Basi di dati II, primo modulo — 6 settembre 2011

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

Cognome _____ **Nome** _____ **Matricola** _____

Domanda 1 (20%) Considerare i due seguenti scenari in ciascuno dei quali due client diversi inviano richieste ad un gestore del controllo di concorrenza. Ciascun client può inviare una richiesta solo dopo che è stata eseguita o rifiutata la precedente (se invece una richiesta viene bloccata da un lock, allora il client rimane inattivo fino alla concessione o allo scadere del timeout). Si supponga che, in caso di stallo, abortisca la transazione che ha avanzato la richiesta per prima. In caso di abort, si ignorino le successive richieste della transazione che ha abortito (senza rilanciarla).

scenario 1

client 1	client 2
read(x)	
	read(x)
x = x + 10	
write(x)	
	x = x + 20
	write(x)
commit	
	commit

scenario 2

client 1	client 2
read(x)	
	read(x)
	x = x + 20
	write(x)
	commit
read(x)	
commit	

Considerare uno scheduler che utilizzi il controllo di concorrenza basato su 2PL e livelli di isolamento SERIALIZABLE e READ COMMITTED. Assumiamo che (come avviene di solito) 2PL preveda

- SERIALIZABLE: lock a due fasi stretto, con lock condivisi per letture e esclusivi per scritture.
- READ COMMITTED: lock condivisi per la lettura senza 2PL (possono essere rilasciati prima della acquisizione di altri lock) ed esclusivi per la scrittura con 2PL stretto (mantenuti fino a commit o abort).

Mostrare il comportamento dello scheduler nei due casi seguenti, supponendo che il valore iniziale dell'oggetto x sia 100. Indicare le operazioni che vengono eseguite nell'ordine con, per ciascuna, il valore che viene letto o scritto. In conclusione, per ciascun caso, dire se si verificano o meno anomalie.

Scenario 1 SERIALIZABLE	Scenario 2 READ COMMITTED

Domanda 2 (20%) 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 il fattore di blocco del file sia $f_R = 50$; trascurare i benefici derivanti dai buffer):

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 3 (15%) 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: 33, 1, 2, 8, 10, 12, 14, 16, 20, 24, 4, 6, 13. Mostrare l'albero dopo l'inserimento di quattro, sette, dieci chiavi e alla fine.

Domanda 4 (25%) Si consideri la seguente base di dati, relativa alle ricette acquisite da un insieme di farmacie:

- Ricette(Numero, CodFarmacia, CFPaziente, Data)
- Farmacie(CodFarmacia, Nome, Via, NumeroCivico, Città)
- ElementiRicetta(NumeroRicetta, CodFarmaco, Quantità)
- Farmaci(Codice, Descrizione, CodMolecola, CodCasa, Prezzo, Fascia)
- Molecole(CodMolecola, Descrizione)
- Pazienti(CF, Cognome, Nome, DataNascita, Via, NumeroCivico, Città)
- CaseFarmaceutiche(CodCasa, Nome)
- ASL(Codice, Nome)
- Territorio(Via, Città, NumeroCivico, ASL)

Ci sono dati che cambiano nel tempo fra cui prezzi e fasce ('A', 'B' o 'C') dei farmaci e indirizzi dei pazienti.
Costruire, in tale contesto, uno schema a stella che permetta di analizzare le prescrizioni (quantità e prezzi complessivi) rispetto a

- data (dimensione standard i cui dettagli possono essere omessi);
- farmaci, con le loro proprietà (molecola e casa farmaceutica);
- prescrizione di farmaci nella stessa ricetta
- ASL di residenza e fascia d'età (ad esempio, 0-3,4-17, 18-30, ...; ma potrebbero variare) dei pazienti;
- ASL della farmacia

Supporre che, per ovvie ragioni di privacy, non possano essere riportati dati che permettano di risalire alle identità dei pazienti (CF, cognome, nome, data di nascita e indirizzo) **Indicare esplicitamente la grana dei fatti.**

Grana dei fatti:

Schema dimensionale:

Domanda 5 (20%) Come noto, esistono varie strategie utilizzabili da parte del gestore del buffer per scegliere la pagina libera (unpinned) da utilizzare (e quindi il blocco da rimpiazzare) per eseguire una pin, fra cui le seguenti:

- *naif*: si sceglie una qualunque pagina libera (ad esempio la prima che si incontra scandendo un array o una lista sempre dall'inizio)
- *FIFO*: si sceglie, fra le pagine libere, quella caricata da più tempo
- *LRU* (least recently used): si sceglie, fra le pagine libere, quella utilizzata meno di recente (cioè quella liberata da più tempo)

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).

Pagina del buffer:	0	1	2	3
numero di pin sulla pagina	1	0	2	0
blocco	50	35	33	47
istante load	1	3	7	5
istante unpin		8		6

Si supponga ora che vengano eseguite (a partire dall'istante 10) le seguenti operazioni (in cui l'argomento del metodo è il blocco di interesse):

10	11	12	13	14	15	16	17
unpin(50),	pin(60),	unpin(60),	pin(50),	unpin(50),	pin(60),	unpin(60),	pin(50)

Riempiendo la tabella seguente, indicare quale pagina del buffer viene utilizzata per ciascuna delle pin e se viene effettivamente eseguita una lettura su disco (si noti che ignoriamo le scritture e infatti non abbiamo indicato se le pagine sono sporche), in ciascuna delle strategie. Commentare brevemente il comportamento che si osserva.

Istante		naif		FIFO		LRU	
		Pagina	Lettura?	Pagina	Lettura?	Pagina	Lettura?
11	pin(60)						
13	pin(50)						
15	pin(60)						
17	pin(50)						

Commento:

Basi di dati II, primo modulo — 6 settembre 2011
Cenni sulle soluzioni

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

Cognome _____ **Nome** _____ **Matricola** _____

Domanda 1 (20%) Considerare i due seguenti scenari in ciascuno dei quali due client diversi inviano richieste ad un gestore del controllo di concorrenza. Ciascun client può inviare una richiesta solo dopo che è stata eseguita o rifiutata la precedente (se invece una richiesta viene bloccata da un lock, allora il client rimane inattivo fino alla concessione o allo scadere del timeout). Si supponga che, in caso di stallo, abortisca la transazione che ha avanzato la richiesta per prima. In caso di abort, si ignorino le successive richieste della transazione che ha abortito (senza rilanciarla).

scenario 1

client 1	client 2
read(x)	
	read(x)
x = x + 10	
write(x)	
	x = x + 20
	write(x)
commit	
	commit

scenario 2

client 1	client 2
read(x)	
	read(x)
	x = x + 20
	write(x)
	commit
read(x)	
commit	

Considerare uno scheduler che utilizzi il controllo di concorrenza basato su 2PL e livelli di isolamento SERIALIZABLE e READ COMMITTED. Assumiamo che (come avviene di solito) 2PL preveda

- SERIALIZABLE: lock a due fasi stretto, con lock condivisi per letture e esclusivi per scritture.
- READ COMMITTED: lock condivisi per la lettura senza 2PL (possono essere rilasciati prima della acquisizione di altri lock) ed esclusivi per la scrittura con 2PL stretto (mantenuti fino a commit o abort).

Mostrare il comportamento dello scheduler nei due casi seguenti, supponendo che il valore iniziale dell'oggetto x sia 100. Indicare le operazioni che vengono eseguite nell'ordine con, per ciascuna, il valore che viene letto o scritto. In conclusione, per ciascun caso, dire se si verificano o meno anomalie.

Scenario 1
SERIALIZABLE

client 1		client 2	
read(x)	legge 100		
		read(x)	legge 100
x = x + 10	x vale 110		
xlock(x)	bloccata		
		x = x + 20	x vale 120
		xlock(x)	bloccata
abort			
		write(x)	scrive 120
		commit	

non c'è anomalia

Scenario 2
READ COMMITTED

client 1		client 2	
read(x)	legge 100		
		read(x)	legge 100
		x = x + 20	x vale 120
		write(x)	scrive 120
		commit	
read(x)	legge 120		
commit			

anomalia: lettura inconsitente

Domanda 2 (20%) 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 il fattore di blocco del file sia $f_R = 50$; trascurare i benefici derivanti dai buffer):

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

costo accesso sequenziale:
$N/f_R = 20.000$

costo accesso diretto:
$p + N/v_1 = \approx 14$

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

costo accesso sequenziale:
$N/f_R = 20.000$

costo accesso diretto:
$p + N/v_2 + \ldots = \approx 50.000$

Domanda 3 (15%) 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: 33, 1, 2, 8, 10, 12, 14, 16, 20, 24, 4, 6, 13. Mostrare l'albero dopo l'inserimento di quattro, sette, dieci chiavi e alla fine.

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

Domanda 4 (25%) Si consideri la seguente base di dati, relativa alle ricette acquisite da un insieme di farmacie:

- Ricette(Numero, CodFarmacia, CFPaziente, Data)
- Farmacie(CodFarmacia, Nome, Via, NumeroCivico, Città)
- ElementiRicetta(NumeroRicetta, CodFarmaco, Quantità)
- Farmaci(Codice, Descrizione, CodMolecola, CodCasa, Prezzo, Fascia)
- Molecole(CodMolecola, Descrizione)
- Pazienti(CF, Cognome, Nome, DataNascita, Via, NumeroCivico, Città)
- CaseFarmaceutiche(CodCasa, Nome)
- ASL(Codice, Nome)
- Territorio(Via, Città, NumeroCivico, ASL)

Ci sono dati che cambiano nel tempo fra cui prezzi e fasce ('A', 'B' o 'C') dei farmaci e indirizzi dei pazienti. Costruire, in tale contesto, uno schema a stella che permetta di analizzare le prescrizioni (quantità e prezzi complessivi) rispetto a

- data (dimensione standard i cui dettagli possono essere omessi);
- farmaci, con le loro proprietà (molecola e casa farmaceutica);
- prescrizione di farmaci nella stessa ricetta
- ASL di residenza e fascia d'età (ad esempio, 0-3,4-17, 18-30, ...; ma potrebbero variare) dei pazienti;
- ASL della farmacia

Supporre che, per ovvie ragioni di privacy, non possano essere riportati dati che permettano di risalire alle identità dei pazienti (CF, cognome, nome, data di nascita e indirizzo) **Indicare esplicitamente la grana dei fatti.**

Grana dei fatti: la grana scelta è "singole prescrizioni"

Schema dimensionale:

- FattiPrescrizioni(KData, KFarmaco, KRicetta, KASLfarmacia, KASLpaziente, KFasciaEtà, Quantità, Importo)
- Farmaci(KFramaco, Codice, Descrizione, CodMolecola, DescrizioneMolecola, CodCasa, NomeCasa, Fascia)
- ASL(KASL, CodiceASL, Nome)
- FasciaEtà(KFasciaEtà, ...)
- Data(KData, ...)

Commenti:

- sono indicate chiavi ad hoc per le dimensioni
- *Ricetta* è una dimensione "degenere," cioè senza attributi
- per la privatezza, si eliminano tutte le informazioni personali, indicando solo ASL del paziente e fascia d'età; quindi la grana scelta è "singole prescrizioni;" in effetti, vista la presenza di *KRicetta*, tutte le dimensioni, a parte *Ricetta* e *Farmaci*, sono secondarie

Domanda 5 (20%) Come noto, esistono varie strategie utilizzabili da parte del gestore del buffer per scegliere la pagina libera (unpinned) da utilizzare (e quindi il blocco da rimpiazzare) per eseguire una pin, fra cui le seguenti:

- *naif*: si sceglie una qualunque pagina libera (ad esempio la prima che si incontra scandendo un array o una lista sempre dall'inizio)
- *FIFO*: si sceglie, fra le pagine libere, quella caricata da più tempo
- *LRU* (least recently used): si sceglie, fra le pagine libere, quella utilizzata meno di recente (cioè quella liberata da più tempo)

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).

Pagina del buffer:	0	1	2	3
numero di pin sulla pagina	1	0	2	0
blocco	50	35	33	47
istante load	1	3	7	5
istante unpin		8		6

Si supponga ora che vengano eseguite (a partire dall'istante 10) le seguenti operazioni (in cui l'argomento del metodo è il blocco di interesse):

10	11	12	13	14	15	16	17
unpin(50),	pin(60),	unpin(60),	pin(50),	unpin(50),	pin(60),	unpin(60),	pin(50)

Riempiendo la tabella seguente, indicare quale pagina del buffer viene utilizzata per ciascuna delle pin e se viene effettivamente eseguita una lettura su disco (si noti che ignoriamo le scritture e infatti non abbiamo indicato se le pagine sono sporche), in ciascuna delle strategie. Commentare brevemente il comportamento che si osserva.

Istante		naif		FIFO		LRU	
		Pagina	Lettura?	Pagina	Lettura?	Pagina	Lettura?
11	pin(60)	0	sì	0	sì	3	sì
13	pin(50)	0	sì	1	sì	0	no
15	pin(60)	0	sì	0	no	3	no
17	pin(50)	0	sì	1	no	0	no

Commento:

Si noti che naif riusa sempre la stessa pagina e quindi fa più letture