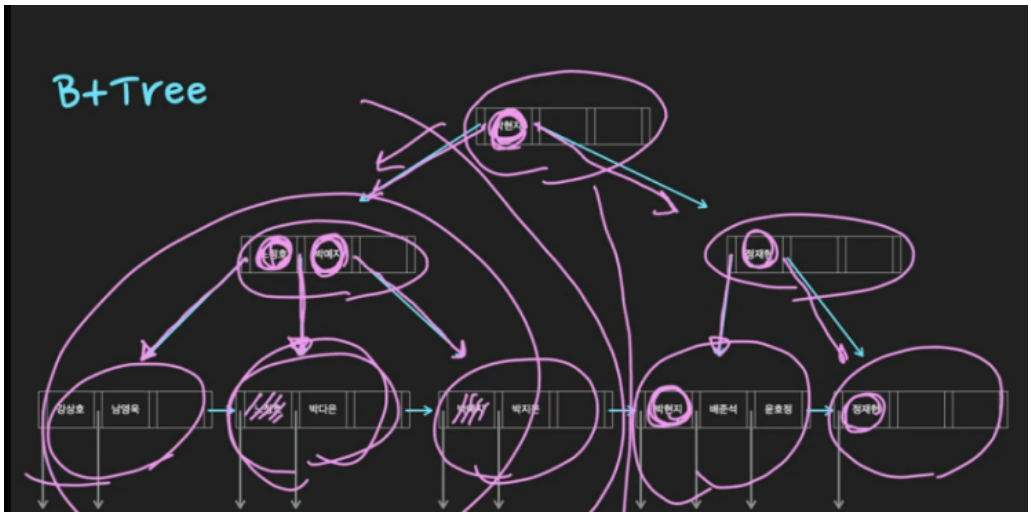
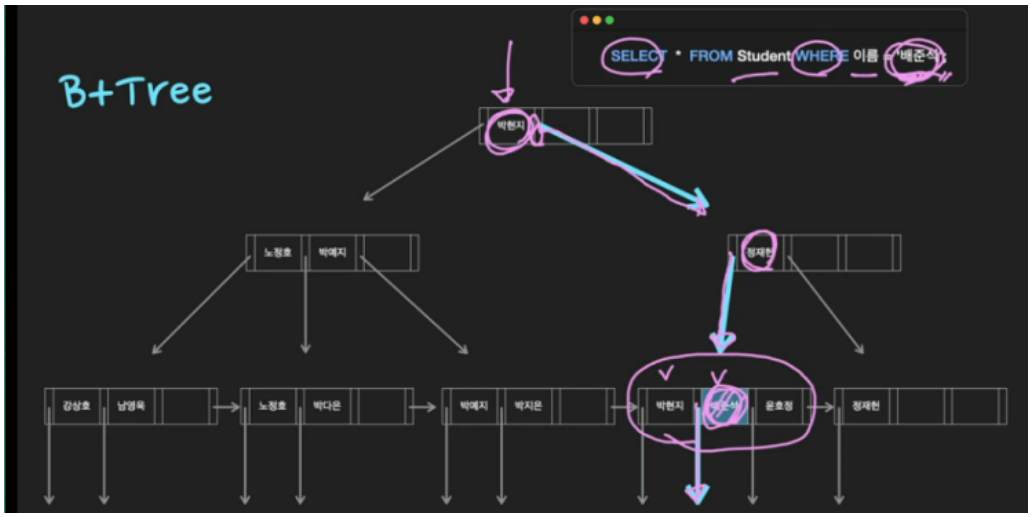
● 데이터를 검색을 할 때 hash table의 시간복잡도는 O(1)이고 b+tree는 O(logn)으로 더 느린데 왜 index는 hash table이 아니라 b+tree로 구현되나?

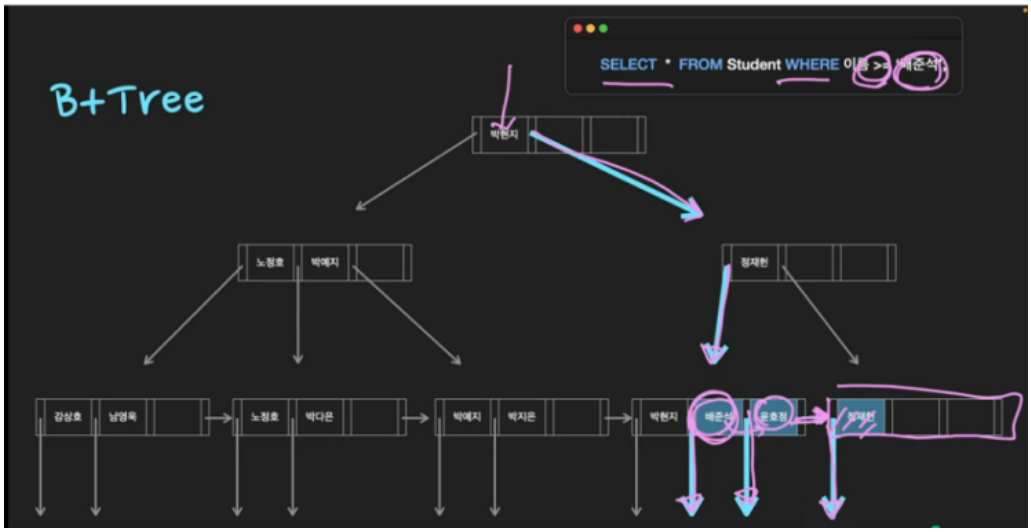
▶ 하나의 데이터만 탐색하는 시간은 Hash table이 O(1)으로 b+tree(의O(logn))보다 더 빠르지만, Hash table은 데이터의 값이 정렬되어 있지 않기 때문에, 부등호를 사용하는 query에 대해서는, 즉, 일정 범위의 값들을 찾을 때는 매우 비효율적이게 되기 때문에, 데이터를 정렬해서 저장하는 b+tree를 이용하여 index를 구현합니다.

▶ B+three가 DB index를 위한 자료구조로 적합한 이유는, 첫째, 항상 정렬된 상태를 유지하여 부등호 연산에 유리하기 때문입니다. 그리고 둘째, 데이터 탐색 뿐 아니라, 저장, 수정, 삭제 시에도, 항상 O(logN)의 시간 복잡도를 갖기 때문입니다. 반면, hash index는 빠든 데이터 검색이 필요할 때 유용하지만, index로써 hash index가 사용되는 경우는 제한적인 이유가, hash index는 등호(=)연산에만 특화되어 있기 때문입니다. 데이터가 조금이라도 달라지면 hash function은 완전히 다든 hash 값을 생성하는 특징 때문에, 부등호 연산(>, <)이 자주 사용되는 DB 검색에는 hash index가 적합하지 않습니다.



B+ Three (Balanced three 균형 트리)





Resources: inflearn 개발남 노씨