Basi di dati II
Esercizi di autovalutazione — 2 aprile 2012
Cenni sulle soluzioni

Domanda 1 Come noto, esistono tecniche leggermente diverse per il recovery, che prevedono solo redo (**"redo-only"**), solo undo (**"undo-only"**) oppure tanto undo quanto redo (**"undo-redo"**). Le tecniche differiscono oltre che nella procedura di recovery, anche nella modalità di esecuzione delle scritture effettive su disco (operazioni di "flush" delle pagine di buffer), che possono essere le seguenti (con riferimento ad un utilizzo di checkpoint non quiescente):

flush-nel-commit le pagine modificate da una transazione vengono scritte su disco (subito) prima del commit della transazione stessa;

flush-all-nel-ckpt nel corso del checkpoint (subito prima della sua conclusione), vengono scritte su disco le pagine modificate da tutte le transazioni;

flush-committed-nel-ckpt nel corso del checkpoint (subito prima della sua conclusione), vengono scritte su disco le pagine modificate dalle transazioni andate in commit.

Nella tabella seguente, indicare per ciascuna tecnica quali approcci alla flush possono o debbono essere utilizzati (indicare il nome sintetico e fornire una breve spiegazione; si noti che potrebbe essere utilizzabile una sola tecnica oppure più di una; in questo secondo caso, indicare e spiegare quale sarebbe preferibile).

1. **"undo-only"**

flush-nel-commit, in quanto garantisce che non ci sia mai bisogno di redo (i dati sono su disco al momento del commit); non si può aspettare il checkpoint, perché se ci fosse un crash prima di esso allora servirebbero redo

2. **"undo-redo"**

flush-all-nel-ckpt: serve per garantire che le transazioni concluse al checkpoint non abbiano bisogno di essere ribadite; può portare ad undo per transazioni non concluse, ma è più semplice da gestire delle altre due, che pure sarebbero corrette

3. **"redo-only"**

flush-committed-nel-ckpt: serve a garantire che le transazioni andate in commit prima del checkpoint non siano da rifare; non scrive altro su disco e quindi non c'è necessità di undo. Non va bene la **flush-nel-commit** perché ci potrebbe essere un crash fra la flush e il commit e allora servirebbe l'undo. Analogamente, non va bene la **flush-all-nel-ckpt** perché si scriverebbero su disco dati non confermati da commit e quindi portebbero servire redo.

Domanda 2 Si considerino un sistema con blocchi di dimensione $B = 1000$ byte e puntatori ai blocchi di $P = 2$ byte e una relazione $R(\underline{A},C,D)$ di cardinalità pari circa a $T = 2.000.000$, con ennuple di $L = 20$ byte e campo chiave A di $L_A = 5$ byte e campo C di $L_C = 8$ byte. Il campo C non è chiave e ogni suo valore è presente, mediamente, in $e = 4$ ennuple. Valutare i pro e i contro relativamente alla presenza di un indice secondario sulla chiave A e di un altro, pure secondario, su C, in presenza del seguente carico applicativo:

1. inserimento di una nuova ennupla (con verifica del soddisfacimento del vincolo di chiave), con frequenza $f_1 = 20.000$
2. ricerca di una ennupla sulla base del valore della chiave A, con frequenza $f_2 = 10.000$
3. ricerca di ennuple sulla base del valore di C, con frequenza $f_3 = 10$

Ragionare in termini di costo degli accessi a memoria secondaria, assumendo disponibilità di buffer che permettano di mantenere stabilmente in memoria due livelli per ciascun indice e considerando che la relazione possa essere memorizzata in forma contigua (assumendo che il tempo di posizionamento della testina sia $r = 100$ volte maggiore del tempo di lettura). Trascurare le problematiche relative alla concorrenza e considerare il costo della lettura pari a quello della scrittura. Rispondere negli spazi sottostanti, in forma sia simbolica sia numerica.

Costo scansione sequenziale (SEQ) in multipli del tempo di posizionamento
numero di blocchi: NB $= \frac{T}{B/L} = 40.000$
numero di accessi: $1 + (\text{NB} - 1) \times \frac{1}{r} = \sim 400$

Numero livelli indice su A (PA)
il fattore di blocco dell'indice è 1.000/7, circa 160 e quindi, con un riempimento di circa il 60-70%, il fan-out è circa 100 e quindi i livelli necessari sono 4

Numero livelli indice su C (PC)
il fattore di blocco dell'indice è 1.000/10, circa 100 e quindi, con un riempimento di circa il 60-70%, il fan-out è circa 65 e quindi i livelli necessari sono pure 4

	nessun indice	indice solo su A	indice solo su C	indice su A e su C
Costo unit. Op. 1	SEQ = 400	PA - 2 + 1 + 1 = 4	SEQ + ... = $\sim$ 400	PA - 2 + PC - 2 + 3 = 7
Costo unit. Op. 2	SEQ = 400	PA - 2 + 1 = 3	SEQ = 400	PA - 2 + 1 = 3
Costo unit. Op. 3	SEQ = 400	SEQ = 400	PC - 2 + e = 6	PC - 2 + e = 6
Costo complessivo	ca 12.000.000	ca 110.000	ca 12.000.000	ca 170.000