Basi di dati II, primo modulo Prova parziale — 18 aprile 2011— Compito A
Cenni sulle soluzioni (per i compiti A e B, gli altri sono simili)

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

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

Domanda 1 (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=20
  - b=200 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
```

$$\frac{N_1}{f_1} + N_1(p+1-2) = 6.100.000 \text{ (nei compiti B e D: } 3.100.000)$$

- scansione di R1: $\frac{N_1}{f_1}$
- un accesso diretto a R2 per ogni ennupla di R1

```
select *
from R1 join R2 on D=E
where B=20
```

$$\frac{N_1}{f_1} + \frac{N_1}{b}(p+1-2) = 130.000$$

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

```
select *
from R1 join R2 on D=E
where C=4
```

$$p + \frac{N_1}{c} + \frac{N_1}{c}(p+1) = \text{ ca. } 65$$

- 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

Domanda 2 (25%)
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 3 (25%) Considerare uno schema dimensionale relativo agli esami, che utilizzi, come tabella dei fatti e come una delle dimensioni, le relazioni come quelle qui schematizzate:

KStudente	KCorso	KData	Voto	...
301	201	405	25	...
301	202	406	28	...
302	201	405	25	...
302	203	407	22	...
...	...	...	...	...

KCorso	Titolo	Crediti	...
201	Geometria	6	...
202	Chimica	6	...
203	Fisica	10	...
...	...	...	...

Supporre che si presentino le seguenti esigenze di modifica:

- per ogni corso, interessa rappresentare anche il docente, per supportare analisi sugli esami svolti da ciascun docente; i docenti cambiano nel tempo e passano da un corso all'altro (e possono anche tenere più corsi nello stesso momento, ma ogni corso ha, in un certo giorno, un solo docente); è disponibile l'informazione relativa ai docenti dei corsi nel tempo (per tutto il periodo, anche passato, di interesse);
- i corsi cambiano nome nel tempo: per esempio, il corso nella prima ennupla potrebbe ad un certo punto cambiare nome da "Geometria" in "Algebra lineare"; interessano selezioni e aggregazioni relative agli esami tanto con riferimento al nome del corso (al momento dell'esame) quanto alla sua identità (un codice che viene introdotto allo scopo, ma non sempre viene utilizzato, perché alcuni analisti preferiscono fare riferimento al nome corrente del corso); le modifiche sono rare, ma è possibile che ci siano corsi con vari cambiamenti di nome.

Mostrare nuove versioni delle due tabelle che permettano di soddisfare le esigenze sopra citate (mostrare anche i dati, con riferimento a quelli presenti negli esempi sopra, aggiungendo nuovi dati ragionevoli, che permettano di comprendere le modifiche). Indicare in particolare quali attributi e quali ennuple vadano aggiunti alle due relazioni.

La tabella dei fatti può essere estesa aggiungendo una dimensione supplementare, che dipende da Corso e Data e può quindi essere facilmente aggiunta anche ad una tabella dei fatti esistente (i dati, come detto, sono disponibili e le ennuple rimangono le stesse, con un attributo in più; si noti che Kdocente dipende funzionalmente da KCorso e KData):

KStudente	KCorso	KData	KDocente	Voto	...
301	201	405	901	25	...
301	202	406	902	28	...
302	201	405	901	25	...
302	203	407	903	22	...
...	...	...	...	...	...

Per la dimensione può essere utile la tecnica della"slowly changing dimension," con un nuovo elemento (quindi una ennupla nella relazione) per ogni modifica. Viste le specifiche, qui potrebbe essere utile introdurre un codice, che non cambia nel tempo, e avere due attirbuti per il nome, con quello attuale e quello "dell'epoca."

KCorso	Codice	Titolo	TitoloAttuale	Crediti	...
201	GEOM01	Geometria	Algebra lineare	6	...
202	CHIM01	Chimica	Chimica	6	...
203	FIS01	Fisica	Fisica	10	...
...	...	...	...	...	...
209	GEOM01	Algebra lineare	Algebra lineare	6	...

Domanda 4 (25%) 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	2	0	0
blocco	70	33	35	47
istante load	1	7	3	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):

unpin(70), pin(60), unpin(60), pin(70), unpin(70), pin(60), unpin(60), pin(70)

Riempiendo la tabella seguente, indicare quale pagina del buffer viene utilizzata per ciascuna delle pin e se viene effettivamente eseguita una lettura su disco (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(70)	0	sì	2	sì	0	no
15	pin(60)	0	sì	0	no	3	no
17	pin(70)	0	sì	2	no	0	no

Commento:

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

Basi di dati II, primo modulo Prova parziale — 18 aprile 2011— Compito B
Cenni sulle soluzioni (per i compiti A e B, gli altri sono simili)

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

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

Domanda 1 (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
  - L_1=1.000.000 ennuple e fattore di blocco f_1=10
  - b=100 valori diversi sull'attributo B (tutti gli interi compresi fra 1 e b)
  - c=100.000 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
  - L_2=2.000.000 ennuple e fattore di blocco f_2=20
  - una struttura disordinata, un indice sulla chiave primaria E

Supponendo che:

- gli indici abbiano tutti i=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
```

$$\frac{L_1}{f_1} + L_1(i+1-2) = 6.100.000 \text{ (nei compiti B e D: } 3.100.000)$$

- scansione di R1: $\frac{L_1}{f_1}$
- un accesso diretto a R2 per ogni ennupla di R1

```
select *
from R1 join R2 on D=E
where B=20
```

$$\frac{L_1}{f_1} + \frac{L_1}{b}(p+1-2) = 130.000$$

- scansione di R1 per la selezione: $\frac{L_1}{f_1}$
- un accesso diretto a R2 per ogni ennupla nel risultato della selezione

```
select *
from R1 join R2 on D=E
where C=4
```

$$i + \frac{L_1}{c} + \frac{L_1}{c}(i+1) = \text{ ca. } 65$$

- accesso diretto a R1 per la selezione: i per l'indice e $\frac{L_1}{c}$ per le ennuple
- un accesso diretto a R2 per ogni ennupla nel risultato della selezione

Domanda 2 (25%)
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
READ COMMITTED

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

anomalia: perdita di aggiornamento

Scenario 2
SERIALIZABLE

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

nessuna anomalia

Domanda 3 (25%) Considerare uno schema dimensionale relativo agli esami, che utilizzi, come tabella dei fatti e come una delle dimensioni, le relazioni come quelle qui schematizzate:

KStudente	KCorso	KData	Voto	...
201	301	405	26	...
201	302	406	28	...
202	301	405	26	...
202	303	407	22	...
...	...	...	...	...

KCorso	Titolo	Crediti	...
301	Geometria	6	...
302	Chimica	6	...
303	Fisica	10	...
...	...	...	...

Supporre che si presentino le seguenti esigenze di modifica:

- per ogni corso, interessa rappresentare anche il docente, per supportare analisi sugli esami svolti da ciascun docente; i docenti cambiano nel tempo e passano da un corso all'altro (e possono anche tenere più corsi nello stesso momento, ma ogni corso ha, in un certo giorno, un solo docente); è disponibile l'informazione relativa ai docenti dei corsi nel tempo (per tutto il periodo, anche passato, di interesse);
- i corsi cambiano nome nel tempo: per esempio, il corso nella prima ennupla potrebbe ad un certo punto cambiare nome da "Geometria" in "Algebra lineare"; interessano selezioni e aggregazioni relative agli esami tanto con riferimento al nome del corso (al momento dell'esame) quanto alla sua identità (un codice che viene introdotto allo scopo, ma non sempre viene utilizzato, perché alcuni analisti preferiscono fare riferimento al nome corrente del corso); le modifiche sono rare, ma è possibile che ci siano corsi con vari cambiamenti di nome.

Mostrare nuove versioni delle due tabelle che permettano di soddisfare le esigenze sopra citate (mostrare anche i dati, con riferimento a quelli presenti negli esempi sopra, aggiungendo nuovi dati ragionevoli, che permettano di comprendere le modifiche). Indicare in particolare quali attributi e quali ennuple vadano aggiunti alle due relazioni.

La tabella dei fatti può essere estesa aggiungendo una dimensione supplementare, che dipende da Corso e Data e può quindi essere facilmente aggiunta anche ad una tabella dei fatti esistente (i dati, come detto, sono disponibili e le ennuple rimangono le stesse, con un attributo in più; si noti che Kdocente dipende funzionalmente da KCorso e KData):

KStudente	KCorso	KData	KDocente	Voto	...
201	301	405	901	26	...
201	302	406	902	28	...
202	301	405	901	26	...
202	303	407	903	22	...
...	...	...	...	...	...

Per la dimensione può essere utile la tecnica della"slowly changing dimension," con un nuovo elemento (quindi una ennupla nella relazione) per ogni modifica. Viste le specifiche, qui potrebbe essere utile introdurre un codice, che non cambia nel tempo, e avere due attirbuti per il nome, con quello attuale e quello "dell'epoca."

KCorso	Codice	Titolo	TitoloAttuale	Crediti	...
301	GEOM01	Geometria	Algebra lineare	6	...
302	CHIM01	Chimica	Chimica	6	...
303	FIS01	Fisica	Fisica	10	...
...	...	...	...	...	...
309	GEOM01	Algebra lineare	Algebra lineare	6	...

Domanda 4 (25%) 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	70	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):

unpin(70), pin(60), unpin(60), pin(70), unpin(70), pin(60), unpin(60), pin(70)

Riempiendo la tabella seguente, indicare quale pagina del buffer viene utilizzata per ciascuna delle pin e se viene effettivamente eseguita una lettura su disco (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(70)	0	sì	2	sì	0	no
15	pin(60)	0	sì	0	no	3	no
17	pin(70)	0	sì	2	no	0	no

Commento:

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