20150527 Hyung Jun Yoon

[CS360] Introduction to Database
Term project #1 report

Description of data and B+ tree

Source : the product dataset of a clothing mall

Schema : Product dataset

- tid : tuple id
- product_idx : the unique number specifies distinct product
- category : category of product such as t-shirts, outers, …
- price : the price of product
- sales : number of items sold
- stock : number of items remaining
- date : first date the item was imported

Number of tuples : 100

Order of the B+ tree : 3

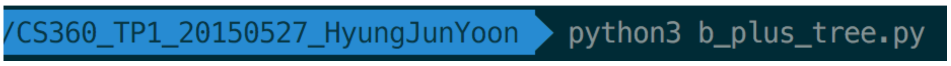
Attributes used for the key of B+ tree :

A pair, <sales, price>, is used as the key of the B+ tree index

Instruction of program

How to run

- b_plus_tree.py is the Python file that performs the given function in problem, and data.csv is the file that contains the data. We must place b_plus_tree.py and data.csv in the same directory for the function to work. Then, in the shell command window, write the python3 b_plus_tree.py command as below to run the program.



```
/CS360_TP1_20150527_HyungJunYoon  python3 b_plus_tree.py
```

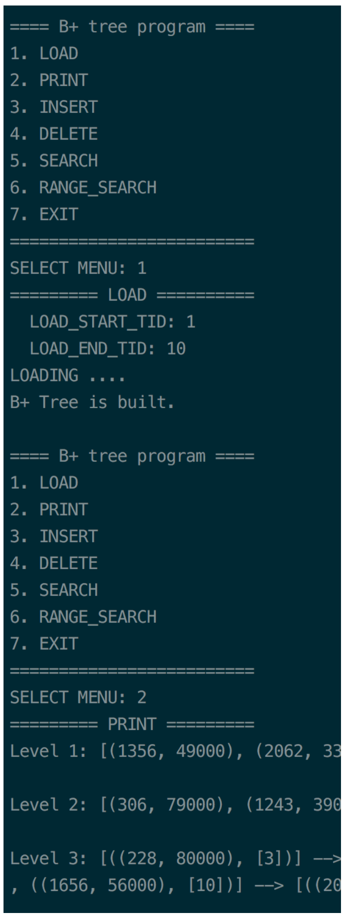
After that, the following screen will appear and you can see that the program is running.

20150527 Hyung Jun Yoon

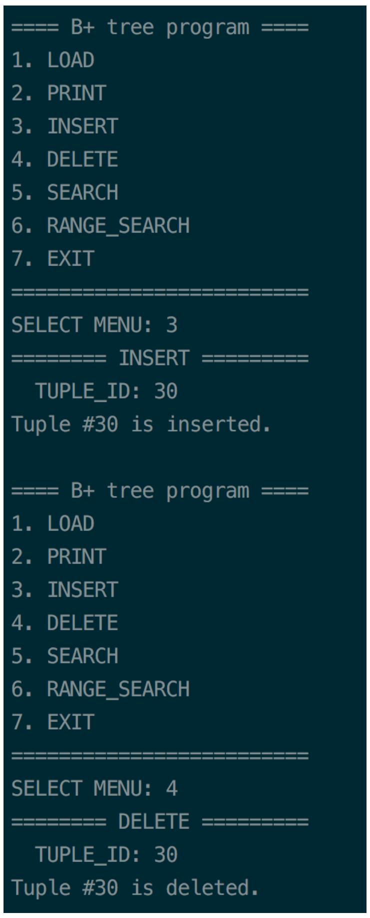
Testing operation

- LOAD : After entering 1 in SELECT MENU, you can load data from data.csv to b+ tree by putting the desired tid value in LOAD_START_TID and LOAD_END_TID.
- PRINT : Enter 2 in SELECT MENU, then you can see the current b+ tree.
- INSERT : Enter 3 in SELECT MENU, then you can insert data from data.csv to b+ tree by putting one tid value(integer) after TUPLE_ID.
- DELETE : Enter 4 in SELECT MENU, then you can delete data in b+ tree by putting one existing tid value(integer) after TUPLE_ID.
- SEARCH : After entering 5 in SELECT MENU, you can search one data in b+ tree by putting one key pair(two integers enclosed in parentheses) after SEARCH KEY.
- RANGE_SEARCH : After entering 6 in SELECT MENU, you can search several data in b+ tree by putting the range of key pairs(two key pairs enclosed in square brackets. For example, [(1175, 21000), (1452, 57000)]) after SEARCH RANGE.

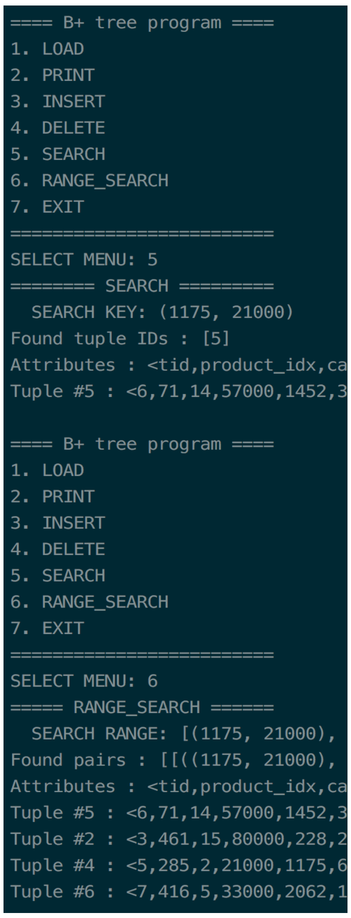
  There are some example screenshots here.



```
==== B+ tree program ====
1. LOAD
2. PRINT
3. INSERT
4. DELETE
5. SEARCH
6. RANGE_SEARCH
7. EXIT
=========================
SELECT MENU: 1
========= LOAD ==========
  LOAD_START_TID: 1
  LOAD_END_TID: 10
LOADING ....
B+ Tree is built.

==== B+ tree program ====
1. LOAD
2. PRINT
3. INSERT
4. DELETE
5. SEARCH
6. RANGE_SEARCH
7. EXIT
=========================
SELECT MENU: 2
========= PRINT =========
Level 1: [(1356, 49000), (2062, 33

Level 2: [(306, 79000), (1243, 390

Level 3: [((228, 80000), [3])] -->
, ((1656, 56000), [10])] --> [((20
```



```
==== B+ tree program ====
1. LOAD
2. PRINT
3. INSERT
4. DELETE
5. SEARCH
6. RANGE_SEARCH
7. EXIT
=========================
SELECT MENU: 3
======== INSERT =========
  TUPLE_ID: 30
Tuple #30 is inserted.

==== B+ tree program ====
1. LOAD
2. PRINT
3. INSERT
4. DELETE
5. SEARCH
6. RANGE_SEARCH
7. EXIT
=========================
SELECT MENU: 4
======== DELETE =========
  TUPLE_ID: 30
Tuple #30 is deleted.
```



```
==== B+ tree program ====
1. LOAD
2. PRINT
3. INSERT
4. DELETE
5. SEARCH
6. RANGE_SEARCH
7. EXIT
=========================
SELECT MENU: 5
======== SEARCH =========
  SEARCH KEY: (1175, 21000)
Found tuple IDs : [5]
Attributes : <tid,product_idx,ca
Tuple #5 : <6,71,14,57000,1452,3

==== B+ tree program ====
1. LOAD
2. PRINT
3. INSERT
4. DELETE
5. SEARCH
6. RANGE_SEARCH
7. EXIT
=========================
SELECT MENU: 6
===== RANGE_SEARCH ======
  SEARCH RANGE: [(1175, 21000),
Found pairs : [[((1175, 21000),
Attributes : <tid,product_idx,ca
Tuple #5 : <6,71,14,57000,1452,3
Tuple #2 : <3,461,15,80000,228,2
Tuple #4 : <5,285,2,21000,1175,6
Tuple #6 : <7,416,5,33000,2062,1
```