1.1 SQL 파싱과 최적화

핵심

- 옵티마이저가 SQL 을 어떻게 처리하는지
- 서버 프로세스는 데이터를 어떻게 읽고 저장하는지

옵티마이저

▼ SQL은 무엇의 줄임말일까?

Structured Query Language

즉 **set-based**, **structured** 하다는 것 (집합적, 선언적)

원하는 결과집합을 구조적, 집합적으로 선언하지만

내부적으로 각 기능이 어떻게 구현되어 있는지는

SQL 을 쓰는 입장에서는 알 필요가 없다

원하는 결과 (데이터를 뽑겠지?) 를 구조적, 집합적으로 선언하지만

그 결과집합을 만드는 과정은 절차적일 수밖에 없는데 (마치 C 같은 언어처럼)

그런 절차적인 프로세스를 처리해주는 **프로시저**가 바로 **옵티마이저**이다!

DBMS 내부 엔진 중 하나인 **옵티마이저**가 데이터로 가는 길을 찾아 주는 네비게이션인 것

> 💡 절차적 프로그래밍 기능을 사용할 수 있는 SQL 이 존재하기는 함.
> 오라클 PL/SQL, SQL Server T-SQL

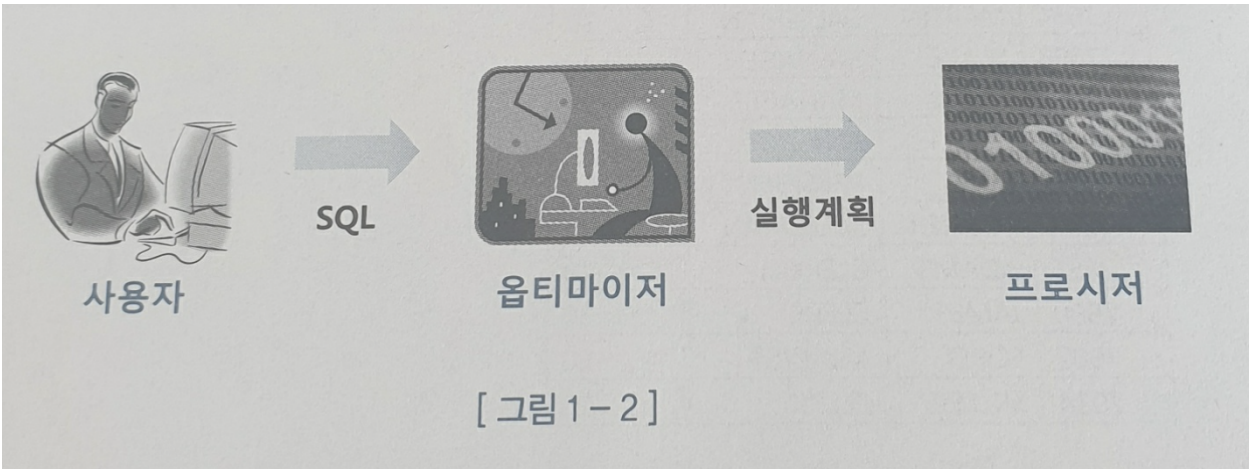
SQL 최적화

DBMS 내부에서 (SQL 사용자인 우리가 하지 않는다!)

프로시저를 작성하고 컴파일해서 실행 가능한 상태로 만드는 전 과정을

SQL 최적화라고 한다.

최적화 과정

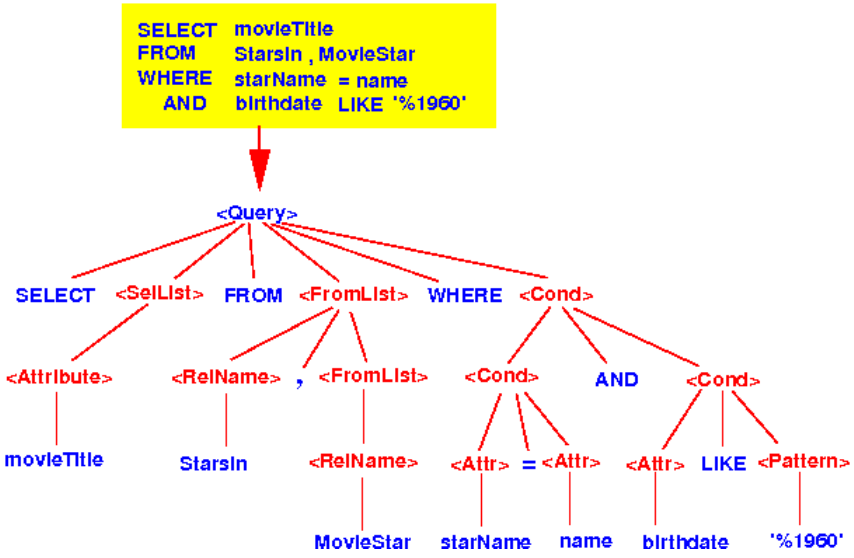


1. 사용자가 SQL을 입력한다.
2. **SQL 파서**가 파싱을 진행한다.
3. **옵티마이저**가 미리 갖고 있는 통계정보를 통해 다양한 실행 경로를 비교 하여 효율적인 경로 하나를 선택한다.
4. **로우 소스 생성기**가 실행경로를 실행 가능한 코드나 프로시저 형태로 포맷팅한다

SQL 파싱과 최적화 과정을 분리해서 얘기하는 경우가 있는데 이 책에서는 2~4를 최적화 과정이라고 얘기한다.

파싱 과정

트리 생성 → Syntax 체크 → Semantic 체크



파싱 트리

- Syntax check? 문법적 오류가 없는지

  사용할 수 없는 키워드가 있는지, 키워드 순서가 올바른지, 누락된 키워드가 있는지

- Semantic check? 의미상 오류가 없는지

  존재하지 않는 테이블이나 컬럼인지, 사용한 오브젝트에 대한 권한이 있는지

SQL 옵티마이저

▼ 옵티마이저?

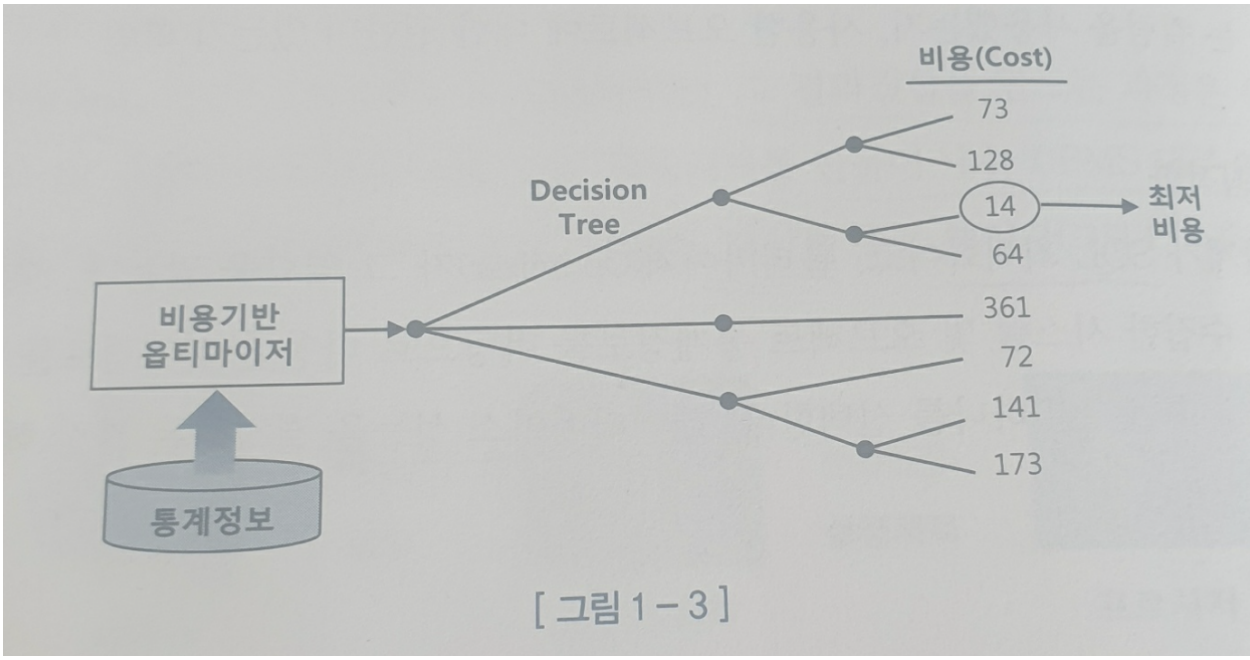
사용자가 원하는 작업을 (= 쿼리에서 원하는 데이터를)

가장 효율적으로 수행할 수 있는 (= 가장 적은 비용으로)

최적의 데이터 액세스 경로를 선택해주는 (= 찾아가는 방법을 알려주는)

DBMS 핵심 엔진

옵티마이저의 최적화 과정



1. 사용자의 쿼리를 수행할만한 후보 리스트를 뽑는다
2. 미리 수집해둔 통계정보를 이용해 각 후보 리스트의 비용을 산정한다.
3. 최저 비용을 나타내는 실행 계획을 선택한다.

> 💡 통계정보는 데이터 딕셔너리에 수집되며, 오브젝트나 시스템의 통계정보를 의미한다.
> 예를 들면, 테이블의 대강의 행수나 열 수, 각 열의 길이나 데이터 타입, 테이블의 크기, PK나 NOT NULL 제약조건, 열 값의 분산과 편향등이 있다.

실행 계획과 비용

특정 쿼리에 대한 실행 계획을 미리 볼 수 있다.

```
EXPLAIN SELECT *
FROM "user" u JOIN user_detail ud
ON u.id = ud.user_id;
```

```
QUERY PLAN
Hash Join  (cost=1.01..2.08 rows=5 width=1119)
  Hash Cond: (ud.user_id = u.id)
  ->  Seq Scan on user_detail ud  (cost=0.00..1.05 rows=5 width=47)
  ->  Hash  (cost=1.00..1.00 rows=1 width=1072)
        ->  Seq Scan on "user" u  (cost=0.00..1.00 rows=1 width=1072)
```

테이블을 스캔하는지, 인덱스를 스캔하는지 / 인덱스를 스캔한다면 어떤 인덱스를 스캔하는지 / ...

여러 실행 계획 중 비용이 가장 적은 실행 계획을 선택하게 되는데

이 때 비용이란 쿼리를 수행하는 동안 발생할 것이라고 예상하는 IO 횟수나 예상 소요시간을 의미한다.

옵티마이저 힌트

옵티마이저가 얼을 타는 경우, 사용자가 실행 계획을 지정할 수 있다.

SQL 이 복잡할수록 옵티마이저가 실수할 가능성이 커진다!

but 웬만하면 옵티마이저의 계획을 따르는 것을 권고하고 있다.

> 💡 PostgreSQL 의 경우 Oracle 과 같은 Hint 를 제공하지는 않고,
> PG_HINT_PLAN 이라는 기능을 제공한다.
>
> Hint 의 경우 syntax 나 semantic 상 오류가 있을 때 무시될 수 있지만
> PG_HINT_PLAN 은 Plan Tree 자체를 변경하는 기법으로 무시될 수 없다.