OLTP DBMS의 성능 향상을 위한 디스크 입출력 방식에 대한 연구

강동필, 문주희
aniphile@hanaro.com, jhmoon@sejong.ac.kr
세종대학교 정보통신대학원 정보통신학과
서울시 동작구 신대방동 신대방빌딩 9층, 011-411-7001

키워드 : DBMS, OLTP, I/O, 입출력

고속의 트랜잭션 처리가 요구되는 OLTP DBMS에서 가장 중요한 성능 향상 요소는 디스크 입출력 시스템이다. 본 연구에서는 DBMS의 디스크 입출력의 구성방식을 디스크 배열, 데이터 파일 구성 방식 측면에서 성능 저하 요인을 분석하고 이에 대한 개선 방안을 제시하였다. 또한 모의 성능 시험을 통하여 제시한 방안에 대한 효과를 검증하였다

디스크 배열(Disk Array)은 다양한 구성 방식이 제안되고 있으며, 또한 각 Level 별로도 새로운 캐싱 방법 등의 다양한 개선 방안이 시도되고 있다. 그러나 실제 상용 시스템에 적용할 수 있는 방식은 RAID Level 1, 5, 10 이며, OLTP 시스템에는 Striping을 적용하는 RAID Level 5나 Level 10을 사용하는 것이 좋으며, 경제성을 고려할 때는 디스크 사용효율이 높은RAID Level 5를 사용하고, 데이터의 고가용성이 요구되는 시스템에는 RAID Level 10을 적용하는 것이 바람직하다.

DBMS 데이터 파일 구성 방식은 장치(Device)의 입출력 방식에 따라 나누어진다. 유닉스(UNIX) 운영체제에서 장치(Device)와의 입출력(I/O)은 두가지 방식으로 이루어진다. 블록 장치(Block Device)는 블록 단위로 입출력이 이루어지며, 커널(Kernel)이 제공하는 시스템 버퍼(System Buffer)를 사용한다. 문자 장치(Character Device)는 문자 단위로 입출력이 이루어지며, 커널이 제공하는 시스템 버퍼를 사용하지 않고, 입출력 장치 각각의 버퍼 또는 큐를 사용한다. 입출력 장치의 고유 버퍼와 프로세스(Process) 사이에 직접 데이터 전송이 이루어진다. 단말기의 입출력, 프린터의 출력, 디스크나 자기 테이프의 Raw 입출력이 해당된다.

DBMS의 데이터 파일을 구성하는 방식도 블록 장치(Block Device)방식으로 이루어지는 파일 시스템(Files System)과 문자 장치(Character Device)방식으로 이루어지는 Raw Device로 나누어진다.

파일 시스템은 전체적인 파일 시스템 트리 또는 디렉토리 구조로 사용자 입장에서는 정보를 조직화하고 저장하기 위해 사용되는 파일과 디렉토리의 집합, 운영체계에서의 파일시스템은 파일과 디렉토리의 위치를 정의하고 있는 테이블 구조를 포함하는 파티션을 의미한다. 데이터베이스 관리자 입장에서 보면, 파일 시스템의 다양한 기능을 사용할 수 있으므로, 백업 및 복구, 데이터 파일의 생성과 확장 등 데이터베이스를 운영하는데 있어서, 관리가 매우 편리하다. Berkeley Fast File System에 기초한 Unix File System(UFS)과 Veritas File System(VxFS) 등이 있다.

로디바이스(Raw Device)는 파일시스템(File System)을 통해 마운트되거나, 쓰여지지 않고 Character Special 디바이스 드라이버를 통해 억세스되는 Disk Partition이다. 어플리케이션 측면에서 보면, 파일 시스템이 없기 때문에 데이터가 어떻게 쓰여질지는 어플리케이션에 달려있다. 디스크에 데이터가 직접 쓰여지기 때문에 Data 손실의 가능성이 낮다. 그러나, 다루기가 매우 까다롭다.

디스크 입출력 성능 저하 요인으로는 버퍼링(Double Buffering)과 Single Writer Lock을 들 수 있다. 대부분의 DBMS는자체적인 버퍼 캐쉬를 가지고 있으므로, 파일 시스템의 버퍼 캐쉬는 불필요하다. 파일 시스템 버퍼에 캐쉬된 DBMS 데이터는 중복되고, 결국 메모리의 낭비와 버퍼를 관리하기 위한 CPU 자원을 추가적으로 사용해야 한다. Direct I/O를 사용함으로써, 데이터는 DBMS 캐쉬와 디스크 간에 직접 쓰여지

므로, CPU자원의 이용을 줄이고, 메모리의 소모를 줄여 트랜잭션 처리 능력을 향상시킬 수 있는 DBMS의 버퍼 캐쉬로 추가로 사용될 수 있다.

파일시스템과 관련하여 DBMS의 쓰기 성능을 제한하는 가장 중요한 요소는 Single writer lock 이다. 일반 UNIX 파일 시스템은 파일의 무결성을 유지하기 위해 한 시점에 하나의 프로세스가 Lock을 제어하여 쓰기 작업을 수행한다. 따라서, DBMS 차원에서 쓰기 병행성을 증가시켜도 파일시스템에서 그러한 쓰기를 순차적으로 처리한다면 아무 소용이 없게 된다.

DBMS는 자체적으로 데이터에 대해서 로우 레벨과 블록 레벨의 Lock을 제어한다. DBMS에서 제어하는 Lock과 중복하여 파일 시스템의 단일시점 쓰기 Lock에 의한 제한을 받아 실제적으로는 한 시점에 한 프로세스에 의한 쓰기 작업만이 수행된다. 따라서, 이러한 파일시스템의 Single writer lock은 불필요하며, DBMS의 입출력 성능을 저해하는 요소가 된다.

입출력 동기방식도 성능과 관련한 중요한 요소이다. 비동기식 입출력(Asynchronous I/O)은 시스템 호출(System Call)이 비동기 방식이다. 시스템 호출(System Call) 호출 후 다음 I/O 시스템 호출(System Call)을 수행하며, Return을 요구하지 않는다. Non-blocking System Level 읽기와 쓰기를 수행하여, System이 병렬 I/O 요청을 동시에 처리할 수 있도록 해준다. Solaris와 같은 UNIX 운영체제는 Raw Device에 대한Asynchronous I/O를 커널에서 지원한다. 그러나, 파일 시스템에 대하여는 Asynchronous I/O를 지원하지 않으므로, DBMS에서 Asynchronous I/O를 지원하여도 Asynchronous I/O로 동작하지 않는다. 동기식 입출력(Synchronous I/O)은 시스템 호출(System Call)이 동기 방식이다. 시스템 호출(System Call) 수행 후 Return을 받은 후 다음 I/O 시스템 호출(System Call) 수행한다. 동기 방식에 따른 DBMS의 처리 방법을 보면, 비동기식 입출력(Asynchronous I/O)일 경우 데이터 파일로의 쓰기 작업이 빈번한 경우나, 단일 쓰기 작업 시의 성능 향상이 가능하고, 사용자가 많은 경우 쓰기 경합에 대한 지연을 제거하여 성능 향상을 가져온다. Raw Device나 VxFS Quick I/O에서 Kernel Asynchronous I/O를 사용할 경우 쓰기 위주의 벤치마크에서 10~12%의 성능 향상을 가져오는 것으로 알려지고 있다.

DBMS 데이터 파일 구성 방식에 측면에서 보면, 일반적인 UFS 파일 시스템의 경우 DBMS 자체의 캐쉬와 운영체제의 버퍼가 이중으로 버퍼링되는데 따른 성능 저하와 DBMS의 쓰기 성능을 제한하는 유닉스 운영체제의 Single Writer Lock으로 인하여 OLTP DBMS에서 성능 저하가 일어난다. 성능 측면에서는 Raw Device를 사용하는 것이 바람직하나, 사용이 까다롭기 때문에 파일시스템에서도 Raw Device와 같은 성능을 낼 수 있는 새로운 파일시스템 들이 제안되고 있다. 실제 성능 시험 결과에서도 기존의 UFS 파일 시스템에 비하여 우수한 처리 능력과 안정적인 입출력 편차를 보여주고 있다. 이러한 새로운 파일시스템을 업무 특성 및 이에 따른 입출력 특성에 맞도록 적용한다면 디스크 입출력의 병목 현상을 제거하고 CPU와 같은 다른 시스템 자원을 효율적으로 사용할 수 있을 것으로 보인다.

본 연구에서는 각 입출력 방식별로 순수한 입출력 성능만을 시험하였다. 이러한 한계로 인하여 OLTP 시스템에서도 다양한 입출력 특성을 가지는 각 실제 적용 업무들에 대한 성능 개선 효과를 측정하는데는 한계가 있는 것으로 보인다. 따라서, 향후 연구 과제로 이러한 다양한 입출력 특성에 맞는 효과적인 성능 측정 방안에 대한 추가적인 연구가 필요할 것으로 보인다.