PRJ 1-1 DB Parser Report

2016-12299 컴퓨터공학부

박용주

**1. 핵심 모듈과 알고리즘에 대한 설명, 구현한 내용에 대한 간략한 설명**

Create table에 대한 뼈대코드를 중심으로 나머지 쿼리도 추가 구현하였다. 주로 주어진 input을 어떤 토큰으로 나누어 파싱할지가 구현의 대부분이었고, console에 올바르게 print하는 부분도 구현하였다.

1) **public** **static** **final** **int** PRINT\_DROP\_TABLE = 2; 쿼리별로 console print를 위한 enum을 정의하였다

2) **SKIP** : **{** " " **|** "\r" **|** "\t" **|** "\n" **}** skip할 문자를 정의한다.

3) *TOKEN* : /\* Keywords \*/ 함수에서 사용할 token을 정의한다.

4) command(): 쿼리, 또는 exit; 를 읽어서 queryList()를 호출하거나 System.exit(0)을 호출한다.

5) queryList(): 세미콜론으로 구분된 쿼리의 나열을 읽어 하나씩 query()로 처리한다.

6) query(): token에 맞는 쿼리함수를 부르고 q = PRINT\_CREATE\_TABLE; 등으로 q를 설정후 return한다.

- 뼈대코드인 create query에 대한 부분은 생략-

7) dropTableQuery() : < DROP\_TABLE > token과 tableName()으로 파싱한다.

8) descQuery(): < DESC > token과 tableName()으로 파싱한다.

9) showTablesQuery(): < SHOW\_TABLES > token을 파싱한다.

10) selectQuery(): < SELECT > {select할 요소} {tableExpression} 으로 파싱한다.

11) selectList(): {select할 요소}에 해당하며 \* 또는 column의 나열이다.

12) selectColumn(): column의 나열에 해당한다. LOOKAHEAD를 통해 compOperand와 nullPredicate와의 충돌을 막았다.

13) tableExpression(): {tableExpression}에 해당하며 from절과 where절로 파싱한다.

14) fromClause(): from절에 해당하며 < FROM >token과 tableReferenceList()로 파싱한다.

15) whereClause(): where절에 해당하며 < WHERE >과 booleanValueExpression() ({명제}의 OR 나열)로 파싱한다.

16) booleanTerm(): {명제}에 해당하며 {명제요소}의 AND 나열로 파싱한다.

17) booleanFactor(): {명제요소}에 해당하며 {명제테스트}로 파싱한다.

18) booleanTest(): predicate() 또는 (booleanValueExpression) 으로 파싱한다.

19) predicate(): salary > 4 (comparisonPredicate) 이나 is null, is not null (nullPredicate)으로 파싱한다.

20) insertQuery(): < INSERT\_INTO > token과 tableName()(target table) 과 {넣을소스}로 파싱한다.

21) insertColumnAndSource(): {넣을소스}에 해당하며 columnNameList와 valueList를 파싱한다.

22) valueList(): value의 나열을 파싱한다.

23) deleteQuery(): < DELETE\_FROM >과 tableName()과 where절로 파싱한다.

**2. 가정한 것들**

1) 빨간색으로 표시된 create, table, delete, from, show 등의 키워드도 illegal한 단어일 것이다.

2) nullOperation에서 is not null과 is not null 모두 correct한 문법일 것이다.

**3. 컴파일과 실행 방법**

1) eclipse에서 SimpleDBMSParser.jj 오른쪽 클릭 -> Compile with javacc -> run

2) cmd에서 jar파일 directory로 들어감 -> java –jar PRJ1-1\_2016-12299.jar 로 실행

**4. 프로젝트를 하면서 느낀 점**

이번 프로젝트를 진행하면서 처음으로 JavaCC를 사용해보았다. 처음 사용한 언어였지만 초기 스켈레톤 코드에 create table에 대한 코드가 미리 짜여져 있어서, 그 코드와 문법 설명을 비교하며 공부하니 익히기에 매우 용이했다.

덕분에 나머지 쿼리들도 문법에 따라 어렵지 않게 작성할 수 있었고, 오류가 나는 부분만 LOOKAHEAD라는 새로운 키워드로 해결하여 프로젝트를 잘 마무리할 수 있었다.

JAVACC를 통해 처음으로 이렇게 간편하게 PARSER를 구현해보았고, 훗날에도 JAVACC와 정규표현식을 이용할 일이 많을 것이라고 생각된다. 앞으로 나올 1-2, 1-3 프로젝트도 기대된다.