Basi di dati II
Esercizi di autovalutazione — 14 marzo 2013

Domanda 1 Si consideri una relazione *R(CodiceCliente,Cognome,Nome,Categoria)* con $N = 1.000.000$ ennuple. Con riferimento alla ricerca di tutti i clienti di una certa categoria, indicare il costo dell'accesso sequenziale e di quello diretto con indice su *Categoria* nei due casi seguenti (mostrare formule e valori numerici; supporre che l'indice abbia profondità $p = 4$ e che i fattori di blocco del file e dell'indice siano rispettivamente $f_R = 50$ e $f_C = 200$):

1. campo selettivo ($v_1 = 100.000$ valori diversi per *Categoria*)

costo accesso sequenziale:
costo accesso diretto:

2. campo poco selettivo ($v_2 = 20$ valori diversi per *Categoria*)

costo accesso sequenziale:
costo accesso diretto:

Domanda 2 Si consideri un B-tree con nodi intermedi che contengono due chiavi e tre puntatori e foglie con due chiavi, in cui vengano inserite chiavi (a partire dall'albero vuoto) nel seguente ordine: 12, 22, 32, 42, 52, 13, 14, 15, 16, 17, 18. Mostrare l'albero dopo l'inserimento di tre, cinque, sette chiavi e alla fine.

Esercitazioni pratiche, da consegnare
Svolgere sul DBMS didattico SimpleDB (`http://www.cs.bc.edu/ sciore/simpledb/`) i seguenti esercizi e consegnarli (su Moodle, vedere il sito) entro il 24 marzo (per chi intende sostenere le prove parziali) o tre giorni prima dell'esame (altrimenti).

Domanda 3
Come illustrato a lezione, SimpleDB utilizza la strategia naif per il rimpiazzamento dei buffer.
Implementare almeno una delle altre strategie (FIFO, LRU, clock) e mostrare che si ottengono così significativi benefici. Notare anche che la versione scaricabile di SimpleDB utilizza pochissimi buffer (8), e che per migliorare le prestazioni è necessario aumentarli; valutare quale può essere un numero adeguato per ottenere benefici.
Individuare opportune modalità per mostrare le differenze di prestazioni. Si suggerisce di lavorare con un programma client, documentando il numero di letture di blocchi effettuate. Allo scopo, si può procedere come segue:

- Modificare la classe FileMgr in modo che supporti statistiche sul numero di blocchi letti (aggiungere uno o più metodi), con riferimento a ciascuno dei file usati da simpleDB (così si potrà capire dove è l'overhead)
- Modificare il metodo commit della classe RemoteConnectionImplementation (in simpledb.remote) in modo che stampi queste statistiche (si deve accedere al FileMgr che può essere ottenuto con il metodo SimpleDB.fileMgr, nel package simpledb.server)

Si possono vedere risultati interessanti anche eseguendo attraverso il client SQLInterpreter una semplice interrogazione, quale `SELECT SName FROM Student` sulla base di dati studentdb fornita con il sistema.
Consegnare le classi modificate e una breve relazione che illustri le modifiche e descriva i test effettuati, mostrando i benefici ottenuti.
Domanda 4
Utilizzando SimpleDB (o meglio, i suoi moduli di livello più basso), scrivere una classe di test che esegue alcune operazioni su un file. Avvertenze:

- porre la classe nel package `simpledb.record`
- la classe deve disporre di alcuni degli oggetti del server; allo scopo, includere una chiamata al metodo init di SimpleDB:
  `SimpleDB.init("studentdb");`
- RecordFile fa riferimento ad una transazione, che deve essere creata allo scopo:
  `Transaction tx = new Transaction();`
- RecordFile fa riferimento allo schema logico e a quello fisico della tabella che viene implementata; vanno entrambi creati nella classe di test:
  `Schema sch = new Schema();`
  `sch.addIntField( ..... );` e/o `sch.addStringField( ..... );` (due o tre campi, non di più)
  ...
  `TableInfo ti = new TableInfo("prova",sch);`

Nella classe di test, effettuare

1. l'inserimento di una sequenza di 10000 record (con valori generati casualmente)
2. la lettura di tutti i record
3. l'eliminazione di una parte dei record (orientativamente il 50% dei record; ad esempio, quelli per cui il valore di un campo soddisfa una certa condizione)
4. la scansione di tutti i record (con la lettura di un campo numerico e un qualche calcolo, ad esempio il valore medio)
5. l'inserimento di altri 7000 record (sempre con valori generati casualmente)
6. nuovamente la lettura di tutti i record

Per ciascuna delle operazioni, stampare il numero di record letti e scritti e il numero di accessi a memoria secondaria per il file in questione (anche in questo caso possono servire metodi che stampano le statistiche, che vanno però richiamati nella classe di test).
Anche in questo caso, consegnare le classi scritte o modificate e una breve relazione che illustri il codice sviluppato e descriva i test effettuati, commentando il comportamento rilevato, ad esempio l'evoluzione del numero di blocchi del file.