Real Alternative DBMS ALTIBASE, Since 1999

ALTIBASE 디스크I/O 병목을 고려한 볼륨구성 가이드

ALTIBASE 6

2015. 11

Document Control

Change Record

Date	Author	Change Reference
2009-12-23	lim272	Created
2015-11-11	omegaman	changed

Reviews

Date	Name (Position)
2010-01-04	dplee(TS)
2010-01-06	bluetheme(TS)
2010-01-07	omegamen(SC), hychoi(TC), kobul(OC)

Distribution

Name	Location

목차

개요

일반적으로 DBMS는 트랜잭션을 처리할 경우 WAL(Write Ahead Logging) 프로토콜에 기반을 둔 복구기법을 위해 리두로그를 기록한다. 리두로그를 기록한 후 데이터를 변경/저장/삭제하는 것이 일반적인 트랜잭션 처리 과정이다. 리두로그와 데이터 모두 영속성을 지닌 물리적인 저장공간을 필요로 한다. 각각의 저장공간은 동일한 디스크에 구성할 수도 있고 물리적으로 분리된 디스크에 구성을 할 수도 있다.

문제는 동일한 디스크에 리두로그와 데이터 저장영역을 구성 하였을 경우 동시에 수많은 트랜잭션을 처리하는 과정에서 각 트랜잭션의 리두로그 기록단계와 데이터 변경단계는 불가피하게 디스크 접근에 대한 I/O경합을 발생시킬 수 있다.

본 문서는 ALTIBASE DBMS가 이러한 리두로그의 기록단계와 데이터의 기록단계를 어떻게 처리하는지 알아보고 디스크병목을 최소화할 수 있는 바람직한 디스크 구성방법을 제시한다.

본 문서는 ALTIBASE 버전 5 이상을 기준으로 작성되었다.

디스크I/O의 발생유형

리두로그

ALTIBASE의 리두로그는 트랜잭션을 처리하는 과정에서 발생하는 모든 변경사항을 기록한다. 즉, 데이터를 복구 시켜야 하는 경우를 대비하여 필요한 모든 정보를 기록한다.

이러한 정보에는 트랜잭션을 처리하는데 필요한 리소스의 변경 내역과 데이터의 변경과정에서 발생하는 모든 정보를 포함한다.

ALTIBASE의 리두로그는 mmap에 의해 메모리상에 맵핑된 영역에 우선 기록되며 LogSyncThread에 의해 주기적으로 리두로그파일에 저장된다. mmap에 맵핑된 메모리의 변경내역은 호출한 프로세스가 비정상 종료되어도 OS가 파일로 저장되는 것을 보장한다. 따라서, 정전이나 OS hang과 같은 상황이 아니라면 로그의 유실은 발생하지 않는다. (장애 발생 시 복구에 대해서는 장애대처 가이드를 참고한다.)

ALTIBASE DBMS는 성능본위의 제품이기 때문에 여러 성능저하의 요인 중 하나인 빈번한 디스크I/O를 최소화 하기 위해 위와 같은 처리방법을 기본설정으로 채택하고 있다. 매우 높은 처리성능을 요구하지 않을 경우 사용자는 리두로그 기록방법을 프로퍼티를 통해 변경할 수 있다.

체크포인트

ALTIBASE는 메모리DB와 디스크DB를 동시에 지원하는데 메모리DB의 경우 어떤 단계에서 디스크I/O가 발생하는지 알아보도록 한다.

구동단계에서 메모리DB는 디스크에 저장된 모든 데이터를 메모리상에 상주시킨다. 메모리에 데이터를 올리는 공간은 페이지 단위로 관리하며 한 페이지의 크기는 32K로 고정되어 있다. 각 페이지는 페이지가 속한 테이블의 레코드 크기에 따라 n개의 슬롯(Slot)으로 분할되며 각 슬롯마다 데이터가 저장되는 구조이다.

트랜잭션이 진행되는 과정에서 데이터가 변경/저장/삭제가 되면 해당 데이터가 저장된 페이지는 내부적으로 관리하는 더티페이지(Dirty Page) 리스트에 등록이 된다. 이 더티페이지 리스트에 관리되는 페이지들을 물리적인 파일에 저장하는 과정을 체크포인트 라고 부른다.

메모리는 휘발성인 특성을 갖고 있기 때문에 체크포인트와 같은 과정을 수행하지 않으면 정전과 같은 상황에서 데이터는 영속성을 가질 수 없다.

체크포인트와 관련한 좀 더 자세한 사항은 체크포인트 문서를 참조하도록 한다. 본 문서에는 고려해야 할 부분은 주기적인 체크포인트의 과정을 통해 메모리DB 역시 디스크의 데이터파일로 저장한다는 점이다.

메모리DB에 대한 트랜잭션 처리량이 많을수록 체크포인트 단계에서 저장해야 할 더티페이지 개수도 많아지게 된다. 이때 리두로그를 저장하는 공간과 메모리DB의 데이터파일을 동일한 물리적 디스크 공간으로 설정 하였다면 체크포인트 진행단계에서 발생하는 트랜잭션들은 리두로그 기록단계와 함께 디스크 쓰기병목을 불러 일으키기 때문에 물리적으로 각각 분리된 디스크에 구성을 하는 것이 바람직하다.

다음과 같이 예를 들어 구성할 수 있다.

분류	디스크구성

리두로그 공간	/ALTIBASE_REDO_LOG
데이터 공간	/ALTIBASE_DATA

디스크DB

앞서 리두로그 기록단계와 메모리DB의 체크포인트 단계에서 디스크병목 발생의 가능성을 설명하고 분리할 것을 권고하였다. 다음으로 디스크DB의 디스크I/O 발생단계를 살펴보고 어떻게 구성을 할지 알아보도록 한다.

디스크DB는 모든 데이터를 디스크에 저장된 파일에 보관하기 때문에 트랜잭션에 의해 데이터를 접근해야 하는 경우 디스크에 저장된 데이터를 매번 탐색해야 한다. 이러한 디스크 접근비용은 성능저하를 유발하기 때문에 일반적으로 디스크DB를 지원하는 DBMS는 모두 버퍼라는 임시 메모리 저장공간을 설정하여 활용하고 있다. 자주 접근하는 데이터는 버퍼에 저장하여 매번 디스크를 탐색하는 비용을 줄이는 것이다.

버퍼는 사용자가 설정한 크기만큼 메모리에 설정되며 8K 단위의 페이지 형태로 관리된다.

디스크DB에 대한 트랜잭션이 발생하면 가장 먼저 버퍼를 탐색하여 필요한 데이터가 존재 하는지 확인하고 없을 경우 데이터파일로부터 버퍼에 적재시킨다. 버퍼에 변경된 데이터가 속한 페이지는 플러쉬리스트(Flush List)에 등록이 되고 플러쉬가 발생하는 경우 여기에 등록된 페이지들은 디스크에 저장된다. 혹은, 버퍼가 더 이상 사용할 수 있는 공간이 없는 경우 LRU알고리즘에 의해 접근빈도가 낮은 페이지들은 디스크에 저장된 후 빈 공간으로 설정되어 재사용된다. 이러한 과정은 버퍼리플레이스 (Buffer Replace) 라고 표현한다.

동일한 디스크에 메모리DB 및 디스크DB를 할당하여 사용한다면 위에서 설명한 체크포인트와 디스크DB의 트랜잭션 처리과정에서 발생하는 플러쉬, 버퍼리플레이스가 빈번하게 디스크I/O를 유발하고 디스크병목에 의한 성능저하까지 발생할 수 있다.

따라서, 메모리DB와 디스크DB를 혼용하여 사용할 경우 각각의 저장공간 역시 물리적으로 분리된 디스크로 구성하는 것이 바람직하다.

다음과 같이 예를 들어 구성할 수 있다.

분류	디스크구성
리두로그 공간	/ALTIBASE_REDO_LOG
데이터 공간 (메모리DB)	/ALTIBASE_MEMORY_DATA
데이터 공간 (디스크DB)	/ALTIBASE_DISK_DATA

언두 테이블스페이스

변경트랜잭션이 발생할 때 메모리DB는 복구를 위한 언두이미지를 메모리상에 관리한다. 원본데이터를 별도의 공간에 복제한 후 복제본에 변경연산을 적용하는 out-place update 방식을 이용한다. (이와 같은 방식을 MVCC기법이라고 하며 자세한 사항은 MVCC 가이드를 참조한다.)

디스크DB의 경우 원본데이터를 언두 테이블스페이스에 복제한 후 원본데이터 영역에서 변경트랜잭션을 수행한다. 이러한 방식을 in-place update 방식이라 한다.

복구가 필요한 경우 언두 테이블스페이스 저장된 원본데이터를 다시 원래 위치에 복제하여 복구하는 방법을 취한다. 이 과정에서 언두 테이블스페이스에는 디스크DB의 변경트랜잭션이 발생할 때마다 복제가 발생하기 때문에 가능한 디스크DB와 별도의 디스크공간으로 구성을 하여 물리적인 디스크병목을 회피하는 것이 바람직하다.

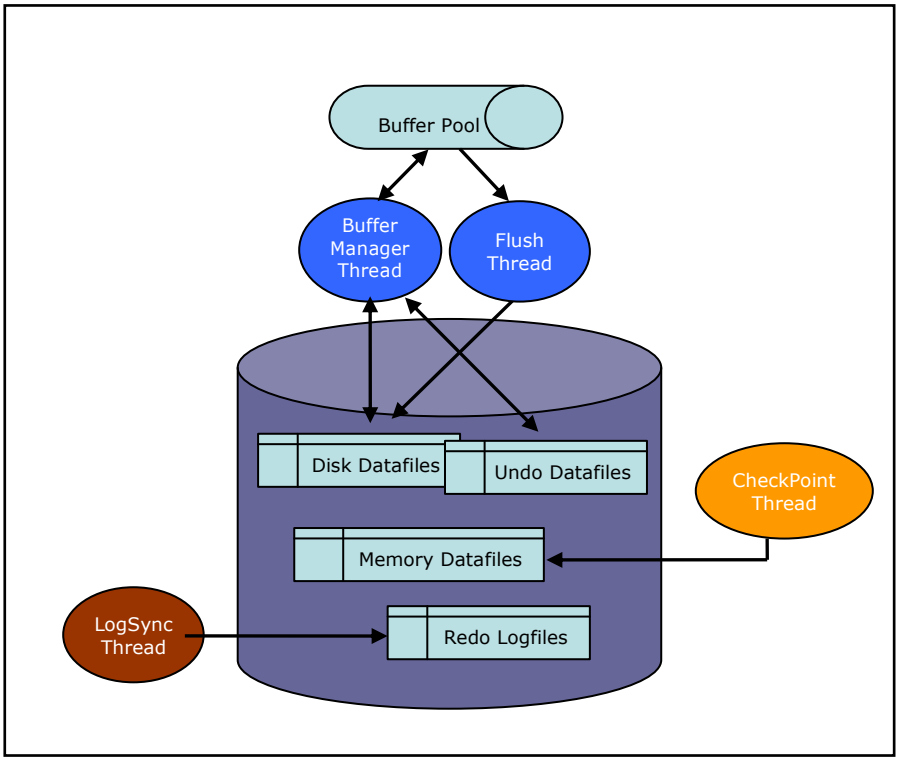
다음과 같이 예를 들어 구성할 수 있다.

분류	디스크구성
리두로그	/ALTIBASE_REDO_LOG
데이터 공간 (메모리DB)	/ALTIBASE_MEMORY_DATA
데이터 공간 (디스크DB)	/ALTIBASE_DISK_DATA
디스크DB 언두 공간	/ALTIBASE_DISK_UNDO

효과적인 데이터 파일 구성 방안

앞에서 살펴본 바와 같이 트랜잭션 처리과정에서 발생하는 리두로그의 기록과 메모리DB의 영속성을 위한 체크포인트 단계 및 디스크DB의 트랜잭션 처리를 위한 디스크I/O는 동일 디스크에 구성할 경우 디스크병목을 발생시키기 때문에 이로 인한 성능감소를 최소화하기 위해 리두로그 및 각각의 영역을 물리적으로 분리된 디스크로 구성하는 것이 바람직하다.

디스크I/O



위의 그림처럼 리두로그의 기록, 체크포인트, 버퍼관리자에 의한 디스크I/O가 하나의 디스크에 몰려 있을 경우 디스크병목은 회피할 수 없음으로 사용자는 본 문서를 참고로 하여 효율적인 디스크구성을 하는 것이 바람직하다.

구성예제 (1)

좀 더 세분화 하여 다음과 같이 구성을 권고하며 시스템의 환경 상 구성이 어려울 경우라도 리두로그와 그 외 영역은 반드시 물리적으로 분리된 디스크로 구성하도록 한다.

분류	디스크구성
ALTIBASE HOME	/ALTIBASE
리두로그	/ALTIBASE_REDO_LOG
데이터 공간 (메모리DB)	/ALTIBASE_MEMORY_DATA
데이터 공간 (디스크DB)	/ALTIBASE_DISK_DATA
데이터 공간 (디스크인덱스)	/ALTIBASE_DISK_INDEX

디스크DB 언두 공간	/ALTIBASE_DISK_UNDO

- ALTIBASE HOME은 개발 및 운영을 위한 바이너리, 헤더, 라이브러리와 같은 파일을 저장하는 공간뿐 아니라 운영 중에 발생하는 중요한 trace log를 기록하기 때문에 별도의 공간으로 분리하여 설정하는 것을 권장한다.
- 메모리 인덱스의 경우는 별도의 디스크I/O는 존재하지 않는다. 이는 ALTIBASE 구동단계에서 메모리 DB를 모두 적재한 후 메모리상에서 인덱스 재구성을 진행하기 때문이다. (메모리 인덱스 변경에 대한 별도의 로깅은 하지 않는다.)
- 디스크DB 및 디스크인덱스를 분리할 것을 권장하는 것은 각각의 구성된 데이터파일의 확장으로 인한 I/O나 DB형상 변경작업등을 수행할 때 디스크I/O의 경합을 최소화 하기 위한 목적이다.
- 백업과 관련된 디스크 고려사항은 본 문서에서 기술하지 않으며 백업/복구 가이드 문서를 참조하도록 한다.

구성예제 (2)

최소의 DISK를 갖고 있다면 다음과 같이 최소한의 분리를 권장한다.

분류	디스크구성
리두로그	/ALTIBASE_REDO_LOG
ALTIBASE HOME 및 데이터	/ALTIBASE

이와 같은 구성은 메모리DB를 위주로 사용할 경우, 혹은 변경작업이 많지 않은 디스크DB만의 서비스 환경에 한하여 디스크I/O경합을 감소시킬 수 있다.

구성예제 (3)

메모리DB는 공통코드와 같은 유형의 소량의 데이터만으로 사용하고 디스크DB에 다양한 업무와 대량의 데이터를 활용하는 형태로 서비스를 할 경우에는 다음과 같이 고려해 볼 수 있다.

분류	디스크구성
리두로그	/ALTIBASE_REDO_LOG
ALTIBASE HOME 및 메모리DB	/ALTIBASE
디스크DB-1 (복잡업무)	/ALTIBASE_DISK_COMPLEX
디스크DB-2 (단순업무)	/ALTIBASE_DISK_SIMPLE

복잡한 쿼리 수행과 관련된 테이블스페이스의 위치와 단순처리를 위주로 하는 테이블스페이스의 물리적인 데이터파일을 분리된 디스크로 구성된 볼륨에 위치시키는 방법도 디스크I/O를 분산시키는 의미에서 유효할 수 있다. 다만, 이와 같은 구성도 버퍼리플레이스가 빈번한 환경에서는 기대한 효과를 얻기 어려울 수 있다.

파일 시스템

ALTIBASE HDB 가 지원하는 파일시스템의 종류와 특성 그리고 필요한 설정에 관해서 알아 본다.

지원 파일시스템

ALTIBASE HDB는 일부 mmap 및 Direct I/O를 지원하지 않는 일부 파일 시스템은 제외하고 주요 파일시스템을 모두 지원한다.

Direct I/O 를 사용할 경우 해당 파일시스템에서 direct I/O 를 지원하도록 mount Option 변경이 필요할 수 있다. Direct I/O 를 지원하지 않는 파일시스템에서 사용할 경우에는 ALTIBASE HDB 프라퍼티를 변경해야 한다. 자세한 설정은 매뉴얼 또는 altibase.properties 파일을 참조한다.

OS	File System	특징
Solaris	UFS	Direct I/O 사용시 mount Option 변경이 필요함
	VxFS	
	ZFS	Direct I/O 을 지원하지 않아 DB 프라퍼티 변경이 필요함.
HP	HFS	
	JFS	Direct I/O 사용시 mount Option 변경이 필요함
	VxFS	Direct I/O 사용시 mount Option 변경이 필요함
AIX	JFS	
	VxFS	
Windows	NTFS	
	FAT32	
Linux	Ext2/Ext3/Ext4	

지원하지 않는 파일시스템

mmap 을 지원하지 않거나 Direct I/O 를 지원하지 않는 일부 아래의 파일시스템에서 ALTIBASE HDB 사용시 문제가 발생할 수 있다.

- Raw Storage Device

  ALTIBASE HDB 는 raw device로 구성된 파일에 접근할 수 없으므로 사용할 수 없다. DB에서 raw device 를 사용하려는 목적은 OS의 file cache 기능을 DB에서 직접 control 하기 위해서이다. 이러한 목적을 위해서라면 ALTIBASE HDB는 raw device 대신 Direct I/O를 지원하는 파일시스템을 사용할 수 있다.

- 일부 NFS(Network File System), NAS (Network attached Storage)

  mmap 을 지원하지 않는 일부 NFS/NAS 파일시스템에서는 데이터 파일 및 로그 파일을 생성하는 DB의 생성단계에서 오류가 발생할 수 있다.

Disk I/O 최적화

Database 의 성능은 Storage 의 Disk I/O 성능과 밀접한 관련이 있다. Storage 에서 Disk I/O 성능을 높일 수 있는 몇가지 방법에 대해서 설명한다.

스트라이핑(Striping)

스트라이핑(Striping) 이란 파일의 블록을 여러 장의 disk 에 분산해서 저장하는 것을 말한다. 동시 파일 입출력 속도를 크게 향상시킬 수 있어 DB의 성능을 크게 높일 수 있다.

디스크의 성능은 스트라이핑 구성 방식에 따라서 큰 차이를 보인다.

일반적으로 속도와 안정성을 위해서 RAID 0+1 (Striping + mirroring) 또는 RAID 5 방식이 주로 사용된다.

RAID 구성방식에 따라서 사용되는 디스크의 수량이 달라질 수 있으므로 데이터베이스의 크기와 가용한 디스크 수량에 따라서 Storage 전문가와 협의하여 적정한 방식을 선택한다.

OS의 File Cache 설정 변경

적절한 파일캐시(file cache) 설정으로 ALTIBASE가 사용하는 메모리 영역에 대한 스왑아웃(swap out) 발생요건을 억제, 스왑핑(swapping)으로 인한 운영체제 계층의 Disk I/O 지연시간이 ALTIBASE의 성능저하로 이어지는 현상을 최소화하기 위해 권고하는 사항이다.

파일캐시란 주기억장치와 보조기억장치간의 속도차이로 인한 병목을 해소하기 위해 운영체제차원에서 관리되는 일종의 시스템버퍼이다. 이러한 파일캐시는 운영체제 저마다의 고유한 정책(policy)에 의하여 관리되나 공통적으로 스왑정책과 직접적인 상관관계를 가진다. 스왑핑이란 자체는 주기억장치보다 더 큰 응용프로그램이나 데이터 파일을 다룰 수 있게 하는 유용함을 가지고 있으나, DBMS와 같이 장기상주 형의 응용프로그램이 운영되는 시스템에서는 스왑핑으로 인한 운영체제 계층의 Disk I/O 지연시간으로 인해 DBMS의 응답시간이 불규칙해지거나 최악의 경우에는 행(hang)과 같은 현상이 발생될 수 있기에 시스템 용도에 따라 고려가 요소가 된다.

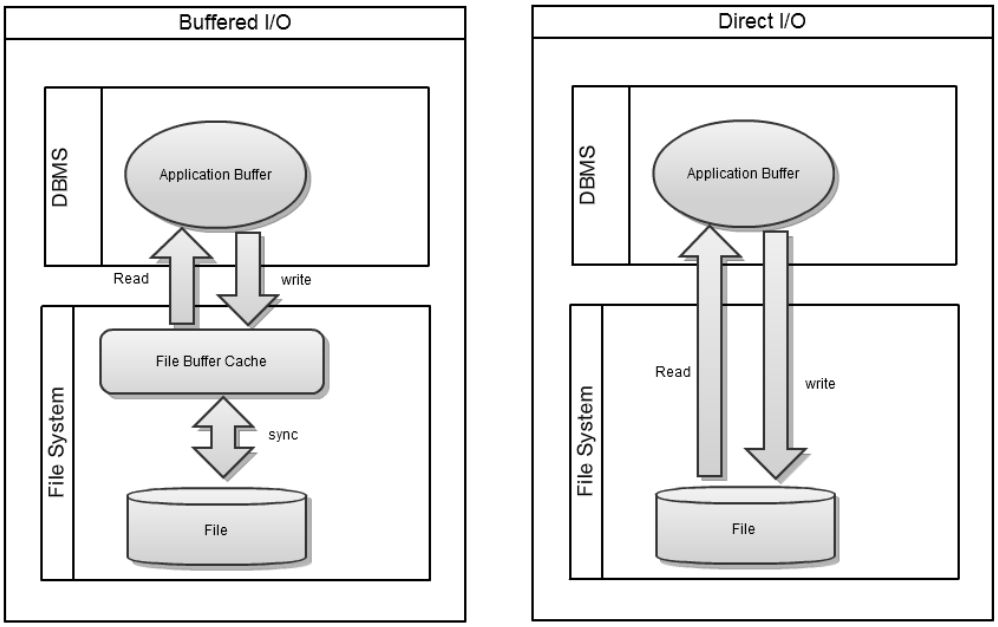
따라서, ALTIBASE의 일관적인 응답시간 보장을 위해서는 스왑이 최대한 발생하지 않도록 관련 파일캐시 및 스왑 커널들을 사전에 설정하는 것이 바람직하다

적절한 파일 캐시 설정 값을 위한 가이드는 아래의 문서를 참조하길 추천한다.

- ALTIBASE 운영을 위한 HPUX 설정 가이드
- ALTIBASE 운영을 위한 AIX 설정 가이드
- ALTIBASE 운영을 위한 Solaris 설정 가이드
- ALTIBASE 운영을 위한 Linux 설정 가이드

Direct I/O

OS의 파일시스템은 아래 그림의 Buffered I/O 형태 처럼 File Buffer Cache 라는 메모리 영역을 두고 파일 접근시 접근한 파일블럭을 캐쉬해둠으로써 느린 디스크에 대한 접근 성능을 향상시키는 구조를 갖고 있다.



그러나 DBMS와 같이 application level 에서 직접 데이타를 캐쉉하는 경우는 디스크에서 file buffer cache 로 그리고 다시 DB의 자체 buffer cache 로 데이타가 이동해야 하는 오버헤드가 발생할 수 있다.

이러한 것을 "double copying" 이라고 하며 이런 경우 CPU와 메모리를 더 소모하는 결과가 나타날 수도 있다.

이러한 경우 우측의 그림처럼 OS의 file cache를 경유하지 않는 파일 입출력 방식인 Direct I/O를 사용하면 DBMS가 사용하는 CPU와 메모리 사용량을 낮출 수 있다.

ALTIBASE 가 Direct I/O 로 데이터 파일 및 로그파일 입출력을 하기 위해서는 다음의 ALTIBASE 프라퍼티가 설정되어 있어야 한다.

- DIRECT_IO_ENABLED = 1 # 0: Buffered I/O, 1:Direct I/O
- DATABASE_IO_TYPE = 1 # 0: Buffered I/O, 1:Direct I/O
- LOG_IO_TYPE = 1 # 0: Buffered I/O, 1:Direct I/O

일부 OS 또는 파일 시스템은 파일에 대한 Direct I/O 를 지원하지 않거나 또는 Application Level 에서의 Direct I/O 를 지원하지 않을 수 있으며 Direct I/O 를 사용하기 위해서 몇가지 작업을 필요로 하기도 한다.

이러한 경우에는 아래의 예와 같이 특정 옵션을 사용하여 파일 시스템을 Mount 하여야 한다.

OS	File System	Required Action
Solaris	UFS	None
	VxFS	mount with convosync=direct

	ZFS	Direct I/O 를 지원하지 않음.
HP	HFS	None
	JFS	None
	VxFS	mount with convosync=direct
AIX	JFS	mount with use -o dio
	VxFS	mount with convosync=direct
Windows	NTFS	None
	FAT32	None
Linux(2.4 > K)	Ext2/Ext3/Ext4	None

Direct I/O 를 사용하는 것이 유리한 경우

데이터 베이스의 크기가 시스템 메모리의 크기보다 크고 Disk Buffer의 크기가 큰 경우에는 Direct I/O 를 사용하는 것이 유리할 수 있다.

DB 의 크기가 크고 대량의 변경작업이 빈번하게 발생하는 DB에서는 checkpoint 과정 중에는 대량의 Disk I/O 가 발생하는 데 이때 OS File cache 와 DB의 buffer cache에 이중으로 copy 하는 비용으로 인해서 CPU 사용량과 메모리 사용량이 급증하는 현상이 보일 수 있다.

이런 경우 Direct I/O 를 사용하는 것이 문제 해결을 위한 방법이 될 수 있다.

Buffered I/O 를 사용하는 것이 유리한 경우

보고된 바에 의하면 Buffered I/O 방식을 사용하는 것이 Direct I/O를 사용하는 것에 비해서 성능상 유리하다.

OS의 Buffered I/O는 multi block 으로 read 하여 필요한 디스크 페이지를 prefetch 하여 입출력 속도를 향상 시킬 수 있어 일반적으로 Buffered I/O 로 사용할 때 성능이 향상되는 경향이 있다.

Page size

ALTIBASE에서는 block size 대신 page size 란 용어를 사용한다.

ALTIBASE의 page 크기는 메모리 테이블은 page당 32K, 디스크 테이블은 page당 8 K의 고정적인 값을 가지고 있으며 변경할 수 없다.

일반적으로 다른 DB 벤더제품에서는 DB의 파일 입출력 효율을 위해서 OS의 block 사이즈를 고려하여 DB의 page size 크기를 DB 생성시에 조정하는 경우가 있으나 ALTIBASE는 page 크기가 고정적이며 변경할 수 없다.

또한 ALTIBASE는 DB의 page 크기에 맞추어서 OS의 block size 변경을 권장하지 도 않는다.

ALTIBASE page 크기와 OS block size 크기의 불일치로 인한 성능 저하 현상은 보고된 바 없다.

알티베이스㈜
서울특별시 구로구 구로 3 동 182-13
대륭포스트 2 차 1008 호
02-2082-1000
http://www.altibase.com

솔루션센터
02-2082-1114
http://support.altibase.com