스

서울특별시 구로구 구로3동 182-13
대릉포스트 2차 1008호
대표전화 : 02-2082-1000
FAX : 02-2082-1099

대전사무소

대전광역시 서구 둔산동 921
주은리더스텔 901호
대표전화 : 042-489-0330
FAX : 042-489-0081

제품구입 및 기술지원

02-2082-1114
support@altibase.com

<http://www.altibase.com>
<http://adc.altibase.com>