MiniDb – Indexing and Primary key

Επιμέλεια: Ιωακείμ Ελ-Χαττάμπ Μπριστογιάννης, Παρασκευή Παλιούρα, Μυρτώ – Μαρία Σπάθα, ΑΜ: Π19048, Π19129, Π19156

Όνομα Ομάδας: Dream Team

Στην παρούσα εργασία υλοποιήθηκαν οι παρακάτω λειτουργίες της miniDB:

1. Υποστήριξη πολλαπλών πρωτευόντων κλειδιών σε έναν πίνακα σχέσεων.
2. Υποστήριξη ευρετηρίου μέσω B+ δέντρου σε μία ή περισσότερες στήλες με διπλότυπα (ενέργειες δημιουργίας ευρετηρίου και επιλογής ευρετηριασμένων χαρακτηριστικών).
3. Υποστήριξη ευρετηρίου μέσω B+ δέντρου σε πολλαπλές στήλες ενός πίνακα (ενέργειες δημιουργίας ευρετηρίου και επιλογής ευρετηριασμένων χαρακτηριστικών).
4. Δημιουργία ευρετηρίου κατακερματισμού.

Παρακάτω, παρουσιάζονται αναλυτικότερα οι λειτουργίες που αναπτύχθηκαν για κάθε ενέργεια.

<u>Ενέργεια δημιουργίας πίνακα με πολλαπλά πρωτεύοντα κλειδιά:</u>

Για την υποστήριξη αυτής της ενέργειας έγιναν τροποποιήσεις στην επεξεργασία του query δημιουργίας πίνακα στη μέθοδο create_query_plan του αρχείου mdb.py, στη μέθοδο __init__ της κλάσης table και στη μέθοδο show της κλάσης table.

Πιο συγκεκριμένα:

- Μέθοδος create_query_plan: Αρχικά, γίνεται αναγνώριση της μορφής του query και στη συνέχεια αποθηκεύεται το κάθε keyword σε ένα dictionary.
- Μέθοδος __init__: Το index του primary key είναι μια λίστα που αποθηκεύει το κάθε column του primary key.
- Μέθοδος show: Χρήση του καινούργιου primary key index για την εμφάνιση του κάθε column που ανήκει στο primary key.

<u>Ενέργεια Δημιουργίας Ευρετηρίου:</u>

Για την υποστήριξη αυτής της ενέργειας έγιναν τροποποιήσεις στην επεξεργασία του query δημιουργίας ευρετηρίου (‘create index *index_name* on *table_name(columns_names)* using *btree/hash*’) στη μέθοδο create_query_plan του αρχείου mdb.py και των μεθόδων δημιουργίας ευρετηρίου create_index, construct_index της κλάσης database.

Επιπλέον, για τη δημιουργία ευρετηρίου μέσω B+ δέντρου τροποποιήθηκαν οι μέθοδοι __init__, insert της κλάσης btree, ενώ για τη δημιουργία ευρετηρίου κατακερματισμού έχει δημιουργηθεί το αρχείο hash.py.

Πιο συγκεκριμένα:

- Μέθοδος create_query_plan: Έχει προστεθεί το action ‘create_index’, στο οποίο αναγνωρίζεται η μορφή του query και οι στήλες που βρίσκονται μετά το ‘on’ εισάγονται σε ένα dictionary.
- Μέθοδος create_index: Παίρνει ως είσοδο το όνομα του ευρετηρίου, το όνομα του πίνακα, τα ονόματα των στηλών πάνω στα οποία θα δημιουργηθεί το ευρετήριο και τον τύπο του ευρετηρίου.
  Σε περίπτωση που ο τύπος του ευρετηρίου είναι btree, εισάγονται τα στοιχεία του ευρετηρίου (όνομα ευρετηρίου, όνομα πίνακα , όνομα ευρετηριασμένων στηλών) στον πίνακα meta_indexes σε μορφή string και γίνεται έλεγχος αν οι στήλες περιέχουν διπλότυπα. Σε περίπτωση που υπάρχουν διπλότυπα, στον αρχικό πίνακα προστίθεται μια κρυφή στήλη για κάθε στήλη που ευρετηριάζεται (η κρυφή στήλη περιέχει ακεραίους και είναι auto incrementing). Στη συνέχεια καλείται η μέθοδος construct_index και αποθηκεύεται η βάση.
  <u>Σημείωση:</u> Στήλες με διπλότυπα χαρακτηρίζονται όλες οι στήλες που δεν είναι primary key.
  Σε περίπτωση που ο τύπος του ευρετηρίου είναι hash, εισάγονται τα στοιχεία του ευρετηρίου (όνομα ευρετηρίου, όνομα πίνακα , όνομα ευρετηριασμένων στηλών) στον πίνακα meta_indexes, καλείται η μέθοδος construct_index και αποθηκεύεται η βάση.
- Μέθοδος construct_index: Παίρνει ως είσοδο το όνομα του ευρετηρίου, το όνομα του πίνακα, τα ονόματα των στηλών πάνω στα οποία θα δημιουργηθεί το ευρετήριο, τον τύπο του ευρετηρίου και μια λογική μεταβλητή για τον καθορισμό διπλοτύπων (True σε περίπτωση που υπάρχουν διπλότυπα).
  Σε περίπτωση που ο τύπος του ευρετηρίου είναι btree, φτιάχνεται ένα αντικείμενο της κλάσης btree με branching factor 3. Στη συνέχεια, δημιουργούμε μια λίστα που περιέχει τις εγγραφές για κάθε στήλη που ευρετηριάζεται , καθώς και την τιμή της κρυφής στήλης για κάθε εγγραφή σε περίπτωση που υπάρχουν διπλότυπα. Τέλος, για κάθε στοιχείο αυτής της λίστας καλείται η μέθοδος insert της κλάσης btree.
  Σε περίπτωση που ο τύπος του ευρετηρίου είναι hash, φτιάχνεται ένα αντικείμενο της κλάσης hash_table και για κάθε εγγραφή στο primary key του πίνακα, εισάγουμε την τιμή του και το δείκτη του στο αντικείμενο του hash table. Τέλος, αποθηκεύεται το ευρετήριο.
  <u>Σημείωση:</u> Το ευρετήριο κατακερματισμού δημιουργείται μόνο στο μονοδιάστατο πρωτεύον κλειδί του πίνακα.
- Αρχείο hash.py: Αποτελείται από μια κλάση (τη hash_table), τα αντικείμενα της οποίας είναι οι πίνακες κατακερματισμού. Περιέχει τις παρακάτω μεθόδους:
  - __init__: Ορίζεται το μέγεθος του πίνακα που αντιστοιχεί στον μέγιστο αριθμό εγγραφών που αποθηκεύονται σε αυτόν (εδώ, έχουμε το πολύ 23 εγγραφές). Αρχικοποιείται ο πίνακας με τιμές None για κάθε εγγραφή. Ορίζεται επίσης ο αριθμός των γεμάτων

κάδων σε 0, όπως και το κατώφλι για το resize (0.75). Καλείται η συνάρτηση κατακερματισμού (crc32_hash) και το crc32_table. Τέλος, ορίζεται ένας τυχαίος αριθμός a, ο οποίος χρησιμοποιείται όταν το κλειδί είναι string.

- __repr__: Επιστρέφει τα ζευγάρια σε μορφή {key1: value1, key2: value2, ...}.
- Load_Factor: Υπολογίζει πόσο γεμάτος είναι ο πίνακας, ώστε να γίνει resize όταν φτάσουμε το κατώφλι που έχουμε ορίσει.
- Encode: Κωδικοποιεί το κλειδί που στέλνεται στην CRC32_hash, το οποίο υπολογίζεται από τη μαθηματική συνάρτηση polynomial rolling hash function (ορίζεται μέσα στον κώδικα της Encode).
- Crc32_table: Επιστρέφει έναν πίνακα από τιμές οι οποίες χρησιμοποιούν τον αλγόριθμο crc32 ως hash μέθοδο και θα χρησιμοποιηθούν στη crc32_hash.
- Crc32_hash: Πρόκειται για τη συνάρτηση κατακερματισμού. Κατακερματίζει το κλειδί που παράγεται από τη μέθοδο encode(). Στην ουσία, χρησιμοποιεί First Order Least Significant Bit (LSB), ορίζει το αρχικό CRC σε FFFFFFFF και, τέλος, συμπληρώνει το τελικό CRC.
- Resize: Αυξάνει το μέγεθος του πίνακα, μόλις φτάσει στο κατώφλι. Το μέγεθος του πίνακα αλλάζει στον μικρότερο πρώτο αριθμό μεγαλύτερο από το διπλάσιο του τωρινού μεγέθους. Για να βρούμε τον πρώτο αριθμό, εφαρμόζουμε το «Κόσκινο του Ερατοσθένη». Αφού αλλάξουμε το μέγεθος του πίνακα, όλες οι καταχωρίσεις κατακερματίζονται εκ νέου.
- Insert: Παίρνει σαν είσοδο το key, value και χειρίζεται την εισαγωγή στοιχείων στον πίνακα.

Πηγή για τον αλγόριθμο crc32: https://www.section.io/engineering-education/hashtables-implementation-using-crc32-algorithm/?fbclid=IwAR3EPcre8bxixTYCq3si6IFHJgzeIq5gJDWT_2TCzFVwHhADtQyC07ADYik

Ενέργεια επιλογής ευρετηριασμένων χαρακτηριστικών:

Η επιλογή ευρετηριασμένων χαρακτηριστικών υποστηρίζεται για ευρετήριο μέσω B+ δέντρου.

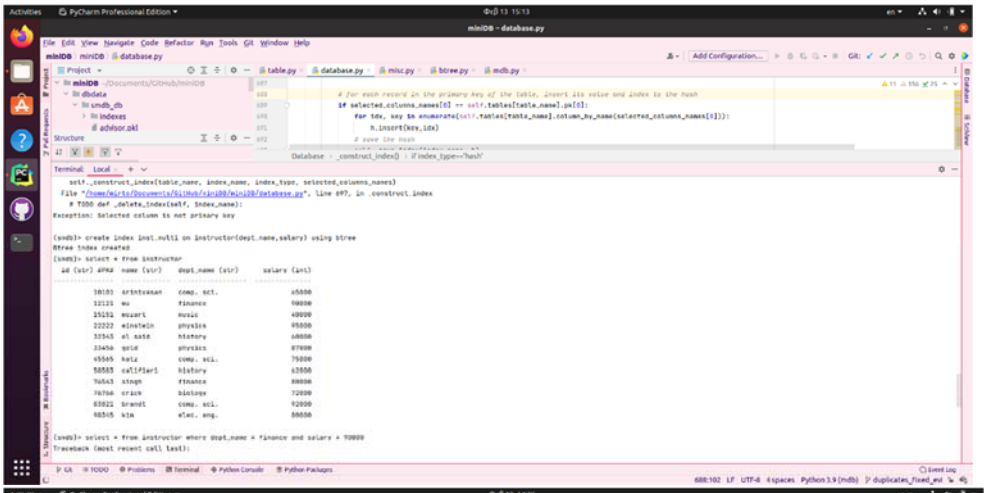
Για την υποστήριξη αυτής της ενέργειας έγιναν τροποποιήσεις στις μεθόδους select της κλάσης database, select_with_btree της κλάσης table, split_condition του αρχείου misc.py, has_index της κλάσης database και find της κλάσης btree.

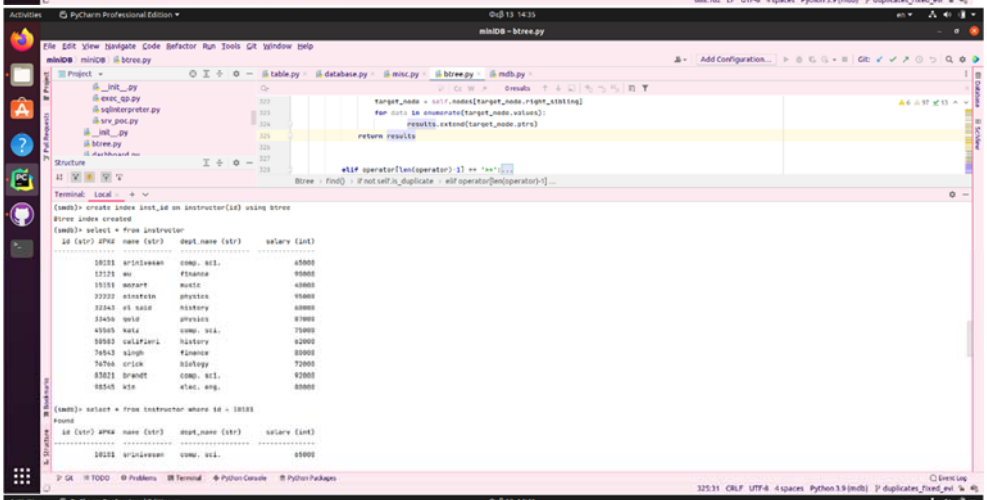
Πιο συγκεκριμένα:

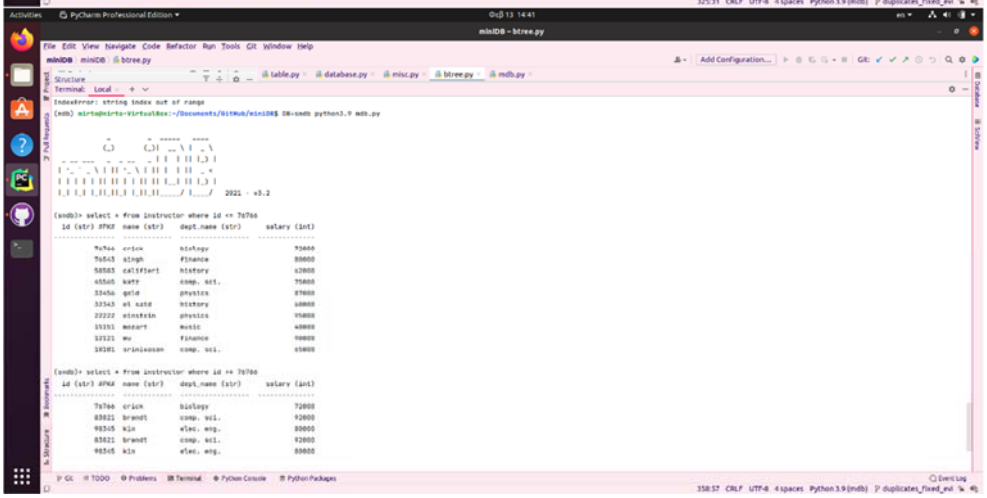
- Μέθοδος select: Έχει προστεθεί ο έλεγχος των στηλών του πίνακα που γίνεται η επιλογή για το αν είναι ευρετηριασμένες (μέσω της μεθόδου has_index). Σε περίπτωση που είναι ευρετηριασμένες, εκτελείται η μέθοδος select_with_btree.

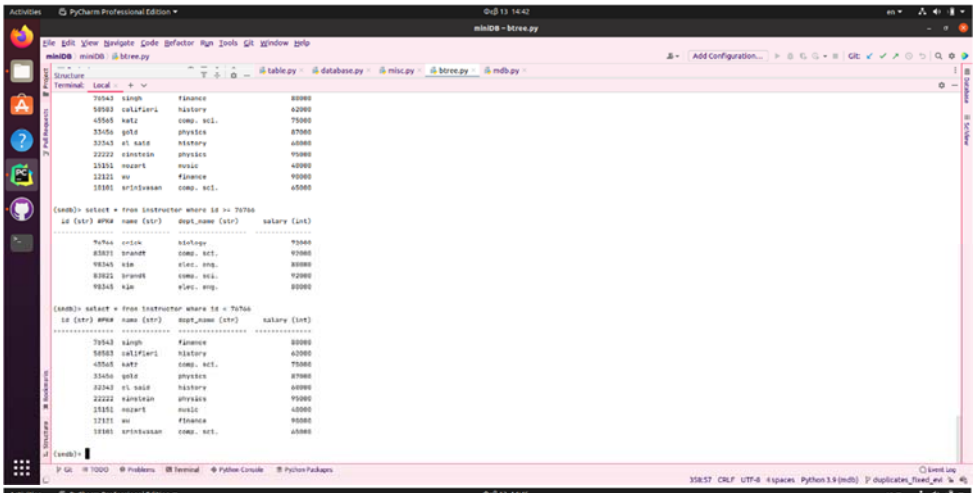
- Μέθοδος select_with_btree: Έχει προστεθεί η διάκριση του query επιλογής στις παρακάτω περιπτώσεις:
    - Σε περίπτωση που η επιλογή γίνεται σε δύο στήλες και η πρώτη στήλη κάνει ερώτηση ταυτότητας, εκτελείται η μέθοδος find.
    - Σε περίπτωση που η επιλογή γίνεται σε μία στήλη, εκτελείται η μέθοδος find.
    - Σε οποιαδήποτε άλλη περίπτωση, εκτελείται η αναζήτηση χωρίς ευρετήριο.
- Μέθοδος split_condition: Χωρίζει τη συνθήκη της επιλογής (πρόταση μετά το where) στα συνθετικά της (όνομα στήλης, τελεστής σύγκρισης, τιμή χαρακτηριστικού). Έχει προστεθεί υποστήριξη για επιλογή πολλών στηλών.
- Μέθοδος has_index: Ελέγχεται αν υπάρχει ευρετήριο για τις στήλες που παίρνει ως είσοδο και επιστρέφει True και το όνομά του σε περίπτωση που υπάρχει. Ο έλεγχος γίνεται εκτελώντας τη μέθοδο select για τον πίνακα meta_indexes αναζητώντας το όνομα του πίνακα και τα ονόματα των στηλών που θέλουμε να ευρετηριάζονται.
- Μέθοδος find: Κάνει αναζήτηση στο ευρετήριο και επιστρέφει τα αποτελέσματα. Διακρίνονται οι παρακάτω περιπτώσεις:
    - Σε περίπτωση που το ευρετήριο δεν έχει δημιουργηθεί πάνω σε στήλες με διπλότυπα (δηλαδή είναι στο primary key), γίνεται αναζήτηση της ιδανικής θέσης της επιθυμητής τιμής μέσα στο δέντρο. Στη συνέχεια, ανάλογα με τον τελεστή σύγκρισης που παρέχεται από το query, επιστρέφονται τα κατάλληλα αποτελέσματα.
    - Σε περίπτωση που το ευρετήριο έχει δημιουργηθεί πάνω σε στήλες με διπλότυπα, στην τιμή αναζήτησης προστίθεται μια τιμή για την κρυφή στήλη (0 ή 20000000, ανάλογα με τον τελεστή σύγκρισης) και γίνεται αναζήτηση της ιδανικής θέσης της επιθυμητής τιμής μέσα στο δέντρο. Στη συνέχεια, ανάλογα με τον τελεστή σύγκρισης που παρέχεται από το query, επιστρέφονται τα κατάλληλα αποτελέσματα, αφού πρώτα έχει γίνει ο ίδιος έλεγχος για τα δεξιά ή αριστερά ‘αδέρφια’ του κόμβου που βρέθηκε (για να προσθέσουμε στα αποτελέσματα και αυτά).

Ακολουθούν Screenshots από την εκτέλεση της εφαρμογής:

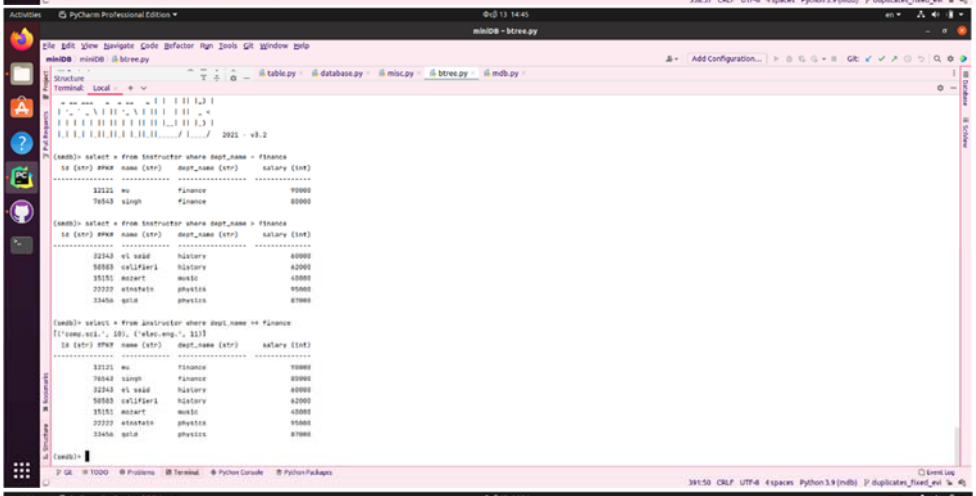

miniDB – database.py
Terminal: Local
(smdb)> create index inst_multi on instructor(dept_name,salary) using btree
Btree index created
(smdb)> select * from instructor
Exception: Selected column is not primary key
(smdb)> select * from instructor where dept_name = finance and salary > 70000


miniDB – btree.py
Terminal: Local
(smdb)> create index inst_id on instructor(id) using btree
Btree index created
(smdb)> select * from instructor
(smdb)> select * from instructor where id = 10101
Found

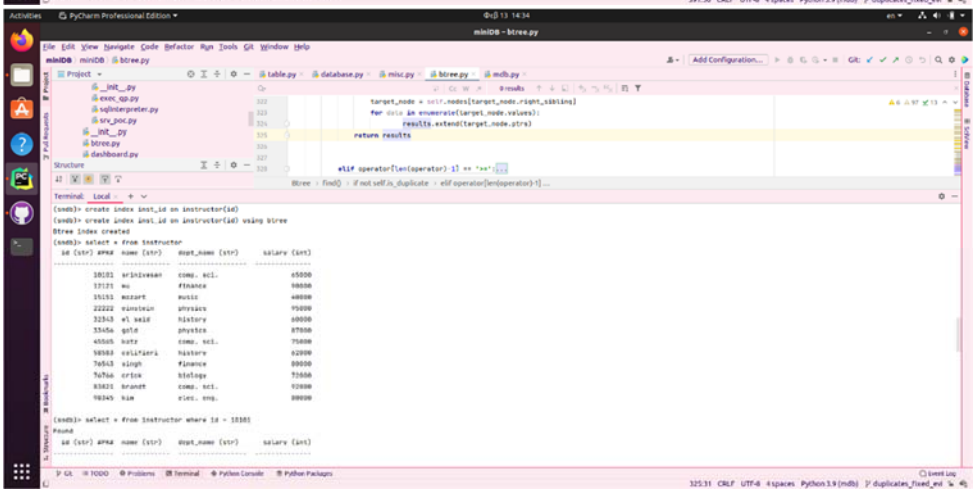

miniDB – btree.py
Terminal: Local
IndexError: string index out of range
(smdb) mirto@mirto-VirtualBox:~/Documents/GitHub/miniDB$ DB=smdb python3.9 mdb.py
(smdb)> select * from instructor where id <= 76766
(smdb)> select * from instructor where id >= 76766



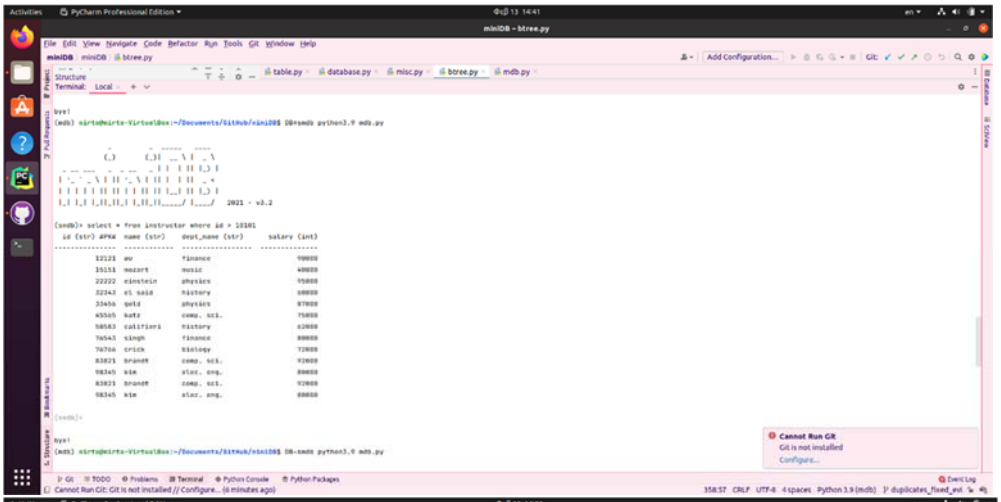
```
(sendb)> select * from instructor where id >= 76766
  id (str) #PK#  name (str)    dept_name (str)      salary (int)
---------------  ------------  -----------------  --------------
        76766    crick         biology                    72000
        83821    brandt        comp. sci.                 92000
        98345    kim           elec. eng.                 80000

(sendb)> select * from instructor where id < 76766
  id (str) #PK#  name (str)    dept_name (str)      salary (int)
---------------  ------------  -----------------  --------------
        76543    singh         finance                    80000
        58583    califieri     history                    62000
        45565    katz          comp. sci.                 75000
        33456    gold          physics                    87000
        32343    el said       history                    60000
        22222    einstein      physics                    95000
        15151    mozart        music                      40000
        12121    wu            finance                    90000
        10101    srinivasan    comp. sci.                 65000
(sendb)>
```

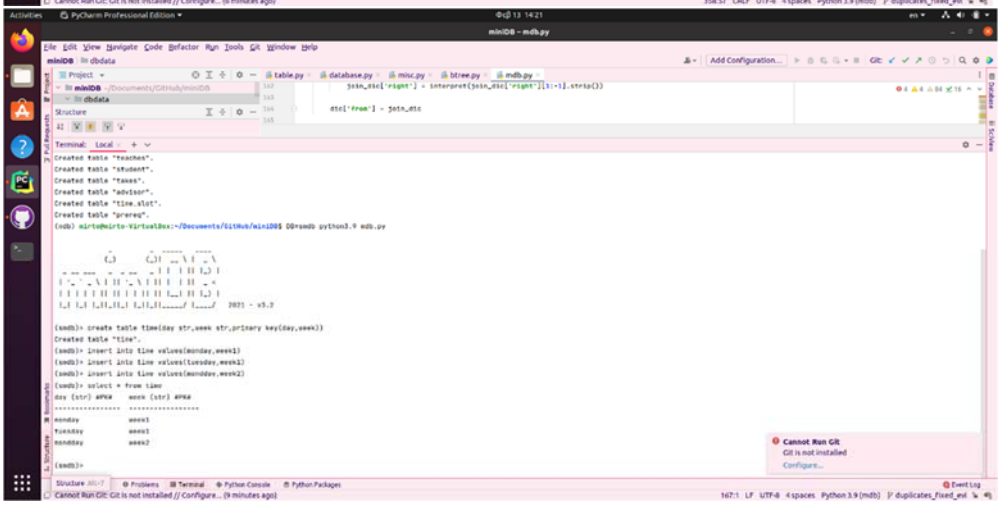


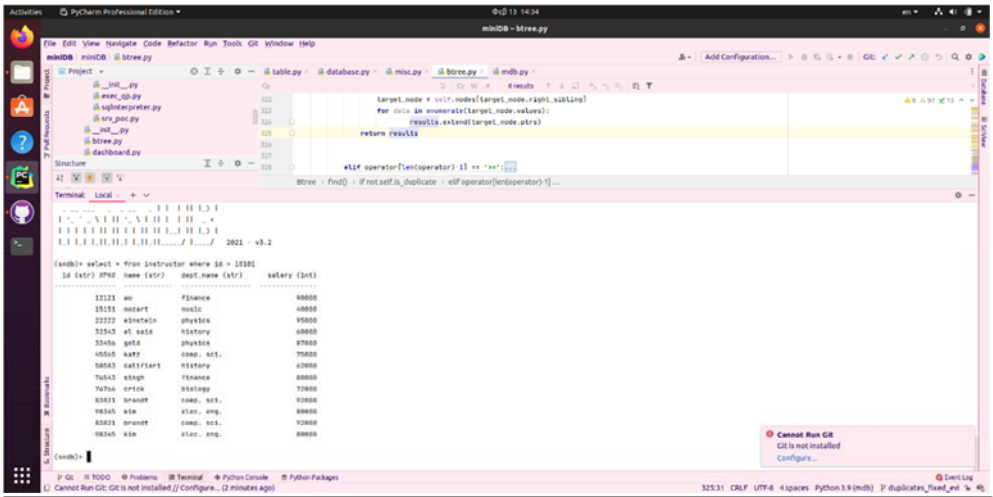
```
(sendb)> select * from instructor where dept_name = finance
  id (str) #PK#  name (str)    dept_name (str)      salary (int)
---------------  ------------  -----------------  --------------
        12121    wu            finance                    90000
        76543    singh         finance                    80000

(sendb)> select * from instructor where dept_name > finance
  id (str) #PK#  name (str)    dept_name (str)      salary (int)
---------------  ------------  -----------------  --------------
        32343    el said       history                    60000
        58583    califieri     history                    62000
        15151    mozart        music                      40000
        22222    einstein      physics                    95000
        33456    gold          physics                    87000

(sendb)> select * from instructor where dept_name >= finance
[('comp. sci.', 10), ('elec. eng.', 11)]
  id (str) #PK#  name (str)    dept_name (str)      salary (int)
---------------  ------------  -----------------  --------------
        12121    wu            finance                    90000
        76543    singh         finance                    80000
        32343    el said       history                    60000
        58583    califieri     history                    62000
        15151    mozart        music                      40000
        22222    einstein      physics                    95000
        33456    gold          physics                    87000
(sendb)>
```

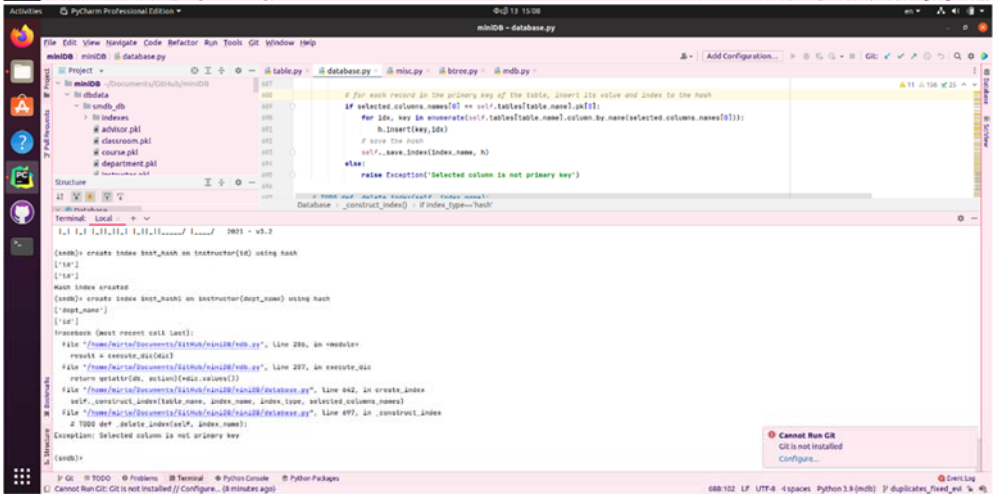


```
(sendb)> create index inst_id on instructor(id)
(sendb)> create index inst_id on instructor(id) using btree
Btree index created
(sendb)> select * from instructor
  id (str) #PK#  name (str)    dept_name (str)      salary (int)
---------------  ------------  -----------------  --------------
        10101    srinivasan    comp. sci.                 65000
        12121    wu            finance                    90000
        15151    mozart        music                      40000
        22222    einstein      physics                    95000
        32343    el said       history                    60000
        33456    gold          physics                    87000
        45565    katz          comp. sci.                 75000
        58583    califieri     history                    62000
        76543    singh         finance                    80000
        76766    crick         biology                    72000
        83821    brandt        comp. sci.                 92000
        98345    kim           elec. eng.                 80000

(sendb)> select * from instructor where id = 10101
Found
  id (str) #PK#  name (str)    dept_name (str)      salary (int)
---------------  ------------  -----------------  --------------
```


miniDB – btree.py
bye!
(mdb) mirto@mirto-VirtualBox:~/Documents/GitHub/miniDB$ DB=mdb python3.9 mdb.py
(mdb)> select * from instructor where id > 10101
id (str) #PK# name (str) dept_name (str) salary (int)
10101 srinivasan comp. sci. 65000
12121 wu finance 90000
15151 mozart music 40000
22222 einstein physics 95000
32343 el said history 60000
33456 gold physics 87000
45565 katz comp. sci. 75000
58583 califieri history 62000
76543 singh finance 80000
76766 crick biology 72000
83821 brandt comp. sci. 92000
98345 kim elec. eng. 80000
bye!
(mdb) mirto@mirto-VirtualBox:~/Documents/GitHub/miniDB$ DB=mdb python3.9 mdb.py
Cannot Run Git
Git is not installed
Configure...


miniDB – mdb.py
Created table "teaches".
Created table "student".
Created table "takes".
Created table "advisor".
Created table "time_slot".
Created table "prereq".
(mdb) mirto@mirto-VirtualBox:~/Documents/GitHub/miniDB$ DB=mdb python3.9 mdb.py
(mdb)> create table time(day str,week str,primary key(day,week))
Created table "time".
(mdb)> insert into time values(monday,week1)
(mdb)> insert into time values(tuesday,week1)
(mdb)> insert into time values(monddey,week2)
(mdb)> select * from time
day (str) #PK# week (str) #PK#
monday week1
tuesday week1
monddey week2
Cannot Run Git
Git is not installed
Configure...


miniDB – btree.py
(sndb)> select * from instructor where id > 10101
Cannot Run Git
Git is not installed
Configure...


miniDB – database.py
(sndb)> create index inst_hash on instructor(id) using hash
Hash index created
(sndb)> create index inst_hash1 on instructor(dept_name) using hash
Exception: Selected column is not primary key
Cannot Run Git
Git is not installed
Configure...