Database Index + Cardinality

데이터베이스 인덱스의 정의 및 효용성

RDBMS 및 Entity에서의 인덱스 설정

Index ?

= 색인

사전적 정의

원 정보 내용을 적절히 나타내는 정보를 추출하고,

원 정보 위치를 가리키는 참조 정보와 함께 나타낸 것

예시

이름, 주소, 주민번호 등

Index ?

데이터베이스 내 정의

데이터베이스 테이블에 대한 검색 성능의 속도를 높여주는 자료구조

정렬 후 별도의 메모리 공간에 해당 데이터의 물리적 주소를 저장 (K/V)

어떻게 검색 성능을 개선하는가?

1. 데이터베이스 관련 속도 저하는 주로 조건문(where) 절에서 발생
2. 테이블 생성 후 데이터는 내부적으로 별도의 정렬 없이 저장됨
3. 따라서 조건부 질의 시 내부 데이터 전체를 조회함 (Full Table Scan)
4. Index를 설정하면 데이터 저장 단계에서 정렬됨

Cardinality / Selectivity

데이터베이스 내 정의

Selectivity : 데이터 집합 내 값을 얼마나 특정할 수 있는지에 대한 지표

Cardinality / Total number of records (range: 0 ~ 1)

Cardinality : 특정 데이터 집합 내 유일성의 척도

select count(distinct ({ COLUMN })) from { TABLE };

예시

회원 수가 40명인 경우

주민번호 : Cardinality 40 / Selectivity : 1

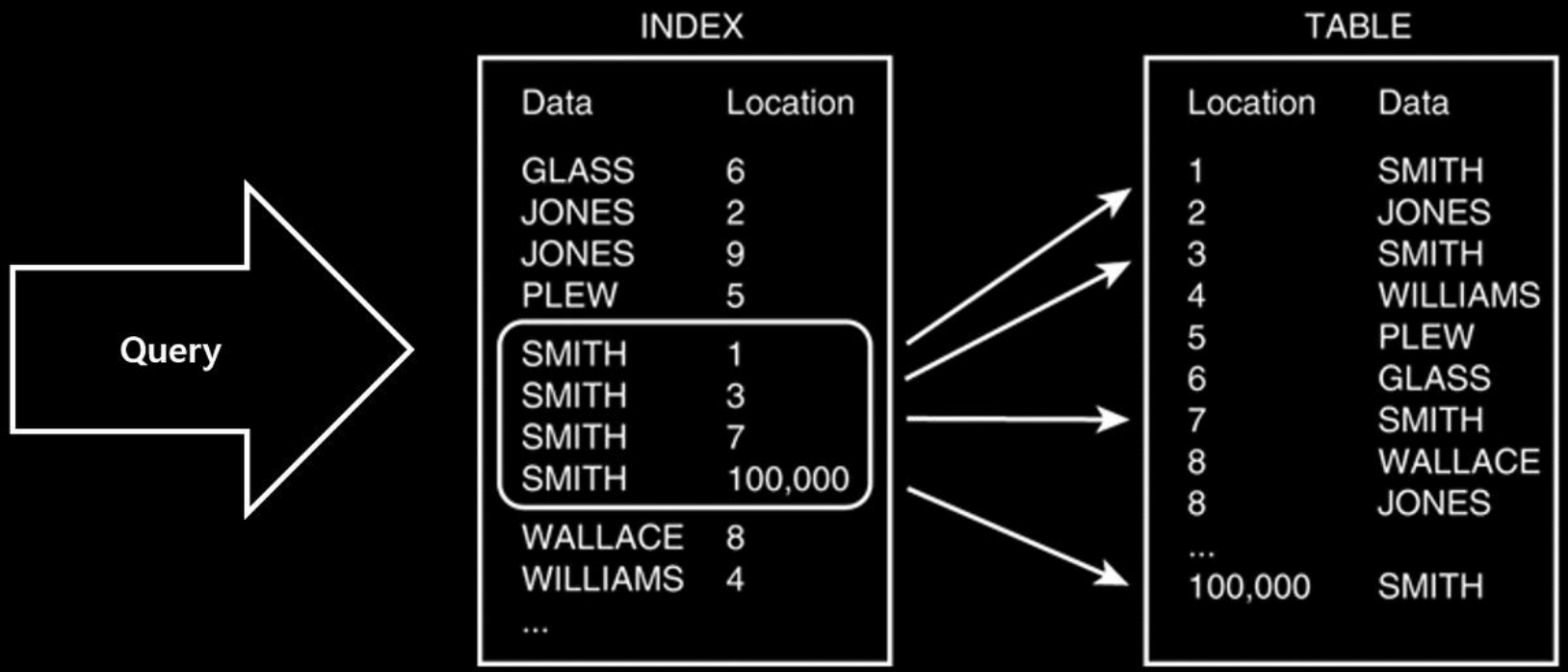
성별: Cardinality 2 / Selectivity : 2/40 = 0.05

Index 설정의 효용성(1)

Full Table Scan 방지

Location 8에 해당되는 값을 찾기 위해 테이블 내 데이터 전체 조회

정렬된 WALLACE, JONES를 조회 후 질의 종료



Index 설정의 효용성(2)

Order by (Sort)에 의한 부하를 방지

1. Order by는 1차적으로 메모리에서 정렬 실행
2. 메모리를 초과하는 경우 디스크 I/O가 추가적으로 발생
3. 디스크를 경유한 입출력은 메모리를 통한 입출력보다 10,000배 이상 느림
4. 디스크 I/O가 발생하면 서버 프로세스가 디스크에서 원하는 블록을 추출
5. 블록 추출 과정에서 경합이 발생하여 총 질의 시간이 지연됨

M IN / MAX의 효율적인 처리

1. 대상 값이 이미 정렬되어 있어 첫 값과 끝 값만 가져오면 됨
2. 테이블 전체를 Full Scan하지 않아도 되어 질의 속도 향상 효과

Index 설정의 고려 사항(1)

PK, Unique 컬럼은 index가 자동으로 설정된다.
1. 별도의 메모리 공간을 필요로 하므로 추가 저장 공간이 필요하다.
2. DML에 대한 개별적인 설정에 따른 성능 저하가 발생한다.*
3. 데이터의 중복도가 높은(낮은 Cardinality)의 컬럼은 인덱스 효율이 낮다.

DM L에 대한 개별적 설정
1. INSERT : 새로운 데이터에 대한 인덱스 추가
2. DELETE : 삭제하는 데이터의 인덱스를 사용하지 않도록 설정
3. UPDATE : 기존 인덱스의 비활성 처리, 갱신된 데이터 인덱스 추가

UPDATE, DELETE가 발생하면 기존 인덱스는 UNUSABLE 처리된다.
즉, 기존 인덱스가 삭제되지 않고 잔류하므로 성능적 이슈를 야기한다.

Index 설정의 고려 사항(2)

인덱스가 존재하나 질의 시 인덱스를 타지 않는 경우
1. 외부적인 가공 또는 연산, 묵시적 형 변환 데이터를 질의하는 경우*
2. 부정형 연산자를 사용하는 경우 (NOT, NOT IN, !=)
3. IS NULL, IS NOT NULL을 사용하는 경우
4. Like 질의 시 대상 문자열의 좌변에 와일드 카드(%)이 삽입된 경우

왜?
1. 질의문의 처리 순서는 F"W"GH"S"O 이므로 조회 이전 가공이 선실행됨
2. 부정형 조건 검색은 넓은 범위의 값이 도출되므로 FTS이 실행됨
3. B+TREE 구조의 인덱스에는 NULL 여부가 저장되지 않음
4. 대상 조건의 첫 번째 값을 모른다면 인덱스 참조가 불가함

Summary

인덱스를 사용해야 하는 경우

1. 데이터의 내부/외부 가공이 필요하지 않으며 조회 빈도가 높은 경우
2. DML이 자주 일어나지 않는 경우
3. 검색 결과 데이터의 범위가 적은 경우 (2~4% 권장, 15% 이내)
4. 테이블의 행의 개수가 많고 중복도가 낮은 경우
5. 오름차순 또는 내림차순의 정렬이 필요한 경우 (Order by 대체)

왜?

1. 조회 결과의 범위가 전체 데이터 중 다수를 차지하면 FTS이 유리함
2. DML이 자주 발생하는 경우 인덱스의 수정 또한 빈번히 일어 남

Index 생성과 설정 (Database : Oracle / InnoDB)

Create Index

```
CREATE INDEX { INDEX-NAME } ON { TABLE-NAME }( { COLUMN-NAME } )
```

Rebuild Index

```
ALTER INDEX { INDEX-NAME } REBUILD; / ANALYZE TABLE { TABLE-NAME }
```

Rename Index

```
ALTER INDEX { INDEX-NAME } RENAME TO { NEW-INDEX-NAME }; / -
```

Disable Index

```
ALTER INDEX { INDEX-NAME } UNUSABLE;
```

Drop Index

```
DROP INDEX { INDEX-NAME };
```

Index 생성과 설정 (JPA)

Create Index

@Entity
@Table(name = { COLUMN-NAME }, indexes = {
@Index(name = { INDEX-NAME }, columnList = { COLUMN1, COLUMN2 ··· }, unique = { T/F })
})

```
@Table(name = "twins", indexes = { @Index(name = "IDX_TWINS_KOR-TITLE", columnList = "kor_title"),
                                   @Index(name = "IDX_TWINS_ENG-TITLE", columnList = "eng_title") })
@JsonIdentityInfo(generator = ObjectIdGenerators.PropertyGenerator.class, property = "id")
public class Twins extends UserDateAudit {
```

```
2022-02-03 02:40:36.504 DEBUG 1056 --- [  restartedMain] org.hibernate.SQL                        : create index "IDX_TWINS_KOR-TITLE" on "twins" ("kor_title")
[Hibernate] create index "IDX_TWINS_KOR-TITLE" on "twins" ("kor_title")
2022-02-03 02:40:36.505 DEBUG 1056 --- [  restartedMain] org.hibernate.SQL                        : create index "IDX_TWINS_ENG-TITLE" on "twins" ("eng_title")
[Hibernate] create index "IDX_TWINS_ENG-TITLE" on "twins" ("eng_title")
```

Reference

SQL Index Maintenance

https://www.sqlshack.com/sql-index-maintenance/

Disc I/O and SQL Performance

https://www.sqlshack.com/sql-server-monitoring-tools-for-disk-i-o-performance/

Disc Access Ordering

https://pkolaczk.github.io/disk-access-ordering/

Database Memory vs Disc I/O

https://12bme.tistory.com/330

인덱스를 사용하지 못 하는 경우

https://brightestbulb.tistory.com/145

Look up to list of table indexes with data dictionary (MariaDB)

https://dataedo.com/kb/query/mariadb/list-table-indexes

인덱스 관련 쿼리 모음(Oracle)

http://dbcafe.co.kr/wiki/index.php/ORACLE_%EC%9D%B8%EB%8D%B1%EC%8A%A4