Project 1-2 Report

2018-10371 컴퓨터공학부 최석우

핵심 모듈과 알고리즘에 대한 설명 + 구현한 내용에 대한 간략한 설명

→ 레이어드 아키텍쳐 형식으로 4단계의 레이어가 상호작용하면서 데이터베이스를 구현한다.

- run.py
  - 가장 바깥에서 사용자와 직접 상호작용하는 역할을 한다.
  - 앞선 프로젝트에서 이미 구현해두었지만, 쿼리를 파싱한 뒤에는 print만 하는 것이 아닌 해당 트리를 database 클래스로 넘겨주어 실제로 처리될 수 있도록 하였다.
- database.py
  - 트리의 형태로 파싱되어 들어온 오브젝트들을 개발자의 편의를 위해 정의된 클래스로 변환하는 역할을 맡는다
  - Parser class
    - 트리로 들어오는 쿼리의 형태를 가공하기 쉬운 형태의 클래스로 변형시켜준다. 변형된 클래스는 나중에 테이블을 만들고 분석하는데 사용된다.
  - Database class
    - run.py의 query들과 일대일대응이 되는 함수들을 가지고 있다.
    - Parser 클래스를 호출해서 클래스의 형태로 변환하고 databaseRepository가 이해하는 형태로 전달하는 Interface의 역할을 한다.
- DatabaseRepository.py
  - 쿼리의 실질적인 분석과 실행이 일어나는 곳이다. TableConstraints, ColumnDefinition, Table과 같은 클래스들이 있으며 이 클래스를 기반으로 쿼리를 처리한다.
  - run.py가 실행되면 DatabaseRepository 클래스는 하나만 생성되며 해당 클래스가 db의 모든 내용을 관리한다.
  - 이 클래스는 dictionary로 항상 치환이 가능하며 이 dictionary는 나중에 binary의 형태로 변경되어 berkeley db에 저장된다.

- DatabaseInstance.py
  - berkeley db와 실제로 소통하는 레이어이다. 테이블을 저장하고 삭제하는 것 외에는 다른 기능을 수행하지 않는다.
  - berkeley db의 key/value pair는 본 구현에서 table-name/table-content에 해당한다. 따라서 berkeley db의 모든 key/value pair를 순환하면 모든 테이블의 모든 정보를 확인해볼 수 있다. table-content에는 테이블 이름부터 스키마, row들이 모두 저장되어 있다.
    - table-content는 테이블의 정보를 dictionary로 변환 하여 그것의 binary form을 가지고 있다. 따라서 decode할 때에는 binary를 불러온 뒤 json.dump를 이용해서 다시 내용을 꺼내오면 된다.

구현하지 못한 내용

- 없다.

프로젝트를 하면서 느낀 점 및 기타사항

- key-value pair로만 database를 구현해야 해서 berkeley db는 단지 텍스트 정보를 저장해주는 저장소 역할로만 사용이 가능했고, 핵심 기능들은 python 문법을 통해서 구현해야 했다.

  → 따라서 필요한 부분에 따라 클래스를 정의하고, 각 클래스가 잘 상호작용할 수 있는 구조를 고안하였다.