Departamento de Cs. e Ingeniería de la Computación

Universidad Nacional del Sur

Elementos de Bases de Datos

Segundo Cuatrimestre de 2014

Trabajo Práctico N° 8

Protocolos para Control de Concurrencia

Ejercicios

1. ¿Por qué los SMBD con manejo de transacciones requieren de Protocolos de Control de Concurrencia?

2. Definir

   a) Bloqueo (*lock*).

   b) Bloqueo compartido (*lock-S*).

   c) Bloqueo exclusivo (*lock-X*).

   d) Compatibilidad entre tipos de bloqueo

3. Dadas las siguientes transacciones:

i)

T_1	T_2
Read(A)	Read(B)
Write(A)	Read(A)
Read(B)	Write(B)

ii)

T_3	T_4	T_5
Read(A)	Read(A)	Read(B)
Write(A)	Read(B)	Write(B)
Read(B)		

   Para los incisos i) y ii) se pide:

   a) Encontrar, si es posible, una planificación concurrente resultante de aplicar el protocolo de dos fases con sólo locks exclusivos.

   b) Encontrar, si es posible, una planificación concurrente resultante de aplicar el protocolo de dos fases con posibilidades de locks exclusivos y compartidos.

   c) Encontrar, si es posible, una planificación concurrente resultante de aplicar el protocolo de dos fases con posibilidades de locks compartidos, upgrade y downgrade.

   d) Encontrar una planificación concurrente resultante de aplicar alguno de los protocolos de dos fases que resulte en deadlock.

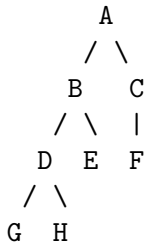
   e) Encontrar una planificación concurrente resultante de aplicar a los requerimientos de entrada un protocolo de locking sin imponer dos fases. Es serializable? Es posible llegar a una planificación no serializable?

   f) Analizar las diferentes alternativas de bloqueo para los protocolos de dos fases en cuanto a serializabilidad de las planificaciones, nivel de concurrencia y posibilidad de deadlock.

4. ¿Qué ventajas y desventajas proporciona la alternativa de bloqueo de dos fases estricto?

5. ¿El protocolo de bloqueo garantiza planificaciones serializables? Justificar.

6. Dada la siguiente transacción: T_0 = req(C); req(D); req(G) (por *req(X)* entiéndase, requiere X) y el siguiente grafo de precedencia:



Mostrar la secuencia de locks necesarios para poder ejecutar T_0.

7. Analice los protocolos basados en árbol en cuanto los siguientes puntos. Justifique.
   - Serializabilidad de las planificaciones.
   - Nivel de concurrencia.
   - Posibilidad de deadlock.
   - Posibilidad de inanición.
8. ¿Qué representa la estampilla de una transacción?.
9. ¿Los protocolos de control de concurrencia basados en estampillas de tiempo resuelven el problema de *deadlock*? ¿Tienen nuevos problemas que no tengan los protocolos basados en bloqueos?.
10. Dadas las siguientes transacciones:

    T_0 = start; read(B); write(B); read(A); write(A); finish
    T_1 = start; read(B); read(C); write(A); write(C); finish
    T_2 = start; read(A); write(A); write(B); finish

    Con 0 <ts(T_0)<ts(T_1)<ts(T_2) y R-ts(A)=R-ts(B)=R-ts(C)=W-ts(A)=W-ts(B)=W-ts(C)=0

    a) Encontrar, si es posible, una planificación concurrente donde al menos una de las transacciones retroceda aplicando el algoritmo de estampilla de tiempos.
    b) Encontrar, si es posible, una planificación concurrente donde ninguna transacción retroceda aplicando el algoritmo de estampilla de tiempos.
    c) Encontrar, si es posible, una planificación concurrente donde deba retroceder en cascada más de una transacción aplicando el algoritmo de estampilla de tiempos.
    d) Encontrar, si es posible, una planificación concurrente donde la falla de una transacción resulte en una planificación no recuperable.
    e) Encontrar, si es posible, una planificación donde se pueda aplicar la regla de escritura de Thomas.

    Para mostrar las planificaciones encontradas utilice, por ejemplo, el siguiente formato:

	T_0	T_1	T_2	Estampillas
1.	start			
2.		start		
3.		Read(B)		<B, R-ts=ts(T_1), W-ts=0>
4.	...	...	...	...

11. Para la siguiente planificación de entrada:

	T_0	T_1	T_2
1.	Read(A)		
2.		Write(A)	
3.	Write(A)		
4.			Write(A)

Con 0 <ts(T_0)<ts(T_1)<ts(T_2) y R-ts(A)=R-ts(B)=R-ts(C)=W-ts(A)=W-ts(B)=W-ts(C)=0.

a) Verificar si es serializable en conflictos y en vistas.

b) ¿Puede resultar esta planificación de aplicar el protocolo de dos fases?

c) ¿Puede resultar esta planificación de aplicar el protocolo de estampillas de tiempo tradicional?

d) ¿Puede resultar esta planificación de aplicar el protocolo de estampillas con la regla de escritura de Thomas?

Nota: Para los protocolos de estampillas, indique para cada instante de tiempo como se actualizan las estampillas correspondientes. Por ejemplo: 1. <A, R-ts=ts(T_0), W-ts=0>.

12. Para la siguiente planificación de entrada:

	T_1	T_2	T_3
1.		Read(A)	
2.			Write(A)
3.	Read(B)		
4.	Write(B)		
5.		Write(B)	

Con 0 <ts(T_1)<ts(T_2)<ts(T_3) y R-ts(A)=R-ts(B)=R-ts(C)=W-ts(A)=W-ts(B)=W-ts(C)=0.

a) ¿ Es serializable en conflictos?

b) ¿Puede resultar esta planificación de aplicar el protocolo de dos fases?

c) ¿Puede resultar esta planificación de aplicar el protocolo de estampillas de tiempo?

13. Considere la siguiente planificación para las transacciones T_1, T_2 y T_3, con $ts(T_1) = 1$, $ts(T_2) = 2$ y $ts(T_3) = 3$, y suponga que inicialmente existen las siguientes versiones:
<A_0, 11, R-ts=0, W-ts=0>, <B_0, 12, R-ts=0, W-ts=0>

	T_1	T_2	T_3
1.	read(A)		
2.	read(B)		
3.		read(A)	
4.		A:=A+10	
5.		write(A)	
6.			read(A)
7.	B:=A+B		
8.	write(B)		
9.		B:=A	
10.		write(B)	
11.			read(B)

a) Analizar el resultado de aplicar el protocolo de multiversión.

b) Encontrar, si el posible, otra planificación que al aplicar el protocolo de multiversión resulte en algún retroceso.

Nota: Indique para cada instante de tiempo como se actualizan las estampillas y los valores de las versiones correspondientes. Por ejemplo: 1. <A_0, 11, R-ts=ts(T_0), W-ts=0>

14. Para las transacciones T_0, T_1 y T_2 y la siguiente planificación, analizar el resultado de aplicar el protocolo de validación. **Nota:** Las operación Write modifica los datos locales de la transacción y la operación *output* vuelca los valores de estos datos en la base de datos.

T_0	T_1	T_2
start		
Read(A)		
Read(B)		
	start	
	Read(A)	
Write(B)		
	Read(C)	
		start
		Read(A)
	Write(B)	
		Read(B)
valid		
output(B)		
finish		
	valid	
	output(B)	
	finish	
		valid
		finish

15. Protocolo de concurrencia por validación. Suponga que existen las transacciones T_1, T_2 y T_3 y los elementos de base de datos A, B y C. Los conjuntos de lectura (RS) y escritura (WS) para las transacciones son los siguiente:

$$\begin{array}{ll} \text{RS}(T_1) = \{\text{A, B}\} & \text{WS}(T_1) = \{\text{B}\} \\ \text{RS}(T_2) = \{\text{A, C}\} & \text{WS}(T_2) = \{\text{A}\} \\ \text{RS}(T_3) = \{\text{C}\} & \text{WS}(T_3) = \{\text{B, C}\} \end{array}$$

Utilizaremos S_i,V_i y F_i para representar cuando una transacción T_i comienza, intenta validar y termina respectivamente. Dada la siguiente secuencia de eventos: $\underrightarrow{S_1S_2V_1S_3V_3F_1V_2F_2F_3}$ Determine cuales transacciones validan y cuales retroceden, justificando en cada caso.

16. Para las transacciones:
T_0 = read(A); read(B); A:= A+10; write(A); B:=B+20; write(B)
T_1 = B:=22; write(B); C:=23; write(C)
T_2 = read(C); read(B); C:=C +10; write(C); B:=B+20; write(B)

Encontrar una planificación concurrente (no en serie) resultante de aplicar:

a) el protocolo de bloqueo de dos fases.

b) el protocolo de estampillas.

c) el protocolo de estampillas con regla de escritura de Thomas.

d) el protocolo de árbol. Defina un árbol de precedencia a su elección.

e) el protocolo de multiversión. Suponga que inicialmente existen las siguientes versiones: <A_0, 11, R-ts=0, W-ts=0>, <B_0, 12, R-ts=0, W-ts=0>, <C_0, 13, R-ts=0, W-ts=0>

f) el protocolo de validación.

17. Para las transacciones T_1 y T_2 que se muestran a continuación, con valor inicial de $A = 100$, $B = 200$, $C = 300$ y $D = 400$.

    T_1= Read(B); $B = B * 4$; Write(B); Read(C); Read(D); $C = B + D$; Write(C)
    T_2= Read(A); $A = A + 100$; Write(A); Read(B); Read(C); $B = C$; Write(B)

    a) Dar una planificación concurrente serializable usando el protocolo de bloqueos de dos fases con locks compartidos y exclusivos que no caiga en deadlock. ¿Cuál es la serie equivalente? ¿Cuáles son los valores finales de A, B, C y D?

    b) Dar una planificación concurrente serializable usando el protocolo de bloqueos de dos fases con locks compartidos y upgrade que caiga en deadlock. Indicar claramente el punto donde se produce el deadlock.

    c) ¿Es posible encontrar una planificación concurrente serializable usando el protocolo de dos fases estricto?

18. ¿Es posible encontrar una planificación concurrente serializable en conflictos que no sea vÃ¡lida bajo el protocolo de bloqueo de dos fases, ni por el protocolo de estampillas de tiempo tradicional?