데이터베이스 튜닝도구 개발

안 기 덕[†]·오 정 석[††]·이 상 호[†††]

요 약

데이터베이스 시스템의 사용이 증가함에 따라 데이터베이스 시스템을 효율적으로 사용하게 하는 데이터베이스 튜닝 기술에 대한 중요성이 부각되고 있다. 본 논문은 관계형 데이터베이스 시스템과 객체관계형 데이터베이스에 대한 튜닝 도구를 제안한다. 본 도구는 사용자나 시스템 관리자에게 데이터베이스 시스템의 성능에 영향을 주는 요소들을 효과적으로 파악하여 데이터베이스 시스템의 튜닝 방법을 제시한다. 본 논문에서는 튜닝 도구의 설계원칙, 시험 데이터베이스 및 시험 질의들을 기술한다. 본 시험 도구는 총 10개의 카테고리에서 총 65개의 질의어를 제공한다. 본 시험 도구는 일반 상용 제품에 적용하였으며, 시험 결과도 논문에서 보인다.

Development of a Tuning Aid for Database Systems

Ki-Duk Ahn[†] · Jeong-Seok Oh[††] · Sang-Ho Lee[†††]

ABSTRACT

As the usage of database systems increases, the importance of database tuning techniques, which enable users to use database systems effectively, is emerging. This paper proposes a tuning aid for relational and object-relational database systems. The objective of the tuning aid is to guide users and/or system administrators to identify system performance characteristics of relational database systems and object-relational systems effectively, so that database systems could be tuned adequately. The design philosophy, test database, and tuning queries of the tuning aid are presented. It offers sixty-five queries in ten different categories. It has been applied to a commercial database system and the experimental results are also reported.

1. 서 론

현대 사회가 정보화 사회로 발전하면서 이용되는 데이터의 양은 급격히 증대되고, 데이터베이스 적용 환경 구조가 복잡 및 다양화되면서 데이터베이스의 대형화를 초래하였다. 데이터베이스의 대형화는 시스템 자원들의 작업부하를 야기 시켰고, 데이터베이스의 효율적인 관리와 사용을 중요한 논점으로 부각시켰다. 이를 위해, 데이터베이스 관리자는 시스템의 버퍼 풀 크기, 잠금 테이블, 자료구조 다양한 요소들에 기초하여 튜닝을 결정한다. 데이터베이스 튜닝은 데이터베이스 시스템 자원이 활용되는 방식을 개선하여 보다 향상된 성능을 낼 수 있도록 조정하는 작업이다.

데이터베이스 튜닝은 데이터베이스 성능평가(benchmark)와 구분된다. 데이터베이스 성능평가는 두 개 이상의 데이터베이스 시스템들에 대한 성능을 정밀하게 비교하여 구매자의 요구 사항에 적합한 데이터베이스 시스템을 구매하도록 결정하는데 도움을 주는 것이다[4]. 반면에, 데이터베이스 튜닝도구는 다른 데이터베이스 시스템과 성능을 비교하지 않으며, 임의의 데이터베이스 환경에서 데이터베이스 시스템에 대한 성능을 향상함을 목적으로 한다.

데이터베이스 튜닝에서 SQL 튜닝은 향상된 성능을

† 준 회 원 : 다우 기술(주) 연구원
†† 준 회 원 : 숭실대학교 대학원 컴퓨터학과
††† 정 회 원 : 숭실대학교 컴퓨터학부 교수
논문접수 : 2000년 7월 7일, 심사완료 : 2000년 11월 11일

얻기 위한 중요한 부분이다[2]. SQL 튜닝이란 SQL 문장에 대한 튜닝으로 데이터베이스를 효율적으로 사용하도록 SQL 문장을 수정하고 작성하는 것이다. 동일한 결과 값을 갖는 질의라도 SQL 문장의 형태에 따라 그 응답시간은 상이하게 다를 수 있다. 테이블에 인덱스가 적절하게 생성되었어도 인덱스를 사용하지 못하는 SQL 문장으로 질의한다면 데이터베이스 성능은 저하될 것이기 때문이다. 반면에 특정한 경우에는 인덱스를 사용하지 않는 것이 더 효율적이기 때문에 SQL 튜닝을 통해 인덱스를 적절하게 선택하고 사용하는 것이 중요하다. 데이터베이스 인덱스를 효과적으로 선택하고 사용하기 위해서는, 인덱스 생성 시점, 인덱스를 생성할 속성 선정, 생성된 인덱스를 사용할 수 있도록 SQL 문장 작성 방법 등을 고려해야 한다[2].

본 논문에서는 관계형 데이터베이스 시스템과 객체관계형 데이터베이스 시스템에 적용할 수 있는 튜닝도구를 제안한다. 본 도구는 사용자가 데이터베이스 시스템을 효율적으로 사용할 수 있도록 관리자가 시스템을 올바르게 튜닝할 수 있도록 도와주며, 사용자에게는 데이터베이스 시스템을 효율적으로 사용할 수 있도록 도와준다. 관계형 데이터베이스 시스템에 대한 튜닝도구의 기능은 인덱스의 생성 관계와 질의 선택율(selectivity)에 의거하며, 데이터베이스에 대한 성능상의 특징을 효과적으로 분석하여 데이터베이스 시스템에 대한 인덱스 생성 가이드를 제공한다. 객체관계형 데이터베이스 시스템에 대한 튜닝도구의 기능은 객체지향적 특징을 이용하는 SQL로 질의했을 때와 객체지향적 특징을 이용하지 않는 SQL로 질의했을 때를 비교, 분석하여 SQL 형태에 따른 데이터베이스 시스템의 객체지향 기능 효율성을 알아볼 수 있다. 본 논문에서 제안하는 튜닝도구는 총 10개의 카테고리와 65개의 시험 질의로 구성되어 있다. 또한 제안하는 튜닝도구를 상용 데이터베이스에 적용하였으며, 그 시험 결과도 보고한다.

본 논문의 구성은 다음과 같다. 제2장에서는 본 튜닝도구의 설계 원칙 및 시험 데이터베이스의 스키마와 인스턴스에 대한 개괄적 정보를 소개한다. 제3장에서는 시험 질의 형태를 보여준다. 제4장에서 튜닝도구를 상용 데이터베이스 시스템에 적용한 시험결과를 제시하고, 마지막으로 제5장에서 결론을 맺는다.

2. 설계 원칙 및 시험 데이터베이스

데이터베이스 시스템 튜닝 방법에 관련되는 요소로는, 광의의 관점에서는 하드웨어 구성, 운영체제 특성, 데이터베이스 구조 및 배치(deployment) 등이며 협의의 관점에서는 데이터베이스에서 사용하는 다양한 자원(예를 들면, 데이터 버퍼 크기, 인덱스 버퍼 크기, 데이터 및 인덱스 페이지 크기, 잠금 테이블 크기, 세마포어(semaphore) 크기, 로그 버퍼 크기, 회복 기법 등)에 대한 결정 및 조정 등이 있다. 데이터베이스 튜닝에 관련되는 각각의 요소들은 그 성격상 상호 밀접한 연관성을 가진다.

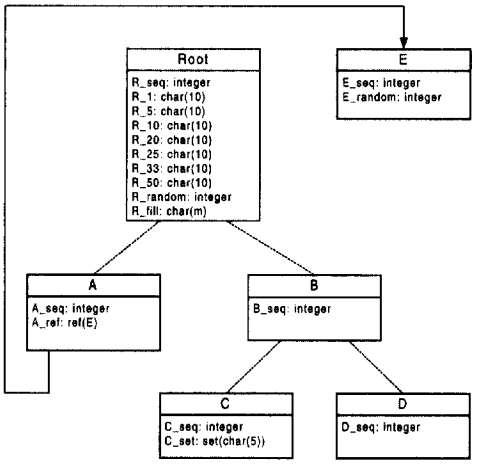
데이터베이스 튜닝 요소 각각에 대한 조정 기법은 문헌상에서 데이터베이스 연구자 및 개발자에 의해 지속적으로 연구/개발되고 있다. 그러나, 단일 요소를 고려하는 튜닝 기법을, 모든 튜닝 요소가 서로 연관되어 운영되는 실제 데이터베이스 시스템 환경에서 적용하기에는 근본적인 한계가 있다. 이론적으로도 상호 연관성이 있는 모든 튜닝 요소를 단일 모델 하에서 튜닝하는 기법은 현재까지 정립되어 있지 않다. 이러한 이론적 한계점들을 극복하기 위해 본 논문에서는 "실질적인" 접근 방식을 취하였다. 실질적인 접근 방식이란 실제 데이터베이스 시스템이 사용되는 유사한 상황하에서 다양한 형태의 질의어를 수행하여 해당 데이터베이스 시스템의 성능 특성을 파악하여 효과적인 튜닝을 도와주는 정보를 제공하고자 하였다.

데이터베이스 크기는 데이터베이스 시스템의 성능과 많은 관계가 있다. 본 도구에서는 효과적으로 데이터베이스 시스템의 특징을 알아보기 위해서 데이터베이스 크기, 각 행의 크기 등을 사용자가 임의로 정하게 하였다. 이러한 방식은 튜닝 도구에 확장성(scalability)을 제공한다. 또한 본 튜닝 도구의 효과성(effectiveness)을 고려하여, 시험 질의는 하나의 튜플, 1%, 5%, 10%, 20%, 25%, 33%, 50%의 질의 선택율을 기준으로 설계되었고, 인덱스의 생성 관계, 객체관계형 데이터베이스 시스템과 관계형 데이터베이스 시스템의 모델 차이 등을 고려하여 사용자와 데이터베이스 관리자에게 데이터베이스 시스템에 대한 성능상의 특징들을 효과적으로 인식시켜 줄 수 있도록 하였다.

시험 데이터베이스의 스키마와 인스턴스는 데이터의 분포에 따른 시스템의 성능을 효과적으로 파악할 수 있고, 객체관계형 시스템의 기본적인 기능들을 포함할 수 있도록 구성되었다. 본 튜닝도구는 관계형 데이터베이스 시스템 질의군과 객체관계형 데이터베이스 시스템 질의군으로 나뉘어지고, 시험 데이터베이스 스키마는 관계형 데이터베이스 시스템 질의군과 객체관계

형 데이터베이스 시스템 질의군에서 다르게 해석된다.

(그림 1)은 시험 데이터베이스에 대한 스키마를 도식화 한 것이다. 객체관계형 데이터베이스 시스템 질의군의 경우 얇은 선의 직사각형은 인스턴스가 없는 클래스를 나타내고, 굵은 선의 직사각형은 인스턴스를 가지는 클래스를 나타낸다. 또한, 점선은 상속의 관계를 나타낸다. 예를 들어 클래스 *C*는 클래스 *B*로부터 상속받고, 클래스 *B*는 클래스 *ROOT*에서 상속받음을



(그림 1) 시험 데이터베이스 스키마

의미한다. 시험 데이터베이스 스키마에서 보이는 상속은 클래스의 속성에 대한 상속이며, 실제적인 인스턴스를 상속받는 것을 의미하지는 않는다. 그리고 화살표는 클래스의 참조를 나타낸다. 클래스를 나타내는 직사각형은 2개의 작은 직사각형으로 나뉘어지는데, 위의 직사각형에는 해당 클래스의 이름이 표시되어 있고, 아래의 직사각형에는 해당 클래스가 포함하는 속성 이름과 속성 자료형의 쌍으로 표기된다. 반면에, 관계형 데이터베이스 시스템 질의군을 위해서는 위의 객체관계형 데이터베이스 시스템 질의군을 위한 스키마에서 설명한 클래스 계층의 상속과 클래스 참조, 클래스 *A, B, C, D, E*는 무시하며 테이블 *ROOT*만을 독립적인 테이블로 간주하여 사용한다.

<표 1>은 생성되는 속성 값과 데이터의 분포, 결과적으로 몇 개의 고유한 속성 값이 만들어지는지 보여주며, NUM은 시험 데이터베이스의 튜플의 개수이다. 시험 데이터베이스의 *{ROOT, A, B, C, D E}_seq* 속성은 주 키(primary key) 속성을 가지며 순차적인 정수 값이다. 또한 *R_1, R_5, R_10, R_20, R_25, R_33, R_50* 속성은 각각 1%, 5%, 10%, 20%, 25%, 33%, 50%의 고유 값 분포(unique value distribution)를 갖는다. *{R, E}_random* 속성은 임의의 정수 값으로 생성된다. 또한, *C_set* 속성은 객체관계형 데이터베이스 시스템의 집합 자료형으로서 1개부터 10개까지의 원소로 구성된다.

본 튜닝도구에서는 시험 데이터베이스 크기의 사용자 정의가 가능하다. 객체관계형 데이터베이스 시스템의 경우 각 클래스 내의 튜플 개수를 N(클래스)라 정의하고, 관계형 데이터베이스 시스템의 경우 각 테이블 내의 튜플 개수를 N(테이블)이라 정의한다. 튜닝도구 사용자는 각각의 응용분야에 알맞은 데이터베이스의 크기를 선택하기 위해 N(클래스 또는 테이블)을 자유롭게 정의할 수 있지만, 다음과 같은 제약사항이 요구된다.

〈표 1〉 시험 데이터베이스의 데이터 분포

속성	값	유일한 값의 개수	유일한 값 당 튜플의 개수	데이터의 분포
{}_seq	1,2,3,...,NUM	NUM	1	순차적 값
R_1	'aaaaaaa001', ... , 'aaaaaaa100'	100	NUM/100	1% 고유 값 분포
R_5	'bbbbbbb001', ... , 'bbbbbbb020'	20	NUM/20	5% 고유 값 분포
R_10	'ccccccc001', ... , 'ccccccc010'	10	NUM/10	10% 고유 값 분포
R_20	'ddddddd001', ... , 'ddddddd005'	5	NUM/5	20% 고유 값 분포
R_25	'eeeeeee001', ... , 'eeeeeee004'	4	NUM/4	25% 고유 값 분포
R_33	'fffffff001', ... , 'fffffff003'	3	NUM/3	33% 고유 값 분포
R_50	'ggggggg001', 'ggggggg002'	2	NUM/2	50% 고유 값 분포
R_fill	임의의 문자열	알수 없음	알수 없음	임의의 값 분포
{}_random	임의의 정수값	알수 없음	알수 없음	임의의 값 분포
C_set	{'aaaaa', 'baaaa', ... , 'jaaaa'}	10	NUM/10	1개부터 10개까지의 원소

· N(클래스) = 1000 * *scaling factor*
· N(테이블) = 1000 * *scaling factor*

시험 데이터베이스 튜플 개수의 제약사항에서 *scaling factor*는 시험 데이터베이스 튜플의 개수를 1,000의 배수로 제약하기 위해 사용자가 정의하는 인수로서 양의 정수이다. 예를 들어, 튜닝도구 사용자가 시험하고자 하는 데이터베이스 테이블의 튜플 개수가 1,000,000개라고 하면, *scaling factor*를 1,000으로 설정하면 된다. 또한, 테이블 *ROOT*에서 각각의 튜플 크기를 M이라 정의하여 튜닝도구 사용자는 알맞은 튜플 크기를 선택하기 위해 M을 자유롭게 정의할 수 있다. 테이블 *ROOT*의 튜플 크기 M을 식으로 나타내면 식 (1)과 같으며, 식 (1)에서 *int*는 데이터베이스 시스템의 정수형(integer) 크기(byte)이다.

$$M=(10\times7)+(int\times2)+m \quad (1)$$

식 (1)에서 (10×7)은 테이블 *ROOT*의 문자형(char(10)) 속성인 *R_1, R_5, R_10, R_20, R_25, R_33, R_50*의 크기를 의미하며, (*int*×2)는 테이블 *ROOT*의 정수형 속성인 *R_seq, R_random*의 크기를 의미한다. 또한, m은 테이블 *ROOT*의 *R_fill* 속성의 크기로 튜플의 크기를 정의하기 위해서 튜닝도구 사용자가 설정한다. 예를 들어, 시험하고자 하는 데이터베이스 시스템의 *int*가 4바이트라고 가정하고, 테이블 *ROOT*에서 각 튜플의 크기를 100바이트로 정의하고자 한다면, 튜닝도구 사용자는 m을 22로 설정하면 된다.

3. 시험 질의

본 장에서는 데이터베이스 시스템의 성능 관련 요소를 파악할 수 있는 다양한 질의를 제안한다. 시험 질의는 총 10개의 카테고리로 분류되며, 관계형 데이터베이스 시스템 질의군(카테고리 1부터 카테고리 7)과 객체관계형 데이터베이스 시스템 질의군(카테고리 8부터 카테고리 10)으로 나뉘어진다. 관계형 데이터베이스 시스템 질의군은 사용자에게 데이터의 분포 및 질의 형태에 따라 인덱스의 성능향상을 비율로 표시하는 인덱스의 효율성을 파악하는 질의들로 구성된다. 객체관계형 시스템 질의군은 객체관계형 시스템의 객체지향 기능에 대한 효율성을 파악하며, 시스템에서 제공하는 객체지향형 연산자 및 기능을 이용하는 질의(이하, 객체관계형 질의)와 동일한 의미의 질의로서, 객체지향형 연산자 및 기능을 이용하지 않는 질의(이하, 관계형 질의)들로 구성된다.

3.1 카테고리 1 : 정확 매치에 대한 비클러스터 인덱스의 효율성

비클러스터 인덱스의 사용은 데이터베이스 시스템에서 자료 검색의 효율을 향상시키지만, 검색되는 자료의 질의 선택율에 따라 인덱스의 사용이 검색 효율을 떨어뜨릴 수 있다. 카테고리 1은 테이블의 데이터 분포에 따른 비클러스터 인덱스 효율성을 파악하며, 총 8개의 질의 선택율을 가지는 질의가 제안되었다. 각각의 질의에 대한 결과는 시험 데이터베이스에 각 질의 조건의 속성에 인덱스를 생성하지 않았을 때 측정한 평균 수행시간을 비클러스터 인덱스를 생성했을 때 측정한 평균 수행시간으로 나눈 평균 수행시간의 비율이다. 각 질의의 질의 조건은 <표 2>와 같다.

- **질의 1-1 ~ 1-8**

  select R_seq, R_1, R_5, R_10, R_20, R_25, R_33, R_50 from ROOT
  where 조건;

〈표 2〉 카테고리 1의 질의 조건

질의	조건
질의 1-1	R_seq = 100
질의 1-2 (1%)	R_1 = 'aaaaaaa001'
질의 1-3 (5%)	R_5 = 'bbbbbbb001'
질의 1-4 (10%)	R_10 = 'ccccccc001'
질의 1-5 (20%)	R_20 = 'ddddddd001'
질의 1-6 (25%)	R_25 = 'eeeeeee001'
질의 1-7 (33%)	R_33 = 'fffffff001'
질의 1-8 (50%)	R_50 = 'ggggggg001'

3.2 카테고리 2 : 복합 인덱스의 효율성

카테고리 2는 데이터베이스 관리자에게 복합 인덱스 효율성을 알아볼 수 있게 한다. 결과는 질의 2-1에 대해 세 가지의 상이한 인덱스 생성 조건에서 측정하여 평균 수행시간의 비율을 보고하며, 비율은 두 가지가 보고된다. 첫 번째 비율은 인덱스 생성조건 2에서 측정된 평균 수행시간을 인덱스 생성조건1에서 측정한 평균 수행시간으로 나누어 계산하며, 두 번째 비율은 인덱스 생성조건 3에서 측정한 평균 수행시간을 인덱스 생성조건 1에서 측정한 평균 수행시간으로 나누어

계산한다. 자세한 인덱스 생성 조건은 <표 3>과 같다.

- **질의 2-1**

  select R_seq, R_1, R_5, R_10, R_20, R_25, R_33, R_50 from ROOT
  where 조건;

〈표 3〉 카테고리 2의 질의 및 인덱스 생성 조건

질의	조건	인덱스 생성 조건 1	인덱스 생성 조건 2	인덱스 생성 조건 3
질의 2-1	R_seq=1 and R_1= 'aaaaaaa001'	(*R_seq, R_1*) 속성의 순서로 복합 인덱스를 생성하고 다른 인덱스는 생성하지 않았을 때.	(*R_1, R_seq*) 속성의 순서로 복합 인덱스를 생성하고 다른 인덱스는 생성하지 않았을 때.	*R_seq*과 *R_1* 속성 각각에 비클러스터 인덱스를 생성했을 때.
질의 2-2	R_1 = 'aaaaaaa001' and R_10 = 'ccccccc001'	(*R_1, R_10*) 속성의 순서로 복합 인덱스를 생성하고 다른 인덱스는 생성하지 않았을 때.	(*R_10,R_1*) 속성의 순서로 복합 인덱스를 생성하고 다른 인덱스는 생성하지 않았을 때.	*R_1*과 *R_10* 속성 각각에 비클러스터 인덱스를 생성했을 때.

3.3 카테고리 3 : 정확 매치에 대한 클러스터 인덱스의 효율성

카테고리 3은 데이터베이스 관리자에게 테이블의 데이터 분포에 따른 정확 매치에 대한 클러스터 인덱스 효율성을 고려하며, 총 7개의 질의 선택율을 가지는 질의가 제안되었다. 각각의 질의에 대한 결과는 시험 데이터베이스에 각 질의 조건의 속성에 비클러스터 인덱스를 생성했을 때 측정한 평균 수행시간을 클러스터 인덱스를 생성하였을 때 측정한 평균 수행시간으로 나눈 평균 수행시간의 비율이다. 각 질의의 질의 조건은 <표 4>와 같다.

〈표 4〉 카테고리 3의 질의 조건

질 의	조 건
질의 3-1 (1%)	R_1 = 'aaaaaaa001'
질의 3-2 (5%)	R_5 = 'bbbbbbb001'
질의 3-3 (10%)	R_10 = 'ccccccc001'
질의 3-4 (20%)	R_20 = 'ddddddd001'
질의 3-5 (25%)	R_25 = 'eeeeeee001'
질의 3-6 (33%)	R_33 = 'fffffff001'
질의 3-7 (50%)	R_50 = 'ggggggg001'

- **질의 3-1 ~ 3-8**

  select R_seq, R_1, R_5, R_10, R_20, R_25, R_33, R_50 from ROOT
  where 조건;

3.4 카테고리 4 : 범위 질의에 대한 클러스터 인덱스의 효율성

카테고리 4는 테이블의 데이터 분포에 따른 범위 질의에 대한 클러스터 인덱스 효율성을 고려하며, 총 7개의 질의 선택율을 가지는 범위 질의가 제안되었다. 각각의 질의에 대한 결과는 시험 데이터베이스에 *R_seq* 속성에 비클러스터 인덱스를 생성했을 때 측정한 평균 수행시간을 *R_seq* 속성에 클러스터 인덱스를 생성했을 때 측정한 평균 수행시간으로 나눈 평균 수행시간의 비율이다. 각 질의의 질의 조건은 <표 5>와 같다.

- **질의 4-1 ~ 4-7**

  select R_seq, R_1, R_5, R_10, R_20, R_25, R_33, R_50 from ROOT
  where 조건;

〈표 5〉 카테고리 4의 질의 조건

질 의	조 건
질의 4-1 (1%)	R_seq between 1 and N(ROOT) * 0.01
질의 4-2 (5%)	R_seq between 1 and N(ROOT) * 0.05
질의 4-3 (10%)	R_seq between 1 and N(ROOT) * 0.1
질의 4-4 (20%)	R_seq between 1 and N(ROOT) * 0.2
질의 4-5 (25%)	R_seq between 1 and N(ROOT) * 0.25
질의 4-6 (33%)	R_seq between 1 and N(ROOT) * 0.33
질의 4-7 (50%)	R_seq between 1 and N(ROOT) * 0.5

3.5 카테고리 5 : 'NOT' 술어에 대한 인덱스의 효율성

'NOT' 술어를 사용하는 질의는 경우에 따라 데이터베이스 시스템의 질의 처리속도를 저하시킨다. 카테고리 5는 데이터베이스 시스템에서 'NOT' 술어에 대한 비클러스터 인덱스 효율성을 고려하며, 총 7개의 질의 선택율을 가지는 질의를 제안한다. 각각의 질의에 대한 결과는 시험 데이터베이스에 *R_seq* 속성에 인덱스를 생성하지 않았을 때 측정한 평균 수행시간을 *R_seq* 속성에 비클러스터 인덱스를 생성했을 때 측정한 평균 수행시간으로 나눈 평균 수행시간의 비율이다. 각각 질의의 질의 조건은 <표 6>과 같다.

- **질의 5-1 ~ 5-14**

  select R_seq, R_1, R_5, R_10, R_20, R_25, R_33, R_50 from ROOT

where 조건;

〈표 6〉 카테고리 5의 질의 및 인덱스 생성 조건

질 의	조 건
질의 5-1 (1%)	R_seq NOT between 1 and N(ROOT) - N(ROOT)*0.01
질의 5-2 (5%)	R_seq NOT between 1 and N(ROOT) - N(ROOT)*0.05
질의 5-3 (10%)	R_seq NOT between 1 and N(ROOT) - N(ROOT)*0.1
질의 5-4 (20%)	R_seq NOT between 1 and N(ROOT) - N(ROOT)*0.2
질의 5-5 (25%)	R_seq NOT between 1 and N(ROOT) - N(ROOT)*0.25
질의 5-6 (33%)	R_seq NOT between 1 and N(ROOT) - N(ROOT)*0.33
질의 5-7 (50%)	R_seq NOT between 1 and N(ROOT) - N(ROOT)*0.5

3.6 카테고리 6 : 'OR' 술어에 대한 인덱스의 효율성

'OR' 술어를 사용하는 질의는 경우에 따라 데이터베이스 시스템의 질의 처리속도를 저하시킨다. 카테고리 6은 데이터베이스 시스템에서 'OR' 술어에 대한 비클러스터 인덱스 효율성을 고려하며, 총 6개의 질의 선택율을 가지는 질의가 제안되었다. 질의에 대한 결과는 시험 데이터베이스에 각 질의 조건의 속성에 인덱스를 생성하지 않았을 때 측정한 평균 수행시간을 비클러스터 인덱스를 생성하였을 때 측정한 평균 수행시간으로 나눈 평균 수행시간의 비율이다. 각 질의의 질의 조건은 <표 7>과 같다.

- **질의 6-1**

  select R_seq, R_1, R_5, R_10, R_20, R_25, R_33, R_50 from ROOT
  where 조건;

〈표 7〉 카테고리 6의 질의 조건

질 의	조 건
질의 6-1 (1%)	R_seq = 1 OR R_1 = 'aaaaaaa001'
질의 6-2 (6%)	R_1 = 'aaaaaaa001' OR R_5 = 'bbbbbbb001'
질의 6-3 (10%)	R_5 = 'bbbbbbb001' OR R_5 = 'bbbbbbb002'
질의 6-4 (15%)	R_5 = 'bbbbbbb001' OR R_10 = 'ccccccc001'
질의 6-5 (20%)	R_10 = 'ccccccc001' OR R_10 = 'ccccccc002'
질의 6-6 (30%)	R_10 = 'ccccccc001' OR R_20 = 'ddddddd001'

3.7 카테고리 7 : 접미어 매치 질의에 대한 인덱스 효율성

데이터베이스 시스템에 따라서 와일드카드(wildcard) '%'를 사용한 질의는 인덱스 효율성이 떨어질 수 있다. 카테고리 7은 와일드카드 '%'를 이용한 접미어 매치 질의(suffix match query)에 대한 인덱스 효율성을 고려하며, 총 7개의 질의 선택율을 가지는 질의로 구성되어 있다. 각각의 질의에 대한 결과는 시험 데이터베이스에 각 질의 조건의 속성에 인덱스를 생성하였을 때 측정한 평균 수행시간을 비클러스터 인덱스를 생성했을 때 측정한 평균 수행시간으로 나눈 평균 수행시간의 비율이다. 각 질의의 질의 조건은 <표 8>과 같다.

- **질의 7-1 ~ 7-7**

  select R_seq, R_1, R_5, R_10, R_20, R_25, R_33, R_50 from ROOT
  where 조건;

〈표 8〉 카테고리 7의 질의 조건

질 의	조 건
질의 7-1 (1%)	R_1 like '%aa001'
질의 7-2 (5%)	R_5 like '%bb001'
질의 7-3 (10%)	R_10 like '%cc001'
질의 7-4 (20%)	R_20 like '%dd001'
질의 7-5 (25%)	R_25 like '%ee001'
질의 7-6 (33%)	R_33 like '%ff001'
질의 7-7 (50%)	R_50 like '%gg001'

3.8 카테고리 8 : 클래스 계층에 대한 효율성

카테고리 8은 객체관계형 데이터베이스 시스템의 클래스 계층에 대한 효율성을 파악하며. 객체관계형 질의와 관계형 질의에 대해 총 8개의 질의 선택율을 가지는 질의가 제안되었다. 객체 관계형 질의는 상속성(inheritance)에 의해 클래스 계층을 모두 검색하지만, 관계형 질의는 상속성을 지원하지 못하기 때문에 클래스 계층을 모두 검색하기 위해서는 각각의 클래스를 독립적인 클래스로 간주하고 모두 UNION 해야 한다[5]. 인덱스는 객체관계형 데이터베이스의 클래스 계층에 대해 생성되며, 각각의 클래스에 대한 인덱스는 생성되지 않고, 또한 각 질의 조건에 비클러스터 인덱스로 생성된다. 각각의 질의에 대한 결과는 객체관계형 질의로 측정한 평균 수행시간을 동일한 의미의 관계형 질의로 측정한 평균 수행시간으로 나눈 비율이다. 각 질의의 질의 조건은 <표 9>와 같다.

〈표 9〉 카테고리 8의 질의 조건

질 의	객체관계형/관계형 질의 조건
질의 8-1	R_seq = 100
질의 8-2 (1%)	R_1 = 'aaaaaaa001'
질의 8-3 (5%)	R_5 = 'bbbbbbb001'
질의 8-4 (10%)	R_10 = 'ccccccc001'
질의 8-5 (20%)	R_20 = 'ddddddd001'
질의 8-6 (25%)	R_25 = 'eeeeeee001'
질의 8-7 (33%)	R_33 = 'fffffff001'
질의 8-8 (50%)	R_50 = 'ggggggg001'

- **Query 8-1 ~ 8-8**

객체관계형 질의 : select R_seq, R_1, R_5, R_10, R_20, R_25, R_33, R_50 from ROOT where 객체관계형 질의 조건;

관계형 질의 : select R_seq, R_1, R_5, R_10, R_20, R_25, R_33, R_50 from A where 관계형 질의 조건
UNION ALL
select R_seq, R_1, R_5, R_10, R_20, R_25, R_33, R_50 from C where 관계형 질의 조건
UNION ALL
select R_seq, R_1, R_5, R_10, R_20, R_25, R_33, R_50 from D where 관계형 질의 조건;

3.9 카테고리 9 : 객체 참조에 대한 효율성

카테고리 9는 객체관계형 데이터베이스 시스템의 객체 참조의 효율성을 분석하며, 객체 관계형 질의와 관계형 질의에 대해 총 8개의 질의 선택율을 가지는 질의가 제안되었다. 객체관계형 질의는 다른 클래스를 참조하는 참조 속성(reference attribute)을 사용하여 검색이 가능하지만, 관계형 질의는 참조 속성을 지원하지 못하기 때문에 다른 객체를 참조하기 위해서는 각각의 클래스를 독립적인 클래스로 간주하여 조인을 사용해야 한다. 각각의 질의에 대한 결과는 객체관계형 질의로 측정한 평균 수행시간을 동일한 의미의 관계형 질의로 측정한 평균 수행시간으로 나눈 비율이다. 자세한 인덱스 생성 조건과 질의 조건은 <표 10>과 같다.

- **질의 9-1 ~ 9-8**

객체관계형 질의 : select a.R_seq, a.R_1, a.R_5, a.R_10, a.R_20, a.R_25, a.R_33, a.R_50, a.A_ref->E_seq from A a where 객체관계형 질의 조건;

〈표 10〉 카테고리 9의 질의 및 인덱스 생성 조건

질의	객체관계형 질의 조건	관계형 질의 조건	인덱스 생성 조건
질의 9-1	a.R_seq = 100	a.R_seq = e.E_seq and a.R_seq = 100	클래스 *ROOT*의 *R_seq* 속성, 클래스 *A*의 *A_ref* 속성, 클래스 *E*의 *E_seq* 속성에 비클러스터 인덱스를 생성.
질의 9-2 (1%)	a.R_1 = 'aaaaaaa001'	a.R_seq = e.E_seq and a.R_1 = 'aaaaaaa001'	클래스 *ROOT*의 *R_seq*, *R_1* 속성, 클래스 *A*의 *A_ref* 속성, 클래스 *E*의 *E_seq* 속성에 비클러스터 인덱스를 생성.
질의 9-3 (5%)	a.R_5 = 'bbbbbbb001'	a.R_seq = e.E_seq and a.R_5 = 'bbbbbbb001'	클래스 *ROOT*의 *R_seq*, *R_5* 속성, 클래스 *A*의 *A_ref* 속성, 클래스 *E*의 *E_seq* 속성에 비클러스터 인덱스를 생성.
질의 9-4 (10%)	a.R_10 = 'ccccccc001'	a.R_seq = e.E_seq and a.R_10 = 'ccccccc001'	클래스 *ROOT*의 *R_seq*, *R_10* 속성, 클래스 *A*의 *A_ref* 속성, 클래스 *E*의 *E_seq* 속성에 비클러스터 인덱스를 생성.
질의 9-5 (20%)	a.R_20 = 'ddddddd001'	a.R_seq = e.E_seq and a.R_20 = 'ddddddd001'	클래스 *ROOT*의 *R_seq*, *R_20* 속성, 클래스 *A*의 *A_ref* 속성, 클래스 *E*의 *E_seq* 속성에 비클러스터 인덱스를 생성.
질의 9-6 (25%)	a.R_25 = 'eeeeeee001'	a.R_seq = e.E_seq and a.R_25 = 'eeeeeee001'	클래스 *ROOT*의 *R_seq*, *R_25* 속성, 클래스 *A*의 *A_ref* 속성, 클래스 *E*의 *E_seq* 속성에 비클러스터 인덱스를 생성.
질의 9-7 (33%)	a.R_33 = 'fffffff001'	a.R_seq = e.E_seq and a.R_33 = 'fffffff001'	클래스 *ROOT*의 *R_seq*, *R_33* 속성, 클래스 *A*의 *A_ref* 속성, 클래스 *E*의 *E_seq* 속성에 비클러스터 인덱스를 생성.
질의 9-8 (50%)	a.R_50 = 'ggggggg001'	a.R_seq = e.E_seq and a.R_50 = 'ggggggg001'	클래스 *ROOT*의 *R_seq*, *R_50* 속성, 클래스 *A*의 *A_ref* 속성, 클래스 *E*의 *E_seq* 속성에 비클러스터 인덱스를 생성.

관계형 질의 : select a.R_seq, a.R_1, a.R_5, a.R_10, a.R_20, a.R_25, a.R_33, a.R_50 ,a.A_ref e.E_seq from A a, E e where 관계형 질의 조건;

3.10 카테고리 10 : 집합 자료형의 효율성

카테고리 10은 객체관계형 데이터베이스 시스템의 집합 자료형 중 원소들간의 중복을 허락하지 않는 집합(set)에 대한 효율성을 파악하며, 객체관계형 질의와 관계형 질의에 대해 총 5개의 질의 선택율을 가지는 질의가 제안되었다. 객체관계형 질의는 시스템에서 지원하는 집합 연산자를 사용하여 검색이 가능하지만, 관계형 질의는 객체관계형 데이터베이스의 집합 자료형을 질의하기 위해서 집합 자료형을 구현하기 위해서 제안한 스키마 이외의 *SET* 테이블을 생성하여 조인을 사용해야 한다. 각각의 질의에 대한 결과는 객체관계형 질의로 측정한 평균 수행시간을 동일한 의미의 관계형 질의로 측정한 평균 수행시간으로 나눈 비율이다. 인덱스는 테이블 *C*의 *C_set* 속성과 테이블 *SET*의 *S_element* 속성에 클러스터 인덱스로 생성된다. 각 질의의 질의 조건은 <표 11>과 같으며, 카테고리 10에서만 사용하는 테이블 *SET*의 스키마는 다음과 같다.

TABLE SET (
S_seq integer, S_element char(10));

〈표 11〉 카테고리 10의 질의 및 인덱스 생성 조건

질 의	객체관계형 질의 조건	관계형 질의 조건
질의 10-1 (10%)	'jaaaa' in C_set	c.R_seq = s.S_seq and s.S_element='jaaaa'
질의 10-2 (20%)	'iaaaa' in C_set	c.R_seq = s.S_seq and s.S_element='iaaaa'
질의 10-3 (30%)	'haaaa' in C_set	c.R_seq = s.S_seq and s.S_element='haaaa'
질의 10-4 (40%)	'gaaaa' in C_set	c.R_seq = s.S_seq and s.S_element='gaaaa'
질의 10-5 (50%)	'faaaa' in C_set	c.R_seq = s.S_seq and s.S_element='faaaa'

● 질의 10-1 ~ 10-5

객체관계형 질의 :

select R_seq, R_1, R_5, R_10, R_20, R_25, R_33, R_50 from C
where 객체관계형 질의 조건;

관계형 질의 :

select c.R_seq, c.R_1, c.R_5, c.R_10, c.R_20, c.R_25, c.R_33, c.R_50 from C c, SET s
where 관계형 질의 조건;

4. 시험 결과 및 평가

실행 환경은 128M의 주 메모리를 가진 Sun Ultra2 Sparc 시스템을 사용하였으며, 운영체제는 Solaris 2.5.1을 사용하였다. 시험은 객체관계형 모델을 지원하는 상용 데이터베이스 제품이 사용되었다. 시험을 위해 모든 테이블과 클래스의 *scaling factor*를 1,000으로 설정하였고, 테이블 *ROOT*의 *m*을 22로 설정하였다. 각각의 질의에 대해 5번의 콜드 런을 수행하고, 평균 수행시간의 비율을 보고한다.

<표 12>는 관계형 데이터베이스 시스템의 정확 매치에 대한 비클러스터 인덱스 효율성을 평가하는 카테고리 1의 시험 결과를 나타내며, 결과를 분석해보면 질의 선택율이 10%를 초과하는 질의에 대해서는 테이블 전체 스캔(full scan)보다 비클러스터 인덱스의 효율이 낮았기 때문에, 카테고리 1의 인덱스 생성 가이드는 질의 선택율의 분포가 10% 이하일 때 비클러스터 인덱스를 생성하는 것이 효율적인 방법이다.

〈표 12〉 카테고리 1의 시험 결과

질 의	조건1: 인덱스를 생성하지 않았을 때	조건 2: 비클러스터 인덱스를 생성했을 때	비 율 (조건 1/조건 2)
질의 1-1	12.154	0.560	21.703
질의 1-2 (1%)	25.470	3.435	7.414
질의 1-3 (5%)	30.697	9.657	3.178
질의 1-4 (10%)	38.977	32.900	1.184
질의 1-5 (20%)	100.297	101.490	0.988
질의 1-6 (25%)	148.130	167.651	0.883
질의 1-7 (33%)	164.365	201.033	0.817
질의 1-8 (50%)	217.300	274.006	0.793

카테고리 2는 관계형 데이터베이스 시스템의 복합 인덱스 효율성을 평가하였다. <표 13>은 카테고리 2의 튜닝 질의를 시험한 결과를 나타내며, 비율1과 비율2에서 복합 인덱스 구성 시 낮은 자료 분포를 갖는 속

성을 기준으로 인덱스를 구성하는 것이 높은 자료 분포를 갖는 속성을 기준으로 인덱스를 구성하는 것 보다 높은 효율성을 제공한다. 질의 최적기(query optimizer)의 구문을 분석한 결과에서 각각의 질의는 생성된 인덱스를 사용하였지만 복합 인덱스를 생성했을 때와 비클러스터 인덱스를 생성했을 때의 실행 비용에는 큰 차이가 없었다. 따라서, 복합 인덱스의 생성 가이드는 낮은 분포를 갖는 속성을 기준으로 인덱스를 구성하는 것이 효율적인 방법이다.

〈표 13〉 카테고리 2의 시험 결과

질 의	인덱스 생성 조건 1	인덱스 생성 조건 2	인덱스 생성 조건 3	비율1 (조건 2/ 조건 1)	비율2 (조건 3/ 조건 1)
질의 2-1	0.514	0.929	0.729	1.807	1.418
질의 2-2	102.532	141.435	111.701	1.379	1.089

<표 14>는 관계형 데이터베이스 시스템의 정확 매치에 대한 클러스터 인덱스 효율성을 평가하는 카테고리 3의 시험 결과를 나타낸다. 인덱스 생성 조건은 비클러스터 인덱스를 생성했을 때와 클러스터 인덱스로 생성했을 때로 분류된다. 카테고리 내의 모든 질의에서 클러스터 인덱스가 비클러스터 인덱스에 비해 전반적으로 높은 효율성을 제공한다. 따라서, 시험 데이터베이스 시스템의 정확 매치에 대한 클러스터 인덱스의 생성 가이드는 질의의 모든 질의 선택율에서 클러스터 인덱스를 생성하는 것이 효율적인 방법이다.

<표 14> 카테고리 3의 시험 결과

질 의	조건 1: 비클러스터 인덱스를 생성했을 때	조건 2: 클러스터 인덱스를 생성했을 때	비율 (조건 1/조건 2)
질의 3-1 (1%)	3.435	1.713	2.005
질의 3-2 (5%)	9.657	6.905	1.389
질의 3-3 (10%)	32.900	20.918	1.572
질의 3-4 (20%)	101.490	86.615	1.171
질의 3-5 (25%)	167.651	139.167	1.204
질의 3-6 (33%)	201.033	180.353	1.114
질의 3-7 (50%)	274.006	256.499	1.068

<표 15>는 관계형 데이터베이스 시스템의 범위 질의에 대한 클러스터 인덱스 효율성을 평가하는 카테고리 4의 시험 결과를 나타낸다. 인덱스 생성 조건은 크게 비클러스터 인덱스를 생성했을 때와 클러스터 인덱스를 생성했을 때로 분류된다. 질의의 선택율이 5%를 초과하는 질의에 대해서는 클러스터 인덱스의 효율이 비클러스터 인덱스에 비해 낮은 효율성을 제공한다. 시험 데이터베이스 시스템의 범위 질의에 대한 클러스터 인덱스의 생성 가이드는 질의의 선택율이 5% 이하일 때 클러스터 인덱스를 생성하는 것이 효율적인 방법이다.

〈표 15〉 카테고리 4의 시험 결과

질 의	조건 1: 비클러스터 인덱스를 생성했을 때	조건 2: 클러스터 인덱스를 생성했을 때	비율 (조건 1/조건 2)
질의 4-1 (1%)	3.595	1.806	1.990
질의 4-2 (5%)	17.454	9.553	1.827
질의 4-3 (10%)	21.308	22.497	0.947
질의 4-4 (20%)	92.701	104.529	0.886
질의 4-5 (25%)	117.320	138.713	0.845
질의 4-6 (33%)	150.188	176.332	0.851
질의 4-7 (50%)	221.037	251.359	0.879

<표 16>은 관계형 데이터베이스 시스템의 'NOT' 술어에 대한 비클러스터 인덱스의 효율성을 평가하는 카테고리 5의 시험 결과를 나타낸다. 인덱스 생성 조건은 크게 인덱스를 생성하지 않을 때와 비클러스터 인덱스를 생성했을 때로 분류된다. 질의의 선택율이 33%를 초과하는 질의에 대해서는 비클러스터 인덱스의 효율이 테이블 전체 스캔보다 낮았다. 질의 최적기의 구문을 분석한 결과에서 NOT을 사용한 질의는 인덱스를 사용하지만 인덱스를 생성했을 때의 예상 실행 비용과 인덱스를 생성하지 않았을 때의 예상 실행 비용은 비슷한 것으로 분석되었다. 시험 데이터베이스 시스템의 'NOT' 술어에 대한 비클러스터 인덱스의 생성 가이드는 질의의 선택율이 33% 이하일 때 인덱스를 생성하는 것이 효율적인 방법이다.

<표 17>은 관계형 데이터베이스 시스템의 'OR' 술어에 대한 비클러스터 인덱스의 효율성을 평가하는 카

테고리 6의 시험 결과를 나타낸다. 인덱스 생성 조건은 크게 인덱스를 생성하지 않을 때와 비클러스터 인덱스를 생성했을 때로 분류된다. 카테고리 내의 모든 질의에서 비클러스터 인덱스가 전체 테이블 스캔보다 낮은 효율성을 제공한다. 질의 최적기의 구문을 분석한 결과에서 OR를 사용한 질의는 인덱스를 사용하지만, 인덱스를 사용할 때와 사용하지 않을 때 예상 수행 비용은 인덱스를 사용할 때가 더 높은 것으로 분석되었다. 시험 데이터베이스의 OR 질의에 대한 사용 가이드는 인덱스를 사용하지 않는 것이 효율적인 방법이다.

〈표 16〉 카테고리 5의 시험 결과

질 의	조건 1: 인덱스를 생성하지 않았을 때	조건 2: 비클러스터 인덱스를 생성했을 때	비 율 (조건 1/조건 2)
질의 5-1 (1%)	16.365	2.863	5.716
질의 5-2 (5%)	55.324	13.226	4.182
질의 5-3 (10%)	61.836	18.552	3.333
질의 5-4 (20%)	109.660	67.839	1.616
질의 5-5 (25%)	110.802	68.569	1.615
질의 5-6 (33%)	113.129	97.968	1.154
질의 5-7 (50%)	125.654	232.412	0.540

〈표 17〉 카테고리 6의 시험 결과

질 의	조건1: 인덱스를 생성하지 않았을 때	조건2: 비클러스터 인덱스를 생성했을 때	비 율 (조건 1/조건 2)
질의 6-1 (1%)	139.570	142.764	0.977
질의 6-2 (6%)	208.422	210.325	0.990
질의 6-3 (10%)	224.534	218.521	1.027
질의 6-4 (15%)	242.986	231.917	1.047
질의 6-5 (20%)	277.911	284.273	0.977
질의 6-6 (30%)	362.972	320.434	1.132

<표 18>은 관계형 데이터베이스 시스템의 와일드카드 '%'에 대한 비클러스터 인덱스의 효율성을 평가하는 카테고리 7의 시험 결과를 나타낸다. 인덱스 생성 조건은 크게 인덱스를 생성하지 않을 때와 비클러스터 인덱스를 생성했을 때로 분류된다. 카테고리 내의 모든 질의에서 비클러스터 인덱스의 효율이 전체 테이블 스캔과 비슷하거나 낮은 효율성을 제공한다. 질의 최적기의 구문을 분석한 결과에서 %를 사용한 질의는 인덱스를 생성해도 생성된 인덱스를 사용하지 못하고 테이블 전체 스캔을 하였다. 시험 데이터베이스 시스템의 와일드카드 '%'에 대한 생성 가이드는 비클러스터 인덱스는 생성하지 않는 것이 효율적인 방법이다.

〈표 18〉 카테고리 7의 시험 결과

질 의	조건 1: 인덱스를 생성하지 않았을 때	조건 2: 비클러스터 인덱스를 생성했을 때	비 율 (조건 1/조건 2)
질의 7-1 (1%)	17.485	15.235	1.147
질의 7-2 (5%)	32.911	28.323	1.161
질의 7-3 (10%)	58.088	63.174	0.919
질의 7-4 (20%)	81.018	87.797	0.922
질의 7-5 (25%)	108.196	97.895	1.105
질의 7-6 (33%)	124.369	121.209	1.026
질의 7-7 (50%)	156.907	163.382	0.960

<표 19>는 객체관계형 데이터베이스 시스템의 클래 계층에 대한 효율성을 평가하는 카테고리 8의 시험결과를 나타낸다. 질의의 선택율이 1% 이상인 질의에 대해서는 객체관계형 SQL의 효율성이 비객체지향 SQL과 비슷하거나 낮은 효율성을 제공한다. 시험 데이터베이스 시스템의 클래스 계층에 대한 사용 가이드는 질의가 하나의 행을 검색할 때 객체관계형 SQL질의를 사용하는 것이 데이터베이스 시스템에 효율적인 방법이다.

<표 20>은 객체관계형 데이터베이스 시스템의 집합 자료형에 대한 효율성을 평가하는 카테고리 10의 시험 결과를 나타낸다. 카테고리 내의 모든 질의에서 객체관계형 질의가 관계형 질의보다 높은 효율성을 제공한다. 질의 최적기(query optimizer)의 구문을 분석한 결과에서 객체관계형 질의는 생성된 인덱스를 사용하였고 관계형 질의는 SET 테이블과의 조인연산을 수행할 때 인덱스를 사용하지 못하고 테이블 전체 스캔을 하였다. 시험 데이터베이스 시스템의 집합 자료형에 대

한 질의 가이드는 객체관계형 SQL 질의를 쓰는 것이 데이터베이스 시스템에 효율적인 방법이다.

〈표 19〉 카테고리 8의 시험 결과

질 의	관계형 SQL	객체관계형 SQL	비율(관계형 SQL/ 객체관계형 SQL)
질의 8-1	0.314	0.142	2.211
질의 8-2 (1%)	9.837	11.941	0.823
질의 8-3 (5%)	23.565	23.559	1.000
질의 8-4 (10%)	25.477	26.600	0.957
질의 8-5 (20%)	27.860	29.582	0.941
질의 8-6 (25%)	42.294	31.377	1.347
질의 8-7 (33%)	28.858	44.245	0.652
질의 8-8 (50%)	78.750	90.050	0.874

〈표 20〉 카테고리 10의 시험 결과

질의	관계형 SQL	객체관계형 SQL	비율(관계형 SQL/ 객체관계형 SQL)
질의 10-1 (10%)	1825.488	612.580	2.982
질의 10-2 (20%)	2361.138	748.617	3.514
질의 10-3 (30%)	2774.686	863.044	3.215
질의 10-4 (40%)	3392.859	1049.782	3.232
질의 10-5 (50%)	4549.642	1215.507	3.473

현재 시험하는 데이터베이스 시스템에서 카테고리 9 질의를 지원하지 않아 실험에서 제외하였다.

5. 결 론

일반 사용자나 데이터베이스 관리자가 특정 데이터베이스 시스템의 높은 성능을 유지하기는 쉽지 않으며, 데이터베이스 튜닝을 위해서는 많은 지식이 요구된다. 본 논문은 데이터베이스 시스템에서 높은 성능을 유지할 수 있는 특성을 분석해주는 튜닝도구를 제안하였다. 관계형 데이터베이스 시스템에 대한 튜닝도구는 인덱스의 생성 관계와 질의 선택율에 의거하며, 시스템 대한 성능상의 특징을 효과적으로 분석하여 인덱스 생성 가이드로서 제공된다. 객체 관계형 데이터베이스 시스템에 대한 튜닝도구는 객체 지향 질의와 비객체지향 질의에 대해 비교, 분석하여 질의 형태에 따른 객체 기능 효율성을 분석하는 가이드로서 제공된다. 튜닝도구는 총 10개의 카테고리에서 65개의 시험질의를 제공하고 있다. 본 시험 도구는 특정 시스템에 적용되었으며, 그 시험 결과도 기술되어 있다.

향후 과제로서는, SQL 문장에 대한 튜닝을 확장하여 범용(global)튜닝을 위한 튜닝도구 개발과 객체관계형 시스템의 질의군의 확장이 요구된다.

참 고 문 헌

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

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

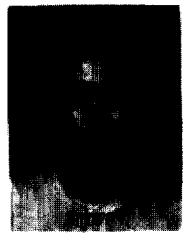
[3] S. Chauduri, E. Christensen, G. Graefe, V. Narasayya, and M. Zwilling, Self-Tuning Technology in Microsoft SQL Server, Data Engineering, Vol.22, No.2, 20-26, 1999.

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

[5] M. Asgarian, M. J. Carey, D. J. DeWitt, J. Gehrke, J. F. Naghton, and D. N. Shah, The BUCKY Object-Relational Benchmark, Proc. of the ACM SIGMOD Conference, 1997

[6] J. M. Hellerstein and M. Stonbraker, Predicate Migration : Optimizing Queries with Expensive Predicates, Proc. of the ACM SIGMOD Conference, 1993

[7] Informix Inc., Informix Performance Guide, 1997.



안 기 덕

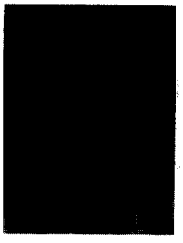
e-mail : kdan@daou.co.kr

1998년 숭실대학교 인공지능학과 졸업

1998년~2000년 숭실대학교 대학원 컴퓨터학과(석사)

2000년~현재 다우기술(주) 연구원

관심분야 : 데이터베이스 성능평가, 튜닝



오 정 석

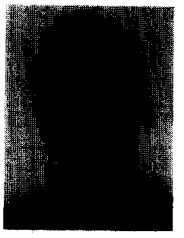
e-mail : dbstar@orion.ssu.ac.kr

1996년 서경대학교 정보처리학과 졸업(학사)

1998년 숭실대학교 대학원 컴퓨터학과(석사)

1998년~현재 숭실대학교 컴퓨터학과 대학원 박사과정

관심분야 : 인터넷 데이타베이스, 데이타베이스 시스템 성능평가, 정보검색



이 상 호

e-mail : shlee@computing.soongsil.ac.kr

1984년 서울대학교 컴퓨터공학과 (학사)

1986년 미국 노스웨스턴 대학교 전산학과(석사)

1989년 미국 노스웨스턴 대학교 전산학과(박사)

1990~1992년 한국전자통신연구원 선임연구원

1992~현재 숭실대학교 컴퓨터학부 부교수

1999~2000년 미국 George mason 대학교 교환 교수

관심분야 : 인터넷 데이타베이스, 데이타베이스 성능평가, 트랜잭션 처리