Basi di dati II

Prova parziale — 28 marzo 2012 — Compito A
Cenni sulle soluzioni
(per il compito A, gli altri sono simili, le varianti del testo sono in rosso)

Rispondere su questo fascicolo.
Tempo a disposizione: un'ora e quindici minuti.

Cognome _____ **Nome** _____ **Matricola** _____ **Ordin.** _____

Domanda 1 (25%)
Considerare una tabella `R` appena creata (e quindi vuota), con le seguenti ipotesi

- `R` è definita su due campi, `A` di lunghezza $a = 4$ byte e `B` di lunghezza $b = 14$ byte, senza vincoli espliciti di chiave (e quindi le operazioni si possono fare senza verifiche particolari);
- la struttura fisica utilizzata per `R` è heap, senza indici, con una memorizzazione a lunghezza fissa (in cui supponiamo che, oltre ai byte necessari per i campi ne servano 2 ulteriori per la memorizzazione) e in cui si marcano come liberi gli spazi dei record eliminati, riutilizzandoli per successivi inserimenti (come avviene in SimpleDB);
- il sistema utilizza blocchi di dimensione $D = 2$ Kbyte (approssimabili a 2000).

In tale contesto, supporre che vengano eseguite, nell'ordine, le seguenti operazioni

1. inserimento di $N = 100.000$ ennuple
2. eliminazione di $N/2 = 50.000$ delle ennuple prima inserite (sulla base di una condizione verificabile durante la scansione)
3. dopo la conclusione e la chiusura della scansione precedente, inserimento di altre N ennuple

Rispondere alle domande seguenti, indicando formule e valori numerici:

Fattore di blocco f per la relazione `R`:

$$f = D/(a + b + 2) = 2000/20 = 100$$

Numero blocchi occupati da `R` dopo la prima serie di inserimenti (punto 1):

$$N/f = 100.000/100 = 1000$$

Numero blocchi occupati da `R` dopo le eliminazioni di cui al punto 2:

$$N/f = 100.000/100 = 1000$$

(lo spazio libero non viene riutilizzato)

Numero blocchi occupati da `R` dopo la seconda serie di inserimenti (punto 3):

$$3/2 \times N/f = 1500$$

(lo spazio libero viene riutilizzato)

Numero di scritture in memoria secondaria (per le pagine dei dati e quelle del log) per la prima serie di inserimenti, supponendo che i record di log abbiano una lunghezza pari a circa il doppio di quella dei record del file, con riferimento ad un programma che utilizzi una transazione separata per ciascun inserimento

- numero di scritture di pagine di log:

  almeno $N = 100.000$, una per transazione

- numero di scritture di pagine di dati:

  con una strategia undo-only, una per transazione, $N = 100.000$; altrimenti, probabilmente di meno, anche solo $N/f = 1000$, una per blocco

Come nel caso precedente, ma con riferimento ad un programma che utilizzi un'unica transazione per tutti gli inserimenti (e supponendo che non vi siano altre transazioni attive)

- numero di scritture di pagine di log:

  dovrebbe essere possibile scrivere tutto il log al momento del commit e quindi (poiché i record sono grandi il doppio di quelli del file): $2 \times N/f = 2000$

- numero di scritture di pagine di dati:

  in ogni caso si dovrebbe avere $N/f = 1000$, una per blocco, o poco più

Domanda 2 (25%)
Si consideri una base di dati con le relazioni

- R1(A,B,C,D) con
  - vincolo di integrità referenziale fra l'attributo D e la chiave E della relazione R2
  - N_1=2.000.000 ennuple e fattore di blocco f_1=200
  - b=20 valori diversi sull'attributo B (tutti gli interi compresi fra 1 e b)
  - c=200.000 valori diversi sull'attributo C (tutti gli interi compresi fra 1 e c)
  - una struttura disordinata, un indice sulla chiave primaria A e un altro sull'attributo C;
- R2(E,F,G) con
  - N_2=1.000.000 ennuple e fattore di blocco f_2=10
  - una struttura disordinata, un indice sulla chiave primaria E

Supponendo che:

- gli indici abbiano tutti p=4 livelli (contando anche radice e foglie) e fattore di blocco massimo f_i=100
- il sistema esegua sempre i join come nested loop
- ogni operazione possa contare su un numero di pagine di buffer pari a circa q=150,

valutare il costo (indicandolo in modo sia simbolico sia numerico) di ciascuna delle interrogazioni seguenti:

```
select *
from R1 join R2 on D=E
```

$$\frac{N_1}{f_1} + N_1(p-2+1) = 6.010.000 \text{ (nei compiti B e D: } 3.010.000)$$

- scansione di R1: $\frac{N_1}{f_1}$
- un accesso diretto a R2 per ogni ennupla di R1: costo unitario profondità p dell'indice, meno 2 per i livelli nel buffer più uno per l'accesso al record

```
select *
from R1 join R2 on D=E
where B=5
```

$$\frac{N_1}{f_1} + \frac{N_1}{b}(p-2+1) = 310.000$$

- scansione di R1 per la selezione: $\frac{N_1}{f_1}$
- un accesso diretto a R2 per ogni ennupla nel risultato della selezione, costo unitario come sopra

```
select *
from R1 join R2 on D=E
where C=502
```

$$p + \frac{N_1}{c} + \frac{N_1}{c}(p-1+1) = \text{ ca. } 55$$

- accesso diretto a R1 per la selezione: p per l'indice e $\frac{N_1}{c}$ per le ennuple
- un accesso diretto a R2 per ogni ennupla nel risultato della selezione, costo unitario come sopra, con la differenza che si può ipotizzare la radice nel buffer, ma non gli altri livelli

Domanda 3 (25%)
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: 1, 3, 5, 7, 9, 11, 13, 15. Mostrare l'albero dopo l'inserimento di cinque chiavi e alla fine.

provare con l'applet suggerita a lezione: `http://slady.net/java/bt/view.php?w=700`

Mostrare poi l'albero dopo l'eliminazione della chiave 3 dall'ultimo albero ottenuto in risposta alla domanda precedente.

provare con l'applet suggerita a lezione: `http://slady.net/java/bt/view.php?w=700`

Domanda 4 (25%)
Nella figura seguente è schematizzato un piccolissimo buffer con quattro pagine (numerate da 0 a 3), il cui stato viene descritto, per ciascuna pagina, da (i) un intero che indica il numero di pin su di essa (quindi 0 indica che la pagina è libera) (ii) un riferimento al blocco che per ultimo è stato caricato nella pagina; (iii) l'istante in cui è stato effettuato l'ultimo caricamento; (iv) l'istante in cui la pagina è stata per l'ultima volta liberata (se è libera) (v) un booleano che indica se la pagina è sporca (1 indica che è stata modificata dopo il caricamento o dopo l'ultima scrittura).

Pagina del buffer:	0	1	2	3
numero di pin sulla pagina	1	2	0	1
blocco	70	33	35	47
istante load	1	7	3	9
istante unpin			8	
dirty	1	0	0	0

Si supponga ora che vengano eseguite (a partire dall'istante 10, dopo che all'istante 9 è stata caricata la pagina 3) le seguenti operazioni (in cui l'argomento delle pin e unpin è il blocco di interesse, con setXXX si indica un aggiornamento di un qualche valore nel blocco e con flushAll il flush dell'intero buffer):

Istante	10	11	12	13	14	15
Operazione	unpin(70),	pin(60),	setXXX(60,...)	unpin(60),	flushAll	setXXX(47,...)

Istante	16	17	18	19	20	21	22
Operazione	unpin(47)	pin(70),	setXXX(70,...)	unpin(70),	pin(60),	unpin(60),	pin(70)

Riempiendo la tabella seguente, indicare, per ciascuna delle strategie e per ciascuna delle pin, (1) quale pagina del buffer viene utilizzata, (2) se il blocco corrispondente viene letto dal disco e (3) se viene eseguita una scrittura su disco.

		naif			LRU			clock		
Istante	Operazione	Pagina	Legge?	Scrive?	Pagina	Legge?	Scrive?	Pagina	Legge?	Scrive?
11	pin(60)	0	sì	sì	2	sì	no	0	sì	sì
17	pin(70)	0	sì	no	0	no	no	2	sì	no
20	pin(60)	0	sì	sì	2	no	no	0	no	no
22	pin(70)	0	sì	no	0	no	no	2	no	no

Eventuali osservazioni Nei compiti B e D si usa la pagina 1 invece che la 2.
Si noti che naif riusa sempre la stessa pagina e quindi fa più letture