SSD 환경의 서브쿼리상에서 조인연산의 연구

이규형, 강운학, 이상원
성균관대학교 정보통신공학부 컴퓨터공학과
e-mail : leekh8412@naver.com

A Study about Join on Sub-Query on SSD Environment

Kyu-Hyoung Lee, Kang-Woon Hak, Sang-Won Lee
Department of Information and Communication Engineering, Sungkyunkwan University

요 약

SSD 가 기존의 데이터베이스 저장매체인 하드디스크를 대체할 것이라는 전망이 나오는 가운데 SSD 환경에서 직접 DBMS 를 가동하여 몇 가지 상황을 제시하고 원하는 결과를 얻을 수 있는 쿼리를 작성하여 성능측정을 한다. 동일한 결과를 얻을 수 있는 쿼리문 들을 비교하여 어떤 방식이 SSD 환경에서 가장 최적의 성능을 발휘하는지를 확인하여 미래의 SSD 데이터베이스 환경에 대비한다.

1. 서론

데이터베이스 관리 시스템(Database Management System : DBMS)는 컴퓨터를 이용하여 데이터를 관리하는 범용 소프트웨어로 여러 개의 프로그램으로 구성되어 있다. 이러한 데이터베이스는 기업, 학교, 개인 등 많은 곳에서 쓰이고 있고 그에 따라 데이터베이스 관리 시스템도 다수 개발되어 왔다.

지금까지 하드디스크 기반의 DBMS 및 쿼리 연구는 많이 이루어져 있지만 차후를 대비한다면 최근 각광받고 있는 SSD 기반의 DBMS 및 쿼리연구도 많이 이루어져야 한다

조인 연산은 2 개의 테이블에 연계된 정보를 산출하는 중요한 연산이며 질의 처리에 있어 데이터 집중적이고 수행시간이 오래 걸리기 때문에 지금까지 효율적인 조인 연산에 대한 구현과 성능 평가에 대해 많은 연구가 진행되었다.

이 연구의 목적은 관계형 데이터베이스를 바탕으로 개발된 다양한 조인 알고리즘들을 SSD 환경에서 비교 분석하여 최적의 성능을 갖는 조인 연산을 찾는 것이 목적이며 그러한 것이 데이터베이스관리자 혹은 개발자들에게 좋은 알고리즘을 선택하도록 도움을 줄 것이다.

2. 평가 데이터베이스 설계

본 연구에서 사용하게 될 테이블은 Microsoft MS-SQL 쿼리분석기를 이용하여 작성하였다. 본 연구에서 사용하는 DB 는 학교 DB 처럼 구현하였으며, 테이블명은 표 1 과 같다.

* 본 연구는 지식경제부 및 정보통신산업진흥원의 대학 IT 연구센터 지원사업(NIPA-2009-(C1090-0902-0046))과 2008 년 정부(교육과학기술부)의 재원으로 한국학술진흥재단 (KRF-2008-0641)의 지원을 받아 수행하였음.

<표 1> 평가 데이터베이스 테이블들

테이블명	설명
Stud	학생 테이블
Dept	학과 테이블
Professor	교수 테이블
Grade	성적 테이블
Course	과목 테이블
Course_Time	과목의 정보
Club	동아리 테이블
Stud_Club	학생과 동아리테이블 연결
Room	건물 테이블
Zipcode	우편번호 테이블
College	단과대학 테이블

생성 테이블 종류는 학생,교수,학과,성적,과목_시간표, 과목,건물,동아리,학생_동아리,우편번호 등의 9 개이다.

3. 테이블 생성

테이블 컬럼에 대한 정의는 컬럼명과 데이터 형식의 제약으로 이루어진다. 제약에는 null 여부, 기본값, 체크조건, 기본 키 등이 있다. 표 2 는 한 예로 학생 테이블을 생성하는 데이터 정의문이다.

<표 2> 학생 테이블 생성 쿼리문

```
Create  table  stud  /*테이블 명*/
(
Sno        char(12) not null primary key
,class     char(2) not null   /*학년*/
,sname     char(10) not null /*학생이름*/
,nid       char(15) null     /*주민번호*/
,sex       char(2) not null /*성별*/
,tel       char(10) null     /*전화번호*/
,hp        char(15) null     /*휴대폰번호*/
,zipcode   char(7) null   /*우편번호*/
,address   char(20) null   /*상세주소*/
,email     char(20) null   /*이메일주소*/
```

```
,dno      char(5) noy null
)
```

성능평가를 위해서는 다량의 데이터가 필요한데 일일히 수작업으로 진행하기에는 비 효율적이므로 난수를 이용한 임의의 데이터 생성기를 작성하여 진행하였다.

4. 테스트 환경

Cpu : AMD Winsor 5200ee++
RAM : 2G
SSD : 하나마이크론 64GB
OS : Microsoft Windows XP professional
DBMS : Micosoft MS-SQL 2005

5. 평가 쿼리문 작성

몇 가지 상황을 제시하고 원하는 결과를 얻을 수 있는 쿼리를 작성하여 성능 측정을 한다. 쿼리는 서브쿼리와 서브쿼리 간의 조인을 할 수 있어야 하고, 하나의 질의에 서브 쿼리만을 사용한 경우와 서브쿼리상에서 조인을 사용한 경우를 비교 분석한다.

<질의 1> 2009 학년도 2 학기 수업을 하고, 주당 10 시간 이상 수업을 하는 컴퓨터공학과 소속 교수의 수업 중에서 가장 많은 학생들이 수강한 수업의 교수명, 과목명, 수업인원, 강의실을 검색하라

- Join 연산만 사용

```
select p.pno
from course_time c, professor p, dept d
where [year]=2009 and term=2 and dname='컴퓨터공학
                                        과'
and c.pno=p.pno and p.dno=d.dno
group by p.pno
having count(c.cno)>10
```

- 서브쿼리만 사용

```
Select pno
from course_time
where [year]=2009 and term=2 and
pno in (select pno from professor
        where dno in (select dno from dept
                      where name='컴퓨터공학과))
```

<표 3> SSD 에서의 질의 1 실행계획 비교

분류	비용	Cpu Time	I/O Time
서브쿼리	8%	0.000024ms	0.0175ms
Join 연산	5%	0.000012ms	0.00075ms

<표 4> HDD 에서의 질의 1 실행계획 비교

분류	비용	Cpu Time	I/O Time
서브쿼리	8%	0.000024ms	0.325ms
Join 연산	5%	0.000012ms	0.0187ms

<질의 2>현재 '컴퓨터공학과' 3 학년 학생 중에서 전공과목을 총 10 학점 이상 수강한 학생들의 평점을 검색하라.

- 서브쿼리+Join 연산

```
select sno,gpa form stud,dept where dept.dname='컴퓨터공
      학과' and stud.dno=dept.dno and stud.sno =
      (select sno from grade a, course b where b.[section]=
            '전공' and a.cno=b.cno group by a.sno
            having count(b.[section])>10)
```

- 서브쿼리만 사용

```
select sno,gpa from stud where sno in (select sno from
    grade where cno in (select cno from course where
      [section]='전공) group by sno having
        count(sno)>10)
```

<표 5>SSD 에서의 질의 2 실행계획 비교

분류	비용	Cpu Time	I/O Time
서브쿼리	10%	0.000034ms	0.0098ms
서브쿼리+Join	8%	0.000028ms	0.0086ms

<표 6> HDD 에서의 질의 2 실행계획 비교

분류	비용	Cpu Time	I/O Time
서브쿼리	10%	0.000034ms	0.0196ms
서브쿼리+Join	8%	0.000028ms	0.0186ms

6. 결론

사용자는 응용프로그램이 신속히 응답하고 보고서가 분석 데이터를 즉시 반환하기를 원한다. 이러한 필요성 때문에 데이터베이스 저장 매체가 하드디스크에서 SSD 로 넘어갈 것이라는 전망도 있다.

일반적인 경우 서브쿼리보다는 Join 쿼리의 성능이 더 좋다. SSD 환경에서도 인덱스가 없고 데이터 사이즈도 적은 경우면 서브쿼리의 수행이 빠를 수 있겠지만 일반적으로 조인이 최선의 전략이라고 할 수 있다. 저장 데이터 용량이 큰 경우에는 서브 쿼리들은 Join 쿼리로 유도하여 해결하는 것이 상대적으로 좋은 전략이라 할 수 있겠다.

SSD 환경에서도 쿼리의 선택에는 큰 차이는 없지만 하드디스크와 비교했을 때 Join 의 선택은 성능 향상 폭이 더 크다. 이 연구의 결과가 각각의 케이스에 대해 동일하다고 단정 지을 수는 없기 때문에 사용자들은 항상 예상 실행계획을 살펴보고 그것을 참고하여 문제를 해결해야 할 것이다.

서브 쿼리의 성능을 높이기 위해 많은 연구가 이루어졌다. 따라서 데이터베이스 사용자들은 서브쿼리의 편리함을 인식하고 또한 서브쿼리의 약점을 잘 파악하면서 자신에게 가장 효과적인 방법을 이용하는 것이 올바른 서브쿼리 접근 전략일 것이다.

참고문헌

[1] 이원조. "관계형 데이터베이스에서 셀프조인 쿼리를 위한 성능평가".2001

[2] 정희진,이상호. "사용자 관점에서의 조인 연산 평가 방법론".2004

[3] 박호진. "플래시 SSD 를 이용한 데이터베이스 로깅 성능에 관한 연구.2008

[4] 오주형. "플래시 SSD 를 이용한 다중 버전 동시성 제어기법 성능평가",2008

[5] Sang-Won Lee and Bongki Moon. "Design of flash-basedbms: An in-page logging approach",2007

[6] Tae-Sun Chung,Dong-JooPark,Dong-Ho Lee,Sang-Won Lee, Ha-Joo Song "System Software for Flash Memory:A Survey", 2006

[7] 권순용. "지금 우리에게 필요한건 SQL 최적화다". 월간마이크로소프트웨어 2009 년 3 월호,p240~241