Tecnologie e architetture per la gestione dei dati — 18 maggio 2023

Tempo a disposizione: un'ora.

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

Tecnologie e architetture per la gestione dei dati — 18 maggio 2023

Domanda 1 (20%)

Considerare lo scenario a fianco in cui tre client diversi inviano richieste ad un gestore della 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 al timeout). Si supponga che, in caso di stallo, abortisca la transazione che ha avanzato la richiesta per prima. In caso di abort, si supponga che il client rilanci la stessa transazione (subito dopo l'esecuzione delle altre azioni in attesa sullo stesso dato). Considerare uno scheduler con controllo di concorrenza basato su **Multversioni** (come in Postgres) e livello di isolamento **SERIALIZABLE** sulle prime due transazioni e **SERIALIZABLE** sulla terza. Mostrare il comportamento dello scheduler, supponendo che il valore iniziale dell'oggetto x sia **500** e quello dell'oggetto y sia **100**. Indicare, nell'ordine, le operazioni che vengono eseguite da ciascun client, specificando, per ciascuna, il valore che viene letto o scritto. In conclusione, dire se si verificano o meno anomalie.

client 1	client 2	client 3
		begin read(x)
begin read(x) x = x + 10 write(x) read(y)		
	begin read(y) y = y + 20 write(y) read(x) x = x – 20 write(x) commit	
y = y – 10 write(y) commit		
		(dopo molto tempo) read(x) commit

client 1	client 2	client 3

Si verificano anomalie?

Domanda 2 (20%)

Considerare nuovamente lo scenario della domanda precedente, ripetuto qui a fianco per comodità. Considerare uno scheduler con controllo di concorrenza basato su **2PL stretto** con livello di isolamento **SERIALIZABLE** sulle prime due transazioni e **READ COMMITTED** sulla terza. Mostrare il comportamento dello scheduler, supponendo ancora che il valore iniziale dell'oggetto x sia **500** e quello dell'oggetto y sia **100**.

client 1	client 2	client 3
		begin read(x)
begin read(x) x = x + 10 write(x) read(y)		
	begin read(y) y = y + 20 write(y) read(x) x = x – 20 write(x) commit	
y = y – 10 write(y) commit		
		(dopo molto tempo) read(x) commit

client 1	client 2	client 3

Si verificano anomalie?

Domanda 3 (20%)
Considerare il rifornimento di benzina presso un distributore self-service come una transazione gestita secondo un protocollo che somiglia al commit a due fasi (ma non è uguale ad esso, anche perchè richiede uno specifico ordine per le operazioni). In effetti, si può pensare che ci sia un coordinatore (il dispositivo attraverso il quale si fanno le richieste), un lettore di carte di credito e una pompa di erogazione. Di solito, sulla carta di credito viene addebitato l'importo corrispondente alla benzina effettivamente erogata, quindi dopo l'erogazione.

1. Illustrare un possibile comportamento del sistema, specificando l'ordine con cui vengono eseguite le operazioni, nel caso in cui vadano tutte a buon fine.

2. Illustrare che cosa si può supporre che succeda in caso di guasto della pompa e quindi di mancata erogazione.

3. Illustrare come si può ipotizzare che venga gestito un guasto (che potrebbe essere serio e lungo, anche se certamente raro) del dispositivo di coordinamento, che avvenga fra il momento in cui si conclude l'erogazione e la comunicazione della quantità di benzina (che la pompa di erogazione deve comunicare al dispositivo di coordinamento).

Tecnologie e architetture per la gestione dei dati — 18 maggio 2023

Domanda 4 (20%)
Considerare un sistema che utilizzi blocchi di lunghezza $D = 8$ KB (approssimabili a 8000 byte) e una tabella R con una struttura fisica heap con record a lunghezza fissa che occupano $L = 20$ byte ciascuno, in cui vengono inserite $T = 50.000$ ennuple, con valori della chiave tutti diversi fra loro e da quelli già nella relazione (quindi il sistema verifica il soddisfacimento del vincolo di chiave e ammette tutte le operazioni).
Rispondere alle domande seguenti, indicando formule e valori numerici:

Indicare il numero di scritture in memoria secondaria necessarie per realizzare i 50.000 inserimenti, supponendo che i record di log abbiano una lunghezza pari a circa il triplo 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:

- numero di scritture di pagine della relazione, assumendo che il sistema utilizzi una strategia undo-redo senza vincoli particolari

Come nel caso precedente, ma con riferimento ad un programma che, per realizzare i 50.000 inserimenti, utilizzi complessivamente $k = 10$ transazioni, ognuna con 5000 inserimenti (e supponendo che non vi siano altre transazioni attive)

- numero di scritture di pagine di log:

- numero di scritture di pagine della relazione, sempre assumendo una strategia undo-redo senza vincoli particolari

Domanda 5 (20%)
Per ciascuno dei due seguenti schedule, verificare se esso è conflict serializable (giustificando la risposta). In caso positivo, mostrare lo schedule seriale equivalente.

1. $r_1(x), r_2(x), w_2(x), w_2(z), r_3(z), w_3(u), r_4(u), w_4(u), r_4(y), w_4(y), r_1(y)$

2. $r_1(z), r_2(x), w_2(x), r_3(x), w_3(x), r_4(x), w_4(x), r_4(y), w_4(y), r_1(y)$

Tecnologie e architetture per la gestione dei dati — 18 maggio 2023
Cenni sulle soluzioni

Tempo a disposizione: un'ora.

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

Tecnologie e architetture per la gestione dei dati — 18 maggio 2023

Domanda 1 (20%)

Considerare lo scenario a fianco in cui tre client diversi inviano richieste ad un gestore della 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 al timeout). Si supponga che, in caso di stallo, abortisca la transazione che ha avanzato la richiesta per prima. In caso di abort, si supponga che il client rilanci la stessa transazione (subito dopo l'esecuzione delle altre azioni in attesa sullo stesso dato). Considerare uno scheduler con controllo di concorrenza basato su **Multversioni** (come in Postgres) e livello di isolamento **SERIALIZABLE** sulle prime due transazioni e **SERIALIZABLE** sulla terza. Mostrare il comportamento dello scheduler, supponendo che il valore iniziale dell'oggetto x sia **500** e quello dell'oggetto y sia **100**. Indicare, nell'ordine, le operazioni che vengono eseguite da ciascun client, specificando, per ciascuna, il valore che viene letto o scritto. In conclusione, dire se si verificano o meno anomalie.

client 1	client 2	client 3
		begin
		read(x)
begin		
read(x)		
x = x + 10		
write(x)		
read(y)		
	begin	
	read(y)	
	y = y + 20	
	write(y)	
	read(x)	
	x = x – 20	
	write(x)	
	commit	
y = y – 10		
write(y)		
commit		
		(dopo molto tempo)
		read(x)
		commit

client 1	client 2	client 3
		begin
		read(x) — legge 500
begin		
read(x) — legge 500		
x = x + 10		
write(x) — scrive 510		
read(y) — legge 100		
	begin	
	read(y) — legge 100	
	y = y + 20	
	write(y) — scrive 120	
	read(x) — legge 500	
	x = x - 20	
	wlock(x) — bloccata	
y = y - 10		
wlock(y) — bloccata		
	...	
	abort	
write(y) — scrive 90		
commit		
	begin	
	read(y) — legge 90	
	y = y + 20	
	write(y) — scrive 110	
	read(x) — legge 510	
	x = x - 20	
	write(x) — scrive 490	
	commit	
		read(x)
		commit
		read(x) — legge 500

Si verificano anomalie?
No

Domanda 2 (20%)

Considerare nuovamente lo scenario della domanda precedente, ripetuto qui a fianco per comodità. Considerare uno scheduler con controllo di concorrenza basato su **2PL stretto** con livello di isolamento **SERIALIZABLE** sulle prime due transazioni e **READ COMMITTED** sulla terza. Mostrare il comportamento dello scheduler, supponendo ancora che il valore iniziale dell'oggetto x sia **500** e quello dell'oggetto y sia **100**.

client 1	client 2	client 3
		begin read(x)
begin read(x) x = x + 10 write(x) read(y)		
	begin read(y) y = y + 20 write(y) read(x) x = x – 20 write(x) commit	
y = y – 10 write(y) commit		
		(dopo molto tempo) read(x) commit

client 1	client 2	client 3
		begin read(x) — legge 500 (rilascia il lock)
begin read(x) — legge 500 x = x + 10 write(x) — scrive 510 read(y) — legge 100		
	begin read(y) — legge 100 y = y + 20 wlock(y) — bloccata	
y = y - 10 wlock(y) — bloccata		
	... abort	
write(y) — scrive 90 commit		
	begin read(y) — legge 90 y = y + 20 write(y) — scrive 110 read(x) — legge 510 x = x - 20 write(x) — scrive 490 commit	
		read(x) — legge 490 commit

Si verificano anomalie?
Lettura inconsistente per il client 3

Domanda 3 (20%)
Considerare il rifornimento di benzina presso un distributore self-service come una transazione gestita secondo un protocollo che somiglia al commit a due fasi (ma non è uguale ad esso, anche perchè richiede uno specifico ordine per le operazioni). In effetti, si può pensare che ci sia un coordinatore (il dispositivo attraverso il quale si fanno le richieste), un lettore di carte di credito e una pompa di erogazione. Di solito, sulla carta di credito viene addebitato l'importo corrispondente alla benzina effettivamente erogata, quindi dopo l'erogazione.

1. Illustrare un possibile comportamento del sistema, specificando l'ordine con cui vengono eseguite le operazioni, nel caso in cui vadano tutte a buon fine.

possibile soluzione

(a) inserimento carta di credito

(b) verifica della validià della carta di credito ("preautorizzazione") e (di solito) restituzione della carta

(c) comunicazione alla pompa di erogazione

(d) erogazione

(e) comunicazione dalla pompa al controllore con l'importo

(f) addebito dell'importo

vspace*1cm

2. Illustrare che cosa si può supporre che succeda in caso di guasto della pompa e quindi di mancata erogazione.

Nessun problema particolare, visto l'addebito viene eseguito dopo l'erogazione

3. Illustrare come si può ipotizzare che venga gestito un guasto (che potrebbe essere serio e lungo, anche se certamente raro) del dispositivo di coordinamento, che avvenga fra il momento in cui si conclude l'erogazione e la comunicazione della quantità di benzina (che la pompa di erogazione deve comunicare al dispositivo di coordinamento).

La pompa deve registrare (in un proprio log) le operazioni svolte, in modo che, al ripristino del dispositivo di coordinamento sia possibile comunicare ad esso l'avvenuta erogazione e possa quindi essere registrato il conseguente addebito

Tecnologie e architetture per la gestione dei dati — 18 maggio 2023

Domanda 4 (20%)
Considerare un sistema che utilizzi blocchi di lunghezza $D = 8$ KB (approssimabili a 8000 byte) e una tabella R con una struttura fisica heap con record a lunghezza fissa che occupano $L = 20$ byte ciascuno, in cui vengono inserite $T = 50.000$ ennuple, con valori della chiave tutti diversi fra loro e da quelli già nella relazione (quindi il sistema verifica il soddisfacimento del vincolo di chiave e ammette tutte le operazioni).
Rispondere alle domande seguenti, indicando formule e valori numerici:

Indicare il numero di scritture in memoria secondaria necessarie per realizzare i 50.000 inserimenti, supponendo che i record di log abbiano una lunghezza pari a circa il triplo 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 $T = 50.000$, una per transazione
- numero di scritture di pagine della relazione, assumendo che il sistema utilizzi una strategia undo-redo senza vincoli particolari : al massimo una per transazione, $T = 50.000$; probabilmente di meno, anche solo $T/f = 125$, una per blocco (dove f indica il fattore di blocco $f = D/L = 400$)

Come nel caso precedente, ma con riferimento ad un programma che, per realizzare i 50.000 inserimenti, utilizzi complessivamente $k = 10$ transazioni, ognuna con 5000 inserimenti (e supponendo che non vi siano altre transazioni attive)

- numero di scritture di pagine di log:
  per ogni transazione si debbono scrivere le pagine di log corrispondenti. Ogni transazione scrive T/k ennuple e quindi le relative scritture su log occupano $T/k \times L \times 3 = 300$KB e cioè $T/k \times L \times 3 * 1/D =$ ca.38 blocchi. In totale, per $k = 10$ transazioni, circa $T \times L \times 3 * 1/D =$ ca.380 scritture
- numero di scritture di pagine della relazione, sempre assumendo una strategia undo-redo senza vincoli particolari non si può dire con precisione, anche solo $T/f = 125$, una per blocco

Tecnologie e architetture per la gestione dei dati — 18 maggio 2023

Domanda 5 (20%)
Per ciascuno dei due seguenti schedule, verificare se esso è conflict serializable (giustificando la risposta). In caso positivo, mostrare lo schedule seriale equivalente.

1. $r_1(x), r_2(x), w_2(x), w_2(z), r_3(z), w_3(u), r_4(u), w_4(u), r_4(y), w_4(y), r_1(y)$

Il grafo dei conflitti (da mostrare) contiene un ciclo

2. $r_1(z), r_2(x), w_2(x), r_3(x), w_3(x), r_4(x), w_4(x), r_4(y), w_4(y), r_1(y)$

Il ggrafo dei conflitti (da mostrare) è aciclico e l'ordinamento topologico dei suoi nodi è 2, 3, 4, 1, quindi lo schedule seriale equivalente è quello che prevede le transazioni in tale ordine