PROJECT 1-2 REPORT

2016-12299

컴퓨터공학부 박용주

1. 핵심 모듈과 알고리즘에 대한 설명, 구현한 내용에 대한 설명

대부분의 내용은 기존에 있던 printMessage 함수안에서 추가하여 구현하였다.

핵심 내용은 Berkeley DB를 이용하여 key, value에 어떤 것을 넣을 지 설계하는 것이었다. 내가 선택한 방법은 schema 하나당 하나의 DB에 하나의 테이블을 저장하는 것이었다.

Create table을 할 때마다 그 쿼리안에 들어있는 모든 내용을 printMessage함수의 인자로 넘겨준다. 그 후, column, pk, fk로 parsing하여 각각에 대한 한줄한줄의 정보를 새로운 DB를 만들어 넣는다.

이렇게해서 만들어진 Schema에 해당하는 DB들 이외에 하나의 DB가 더 있는데, 그것은 main에서 처음부터 만들어져있는 FK_DB이다. FK_DB는 key로 참조하는 테이블, data로 참조되는 테이블을 저장하고 있기 때문에 1-2의 drop table이나 훗날 delete 등을 구현할 때 유용하게 쓰일 것으로 예상된다.

Drop table은 해당하는 DB를 지우는 것으로 구현하였고, desc도 해당 DB를 불러와서 정보를 한줄한줄 모두 읽은 후, column의 내용 해당하는 정보를 출력하는 방식으로 구현하였다. Drop table은 FK_DB와 연동하여 참조되고 있는 table의 삭제를 막았다.

Show tables는 모든 DB를 읽어와(FK_DB를 제외한) 하나씩 출력하는 방식으로 구현하였다.

DB에 스키마를 저장하는 핵심방법은 분류이다. Column이름과 type, null, pk, fk등의 정보를 모두 저장해야 했는데, string을 길게 이어 저장하는 방법도 있지만 나는 a:column_name, a:type, a:not_null, a:primary_key, a:foreign_key를 이용하여 모든 내용을 따로따로 저장하였고, desc에서도 한 column에 해당하는 정보를 들고와서 출력하였다.

또, printMessage에서는 DB는 새로 계속해서 만들고 닫지만, 환경은 유지되어야하기 때문에 환경변수는 MAIN에 선언한 뒤, 인자로 계속해서 넘겨주는 방식으로 구현하였다.

또, case insensitive를 구현하기 위해 column과 table이름을 모두 소문자로 저장하였고, 쿼리를 줄 때도 해당 부분을 소문자로 변환하는 방식으로 구현하였다.

쿼리는 Token을 String으로 변환하여 구현하였는데, string을 계속 더하고 나중에 parsing하는 방법도 있었지만, 나는 ArrayList를 이용하여 원소를 하나씩 저장하였다. 그리고 이 ArrayList를 통째로 create table 을 수행하는 함수에 넘겨준 후, 원소를 하나씩 빼어내는 방식을 취했다

Table name부터 계속해서 String 또는 Arraylist를 return해가면서 query() 함수까지 인자를 넘겨준 후, 쿼리에 알맞게 printMessage에 정보와 환경 2가지를 인자로 넘겨준다..

Cursor에 대한 try catch를 제외한 모든 error처리는 printErrorMessage라는 함수를 이용해 수행하였다.

2. 구현하지 못한 내용

DB를 계속해서 열고 닫다 보니 코드의 중복이 많은데, 이를 아직 캡슐화하지 못하였다. 그래서 코드가 많이 길어진 것 같다.

3. 가정한 것들

Berkeley DB의 data로 배열이 들어갈 수 없을 것이다.

한 테이블에 모든 schema의 내용을 저장하는 것보다 따로 DB를 만드는 것이 나중에 프로젝트가 진행됨에 따라 더 생각하기 편할 것이다.

4. 컴파일과 실행 방법

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

2) cmd에서 jar파일이 있는 directory로 들어감 -> java –jar PRJ1-2_2016-12299.jar 로 실행(같은 경로 안에 db폴더를 만들어야 함)

5. 프로젝트를 하면서 느낀점

Schema를 각각의 DB로 둔 것은 크게 나쁜 설계인 것 같지는 않지만, 쿼리에 담긴 정보를 넘길 때 ArrayList로 하는 방법은 좋지 않은 것 같다. 오히려 String을 연결시켜 인자로 넘겨준 후, parsing을 했다면 훨씬 더 버그의 위험성이 없고 깔끔한 코드가 되었을 것이다.

또, 마지막 제출 전 디버깅 시간이 많이 걸렸는데 오류가 나는 이유를 찾아보니 DB를 열고나서 오류가 나면 break하는 바람에 제대로 닫지 않은 채로 함수를 빠져나가기 때문이었다. 파일이든

DB든 열고나면 꼭 모든 경우를 생각하여 닫는 습관을 들여야 할 것 같다.

그래도 1-3에서 DDM을 구현할 때에는 설계가 되어있기 때문에 더 실수 없이 빠르게 구현할 수 있을 것이라 생각한다.