데이터베이스 PRJ 1-3 리포트

공과대학 컴퓨터공학부 2020-12907 이현우

1. 구현한 내용 : 핵심 모듈 및 알고리즘

구현한 내용을 먼저 간략히 설명하자면, 이전 PRJ 1-2에 이어 DML이 실행가능하도록 코드를 추가하였다. 가장 핵심이 되는 부분은 select와 delete를 할 때 조건을 설정해주는 where절을 parsing하는 것이었고, where절의 condition들을 python내의 dictionary로 저장하여 이를 처리하고자 하였다. 이 where절의 true, false를 판별한 뒤, 해당 조건에 맞는 데이터들을 select, delete 하도록 프로그램을 구현하였다. 그리고 insert를 위해 기존 스키마와 입력된 데이터 타입을 비교하여 insert하는 프로그램을 작성하였다.

sqlTransformer.py 모듈을 먼저 확인하면, 기존에는 drop_query등 tree 구조의 최상단에서 아래 node들에 접근해 원하는 데이터를 얻어 sql_data에 저장하는 방식을 사용하였다. 하지만 이렇게 구현하면 다양한 구조로 입력될 수 있는 where clasue를 파싱하기에 어려움이 있었고, 이 어려움을 해결하기 위해 tree 하단 node부터 원하는 데이터들을 return 하여 상위 method로 전달하는 로직으로 프로그램을 작성하였다. where clause에서 중요한 정보들은 condition의 내용, condition간의 연산자(and, or), condtition들을 감싸는 괄호들이 있었고, 이런 정보들을 모두 파이썬의 dictionary 형태로 저장하여 sql data 내에 저장하도록 하였다. 이 후 다른 정보들을 파싱하여 저장하는 과정은 PRJ 1-2에서 한 방법과 동일하다.

run.py에서 sqlTransformer.py에서 얻은 정보를 이용하여 berkeleydb에 작업을 수행하는 과정은 PRJ 1-2와 동일하게 구성하였다. sql runner로 sql type에 맞는 function으로 sql data를 전달하고, 해당 함수에서 sql data를 이용하여 berkeleydb에 있는 table에 전달하여 작업을 수행하는 logic이 메인 logic이다. berkeleydb에 생성 시간을 기반으로 key값을 datetime 기반으로 생성하고, 데이터들을 파이썬 리스트를 이용해 저장하는 방식을 이용하였다. 이 기본 방식을 기반으로 sql insert에서는 해당 table에 해당하는 db파일에서 schema 정보를 얻은 뒤, 실제 데이터와 schema의 데이터를 비교하여 error checking 및 데이터 값 삽입을 진행하였다.

run.py에서 PRJ 1-3 과정에서 크게 추가된 부분은 where clause를 처리하는 logic이다. where clause 처리를 위하여 이전에 dictionary로 저장한 condition들을 탐색하면서, 먼저 condition의 데이터를 true or false로 판별하는 메소드를 evaluate_condition~ 라는 이름으로 구현하였다. 모든 condtion들의 boolean을 판별하여 원래 condition dictionary가 있던 부분을 replace하고, 모든 condition들의 boolean값이 판별되면 postfix 방식을 이용하여 boolean들을 연산하여 최종 where clause의 boolean 값을 판별하도록 하였다. 이 where clause를 모든 row들에 각각 적용하여 해당 row가 select되거나 delete 되는 대상인지를 판단하도록 하였다.

위의 where clause를 바탕으로, 먼저 delete function은 모든 row들의 key와 data값을 조회 한 후, 각 row들에 대해 where clasue를 판별한 후 true에 해당하는 row들을 delete row 리스트에 저장하여 보관하였다. 모든 row들을 탐색한 후, delete row에 저장된 row를 한번에 db파일에서 삭제하도록 구현하였다. select function은 여러 개의 table을 한번에 조회할 수 있으므로, 먼저 각 테이블의 row들의 cartersian곱을 구해 따로 저장하도록 하였다.

이 후 delete와 동일하게 각 row들에 where clasue를 판별하여 해당되는 row들을 select row에 저장하였고, 출력해야 하는 column들만 따로 뽑아내 최종 출력하도록 구현하였다.

2. 느낀점 및 기타사항

이번 프로젝트를 진행하면서 가장 어려움을 겪었던 부분은 where clause를 처리하는 부분이었다. 이전까지는 tree구조를 parsing하더라도, 어느정도 고정된 structure에서 데이터를 얻었기 때문에 root node에서 충분히 모두 접근이 가능했는데, 이번 where clause에서는 tree 구조가 sql문에 따라 변화할 수 있어 각 tree를 순회할 때 원하는 데이터를 미리 처리하여 상위 node로 전달하는 것이 중요했다. 이런 접근 방식의 코딩은 처음 해보는 것이었는데, 기존 틀에 벗어나 새로운 접근방식으로 프로그램을 구현하는 것이 중요함을 느낄 수 있던 시간이었다. 또한, where절을 처리하면서 where절의 내용을 판별하는 method들의 양이 길어졌고, 이 과정에서 적절한 에러 체크 및 최적화가 필요함을 느꼈다. 또한, 완전한 숏코딩보다는 적절한 유지보수를 위해 코드를 가시적으로 작성하는 것이 중요함을 느꼈다. 과제와 관련없이 향후 나의 프로젝트를 개선한다면, update에 대해서도 구현해보고 싶다는 생각이 들었다.