개인 컴퓨팅 환경에서 LOB 캐쉬를 지원하기 위한 SQL CLI의 확장*

이종민, 강현철
중앙대학교 컴퓨터공학과

Extending SQL CLI To Support LOB Caching in a Personal Computing Environment

JongMin Lee, Hyunchul Kang
Dept. of Computer Science and Engineering, Chung-Ang University

요 약

망 환경에서 PC 사용자들의 멀티미디어 데이타 검색이 점차 증가하고 있다 SQL CLI(Call Level Interface)는 클라이언트-서버 환경에 적합한 데이타베이스 응용 프로그래밍 인터페이스(API)로서, 현재의 표준안에서는 멀티미디어 데이타와 같은 대용량 데이타의 효율적 검색을 위한 기능을 다양하게 제공해주지 못하고 있다. 본 논문에서는 멀티미디어 데이타를 구성하는 LOB(Large OBject)의 빠른 검색을 위하여 SQL CLI 상에서 LOB의 캐쉬를 제안하고, 이를 위한 SQL CLI 함수의 확장을 제안한다 그리고 제안한 함수 중 일부를 SQL CLI를 지원하는 실제 DBMS 상에서 구현하여 LOB 캐쉬의 성능을 평가한다.

1. 서 론

근래의 컴퓨팅 환경은 클라이언트-서버 소프트웨어 구조가 주류를 이루고 있다. 데이타베이스 시스템도 기존 중앙집중 환경에서 운용되는 것보다 망 환경에서 데이타 서버로서 클리이언트의 응용 프로그램을 지원하는 기능이 중요하게 되었다. 데이타베이스 시스템이 제공하는 응용 프로그래밍 인터페이스(API)들 중 클라이언트-서버 DBMS 환경에 적합한 것으로 SQL Call Level Interface(이하, CLI)가 있다[1][2][3][4].

CLI는 응용 프로그램이 데이타베이스에 연결되어 SQL 명령어를 수행하고, 그 결과를 받아보기 위한 함수들로 구성된 인터페이스이다. 그림 1에서 보듯이 CLI를 지원하는 DBMS는 자신의 SQL 처리기의 기능을 호출하는 CLI 함수 라이브러리를 갖추고 있다. 이는 응용 프로그램의 실행 시에 링크 및 적재(load)되는 실행 시간(run time) 라이브러리이다. 따라서 응용 프로그램은 데이타베이스 접근이 필요할 때 해당 CLI 함수를 호출하고 호출된 CLI 함수는 이 라이브러리를 통하여 수행된다. 응용 프로그램은 목적(target) DBMS에 상관없이 독립적으로 개발이 가능하다. 이러한 특성으로 인해 CLI를 이용하여 작성된 응용 프로그램은 동일한 사양의 CLI를 지원하는 서로 다른 DBMS들 간에 실행 모듈 수준의 호환성(binary compatibility)을 갖는다. 이를 위해 필요한 것은 물론 CLI 사양의 표준화이다. X/Open에서는 SQL Access Group과 함께 1992년 표준 사양을 제정 발표하였고[2], ISO/ANSI에서도 1993년 표준 사양을 제정하였다[3]. Microsoft는 ODBC 사양을 발표하였다[4]

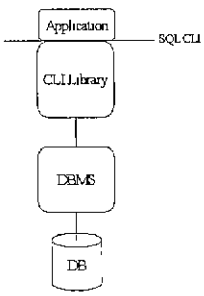

그림 1 SQL CLI의 개념

현재 개인 컴퓨팅 환경에서 멀티미디어 데이타에 대한 사용자의 요구가 증대하는 추세에 있으며, 이에 따라 PC 사용자를 위해 더욱 효율적이고 신속한 검색을 지원해 줄 필요가 있다

기존의 CLI 응용 프로그램에서는 그림, 동영상 등의 멀티미디어 데이타를 구성하는 LOB(Large OBject)을 검색하기 위하여, X/Open이 명시한 CLI 표준의 경우, GetCol이라는 CLI 함수(ODBC의 경우, GetData)를 호출한다. GetCol 함수는 클라이언트 응용 프로그램의 작업 영역에 주어진 버퍼의 크기 제약으로 인해 한

* 본 논문은 정통부의 정보통신우수시범학교 지원 사업에 의한 것임.

번의 호출로 서버로부터 LOB을 검색하지 못할 경우, 여러 번 반복 호출되어 LOB의 연속된 부분을 차례로 검색한다. 이와 같은 기존의 LOB 검색 과정을 더욱 효율적으로 수행하기 위한 방법이 요구된다.

개인 컴퓨팅 환경에서 사용자는 자신의 편의에 맞도록 PC의 환경을 구성하고 사용할 뿐 아니라 검색하는 LOB은 관심영역에 따라 전체 데이타베이스 중 비교적 국한된 부분에 속할 수 있다. 즉, 동일한 LOB의 검색을 반복하는 경우가 많다. 또한 이미지, 비디오 동과 같은 LOB은 비교적 내용이 자주 변하지 않는 특성을 가지고 있다. 이와 같은 점을 고려할 때 자주 사용하는 LOB을 캐쉬함으로써 더 효율적이고 신속한 검색을 지원할 수 있다.

본 논문에서는 CLI 상에서 LOB의 캐쉬를 제안하고, 이를 지원하기 위한 CLI 사양의 확장 방안을 제안한다. 그리고 제안한 확장 내용의 일부를 CLI를 지원하는 실제 DBMS 상에서 구현하고 성능을 평가한 결과를 기술한다.

본 논문의 구성은 다음과 같다. 2절에서는 CLI에서 LOB 캐쉬를 지원하기 위해 해결해야 할 기술적 문제점을 기술한다. 3절에서는 기존의 표준 CLI 사양에 LOB 캐쉬를 위한 새로운 함수의 추가를 제안한다. 4절에서는 제안된 CLI 확장의 일부 구현 및 LOB 캐쉬/검색 성능의 평가 결과를 기술하며, 5절에서 결론과 향후 연구 방향에 대해 기술한다

2. SQL CLI에서 LOB 캐쉬의 지원

SQL CLI에서 LOB 캐쉬를 지원하기 위해서는 다음과 같은 사항을 고려해야 한다.

- 서버 측에 저장된 LOB과 캐쉬된 LOB 간의 일관성 유지
- LOB 캐쉬 교체 정책
- LOB 캐쉬 정보의 효율적 관리
- LOB 캐쉬를 지원하기 위한 새로운 CLI 함수

첫째, 서버에 저장되어 있는 LOB은 갱신될 수 있기 때문에 클라이언트 측에 캐쉬된 LOB을 갱신 여부의 확인 없이 사용한다면, 정확하지 않은 내용의 LOB을 접근하는 결과를 가져올 수 있다. 캐쉬 일관성 유지를 위하여 LOB의 캐쉬 및 검색 과정은 다음과 같이 수행되어야 한다. 우선 서버로부터 응용 프로그램이 LOB을 캐쉬할 때는, 서버 내에서 LOB을 식별하는 식별자와 서버에서 LOB이 최종 갱신된 시점의 타임스탬프를 함께 저장한다. 이후 LOB을 검색할 때는, 먼저 응용 프로그램이 요구한 LOB에 대한 식별자와 타임스탬프를 서버로부터 검색한다. 만약 서버와 캐쉬의 타임스탬프가 일치하면 캐쉬 내의 LOB은 유효하므로 캐쉬로부터 검색한다. 그렇지 않으면 기존의 LOB 검색과 마찬가지로 서버로부터 검색한 후 캐쉬한다.

둘째, LOB 캐쉬 영역이 가득차 있을 때, 새로운 LOB을 캐쉬하는 경우 일부 LOB을 교체해야 한다. 기존 DBMS에서 페이지 단위의 버퍼 교체에 비해 LOB은 크기가 다양하므로 LOB 교체 알고리즘은 이러한 점을 고려해야 한다. 예를 들어 하나의 큰 LOB을 캐쉬하기 위하여 작은 크기의 LOB을 여러 개 교체 할 수 있다

셋째, LOB 캐쉬 정보란 LOB 캐쉬 영역의 최대 용량 및 현재의 가용 용량; 캐쉬 영역이 저장될 하드 디스크의 디렉토리 경로; 캐쉬된 LOB에 대한 인덱스 등의 캐쉬 검색을 위해 필요한 정보; 캐쉬된 LOB 별로 LOB의 식별자, 타임스탬프, 크기와 같은 정보; 크기별, 갱신 시간별, 마지막 참조된 시간별 등 LOB에 대한 통계 정보; 캐쉬에 두고자 하는 LOB 선호도와 같은 선택 정보 등을 의미한다 캐쉬된 LOB에 대한 인덱스는 각 LOB에 대해 하드 디스크에 저장되는 회일 이름, 마지막으로 참조된 시간 등으로 구성된다 이러한 정보가 신속한 검색 및 갱신을 위하여 효과적으로 유지될 필요가 있다

넷째, LOB 캐쉬를 지원하기 위하여 새로운 CLI 함수를 지원한다면 응용 프로그램 수준에서 캐쉬에 대한 정보를 알 수 있으며 적절하게 캐쉬를 관리할 수 있다 기존 응용 프로그램이 LOB 캐쉬 기능을 이용하려면 새롭게 추가된 함수를 이용하여 다시 작성되어야 한다

3. LOB 캐쉬를 위한 새로운 CLI 함수

본 절에서는 2절에서 언급한 LOB 캐쉬 지원을 위한 고려사항 중 CLI에서 캐쉬를 지원하기 위한 새로운 함수를 제안한다 LOB 캐쉬 지원을 위하여 CLI에 추가되어야 할 함수는 기능별로 크게 다음과 같이 분류할 수 있다.

- LOB 캐쉬 핸들(handle)의 할당과 해제
- LOB 캐쉬 영역의 추가 할당과 해제
- LOB 검색과 캐쉬로부터 LOB 삭제
- 캐쉬 정보의 변경과 검색

첫째, LOB 캐쉬 핸들이란 LOB 캐쉬 정보를 저장하는 자료구조를 가리키는 포인터이다. LOB 캐쉬를 위하여 핸들을 할당하고 해제하는 함수가 필요하다. 할당 함수는 LOB 캐쉬 정보의 자료구조를 위한 메모리를 할당하고 초기화한다. 그리고 핸들에 이 자료구조를 가리키는 값을 할당한다. 해제 함수는 이 자료구조를 메모리로부터 해제하고 핸들의 값을 널 값으로 한다.

둘째, 효율적인 LOB 캐쉬를 위하여 LOB 캐쉬 영역을 여러 개 유지할 수 있다 예를 들어 자주 참조하는 LOB은 용량을 크게 할당한 캐쉬 영역에, 자주 참조하지 않는 LOB은 용량을 적게 할당한 다른 캐쉬 영역에 저장함으로써 캐쉬 교체의 빈도를 조절할 수 있다. 캐쉬 영역의 할당 함수는 하드 디스크에 하나의 디렉토리를 설정하고 핸들의 값과 캐쉬 정보를 그에 맞게 갱신한다. LOB 캐쉬 영역 해제 함수는 해당 디렉토리를 지우고 핸들을 삭제한다.

셋째, LOB을 검색하는 기능과 캐쉬로부터 LOB을 삭제하는 기능이 필요하다. LOB 검색 함수는 기존의 CLI GetCol 함수에 캐쉬 유효성 여부 검사, 서버로부터가 아닌 캐쉬로부터 검색하는 기능을 지원해야 한다. 응용 프로그램 내의 지정된 부분에서의 LOB 검색은 모두 캐쉬하도록 설정 및 해제하는 함수도 필요하다. 캐쉬 영역은 한정되어 있기 때문에 캐쉬 영역의 효율적인 사용을 위하여 캐쉬된 LOB의 삭제가 기능해야 하는데, 예를 들어 특정 크기 이상의 LOB, 오래 전에 참조된 LOB의 삭제 등을 수행하는 함수가 필요하다. 캐쉬된 LOB을 찾는 함수도 필요하다.

넷째, LOB 캐쉬를 더욱 효율적으로 사용하기 위하여 LOB 캐쉬에 대한 전체 정보 및 각 LOB별 통계 정보, 선호도 정보를 검색 또는 갱신할 수 있는 함수가 필요하다. 예를 들어 통계 정보를 검색하여 오랫동안 참조되지 않은, 크기가 지정된 값보다 큰 LOB은 삭제할 수 있다. 또한 LOB 캐쉬에서 중요한 사항은 바로 캐쉬 적중률(hit

ratio)을 높이는 것인데, 이를 위하여 LOB 선호도에 관한 정보, 즉 선호하는 LOB과 선호하지 않는 LOB을 적절히 명시함으로써 캐쉬 적중률을 높일 수 있다. 이상의 함수들을 정리하면 표 1과 같다.

CLI 함수	기능
LOB 캐쉬 핸들의 할당과 해제	
AllocLOBCache()	캐쉬 핸들 할당
FreeLOBCache()	캐쉬 핸들 해제
LOB 캐쉬 영역의 할당과 해제	
AllocLOBCacheArea()	새로운 캐쉬 영역을 할당
FreeLOBCacheArea()	캐쉬 영역을 제거
LOB 검색과 캐쉬된 LOB의 삭제	
GetnCacheLOB()	LOB을 검색 및 캐쉬
PurgeLOB()	캐쉬에서 LOB을 삭제
BeginLOBCache()	이 함수 이후의 검색은 캐쉬
EndLOBCache()	이 함수 이후의 검색은 캐쉬하지 않음
FindLOB()	캐쉬된 LOB을 찾음
캐쉬 정보의 변경과 검색	
GetLOBCacheInfo()	캐쉬 정보를 반환
ModifyLOBCacheInfo()	캐쉬 정보를 변경
SetLOBPreference()	선호하는 LOB을 명시

표 1 LOB 캐쉬 지원을 위한 SQL CLI 함수(주)

4. 구현 및 성능평가

본 논문에서는 전 절에서 제안한 LOB 캐쉬를 위한 CLI 함수 중 가장 기본적이라 할 수 있는 GetnCacheLOB 함수를 CLI를 지원하는 바다-II DBMS[5] 상에서 구현하였다. 확장된 CLI 라이브러리는 클라이언트 측에서는 Windows/NT PC에서 MS Visual C++ 6.0으로 실행 시간 라이브러리로 구현하였고, 서버 측에서는 동적 SQL을 사용하는 ESQL/C 프로그램[6]으로 구현하였다. 바다-II DBMS 서버는 Solaris 2.4 환경에서 구동되고, PC 클라이언트와 Ethernet으로 연결되어 있다.

그림 2는 X/Open의 CLI 사양에서 LOB을 검색하는 함수인 GetCol과 이 함수에 캐쉬 기능을 부기하여 확장한 GetnCacheLOB의 성능을 비교한 것이다. 1M, 5M, 10M, 20M, 30M, 40M, 50M 바이트 용량의 LOB에 대한 검색 시간을 각각 측정한 것이며 검색시간의 단위는 초이다. GetnCacheLOB을 쓰는 경우, 캐쉬에 검색하려 하는 LOB이 하나도 없는 경우(캐쉬 적중률=0%)의 검색하려는 LOB이 모두 있는 경우(캐쉬 적중률=100%)의 검색 시간을 측정하였다. 캐쉬 적중률이 0%일 경우에는 LOB 캐쉬 지원 시 최악의 성능을 내는 경우로, 유효성 검사를 위하여 식별자와 타임스탬프를 검색하는 시간과 검색된 LOB을 캐쉬에 저장하기 위한 추가 비용으로 인하여 기존의 GetCol보다 좋지 못한 성능을 나타낸다. 반면에 캐쉬 적중률이 100%인 경우에는 GetCol 비해 상당한 성능 향상이 있음을 알 수 있다. 예를 들어, 50M 바이트 LOB의 경우 캐쉬 적중이 되지 않은 경우에는 GetCol에 비해 약 7%의 성능 저하를 가져왔다. 반면에 캐쉬 적중인 경우에는 GetCol에 비해 약 88%의 성능 향상을 가져왔다. 이상의 실험으로부터 LOB을 캐쉬함으로써 얻을 수 있는 이익이 손해보다 훨씬 크다는 것을 알 수 있다.

(주) 각 함수의 명칭은 모두 SQL로 시작.

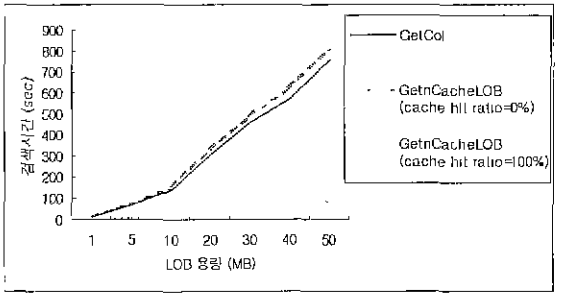

그림 2 GetnCacheLOB의 성능 평가

5. 결론

PC와 같은 개인 컴퓨팅 환경에서 멀티미디어 데이터를 구성하는 LOB의 효율적 검색 기법이 요청되고 있다. 본 논문에서는 SQL CLI에서 LOB의 캐쉬를 제안하고, 이를 위한 새로운 CLI 함수를 제안하였다. 그리고 CLI를 지원하는 실제 DBMS 상에서 그 중 일부 함수를 구현하고 LOB 캐쉬 성능을 평가하였다.

향후 연구과제는 본 논문 3절에서 제안한 CLI 함수를 모두 구현하는 것이다. 이를 위하여 2절에서 기술한 LOB 캐쉬 교체 알고리즘, LOB 캐쉬 영역 및 LOB 캐쉬 정보의 효율적인 관리를 위한 자료구조가 연구되어야 한다.

참고문헌

[1] M. Venkatrao and M. Pizzo, "SQL/CLI - A New Binding Style For SQL," SIGMOD Record, Vol. 24, No. 4, Dec 1995, pp. 72-77.

[2] X/Open Company Ltd., "Data Management: SQL Call Level Interface(CLI)," X/Open Snapshot, Sep. 1992

[3] J. Melton (Ed.), "ISO-ANSI (Working Draft) SQL Call-Level-Interface(CLI)," ISO DBL MUN-005/ANSI X3H2-93-360, Aug. 1993.

[4] Microsoft Corp., "Microsoft ODBC 3.0 Software Development Kit and Programmer's Reference," Microsoft Press, 1997.

[5] 강은지, 이재성, 김한울, 김지현, 강현철, 전성택, "클라이언트-서버 DBMS를 위한 멀티미디어 확장 SQL CLI의 개발," 정보과학회논문지(C), 3권, 4호, 1997. 8, pp. 343-352

[6] 이재성, 장한울, 강현철, 김명준, "망 환경에서 멀티미디어 데이타 처리를 위한 동적 SQL의 확장," 한국 정보과학회 가을 학술발표논문집, 22권, 2호, 1995. 10, pp. 173-176.