How to Run Tests

Main Blocks

- How to test the linear main data storage

  The test codes have been provides below the file $SequentialBlocks.py$ which is for exercise 1 of the computing assignment.

  Note that you can always use the function $printall(name)$ to see the data inside. You can also use the $printTable$ to get a nice visualization of data.

  Don't touch functions like $merge$ and $split$ because they are private.

  Below are the descriptions of its grammar:

  - insertion

    The insertion will require the name of table you want to insert, like the "MOVIE" and also the tuples you want.

    You have to modify the $schema.py$ if you want to insert some new tables because the code has to know the positions of its keys which is defined in the $schema.py$! You can see the constructor of the main data here. You should know that we build the relation by passing the function pointers (defined in $schema.py$) inside it, otherwise it's not possible to compare the keys!

```
table_list = ["MOVIE"]
a = MainData(table_list)
a.insert("MOVIE", ('Titanic',1997,'USA',195,'romance'))
a.insert("MOVIE", ('Shakespeare in Love',1998,'UK',122,'romance'))
a.insert("MOVIE", ('The Cider House Rules',1999,'USA',125,'drama'))
a.insert("MOVIE", ('Gandhi',1982,'India',188,'drama'))
a.insert("MOVIE", ('American Beauty',1999,'USA',121,'drama'))
a.insert("MOVIE", ('Affliction',1997,'USA',113,'drama'))
a.insert("MOVIE", ('Scream 3',2000,'USA',116,'horror'))
```

  - search

    The search is rather simple and see the code below.

```
print(a.find("MOVIE", ('Scream 3',2000)))
```

    Then the output will be:

```
('Scream 3', 2000, 'USA', 116, 'horror')
```

    Remember that the search here doesn't exploit the B+ tree, this is only for exercise 1.

  - deletion

```
a.delete("MOVIE", ('Scream 3',2000))
a.delete("MOVIE", ('Titanic',1997))
```

    You can delete a lot to see whether the $merge$ works.

- What IO functions I used for B+ tree

  As you can see, the $merge$ and $split$ and $\mathrm{delete\_with\_tree}$ are specially made for database and B+ tree. The delete has a default argument called $LinearSearch$ which should be 0 if you call the deletion from class database because the database combines the main block with the B+ tree. It will enable quick search, quick insertion and quick deletion. The B+ tree has an IO called $\mathrm{Search\_for\_node}$ which will return the block pointer you want.

Database

The database is highly integrated.

- IO functions(Not for query engine)
  - delete
  - insert
  - search / search_attributes

    Please don't use these IO functions without reading the definition.
- How to interact with query system
  - The $\mathrm{search\_attributes}$ is the IO to the query engine.

```
def search_attributes(self, name, input_dict):#attribute is a dictionary
# search with primary key for PERSON, the input should be ("PERSON", {"id":
"xxx"})
# For table Movie, the input (primary key) should be ("Movie",
{"title":"xxx", "year":"xxx"})
# How to see if it's primary key or not?
# Make use of the schema.py to check the relation schema.
# eg. for the key of "year" and "title" -> the input of search should be
title, year -> make sure the order is OK
# WARNING: input keys might be primary key or secondary key
# Finally, the return value should be like [(xx,xx,xx,1,10,xx),
(xx,xx,xx,1,10,xx),(xx,xx,xx,1,10,xx)] -> a list of tuples
```

  - $getList(name)$

    return all the content of table (a list of tuples)
  - $getSchema(name)$

    return the relation schema by the name
- How to understand the tests I provided

  There are a lot of tests under the $database.py$, you should know the following to understand the output.

  - $database.data.printall()$ gives all the information of the main data.
  - The for loops are used to print out the data by index structure: to prove the correctness of the tree structure.
  - "checking layer: True" means that you are now in the bottom of the B+ tree and vice versa.
  - Please don't modify the testing code too much because you can see that we print the B+ tree by for loops. If you insert too many or delete too many tuples, the height of the B+ tree will be different, so the for loops may not be correct. For examples, for a very high B+ tree, I have to use may for loops to print out the B+ tree layer by layer.

- Pretty Print

  Use the $\mathrm{printTable}(\mathrm{table\_name})$ to show the whole data in pretty form!

B Tree Alone

In our CA, B-Tree serves as the index structure for secondary keys for query requests, which requires modification as a consequence. Instead of a pointer to data stored in MainData, the data field of a B-Tree node is now a $spider$, which is a list containing all the data with the very same key.

Below are the interface functions we used in database and their grammar:

- insertion

  To insert, we first search the B-Tree to see whether there already has the key we are to insert. If so, just append a pointer to new data to the $spider$ of the corresponding node. Else, then just apply the normal insertion. Generally, the insertion is similar to what have discussed in lectures and labs.

  ```
  tree = Btree()
  tree.insert(14, "data14")
  tree.insert(14, "#data14")
  tree.insert(0, "data0")
  tree.insert(8, "data8")
  ```

- search

  the search method is similar to what have discussed in lectures and labs. The following codes show how to execute it.

  ```
  tree.find(8)
  tree.find(14)
  ```

  and the results will be:

  ```
  ['data8']
  ['data14', '#data14']
  ```

- deletion

  the delete method is also similar to what have discussed in lectures and labs. Just execute the following codes.

  ```
  tree.delete(8)
  tree.delete(14)
  ```

Abstract Stack Machine and Query

- Grammar

See the file $ASMgrammar.pdf$ for more detail. (We also prepared a Chinese version. lol.)

- How to test

  See the $Q2test.py$ and $Q3test.py$ for checking whether the queries are correct.

  Each file consists of testing for Exercise 2 (ii)/(iii) & Exercise 3 (v)

For $Q3test.py$, we provided two different testing data sets. One is the data from BB, which is very big and hard for justification (but you can do it using SQL). Another is the data from us, which is very small, and you can easily check the result by just read few lines of data.

For $Q2test.py$, we only provide a big data set from BB. You can justified the result by using SQL and compare our result with the SQL result.

Some notice in the files are import, so please do not ignore them.

Authors

Group 7

- **Zhu Zhongbo** - *3180111635*
- **Guan Zimu** - *3180111630*
- **Xie Tian** - *3180111631*
- **Yang Zhaohua** - *3180111374*

Contact us

your communication really makes a difference

- Zhongbo.18@intl.zju.edu.cn
- Zhaohua.18@intl.zju.edu.cn
- tianx.18@intl.zju.edu.cn
- zimu.18@intl.zju.edu.cn