관계형 데이타베이스 시스템을 위한 성능향상 보조도구 설계

안 기 덕, 오 정 석, 이 상 호
숭실대학교 컴퓨터학부

Design of a Tuning Aid for Relational Database Systems

Ki Duk Ahn, Jeong Seok Oh, Sang Ho Lee
School of Computing, Soong Sil University

요 약

정보화 사회로 발전하면서 크고, 다양하며, 복잡한 데이타들이 생겨나고 저장, 유지되어 데이터베이스도 대형화되면서, 대용량 데이터베이스의 성능 문제는 매우 중요한 논점이 되었다. 본 논문에서는 관계형 데이타베이스 시스템의 새로운 성능향상 보조도구를 소개한다. 본 도구의 목적은 사용자나 시스템 관리자가 특정 데이타베이스 시스템에서 성능에 영향을 주는 요소들을 효과적으로 파악하여 데이타베이스 시스템이 높은 수준의 성능을 유지할 수 있도록 도와준다. 설계 원리, 시험 데이터베이스, 그리고 튜닝 질의어들이 보여진다. 9개의 카테고리 안의 총 36개의 시험 질의어가 제안되었다.

1. 서 론

1980년대에 관계형 시스템이 발전하여 사용되면서 오늘날 사용되는 상용 데이타베이스 시스템의 약 80%는 관계형이다. 정보화 사회로 발전하면서 크고, 다양하며, 복잡한 데이터들이 생겨나고 저장, 유지되면서 데이터베이스도 대형화되었다. 데이타베이스가 대형화 될수록 시스템의 성능문제는 매우 중요한 논점이 되었다. 그러므로 데이타베이스 시스템을 효율적으로 사용하기 위해서는 적절한 튜닝 가이드가 필요하다. 데이타베이스 튜닝은 지식 집약적 분야이기 때문에 [1], 데이타베이스 튜너는 성능 향상을 위해 시스템의 버퍼 풀 크기, 자료구조 등 여러 가지 요소에 기초하여 튜닝 결정을 내려야 한다. 본 보조도구는 관계형 시스템을 위하여 설계되었고, 데이타베이스 성능평가와 다르다. DBMS 성능평가는 주로 성능을 의미하는 데이타베이스의 용량을 평가하는 측정치들의 집합의 지침서이며, 데이타베이스 성능평가의 목적은 데이타베이스 시스템간의 성능을 정밀하게 비교하는 것이다. 따라서 좋은 데이타베이스 성능평가는 데이타베이스 관리자에게 성능기반상의 구매 결정을 가능하게 한다 [4]. 본 보조도구는 성능평가가 아니므로 다른 시스템과의 성능 비교를 하지 않고 한 시스템 내의 여러 가지 성능적 특징들을 파악할 수 있는 시험들을 한다. 본 연구의 목적은 새로운 관계형 데이타베이스 시스템을 위한 성능향상 보조도구를 설계하여 시용자에게는 시스템을 보다 효율적으로 사용할 수 있도록 도와주는 튜닝 가이드를 제공하고, 튜너에게는 시스템의 특징을 효과적으로 파악하게 하여 시스템이 항상 적절한 성능을 낼 수 있도록 튜닝 되게 한다.

2. 설계 원리

현재 관계형 데이타베이스 시스템은 용량, 성능, 그리고 가격에 대해 많은 변화가 있다. 성능향상 보조도구는 데이타베이스 시스템의 성능상의 특징을 파악할 수 있고, 성능상의 병목현상을 추적 가능하게 하는 시험들의 집합으로 제공되어야 한다 [5]. 데이타베이스 관리자의 가장 중요한 작업중의 하나는 시스템의 작업부하에 맞는 물리적인 데이타베이스 설계를 하는 것이다. 물리적인 데이타베이스 설계의 가장 중요한 요소는 인덱스를 선택하는 것이다. 왜냐하면 적절한 인덱스의 선택은 시스템의 성능에 매우 중요하기 때문이다 [3]. 성능향상 보조도구는 데이타베이스 관리자에게는 시스템의 성능상의 여러 가지 특징에 대한 정보들을 제공하여 시스템 튜닝 가이드로서의 역할을 한다. 테이블에 대한 인덱스는 테이블의 하나 또는 그 이상의 레코드들에 대해 빨리 접근할 수 있게 하는 자료구조이다. 적합한 인덱스의 튜닝은 좋은 성능을 위한 필수요소이다. 적절하지 않은 인덱스의 선택은 다음과 같은 문제를 야기한다 [1]

- 인덱스가 유지되나 한번도 사용되지 않는 경우
- 하나의 레코드가 검색되기 위해서 테이블 전체가 스캔되는 경우
- 끝나지 않는 조인 연산
- 동시성 제어 병목 현상

SQL 문장에 대한 튜닝은 좋은 성능을 얻기 위한 중요한 부분중의 하나이다 [2]. 본 보조도구는 SQL 문장을 통한 시험을 통해 효과적으로 인덱스를 선택하고, 유지하고, 사용하는 길잡이가 된다. 시스템 성능상의 특징은 총 9개의 카테고리를 두어 효과적으로 확인할 수 있게 한다. 이 보조도구의 목표는 데이타베

이스 관리자나 사용자가 데이타베이스를 좀 더 효율적으로 사용할 수 있게 도움을 주는 것에 있다

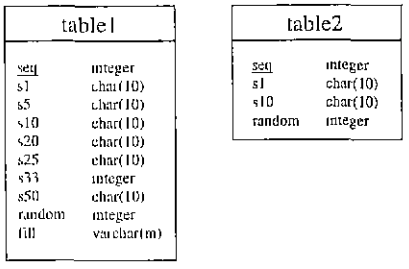
본 도구는 다음과 같은 원리 하에 설계되었다.

- 효과성 (effectiveness) 보조도구는 관계형 시스템이 적절하게 튜닝 되고 사용될 수 있도록 사용자 그리고/또는 데이타베이스 관리자에게 성능상의 특징들을 효과적으로 인식시켜 줄 수 있어야 한다.
- 이식성 (portability). 보조도구는 다른 플랫폼 하의 여러 가지 DBMS에 쉽게 이식될 수 있어야 한다.
- 안내성 (guidance). 대부분의 성능상의 문제는 독자적인 교정 해결책이 있다 보조도구는 대부분의 성능상의 문제유형을 설명하고, 우리가 발생할 수 있는 성능상의 여러 가지 문제점들을 교정할 수 있는 조치를 제안할 수 있어야 한다
- 확장성 (scalability): 보조도구는 확장성이 뛰어나게 설계되어 설계의 수정 없이 다양한 크기의 데이타베이스에 적용 가능해야 한다 시험 데이타베이스의 크기는 데이타베이스 시스템 성능을 좌우한다. 본 보조도구는 어떠한 크기의 데이타베이스에도 사용 가능하여 데이타베이스 크기는 고정되어 있지 않고 완전히 사용자 정의가 가능하다.

36개 각각의 튜닝 질의에 대해 총 5번의 응답시간을 측정하며, 가장 빠른 응답시간과 가장 느린 응답시간을 제외한 3번의 응답시간의 평균으로 계산하고, 각 관련 질의 결과의 비율로 나타낸다. 크기가 매우 큰 의미 없는 파일을 각 질의의 실행 사이에 매인 메모리로 읽어서 콜드 시작환경을 구현한다.

3. 시험 데이타베이스 스키마와 인스턴스

시험 데이타베이스는 두 개의 테이블로 구성되어 있다. 테이블 *table1*은 3개의 정수형 속성과 6개의 10바이트 크기의 문자형 속성과 하나의 가변문자형 속성으로 구성되어 있다 데이블 *table2*는 2개의 정수형 속성과 두 개의 10바이트 크기의 문자형 속성으로 구성되어 있다 [그림 1]은 시험 데이타베이스의 스키마와 칼럼의 자료형을 그림으로 표현한 것이다. 테이블의 밑줄 친 칼럼은 테이블의 키 속성을 나타낸 것이다



[그림 1] 시험 데이타베이스 스키마

사용자 정의가 가능한 시험 데이타베이스의 크기는 다음과 같다. 데이타베이스의 튜플의 개수를 N이라 하고, 각각의 튜플의 크기를 M이라 정의한다 사용자는 각각의 응용분야에 알맞은 데이타베이스를 선택하기 위해 M과 N을 자유롭게 선택할 수 있지만 항상 일아야 할 제한사항이 있다. N은 1000의 배수여야 하고, M은 72보다 커야 한다. 이것을 수식으로 나타내면 아래와 같다

$N = 1000 * n$ (n은 양의 정수)

$M = 72 + m$ (m은 테이블 *table1*의 *fill*속성의 크기)

표 1은 가능한 속성 값과 결과적으로 몇 개의 고유한 속성값이 만들어지는지 보여준다

[표 1] 테스트 데이타베이스의 인스턴스 분포

속성	값	유일한 값의 개수	유일한 값 당 튜플의 개수
seq	1 2,3, ,N	N	1
s1	'aaaaaaa001' . ,'aaaaaaa100'	100	N/100
s5	'bbbbbbb001',...,'bbbbbbb020'	20	N/20
s10	'cccccccc001' ,'cccccccc010'	10	N/10
s20	'ddddddd001'. ,'ddddddd005'	5	N/5
s25	'eeeeeee001',. .,'eeeeeee004'	4	N/4
s33	10000, 20000, 30000	3	N/3
s50	'fffffff001',...,'fffffff002'	2	N/2
random	임의의 정수값	알수 없음	알수 없음
fill	임의의 문자열	알수 없음	알수 없음

본 스키마의 인스턴스의 분포는 다음과 같다. 데이블 *table1*의 *seq, s1, s5, s10, s20, s25, s33, s50* 속성은 각각 1%, 5%, 10%, 20%, 25%, 33%, 50%의 고유한 분포를 갖는다. [표 2]는 시험 데이타베이스의 데이터 분포를 보여준다.

[표 2] 시험 데이타베이스 분포

속성	분포
seq	순차적 값
s1	1% 고유 값 분포
s5	5% 고유 값 분포
s10	10% 고유 값 분포
s20	20% 고유 값 분포
s25	25% 고유 값 분포
s33	33% 고유 값 분포
s50	50% 고유 값 분포
random	임의의 값 분포
fill	임의의 값 분포

테이블 *table1*의 *random* 속성은 임의의 값들로 생성되며 데이타들이 데이타베이스 시스템으로 적재되기 전에 데이블 내의 튜플들을 섞는 데 사용된다 또한 테이블 *table1*의 *fill* 속성은 사용자에 의해 크기가 정의되어 테이블 *table1*의 각각의 튜플의 길이를 결정할 때 사용한다.

4. 튜닝 질의어

이 섹션은 카테고리별로 튜닝 질의어들을 보여 준다. 9개의 카테고리내의 총 36개의 질의어가 정의되어 있다

카테고리 1. 비 클러스터 (non-clustered) 인덱스의 효율성

카테고리 1은 데이타베이스 관리자에게 테이블의 분포에 따른 시스템의 non-clustered 인덱스의 효율 전환점을 일려준다. 하나의 튜플, 1%, 5%, 10%, 20%, 25%, 33%, 50% 의 선택율(selectivity)를 가지는 총 8개의 질의어가 제안되었으며 각각 데이블 *table1*의 *seq, s1, s5, s10. s20, s25, s33, s50* 속성에 대해 비 클러스터 인덱스를 생성했을 때와 인덱스를 생성하지 않았을 때의 수행 시간의 평균을 비교하여 시스템의 비 클러스티 인덱스 효율성을 파악한다 다음은 1% 선택율을 가지는 정확 매치

(exact match) 질의를 예로 보인 것이다

```
select *
from table1
where s1 = 'aaaaaaa001';
```

카테고리 2. 복합 (composite) 인덱스의 효율성

카테고리 2는 시스템의 복합 인덱스의 효율성을 알아볼 수 있으며, 2개의 속성으로 구성된 복합 인덱스와 속성 각각에 대해 생성한 비 클러스터 인덱스에 대한 효율성 평가, 복합 인덱스를 구성하는 속성의 순서에 따른 효율성 평가를 포함한 총 4개의 질의어를 제안하였다. 다음은 2개의 속성으로 구성된 복합 인덱스와 속성 각각에 대해 생성한 비 클러스터 인덱스에 대한 효율성 평가 질의를 예로 보인 것이다.

```
select *
from table1
where s20 = 'ddddddd001' and s33 = 10000;
```

카테고리 3, 4 클러스터 (clustered) 인덱스의 효율성

카테고리 3과 카테고리 4는 정확 매치와 범위 질의 (range query)의 클러스터 인덱스 효율성을 알아볼 수 있으며, 시스템의 클러스터 인덱스의 효율 전환점을 알려준다. 각각 선택율에 따른 8개와 7개의 질의어가 제안되었다. 다음은 클러스터 인덱스와 비 클러스터 인덱스의 효율성 평가 질의로 정확 매치와 범위 질의를 예로 보인 것이다

```
select *                    select *
from table1                 from table1
where seq = 100;            where seq between 1 and 50000;
```

카테고리 5, 6, 7. 인덱스를 사용하지 못하는 술어 (predicate)

시스템에서 인덱스를 사용하지 못하는 술어를 찾는다. 카테고리 5에서는 '<>' 술어와 'OR' 술어에 대한 인덱스 효율성을 알아볼 수 있으며, 카테고리 6에서는 인덱스가 생성되어 있는 속성에 사칙 연산자와 문자열 연결 연산자 '||' 를 적용 시켰을 때의 인덱스 효율성을 알아볼 수 있다. 또한 카테고리 7에서는 와일드카드 '%'의 위치에 따른 인덱스 효율성을 알아볼 수 있다. 다음은 인덱스를 사용하지 못하는 술어에 대한 시스템의 특징을 알아보는 질의 중 인덱스가 생성된 속성에 대해 사칙 연산자를 적용 시켰을 때의 인덱스 효율성 평가 질의를 예로 보인 것이다.

```
select *                    select *
from table1                 from table1
where s33/3 = 10000;        where s33 = 10000 * 3;
```

카테고리 8. 조인 (join) 연산과 중첩 질의 (nested query) 효율성

시스템의 조인 연산과 중첩 질의의 효율성을 알아 볼 수 있는 카테고리로 동일한 인덱스를 생성 시켰을 때 같은 결과 값을 갖는 조인 연산 질의와 중첩 질의를 비교한다. 같은 테이블 내의 검색과 다른 테이블과의 검색 두 개의 질의어가 있다. 다음은 조인 연산 질의와 중첩 질의를 비교하는 질의로 테이블 *table1*과 테이블 *table2*의 조인과 중첩질의를 수행하는 질의를 예로 보인 것이다.

```
select *
from table1 S, table2 R
where S.seq = R.seq and S.s33 =10000;
```

```
select *
from table1
where seq in (select seq from table2) and s33 = 10000;
```

카테고리 9. 불필요한 정렬이나 오버헤드를 수반하는 SQL문장

일반적으로 DISTINCT는 정렬이나 오버헤드를 수반한다. 특히 테이블의 키에 대한 DISTINCT사용은 시스템의 성능에 해가 된다. 카테고리 9에서는 테이블의 키에 DISTINCT를 사용한 질의를 통해 시스템의 특성을 알아 볼 수 있다. 다음은 테이블의 키에 DISTINCT를 사용한 질의로 DISTINCT를 사용하지 않았을 때와 비교하여 시스템의 특성을 알아본다

```
select DISTINCT seq         select seq
from table1                 from table1
where s50 = 'fffffff001';   where s50 = 'fffffff001';
```

5. 결론 및 향후계획

데이타베이스의 크기가 급격히 증가함에 따라 데이타베이스 시스템의 성능이 더욱 절실히 요구된다. 본 논문에서는 관계형 데이타베이스 시스템의 새로운 성능향상 보조도구를 제안하였다. 보조도구의 설계원리, 시험 데이타베이스의 스키마와 구조, 9개의 카테고리 내의 튜닝 질의어가 제안되었다.

향후 연구과제로서는 관계형 데이타베이스 시스템의 성능향상을 위한 보조도구의 질의어를 추가하고, 또한 차세대 데이타베이스인 객체 관계형 데이타베이스 시스템을 위한 성능향상 보조도구를 개발할 예정이다.

참고문헌

[1] D. E. Shasha, Database Tuning A Principled Approach. Prentice Hall, 1992.

[2] J. Frazzini, B. Linden, Oracle7 Server Tuning, Oracle Corporation, 1995.

[3] S. Chauduri, E Christensen, G. Gracfe, V Narasayya, M. Zwilling, Microsoft Corporation, Self-Tuning Technology in Microsoft SQL Server, Data Engineering, Vol 22, No 2, 20-26, 1999.

[4] P ONeil, Database Principles Programming Performance, Morgan Kaufmann Publishers, 1994.

[5] Carolyn Turbyfill, Cyril Orji, Dina Bitton, AS3AP An ANSI Standard Scalable and Portable Benchmark for Relational Database Systems, The benchmark handbook, Morgan Kaufmann, 1993.