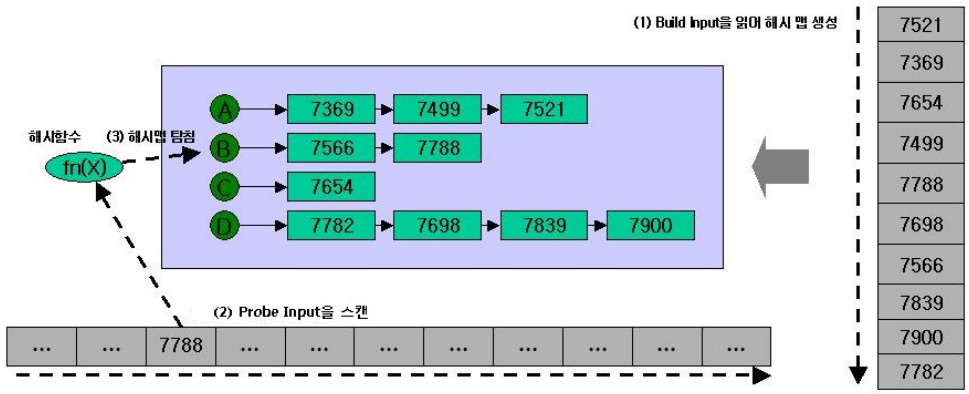
03 해시조인

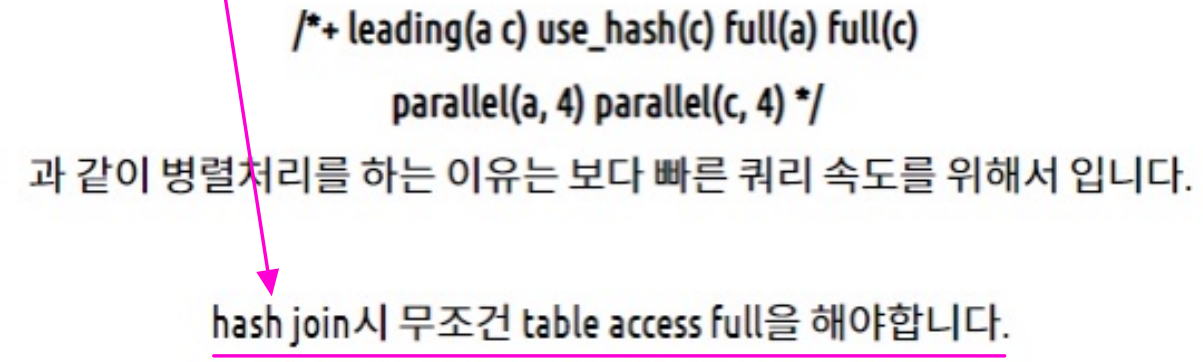
(1) 기본 메커니즘

- 소트머지조인과 NL조인이 효과적이지 못한 상황에 대한 대안으로서 개발되었다.
- 조인대상이 되는 두 집합 중에서 작은 집합(Build Input)을 읽어 Hash Area에 해시 테이블을 생성하고, 큰 집합(Probe Input)을 읽어 해시 테이블을 탐색하면서 조인하는 방식이다.



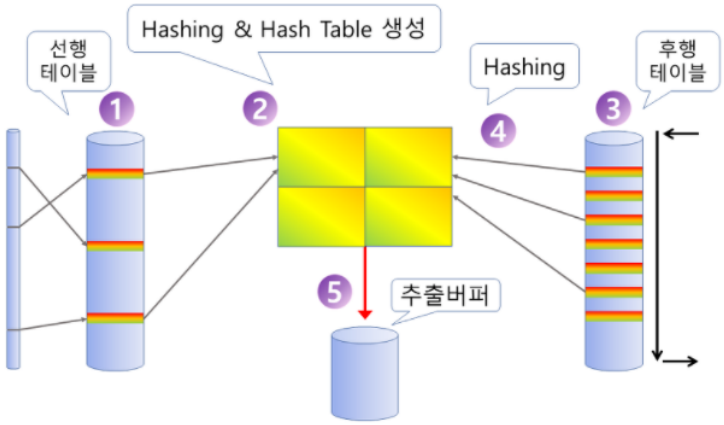
작은 집합(Build Input)을 읽어 해시맵 생성
=> 해시테이블을 생성할 때 해시함수(fn(X))를 사용하고, 해시함수에서 리턴받은 버킷주소(A,B,C,D)로 찾아가 해시체인에 엔트리를 연결한다.
큰 집합(Probe Input)을 스캔
=> 해시테이블을 탐색할 때도 해시함수(fn(X))를 사용하며, 해시함수에서 리턴받은 버킷주소(A,B,C,D)로 찾아가 해시체인을 스캔하면서 데이터를 찾는다.

- 장점: NL조인처럼 조인시 발생하는 Random엑세스 부하가 없다.
  소트머지조인처럼 조인전에 양쪽 집합을 정렬해야 하는 부담이 없다.
- 단점: 해시테이블을 생성하는 비용이 수반
  그러므로, Build Input이 작을 때 효과적이다.(PGA에 할당되는 Hash Area에 담길정도로 충분히 작아야 함)
- 해시키 값으로 사용되는 컬럼에 중복값이 거의 없을 경우 효과적이다.(주후 설명)
  Inner루프로 Hash Area에 생성해둔 해시테이블을 이용한다는 것 외에 NL조인과 유사하다.
  해시테이블 만들 때는 전체범위처리가 불가피하나, Prob Input을 스캔하는 단계는 부분범위 처리가능하다.
  해시조인은 해시테이블이 PGA영역에 할당되므로, NL조인보다 빠르다.
  NL조인은 Outer테이블에서 읽히는 레코드마다 Inner쪽 테이블 버퍼캐시 탐색을 위해 래치획득을 반복하나, 해시조인은 래치 획득과정없이 PGA에서 빠르게 데이터를 탐색할 수 있다.



4) Hash Join은 조인 작업을 수행하기 위해 '해쉬 테이블'을 메모리에 생성한다. → Hash Table의 크기가 메모리에 적재할 수 있는 크기보다 더 커지면 임시 영역(디스크)에 해쉬 테이블을 저장함. 그렇기 때문에 Hash Join을 할 때는 결과 행의 수가 적은 테이블을 선행 테이블로 사용하는 것이 좋다. 선행 테이블의 결과를 완전히 메모리에 저장할 수 있다면 임시 영역에 저장하는 작업이 발생하지 않기도 하고 CPU연산을 조금 덜 수행할 수 있기 때문

1) 다음은 Hash Join의 수행 방식을 단계별로 나타낸 것입니다.



① 선행 테이블에서 주어진 조건을 만족하는 행을 찾는다.
② 해당 행들에 대해서, 선행 테이블의 조인 키(칼럼)를 기준으로 Hash 함수를 적용하여 해쉬 테이블을 생성.
③ 후행 테이블에서 주어진 조건을 만족하는 행을 찾는다.
④ 해당 행들에 대해서, 후행 테이블에 Hash 함수를 적용하여 선행 테이블의 해쉬 테이블에서 맞는 버킷을 찾음.
⑤ JOIN을 수행 & 조인에 성공하면 추출버퍼에 넣는다.
⑥ 후행 테이블의 조건을 만족하는 모든 행에 대해서 3~5번 반복.

해당 버킷에는 선행 테이블 내 로우가 들어있음

2) 해쉬 조인(Hash Join)을 수행할 테이블의 조인 칼럼을 기준으로 Hash 함수를 적용하여 서로 동일한 Hash 값을 갖는 것들 사이에서 실제 값이 같은지를 비교하면서 조인을 수행하는 방식이다.

해쉬 함수를 적용한 실제 값은 어떤 값으로 해쉬(Hashing)될 지 예측할 수 없다. 하지만 해쉬 함수가 적용될 때 동일한 값은 항상 같은 값으로 해쉬됨이 보장된다. 하지만 해쉬 함수를 적용할 때 보다 큰 값이 항상 큰 값으로 해쉬되고 작은 값이 항상 작은 값으로 해쉬된다는 보장은 없다. 그렇기 때문에 Hash Join은 동등 조인(Equi join)에서만 사용할 수 있다는 특징이 있다.

Hash Join – Definition

Consits of build phase and probe phase

Build Phase:

- For each row in table1, calculate hash value of equi join columns
- Store row in a hash table, use calculated hash as key

Probe Phase:

- For each row in table2, calculate hash value of equi join columns
- Check if hash match in hash table, if so check actual columns

→ Output hash match