58. RAC 환경 튜닝 팁 2 - 파티션 테이블 사용하기

RAC 환경에서 파티션 테이블을 사용하게 되면 파티션 와이즈 조인을 할 수 있다.

파티션 와이즈 조인: 파티션끼리 조인하는 것

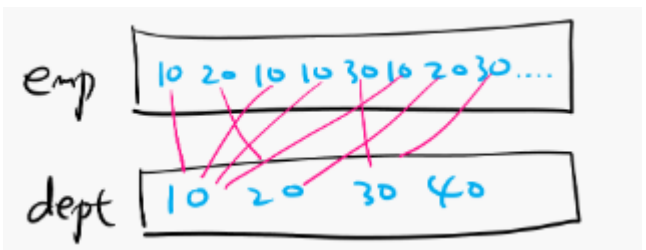
- 일반 heap 테이블을 서로 조인 했을 때의 원리 (왼쪽)

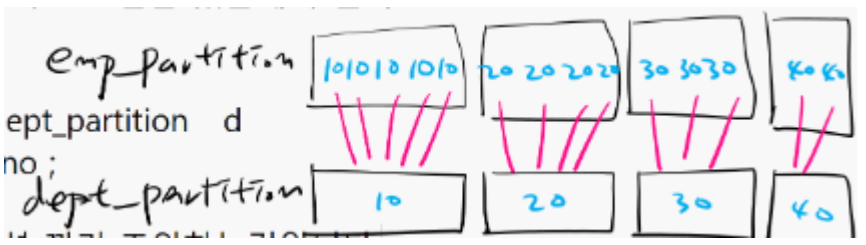
```
select e.ename, d.loc
from emp e, dept d
where e.deptno=d.deptno;
```

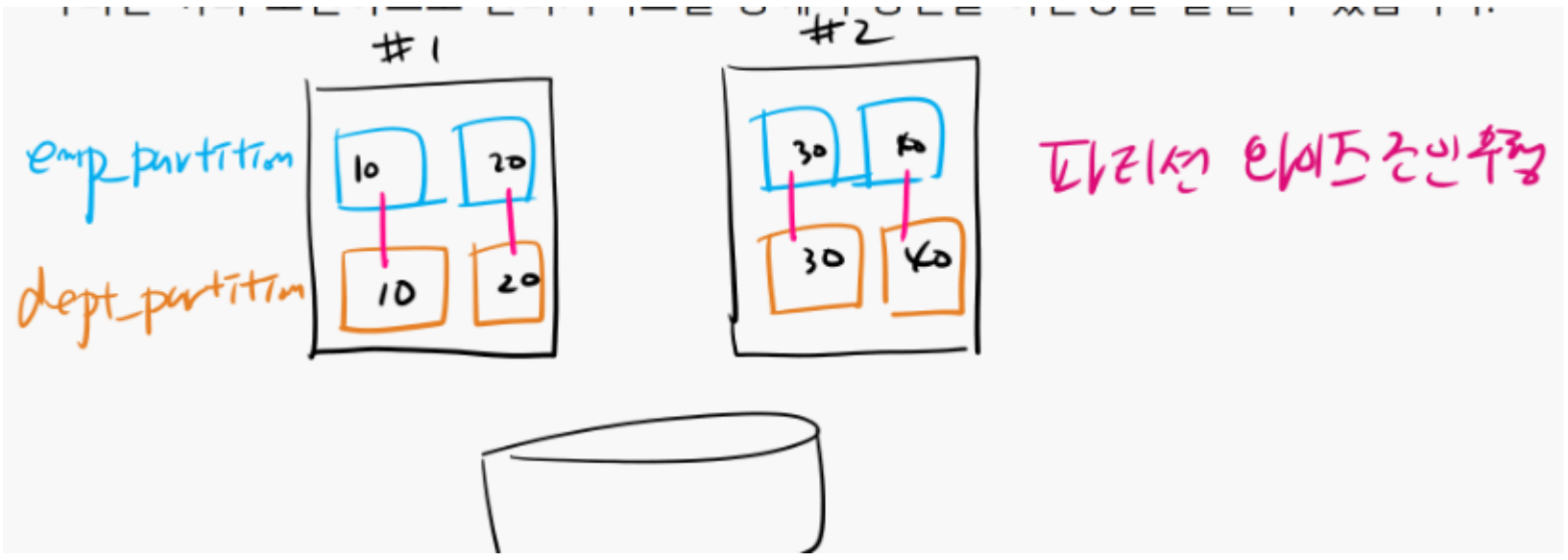
- 파티션 테이블로 파티션 와이즈 조인을 했을 때의 원리 (오른쪽)

```
select e.ename, d.loc
from emp_partition e, dept_partition d
where e.deptno=d.deptno;
```

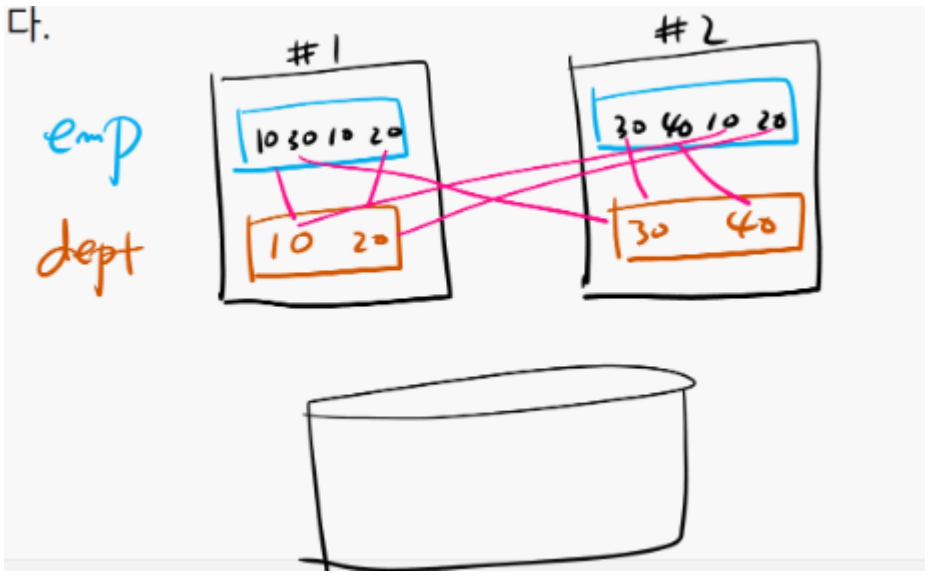
RAC 환경에서 파티션 와이즈 조인을 하게 되면 성능상 유리한 이유
=> 파티션끼리 조인하므로 interconnect를 통해서 통신을 하는 양을 줄일 수 있다.







만약 일반 Heap 테이블로 구성했다면 다음과 같이 interconnect를 통한 통신이 불가피하다.



실습
1. emp 테이블을 부서번호를 기준으로 파티션 테이블로 구성한다.

```
(sql 1 scott)
create table emp_partition
(empno number(4,0) ,
ename varchar2(10),
job varchar2(9),
mgr number(4,0),
hiredate date,
sal number(7,2),
comm number(7,2),
deptno number(2,0)
)
partition by range(deptno)
(partition p_deptno_10 values less than(20),
partition p_deptno_20 values less than(30),
partition p_deptno_30 values less than(40),
partition p_max values less than(maxvalue));

insert into emp_partition
select * from emp;

select ename, sal, deptno from emp_partition;
```

=> 부서번호 별로 모여있다.

```
racdb1(SCOTT) >  select ename, sal, deptno from emp_partition;

ENAME             SAL     DEPTNO
---------- ---------- ----------
KING             5000         10
CLARK            2450         10
MILLER           1300         10
JONES            2975         20
FORD             3000         20
SMITH             800         20
SCOTT            3000         20
ADAMS            1100         20
BLAKE            2850         30
MARTIN           1250         30
ALLEN            1600         30

ENAME             SAL     DEPTNO
---------- ---------- ----------
TURNER           1500         30
JAMES             950         30
WARD             1250         30

14 rows selected.
```

2. dept 테이블을 부서번호를 기준으로 파티션 테이블로 구성한다.

```
create table dept_partition
(deptno number(10),
dname varchar2(14),
loc varchar2(13))
partition by range(deptno)
(partition p_deptno_10 values less than(20),
partition p_deptno_20 values less than(30),
partition p_deptno_30 values less than(40),
partition p_max values less than(maxvalue));

insert into dept_partition
select * from dept;
```

```
3. emp_partition과 dept_partition 테이블에 대해서 통계 정보를 수집한다.
exec dbms_stats.gather_table_stats('scott','emp_partition');
exec dbms_stats.gather_table_stats('scott','dept_partition');

4. emp와 dept 테이블을 파티션 와이즈 조인하고 실행 계획을 본다.
explain plan for
select e.ename, d.loc
from emp_partition e, dept_partition d
where e.deptno=d.deptno;

select * from table(dbms_xplan.display);

=> 실행 계획에 key가 출력 되는데 key 기반으로 필요한 파티션에 대해서만 접근해서 파티션 와이즈
     조인을 했다.
```

```
racdb1(SCOTT) >  select * from table(dbms_xplan.display);

PLAN_TABLE_OUTPUT
--------------------------------------------------------------------------------
Plan hash value: 3969825873

--------------------------------------------------------------------------------
----------------------------

| Id  | Operation                | Name           | Rows  | Bytes | Cost (%CPU)
| Time     | Pstart| Pstop |

--------------------------------------------------------------------------------
----------------------------


PLAN_TABLE_OUTPUT
--------------------------------------------------------------------------------
|   0 | SELECT STATEMENT         |                |    14 |   280 |    90   (0)
| 00:00:02 |       |       |

|   1 |  NESTED LOOPS            |                |    14 |   280 |    90   (0)
| 00:00:02 |       |       |

|   2 |   PARTITION RANGE ALL    |                |     4 |    44 |    52   (0)
| 00:00:01 |     1 |     4 |

|   3 |    TABLE ACCESS FULL     | DEPT_PARTITION |     4 |    44 |    52   (0)
| 00:00:01 |     1 |     4 |

PLAN_TABLE_OUTPUT
--------------------------------------------------------------------------------

|   4 |   PARTITION RANGE ITERATOR|                |     4 |    36 |     9   (0)
| 00:00:01 |   KEY |   KEY |

|*  5 |    TABLE ACCESS FULL     | EMP_PARTITION  |     4 |    36 |     9   (0)
| 00:00:01 |   KEY |   KEY |

--------------------------------------------------------------------------------
----------------------------
```