3η Εργασία (Ομαδική)
Υλοποίηση και Χρήση Βάσεων Δεδομένων
2015-2016

Κουκάς Κωνσταντίνος, Σιούλας Παναγιώτης,Τσιτσιμπίκος Κωνσταντίνος
1115201300075,1115201300161,1115201300187

Σύντομη Περιγραφή:

Η Άσκηση αυτή αφορούσε την υλοποίηση συναρτήσεων του επιπέδου διαχείρισης block και συγκεκριμένα με την Αποθήκευση σε Στήλες (Column Store). Σκοπός των συναρτήσεων ήταν η δημιουργία του επιπέδου αυτού και η διαχείριση αρχείων στα οποία είναι αποθηκευμένα ξεχωριστά τα πεδία των εγγραφών.

Εσωτερικές Δομές:

Όπως υποδείχτηκε για την οργάνωση των αρχείων υπάρχει ένα ξεχωριστό αρχείο πληροφορίας το οποίο θα περιέχει βασικά στοιχεία για την διαχείριση των αρχείων που θα περιέχουν τα δεδομένα του εκάστοτε πεδίου. Το αρχείο αυτό είναι ένα αρχείο Header_Info το οποίο περιέχει έναν πίνακα απο μια δομή (Header_Struct) που ουσιαστικά αποτελεί descriptor των αρχείων στηλών περιέχοντας τον τύπο των δεδομένων του πεδίου, το μέγεθος που έχει το πεδίο,πληροφορία για το άν το αρχείο είναι ήδη ανοιχτό και φυσικά το αναγνωριστικό αρχείου στηλών . Το Header_Info πέραν του πίνακα αυτού περιέχει έναν πίνακα ονομάτων των αρχείων στηλών, το όνομα του φακελού των αρχείων στηλών,και τον αριθμό πεδίων της εγγραφής.

Για την διευκόλυνση της υλοποίησης των ζητούμενων συναρτήσεων και πιο συγκεκριμένα για την αξιοποίηση δυνατότητων πολυμορφισμού, τη διευκόλυνση του χειρισμού των πράξεων και την απλοποίηση του κώδικα με σχεδιασμό υψηλού επιπέδου έχει δημιουργηθεί η κλάση Accessor. Αυτή ουσιαστικά υλοποιεί μια διεπαφή για τον χειρισμό ενός αρχείου στήλης. Έτσι, παρέχει τις πράξεις “Αρχικοποίηση αρχείου” (συνάρτηση InitFile, δημιουργεί το πρώτο block με πληροφορίες για το αρχείο, τα δεδομένα είναι από το block 1 και έπειτα), “Αρχικοποίηση προσπέλασης” (συνάρτηση IteratorSet που αρχικοποιεί τον εσωτερικό iterator του αρχείου πριν το πρώτο στοιχείο), “Επόμενο στοιχείο” (συνάρτηση Next, ο iterator δείχνει στο επόμενο στοιχείο προς προσπέλαση), “Εισαγωγή στοιχείου” (συνάρτηση Insert, εισάγει στο τέλος), “Πλήθος block που διαβάστηκαν” (συνάρτηση GetBlockReadCounter που επιστρέφει το πλήθος block που έχουν προσπελαστεί μέσα από τον Accessor) και οι pure virtual συναρτήσεις “Συγκριση” (συνάρτηση CompareCurrent) ,”Εκτυπωση” (συνάρτηση PrintCurrent). Με χρήση κληρονομικότητας, ορίζουμε τις υποκλάσεις IntAccessor, StrAccessor για τους τύπους δεδομένων που χρειαστήκαμε να χειριστούμε και υλοποιούν τις αφηρημένες συναρτήσεις.

Πάνω από την διεπαφή που παρέχει η Accessor έχει στηθεί μια ακόμα κλάση διεπαφή ColumnAccessor ώστε να υπάρχει σε λογικό επίπεδο διεπαφή χειρισμού αρχείου. Αυτή έχει σε πίνακα τους αναγκαίους Accessor και συντονίζει τις δράσεις τους με βάση τις ανάγκες μας. Έχει δυο στατικές συναρτήσεις OpenColumnFiles και CloseColumnFiles που τερματίζει όλα τα αρχεία που υποδεικνύει το Header_Info. Έχει επίσης τις πράξεις “Αρχικοποίηση” (InitColumnFiles), “Προσδιορισμός αρχείου για συγκρίσεις” (SetFieldCheck), “Προσδιορισμός αρχείων για εκτυπώσεις” (SetFieldCheck), “Αρχικοποίηση προσπέλασης” (IterationSet), “Επόμενο στοιχείο” (Next), “Σύγκριση” (CompareCurrent), “Εκτύπωση εγγραφής” (PrintCurrent), “Εισαγωγή Εγγραφής” (InsertRecord), “Πλήθος block που διαβάστηκαν” (GetBlockReadCounter). Το Iteration γίνεται παράλληλα σε όλα τα αρχεία και προσπελαύνονται τα block μόνο όταν χρειαστεί τιμή σε αυτά.

Με την χρήση των παραπάνω είναι λοιπόν δυνατή μια υψηλού επιπέδου, αφαιρετική διαχείριση των αρχείων αποθηκευμένων κατά στήλες. Αυτά βρίσκονται στο καθορισμένο directory. Το αρχείο κάθε στήλης έχει την εσωτερική δομή:
[elem1]:001 [elem2]:002 [elem3]:003 ….

Περιγραφή & Υλοποίηση Συναρτήσεων:

Σε όλες τις συναρτήσεις έχει χρησιμοποιηθεί έλεγχος σφαλμάτων χρησιμοποιώντας τον naive κανόνα περι επιστροφής αρνητικών τιμών σε περίπτωση οποιοδήποτε σφάλματος και μηδενικής τιμής σε περίπτωση ορθής εκτέλσεσης. Τα σφάλματα έχουν καθοριστεί και οριστεί μέσα στο πρόγραμμα.

- **CS_Init:** Αρχικοποιεί το επίπεδο διαχείρισης στηλών μέσω και της αρχικοποίησης του επιπέδου Block.
- **CS_CreateFiles:** Δημιουργεί τον φάκελο στον οποίο θα τοποθετηθούν τα αρχεία στηλών, δημιουργεί το αρχείο πληροφορίας Header_Info για το οποίο χρησιμοποιούνται οι γνωστές συναρτήσεις επιπέδου Block, ανοίγει το αρχείο αυτό και αποθηκεύει στο πρώτο Block του αρχείου την πληροφορία για τα αρχεία στηλών τα οποία και κατασκευάζει και αρχικοποιεί με τον κατάλληλο τρόπο.
- **CS_OpenFile:** Ανοίγει το αρχείο πληροφορίας που βρίσκεται στο δοσμένο κατάλογο διαβάζει την πληροφορία που περιέχει για τα αρχεία στηλών και ανοίγει και αυτά με τη σειρά του ελέγχοντας παράλληλα αν πρόκειται όντως για αρχεία στηλών ελέγχοντας το αναγνωριστικό που τους έχει δοθεί. Σε περίπτωση σφάλματος επιστρέφονται οι κατάλληλες ενημερώσεις σφάλματος.
- **CS_CloseFile:** Κλείνει τα ανοιγμένα αρχεία στηλών που έχουν ανοίξει κατα την διάρκεια εκτέλεσης του συστήματος. Αυτό γίνεται εκμεταλλευόμενο την μεταβλητή που περιέχεται στα Header_Struct του πίνακα του αρχείου πληροφορία που καθορίζει πότε ένα αρχείο είναι ανοιχτό και πότε όχι.
- **CS_InsertEntry:** Δημιουργεί column store accessor αρχικοποιώντας τον με τη δομή Header_Info. Στη συνέχεια με χρήση της InsertRecord του column accessor εισάγει στα αρχεία των στηλών μια εγγραφή χωρίζοντας την εγγραφή στα πεδία της και αποθηκεύοντας τα δεδομένα στο αντίστοιχο αρχείο στήλης που αντιστοιχεί στο εκάστοτε πεδίο. Ενημερώνει το αρχικό Block των αρχείων στηλών με ενημερώσεις των μεταδεδομένων που περιέχει για την εισαγωγή της νέας Εγγραφής. Επιστρέφει ανάλογα με την ορθότητα εκτέλεσης της συνάρτησης μήνυμα σφάλματος ή επιβεβαίωσης ορθότητας.
- **CS_GetAllEntries:** Δημιουργεί και αρχικοποιεί column store accessor. Καθορίζει, με βάση προηγούμενη είσοδο του χρήστη, το πεδίο προς σύγκριση και τα πεδία για εκτύπωση. Αρχικοποιεί και πραγματοποιεί την αναζήτηση διασχίζοντας παράλληλα τα αρχεία με τη συνάρτηση Next και ελέγχοντας τις εγγραφές με την CompareCurrent. Αν μια εγγραφή ικανοποιεί τις προϋποθέσεις εκτυπώνεται. Τελικά εκτυπώνεται ο αριθμός των μπλοκ δεδομένων που διαβάστηκαν.

Για τη διευκόλυνση του χρήστη έχει υλοποιηθεί μια main ώστε να γίνεται μια διαχείριση με εγγραφές με χρήση των συναρτήσεων των αρχείων στηλών. Στη main αυτή δίνεται στον χρήστη ένα menu με το οποίο μπορεί να κάνει επιλογές και να διαπιστώσει την ορθότητα του προγράμματος. Επισημαίνεται οτι στην εισαγωγή μονοπατιού από το χρήστη, αν έχει τη μορφή path/name, το path πρέπει να είναι υπάρχον μονοπάτι.
Δίνεται η επιλογή για εισαγωγή ξεχωριστών εγγραφών στην Βάση μεμονωμένα ή τη φόρτωση ολόκληρου αρχείου εγγραφών όπως αυτές δόθηκαν στα αρχεία DataSet (που αναρτήθηκαν μαζί με τη εκφώνηση της Εργασίας). Επίσης στη αναζήτηση εγγραφών ζητούνται αρχικά πεδίο και τιμή για σύγκριση (σε περίπτωση που δωθούν null εκτυπώνονται όλες οι εγγραφές) και στη συνέχεια τα πεδία για εκτύπωση. Για παράλειψη των υπόλοιπων πεδίων εκτύπωσης υπάρχει επιλογή newline για ολοκλήρωση της διαδικασίας και συνέχεια στην εκτύπωση των αποτελεσμάτων.

Εντολές Μεταγλώττισης/Εκτέλεσης:

Για την εργασία έχουν φτιαχτεί και 2 makefile με τα οποία γίνεται η μεταγλώττιση του προγράμματος σε 32bit – 64bit συστήματα.

Για μεταγλώττιση σε 32bit σύστημα η εντολή μεταγλώττισης είναι:

make -f makefile32

Ενώ για 64bit συστήματα η εντολή μεταγλώττισης είναι:

make -f makefile64

Μετά την μεταγλώττιση μπορείτε να εκτελέσετε το πρόγραμμα με την εντολή:

./prog