DB proj1-2 보고서

2012-11249
컴퓨터공학부
설재완

1. 핵심 모듈과 알고리즘에 대한 설명

파서를 이용해서 파싱을 한 후, Token.image method 를 통해 column name 과 table name 을 string 형태로 가져온다. 이를 통해 자바코드에서 사용할 수 있다.

```
String tableName() :
{ Token tableName; }
{
  tableName = < LEGAL_IDENTIFIER > { return tableName.image; }
}

String columnName() :
{ Token columnName; }
{
  columnName = < LEGAL_IDENTIFIER > { return columnName.image; }
}
```

비슷한 방법으로 그래마의 정의 부분이 void 형태로 선언된 것을 필요한 자료형으로 return 하게 하여 자바코드에서 사용할 수 있게 하였다.

createTableQuery, dropTableQuery, descQuery, showTablesQuery 에서 각각 core function 을 call 하게 하였다. 특히 이때, database 와 각각의 query 에 필요한 것들을 함수의 argument 로 넘겨주었다.

```
void createTableQuery(Database myDatabase) :
{
  String tableName = null;
  ArrayList<TableElement> tableElementList = null;
  int success;
}
{
  < CREATE_TABLE >
  tableName = tableName()
  tableElementList = tableElementList()
  {
    //create Function
    createTable(myDatabase, tableName, tableElementList);
  }
}
```

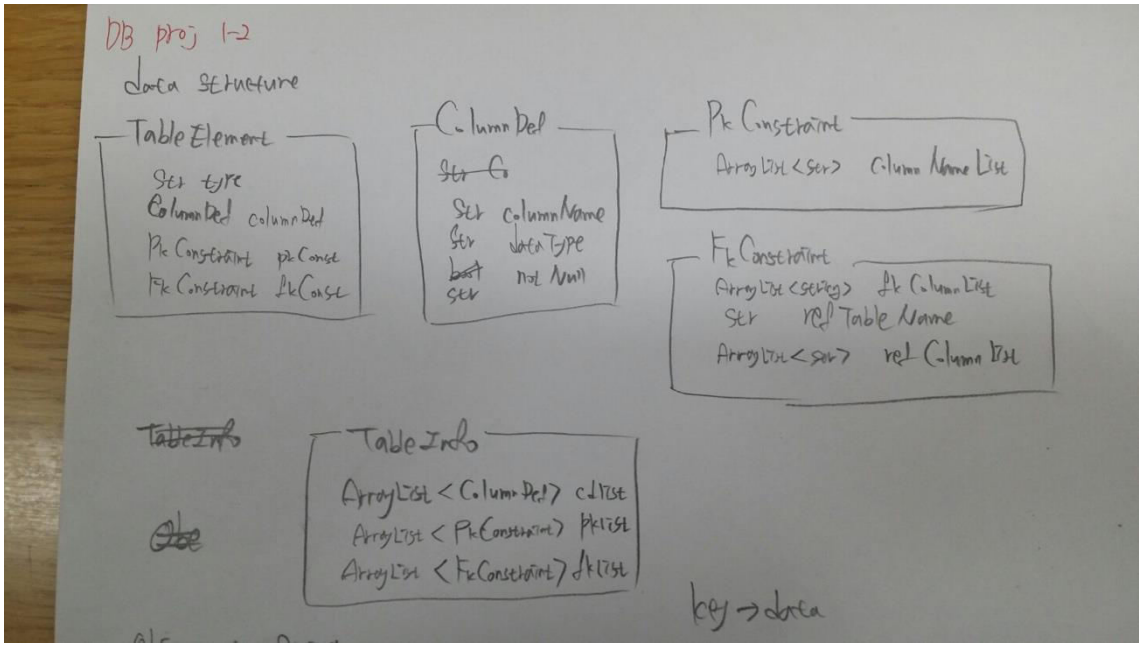
```
void dropTableQuery(Database myDatabase) :
{ String tn; }
{
  < DROP_TABLE >
  tn = tableName()
  { dropTable(myDatabase, tn); }
}

void descQuery(Database myDatabase) :
{ String tn; }
{
  < DESC >
  tn = tableName()
  { desc(myDatabase, tn); }
}

void showTablesQuery(Database myDatabase) :
{}
{
  < SHOW_TABLES >
  { showTables(myDatabase); }
}
```

core function 인 createTable, dropTable, desc, showTables 에서는 파라미터로 넘겨받은 database 와 다른 정보들을 통해 각각 database 에서 읽어오거나, database 에 쓰는 작업을 한다.

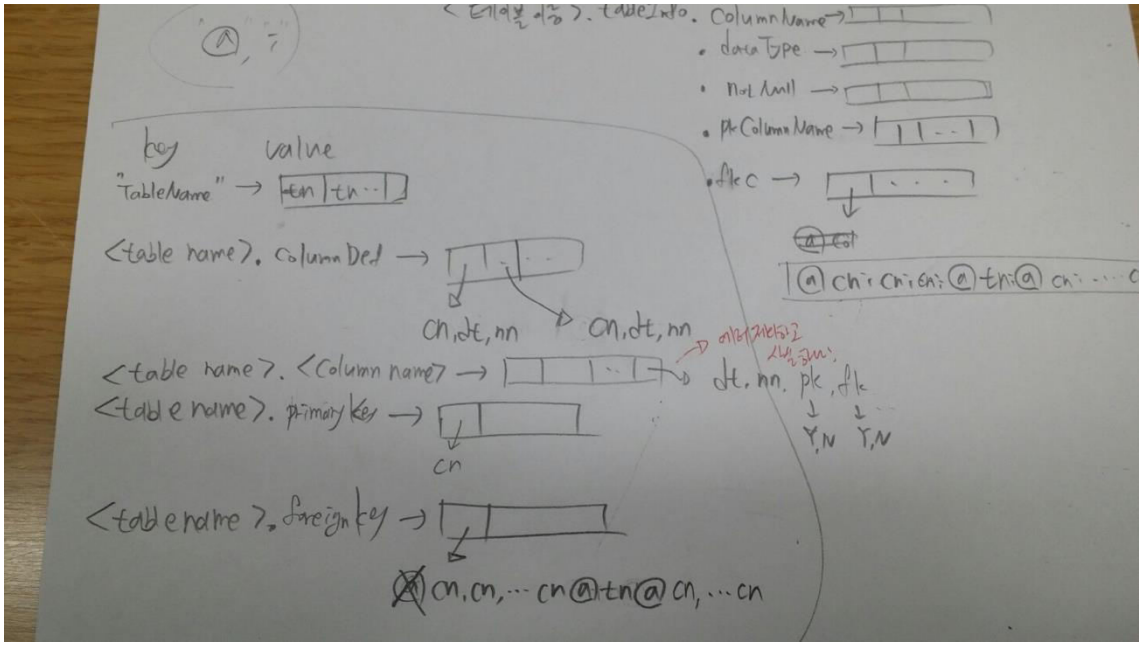
2. 구현한 내용에 대한 간략한 설명

버클리 DB 에서는 key,value 로 write 및 read 하는 것을 지원한다. 그런데 이때에 key 와 value 는 모두 String type 이어야 한다. 하지만 우리가 사용하는 데이터는 하나의 String 이 아니라 여러개의 String 및 ArrayList 를 많이 사용한다. 이를 위해 Serializable interface 를 사용할 수도 있고 String 을 특정 delimiter 를 이용해서 join 하는 방법도 있다. 본인은 후자의 방법을 이용하였다.

다음의 첫번째 그림은 java code 에서 사용한 자료구조이며, 두번째 그림은 key,value 의 구조이다.



Java code 에서 사용한 주요 자료구조



버클리 DB 에 저장한 key,value

각각의 core function 에서는 error 를 check 한 후, error 가 없는 경우에만 일을 진행한다. error check 는 table name, column name 등을 argument 로 넘겨서 is<###>Error 의 함수로 직접 확인하였다.

3. 가정한 것들

새로운 오류 상황을 가정하였다. NonPrimaryKeyError 인데, create table 을 할 때, 명시적으로 primary key 를 정해주지 않는 경우에 발생한다. 이 부분에 대한 명시적인 설명이 없어서 고민했었는데, 2 가지 경우가 가능했다. 전자는 primary key 가 선언되지 않았을 때, 본인처럼 error 를 띄우는 방법이고 후자는 primary key 가 선언되지 않았으면 default 로 모든 column 을 primary key 로 설정하는 방법이다.

Column definition 에서 명시적으로 not null 을 적어주지 않아도 primary key 지정하면 not null constraint 를 갖도록 하였다. 즉 create table student(id int, name char(10), primary key(id))로 할 경우 id column 에 not null constraint 를 추가하였다. 아울러 이는 스펙 문서에 명시된 "Primary key 로 지정된 컬럼은 따로 not null 로 지정하지 않더라도 자동적으로 not null 인 것으로 본다"라는 가정을 따르기 위한 것이다.

프로그램의 종료는 반드시 exit;을 통해서 이루어 진다고 가정하였다. exit;을 할 때 db 를 close 하면서 그때서야 write 하는 경우가 있다. 만약 exit;을 안하고 임의로 종료시에도 반영하게 하려면 sync option 등을 통해서 해결하면 된다.

4. 컴파일과 실행방법

Proj 1-1 과 마찬가지로 컴파일은 eclipse 에서 지원하며, 제출할 파일은 executable jar file 이기 때문에 실행은 jar file 을 실행하면 된다.

```
Last login: Tue Nov  7 16:04:20 on ttys000
bmy4415ui-MacBook-Air:~ bmy4415$ cd Desktop/
bmy4415ui-MacBook-Air:Desktop bmy4415$ java -jar PRJ1-2_2012-11249.jar
Exception in thread "main" java.lang.IllegalArgumentException: Environment home db doesn't exist
        at com.sleepycat.je.log.FileManager.<init>(FileManager.java:394)
        at com.sleepycat.je.dbi.EnvironmentImpl.<init>(EnvironmentImpl.java:537)
        at com.sleepycat.je.dbi.EnvironmentImpl.<init>(EnvironmentImpl.java:462)
        at com.sleepycat.je.dbi.DbEnvPool.getEnvironment(DbEnvPool.java:224)
        at com.sleepycat.je.Environment.makeEnvironmentImpl(Environment.java:288)
        at com.sleepycat.je.Environment.<init>(Environment.java:268)
        at com.sleepycat.je.Environment.<init>(Environment.java:212)
        at SimpleDBMSParser.main(SimpleDBMSParser.java:619)
bmy4415ui-MacBook-Air:Desktop bmy4415$ mv PRJ1-2_2012-11249.jar PRJ1-2_2012-11249
bmy4415ui-MacBook-Air:Desktop bmy4415$ cd PRJ1-2_2012-11249
bmy4415ui-MacBook-Air:PRJ1-2_2012-11249 bmy4415$ ls
PRJ1-2_2012-11249.jar  bin/                   db/                    lib/                   src/
bmy4415ui-MacBook-Air:PRJ1-2_2012-11249 bmy4415$ java -jar PRJ1-2_2012-11249.jar
DB_2012-11249> exit;
bmy4415ui-MacBook-Air:PRJ1-2_2012-11249 bmy4415$
```

단 주의할 점이 있는데, .jar 파일을 project 디렉토리 안에서 실행해야 한다.

아마도 db 라는 디렉토리가 같은 path 에 없기 때문인 것 같다. 그림에서 보는 것처럼 .jar 파일이 project 내부에 있을 경우 잘 실행되는 것을 확인할 수 있다.

5. 프로젝트를 하며 느낀 점

매 숙제마다 새로운 것을 이용하는 것이 쉽지 않았다. 처음 숙제에서 javacc 를 하는 것도 해본 경험이 없는 것이라 쉽지 않았는데, 이번에는 버클리 DB 가 비슷한 역할을 하였다. 특히 반드시 String type 으로 key 와 value 를 가져야해서 자료구조를 잘 설계하는 것이 중요했다. 문제는 String type 으로 key 와 value 를 가져야 한다는 사실을 알게되는데에 오랜 시간이 걸렸다. Reference 를 잘 읽어야하는데, 모든 reference 를 읽기엔 너무 길어서 reference 를 효과적으로 읽는 훈련을 해야겠다고 생각했다. 그럼에도 불구하고 reference 를 읽지 않고 코딩을 할 경우, 해당 플랫폼이 어떤 기능에 제한이 있는지를 모르기 때문에 처음에 설계한 것을 다 뒤집어야 하는 어려움이 있다.

두번째 어려움은 스펙 문서의 이해이다. 명시적이지 않은 상황에 대해서 어떻게 처리해야할지를 확실히 알아야 자료구조를 설계할 수 있는데, 예를 들어 Error message 의 출력(1 개만 출력 또는 모든 error 를 다 출력)등에서 어려움이 있었다. 특히 case insensitive 라는 문장도 실제 모든 data 를 임의적으로 대문자 또는 소문자로 넣어도 되는 것인지 등에 대해서 확신이 없었고 exit;를 통해 명시적으로 프로그램을 종료하지 않는 경우에 지금까지의 기록이 파일에 남아있어야 하는지(sync issue) 등에 대해서도 고민이 많았다.

메일로 대부분의 질문을 해야하는 환경이므로 숙제에 빨리 임하는 것이 중요하다고 생각하였다.

Proj3 도 Proj2 와 연결이 될 것인데, proj2 의 설계상의 문제(key, value 설계 등)가 proj3 에도 영향을 미칠까 두려워 고생을 한 것도 기억이 난다. 대부분 어려운 점을 느꼈으나, 완성을 한 후에는 보람을 느꼈다.