Τμήμα Πληροφορικής και Τηλεπικοινωνιών
Πανεπιστήμιο Αθηνών
Υλοποίηση Συστημάτων Βάσεων Δεδομένων (Κ18)

Περίοδος 2021-2022
Καθηγητής Ι. Ιωαννίδης
2η Εργασία

Ομάδα:
Ονομ/νυμο: Μιχαήλ Δεικτάκης ΑΜ: 1115200800018
Ονομ/νυμο: Ιωάννης Ψαρρός ΑΜ: 1115201800216
Ονομ/νυμο: Φίλιππος-Γεώργιος Μπράτης ΑΜ: 1115201400121

Για εκτελεση:
1)Για attribute surname
    make sht
    ./build/runner
2)Για attribute city
    make newsht
    ./build/runner

Γενικές Παραδοχές:

- Ένα hash file έχει το πρώτο του block με indexing 0 το οποίο κρατάει info. Αυτό στην περίπτωση μας είναι το global depth του hash table, το attribute, το length του attribute και το primary filename.
- Από το 2 ο έως το k ο block με indexing μέσα στο αρχείο 1-MAX_HASH_BLOCKS θα κρατάμε τα directories μας που δείχνουν τα blocks of records (buckets) με τη μορφή int ο οποίος είναι ο αριθμός του εκάστοτε block.
- Από το k+1 έως το N ο block κρατάμε τα buckets όπου 1 block = 1 bucket.
- Η μορφή ενός bucket είναι στα πρώτα BF_BLOCK_SIZE – 8 bytes αποθηκεύονται όσα blocks χωράνε ενώ στα επόμενα 4 bytes κρατάμε το local_depth του bucket και στα τελευταία 4 κρατάμε το πλήθος των records που περιέχει.
- Δεν έχουμε υλοποιήσει dynamic allocating του hash table , όμως δίνουμε τη δυνατότητα στον χρήστη μέσω της σταθερής μεταβλητής MAX_HASH_BLOCKS να ορίσει από την αρχή πόσα blocks θα χρησιμοποιηθούν για το hashing.
- Τέλος το hashing ενός index-key γινεται με προσθεση της τιμης καθε χαρακτηρα και κλήση της hash function που υλοποίησαμε στην πρωτη άσκηση.
- Επειδη δεν εχουμε προσβαση στο index descriptor απο την sht αλλα εχουμε προσβαση στο filename ανοιγουμε το αντιστοιχο αρχειο με καινούργιο descriptor και θεωρούμε οτι ειναι initialized.

SHT_Init

Η SHT_INIT ειναι άδεια καθώς όλες οι απαραίτητες αρχικοποιήσεις γίνονται από την HT_INIT και την main.Αρχικά θέτουμε στο block 0 το global depth, το attribute, το length του attribute και το primary filename που μας
δώθηκαν. Μετά δημιουργούμε τα blocks του hash table σύμφωνα με το MAX_HASH_BLOCKS.

Κάνουμε μετά allocate τα πρώτα SecondaryRecord blocks (buckets) και στη συνέχεια θέτουμε τα directories να δείχνουν με τη σειρά σε κάθε ένα από αυτά τα blocks θέτοντας το local_depth=global_depth και το πλήθος των records = 0. Θεωρούμε πως αν κάποιος φτιάξει ένα αρχείο με σχετικά μεγάλο global depth θα θέλει να αποθηκεύσει πολλά records οπότε τα buckets αυτά θα γεμίσουν.

SHT_OpenSecondaryIndex

Όταν ανοίγουμε ένα index αρχικά αποθηκεύουμε το αναγνωριστικό του αρχείου στον πίνακα με τα ανοιχτά αρχεία.
Μετά θέτουμε το global depth του αρχείου να είναι ίσο με αυτό που έχει αποθηκευτεί στο block 0 έτσι αν ένα αρχείο ανοίχτηκε από άλλο χρήστη και έχει αλλάξει το global depth δεν χάνεται.

SHT_CloseSecondaryIndex

Όταν κλείνουμε ένα αρχείο διαγράφουμε το αναγνωριστικό του από τον πίνακα με τα ανοιχτά αρχεία και θέτουμε το global depth στο block 0 ίσο με αυτό που έχουμε εκείνη τη στιγμή στον πίνακά μας.(Αυτό «πειράζουμε» όταν κάνουμε τα insert οπότε πρέπει να αποθηκευτεί πάλι στο αρχείο

SHT_SecondaryInsertEntry

Αρχικά κοιτάμε αν το global depth που έχουμε είναι μεγαλύτερο από αυτό που μπορεί να φτάσει ο πίνακας με τα directories οπότε αν είναι επιστρέφουμε error.
Μετά κοιτάμε αν το αρχείο στο οποίο προσπαθεί να εισαχθεί το secondary record είναι ανοιχτό.Στη συνέχεια κάνουμε hash το index-key και βρίσκουμε μέσω των directories σε ποιο bucket πρέπει να μπει το record. Επειδή τα blocks του hash table είναι περισσότερα από ένα βρίσκουμε πρώτα σε ποιο block είναι το directory (index)/(BF_BLOCK_SIZE/sizeof(int)) και τη θέση μέσω (index%(BF_BLOCK_SIZE/sizeof(int)).
Αφου βρούμε το bucket κοιτάμε πρώτα αν χωράει και αν ναι το τοποθετούμε.
Αν όχι κοιτάμε και το local depth του bucket:

- Αν είναι ίσο με global depth κάνουμε expand στα directories και μετά split. Το expand γινεται με βοηθητικά arrays όπου αποθηκεύουμε πρώτα τα buckets που δείχνουν, μετά φτιάχνουμε νέο array διπλάσιο και θέτουμε να δείχνει και αυτό σε σωστά buckets και τέλος ξαναγράφουμε το array στο table. Μετά κάνουμε allocate το νέο block και υλοποιούμε το split με βοηθητικό array of secondary records κρατάμε τα records που χρειάζεται να ξαναγίνουν hash

διαγράφουμε τα δεδομένα του bucket και τα ξαναγράφουμε σωστά καλώντας την insert για κάθε ένα από αυτά με το νέο global και local depth (+1).

- Αν είναι μικρότερο από global depth απλά κάνουμε τη διαδικασία του split που περιγράφεται παραπάνω με νέο local depth(+1)

SHT_SecondaryUpdateEntry

Αρχικά η Update πρέπει να καλεστεί από την main function αμέσως μετά την HT_InsertEntry. Προϋπόθεση είναι και πριν την HT insert να έχει κληθεί η συνάρτηση CreateArray για κατάλληλη δέσμευση μνήμης του πίνακα όπως και αφού δεν τον χρειαζόμαστε πλέον να κληθεί η DeleteArray που έχουν συμπεριληφθεί στο Hash_file.c.
Η update παίρνει τον πίνακα με τα αλλαγμένα tupleID και βρίσκοντας τη θέση για το κάθε index_key μέσω του hash function που έχουμε υλοποιήσει,ανάλογα με το attribute του secondary file γράφει στο block το αλλαγμένο tupleID για κάθε record.

SHT_PrintAllEntries

Αρχικα ελεγχουμε αν το αρχειο ειναι ανοιχτο.Επειτα αφου βρουμε στο secondaryFile ολα τα secondaryRecords με το index-key τοτε για καθε ενα μεσω του tupleID βρίσκουμε και μετα τυπωνουμε το αντιστοιχο record στο primaryFile. Ελέγχουμε πρώτα για surnames και μετα για cities

SHT_HashStatistics

Βρίσκουμε το σύνολο των records σε κάθε bucket και επιστρέφουμε το bucket με τον ελάχιστο αριθμό και αυτό με τον μέγιστο , και τέλος εκτυπώνεται και το average πλήθος records που έχουν τα buckets.

SHT_InnerJoin

Αρχικά στην SHT_InnerJoin υλοποιήσαμε μόνο την επιλογή για specific index_key. Παίρνει τη θέση στον πίνακα δυο διαφορετικών secondary files τα οποία αντιστοιχούν σε ένα πρωτεύον αρχείο η κάθε μία. Στην περίπτωση ενός πρωτεύοντος αρχείου με δύο δευτερεύοντα δεν θα φαινόταν κάποια λειτουργικότητα αφού αναγκαστικά από την εκφώνηση όλες οι εγγραφές του πρωτεύοντος εισάγονται και στα δευτερεύοντα αρχεία. Αυτό σημαίνει πως θα είχαν ακριβώς τις ίδιες εγγραφές.
Βρίσκουμε λοιπόν για κάθε εγγραφή με attribute ίσο με το index key που μας δίνεται του πρώτου δευτερεύοντος ευρετηρίου, τη θέση στην οποία βρίσκονται στο πρωτεύον, εκτυπώνουμε τα στοιχεία του και δίπλα εκτυπώνοται όλες οι εγγραφές με

τα στοιχεία τους που βρίσκουμε στο 2ο πρωτεύον ευρετήριο τις οποίες βρίσκουμε με τον ίδιο τρόπο από το 2ο δευτερεύον ευρετήριο. Επιλέξαμε να εκτυπώνονται σε μορφή πίνακα ώστε να είναι ευανάγνωστο, διότι αν τα εκτυπώναμε στη σειρά οι εγγραφές είναι αρκετές και θα δημιουργούσαν “περίεργη” εμφάνιση.