CUBRID Co., Ltd.

CUBRID page 방법

Version

1.0 2007-06-11

(주)큐브리드

135-010 서울시 강남구 논현동 240-10 중앙빌딩 3층. ☏ 02-2015-0600 http://www.cubrid.com

목 차

1. CUBRID에서 page기능 사용하기

1.1 개요

본 문서에서는 다양한 페이지 기법 중 CUBRID를 이용한 최적의 페이지 기법을 전달하고자 한다.

1.1.1 테스트 환경

본 문서에서는 CUBRID 7.0을 이용하여 테스트 한다.

1.1.2 테스트 테이블

본 테스트는 CUBRID 7.0을 설치할 경우 테스트 데이터베이스로 생성이 가능한 subway 데이터베이스 중 running 테이블을 이용한다. running 테이블의 스키마 정보는 아래와 같다.

```
<Class Name>
    running
 <Attributes>
    run_id              : integer NON_NULL
    station_id          : integer
    start_station_id    : integer
    arrival_station_id  : integer
    first_final         : character(4)
    day_of_week         : character varying(6)
    start_time          : character(5)
 <Constraints>
    UNIQUE pk_running ON running(run_id).
```

1.2 CUBRID 페이지 방법

1.2.1 index 와 rownum을 이용한 paging 방법

CUBRID에서는 index와 rownum을 이용한 page방법을 제공하고 있다. 정렬을 위하여 order by를 사용할 경우, 일단 검색된 결과를 정렬 후 그 중 해당 범위(예, rownum < 16)를 검색하게 됨으로 결과 값을 추출하는 것이 느려지게 된다. 따라서 이와 같은 문제를 해결하기 위해서 여러 가지 방법들이 DBMS 별로 제공되고 있으며 CUBRID의 경우 인덱스생성과 rownum을 이용하여 빠른 검색이 가능하도록 하고 있다. 즉 인덱스를 사용하여 검색을 할 경우 검색결과는 인덱스의 순서대로 나오게 되므로 order by 를 이용한 정렬을 피할 수 있으며, 또한 검색결과의 임의 위치에서 임의의 개수를 가져올 수 있는 rownum을 이용하여 편리하게 페이징을 할 수 있다. CUBRID에서의 rownum은 between 11 and 20 과 같이 사용이 가능하다

또한 order by 를 이용하여 정렬시 역순정렬(desc)의 경우는 reverse index를 설정하면 별도의 order by 작업 없이 정렬이 가능하다.

위의 모든 경우에서 index를 반드시 사용해야 하므로 다음의 사항을 반드시 확인해야 한다.

- sqlx.init($CUBRID/admin 경로에서 확인)파일에 parameter 중 index_scan_order=1 로 설정해야 한다.
- index 사용이 가능하도록 where절에 조건에 들어가야 하며 index를 강제로 사용하게 하는 using index 구문을 사용해야 한다.
- running 테이블을 이용한 사용 예는 아래와 같다.

일반적인 사용 방법

```
select run_id
 from (select rownum rnum, run_id from running
          order by id desc) t(rnum, run_id)
 where run_id between 20 and 30
```

CUBRID 를 이용한 방법

```
 select id from running
  order by id for orderby_num() between 20 and 30
```

CUBRID 를 이용하여 튜닝한 방법

1. **Reverse 인덱스 생성(run_id의 값 큰경우를 최근 값으로 봤을 경우):**

```
create reverse index rev_run_id on running(run_id)
```

2. **가장 최근에 기록된 데이터 중 상위 20번째와 30번째사이에 해당하는 값을 가져올 때:**

```
select rownum, run_id  from running
where run_id > 0 and rownum between 20 and 30 using index rev_run_id(+)
```

➔ using index로 rev_run_id 인덱스를 사용함으로 desc 정렬이 필요없이 가장 최근의 값을 가져올 수 있다. 또한 between절을 이용하여 20번째부터 30번째에 해당하는 값만 가져올 수 있었으며 속도는 order by run_id desc 를 이용하는 것 보다 빠르다.

1.2.2 Partition을 이용한 검색 방법

많은 양의 데이터가 게시판에 쌓이게 될 경우 위와 같은 검색으로만 원하는 속도를 내기에 부족할 수 있다. 대용량 테이블의 검색을 용이하게 할 수 있도록 CUBRID에서는 partition 기능을 제공한다.

다음은 partition을 이용하는 예를 설명하도록 한다.

게시판으로 아래와 같은 테이블을 생성한다:

```
create class bbs (
seq_no int primary key,
title varchar(50) not null,
content varchar(2000))
```

➔ 게시판의 번호(seq_no), 제목(title), 내용(content), 구분(section_num)을 기본적으로 생성한다.

Partition을 나누는 방법:

```
alter table bbs
partition by range seq_no) (                  ➔ seq_no의 범위를 기준으로 RANGE분할을 한다.
  partition p0 values less than(1000)         ➔ seq_no가 1000보다 작은 것으로 분할하고
  partition p1 values less than(2000)         ➔ seq_no가 1000과 2000사이의 값으로 분할하고
  partition p2 values less than MAXVALUE)     ➔ 나머지 값을 분할 한다.
```

) ➔ 이렇게 분할 할 경우 해당 seq_no에 해당하는 값은 해당 partition에서만 검색을 하게 되기 때문에 속도의 향상을 가져오게 된다. CUBRID는 RANGE, LIST, HASH와 같이 3가지의 partition방법을 제공하고 있으며, 테이블 구설 설계에 따라 각 방법을 참조하기 바란다.

Partition된 테이블의 검색 방법은 일반 테이블과 동일 하다. where절에 조건에 seq_no을 조건으로 검색할 경우 해당 할당 partition에서 검색이 진행되어 검색의 범위가 줄게 되어 빠르게 검색할 수 있다.