Departamento de Cs. e Ingeniería de la Computación

Universidad Nacional del Sur

Bases de Datos

Segundo Cuatrimestre de 2019

Trabajo Práctico N° 7

Protocolos para Control de Concurrencia

Ejercicios

1. ¿Cuál de las propiedades ACID sobre transacciones aseguran los protocolos de control de concurrencia?

2. 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.

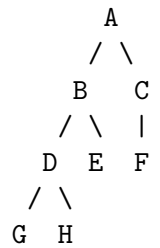
d) Encontrar una planificación concurrente resultante de aplicar alguno de los protocolos de dos fases que resulte en deadlock.

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.

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

4. 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.

5. 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.

6. Dadas las siguientes transacciones:

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

   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 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.	...	...	...	...

7. 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>.

8. Considere la siguiente planificación para las transacciones T_0, T_1 y T_2, con ts(T_0)<ts(T_1)<ts(T_2) 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_0	T_1	T_2
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>

9. 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.

10. 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.

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

11. Para las transacciones T_0, T_1 y T_2 T_3 y T_4, analizar el resultado de aplicar el protocolo de validación para a siguiente planificación, determinando que transacciones validan y cuales retroceden. **Nota:** Las operación Write modifica los datos locales de la transacción y la operacion *output* vuelca los valores de estos datos en la base de datos.

T_0	T_1	T_2	T_3	T_6
start				
Read(B)				
	start			
	Read(A)			
			start	
	Read(B)			
		start		
		Read(C)		
Write(B)				
			Read(C)	
	Write(C)			
		Write(C)		
valid				
output(B)				
$finish$				
	valid			
	output(C)			
	$finish$			
				start
				Read(B)
		valid		
			valid	
		output(C)		
		$finish$		
			$finish$	
				valid
				$finish$

12. Protocolo de concurrencia por validación. Suponga que existen las transacciones T_1, T_2 y T_3 y los 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: $\underline{S_1 S_2 V_1 S_3 V_3 F_1 V_2 F_2 F_3}_{\rightarrow}$ Determine cuales transacciones validan y cuales retroceden, justificando en cada caso.

13. Simule la aplicación del protocolo de estampillas de tiempo pedido para las planificaciones dadas, completando las tablas que se presentan a continuación. Deberá mostrarse como se actualizan las estampillas de tiempo y los valores de los datos. En caso de producirse un retroceso por la violación del protocolo indique:

- en que punto se produce y porque.
- que transacciones retroceden y por que

a) Para las siguientes planificaciones simule la aplicación del Protocolo de estampillas de tiempo **sin** y regla de escritura de **Thomas** suponiendo que :

- los valores iniciales de los datos son: A= 11, B= 12 y C= 13
- las estampillas de tiempo son: 0 < Ts(T1) < Ts(T2) < Ts(T3) e inicialmente R-ts(A) = R-ts(B) = R-ts(C) = W-ts(A) = W-ts(B) = W-ts(B) =0.

i)

T_1	T_2	T_3	Dato	Valor	R-Ts	W-ts
start						
read(A)						
A:=A+10						
write(A)						
		start				
		read(A)				
		A:=A+20				
	start					
	read(B)					
	B:= B+10					
		C:=10				
		write(A)				
		write(C)				
C:=100						
write(C)						
commit						
		commit				
	write(B)					
	commit					

ii)

T_1	T_2	T_3	Dato	Valor	R-Ts	W-ts
start						
read(A)						
A:=A+10						
write(A)						
	start					
	read(B)					
		start				
		read(B)				
		read(A)				
		B:=A+B				
		write(B)				
B:=100						
write(B)						
commit						
		commit				
	B:=B+10					
	write(B)					
	commit					

b) Para la siguiente planificacion simule la aplicación del protocolo de estampillas de tiempo **con** regla de escritura de **Thomas** suponiendo que :

- los valores iniciales de los datos son: A= 11, B= 12 y C= 13
- las estampillas de tiempo son: 0 < Ts(T1) < Ts(T2) < Ts(T3) e inicialmente R-ts(A) = R-ts(B) = R-ts(C) = W-ts(A) = W-ts(B) = W-ts(B) =0.

i)

T_1	T_2	T_3	Dato	Valor	R-Ts	W-ts
start						
read(A)						
A:=A+10						
write(A)						
		start				
		read(A)				
		A:=A+20				
	start					
	read(B)					
	B:= B+10					
		C:=10				
		write(A)				
		write(C)				
C:=100						
write(C)						
commit						
		commit				
	write(B)					
	commit					

c) Para las siguientes planificaciones simule la aplicación del Protocolo de mutiversión suponiendo que :

- las versiones iniciales de los datos son: <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>
- las estampillas de tiempo son: 0 < Ts(T1) < Ts(T2) < Ts(T3)

i)

T_1	T_2	T_3	Verisión	Valor	R-Ts	W-ts
start						
read(A)						
A:=A+10						
write(A)						
		start				
		read(A)				
		A:=A+20				
	start					
	read(B)					
	B:= B+10					
		C:=10				
		write(A)				
		write(C)				
C:=100						
write(C)						
commit						
		commit				
	write(B)					
	commit					

ii)

T_1	T_2	T_3	Dato	Valor	R-Ts	W-ts
start						
read(A)						
A:=A+10						
write(A)						
	start					
	read(B)					
		start				
		read(B)				
		read(A)				
		B:=A+B				
		write(B)				
B:=100						
write(B)						
commit						
		commit				
	B:=B+10					
	write(B)					
	commit					

14. Considere las siguientes planificaciones

T_1	T_2
Lock-X(C)	
	Lock-X(B)
Lock-X(D)	
Unlock(C)	
	Lock-X(C)
	Unlock(B)
Unlock(D)	
	Lock-X(A)
	Unlock(A)
	Lock-X(D)
	Unlock(C)
	Unlock(D)

a)

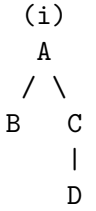
T_3	T_4
Lock-X(A)	
	Lock-X(C)
	Lock-X(D)
	Unlock(C)
Lock-X(C)	
Unlock(A)	
	Unlock(D)
Lock-X(D)	
Unlock(C)	
Unlock(D)	

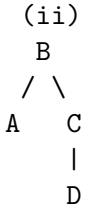
b)

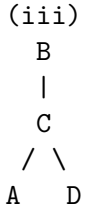
T_5	T_6
Lock-X(B)	
	Lock-X(C)
	Lock-X(D)
	Unlock(C)
Lock-X(C)	
	Unlock(D)
Unlock(C)	
Lock-X(A)	
Unlock(B)	
Unlock(A)	

c)

Siguiendo el protocolo de árbol: ¿Con cual/es de los siguientes árboles es posible obtener cada planificación? Justifique.







15. Suponga que existen las transacciones T_1, T_2, T_3, T_4 y T_5 los datos A, B, C y D. Los conjuntos de lectura (RS) y escritura (WS) para las transacciones son los siguientes:

$$\begin{array}{ll} \text{RS}(T_1) = \{\text{A, B}\} & \text{WS}(T_1) = \{\text{A ,B}\} \\ \text{RS}(T_2) = \{\text{B, C}\} & \text{WS}(T_2) = \{\text{B,C}\} \\ \text{RS}(T_3) = \{\text{A,C}\} & \text{WS}(T_3) = \{\text{C}\} \\ \text{RS}(T_4) = \{\text{D}\} & \text{WS}(T_4) = \{\text{D}\} \\ \text{RS}(T_5) = \{\text{A,D}\} & \text{WS}(T_5) = \{\} \end{array}$$

Utilizaremos S_i,V_i y F_i para representar cuando una transacción Ti comienza, intenta validar y termina respectivamente. Para las siguientes secuencias de eventos:

a) $\underrightarrow{S_1S_2V_1F_1S_4S_3V_2S_5F_2V_3V_5F_3V_4F_5F_4}$

b) $\underrightarrow{S_3S_1V_3S_4F_3S_5V_1V_4F_4V_5S_2F_4F_1V_2F_2}$

Determine cuales transacciones validan y cuales retroceden, justificando en cada caso.