Project 1-1 SQL Parser

2015-10033 김다윤

▣핵심 모듈과 알고리즘

작성한 모듈에서는 SQL의 대표적인 query 구문들의 구조를 분석하여 알맞은 parse tree를 구성하고, 입력에 대해 올바른 SQL 문법 구조를 가지고 있는지 syntax check한다.

▣구현 내용

명세에 적힌 쿼리에 대한 기본 구현

Exit, create table, drop table, desc, select, insert into, delete, show tables

쿼리에 필요한 옵션 구문 구현

primary key, foreign key, from clause, where clause, as

alter table 구문 추가 구현

alter table <TABLE NAME> add <COLUMN NAME> <DATA TYPE>

<TABLE NAME> 테이블에 <DATA TYPE>을 가지는 <COLUMN NAME>이라는 칼럼을 추가한다.

count(), avg(), sum() 함수 추가 구현

기존에 select구문에서 select 다음에 오는 <SELECT LIST>에서 count(), avg(), sum() 함수를 사용할 수 있다. 즉 다음과 같은 구문이 추가로 가능하다

select ..., count(column) [as column_name] ... from ... [as name] [where ...]

select ..., avg(column) [as column_name] ... from ... [as name] [where ...]

select ..., sum(column) [as column_name] ... from ... [as name] [where ...]

각 함수는 해당 칼럼의 row 수, 평균, 합을 각각 반환한다..

▣가정한 것들

non quote special characters는 키보드에서 입력 가능한 특수 문자들 중 따옴표 '와 "를 제외한 전부를 일컫는다. 즉 아래의 것들을 일컫는다.

~ ` ! @ # $ % ^ & * ( ) - _ = + [{ }] ₩ | ; : , < . > / ?

추가로 구현한 구문에서 사용되는 명령어 역시 예약어(keyword)이다. 그러므로 테이블 또는 칼럼 명을 count, avg, sum 따위로 할 수 없다

쿼리의 끝에는 항상 세미콜론이 존재한다고 가정한다.

▣컴파일 및 실행 방법

실행

$cd [jar file location]

$java –jar [filename].jar

종료

$exit;

▣프로젝트 하면서 느낀 점

각 token은 or로 연결되어있어서 token의 제시순서가 파싱하는 우선순위로 적용되었다. 그래서 직접 파싱되는 과정을 생각해보고 우선적으로 검사해야 할 token을 따지는 것이 중요했다.

뿐만 아니라 문법 정의로부터 애매하지 않은 정규식을 만드는 것이 까다로웠다. Grammar definition을 정확히 정의하는 것이 중요하다고 느꼈다.