Enunciado

Dada la siguiente historia en un planificador con timestamp:

st1, st2, r2(X), st3, st4, r1(Y), r4(Z), w3(X), w3(Y), w4(Z), w2(X), w1(Y), r3(Z)

en donde los valores iniciales de X, Y, Z son 0 y sucede que:

t1 escribe Y = 1

t2 escribe X=2

t3 escribe X=3, Y=30

t4 escribe Z = 4

- Decir que pasa en cada acción y que valores quedan en X,Y,Z si el planificador no utiliza la técnica multiversión

- Decir que pasa en cada acción y que valores quedan en X,Y,Z si el planificador utiliza la técnica multiversión

Sin usar multiversión

st1, st2, r2(X), st3, st4, r1(Y), r4(Z), w3(X), w3(Y), w4(Z), w2(X), w1(Y), r3(Z)

Tiempo	T1	T2	T3	T4
1	st1=100			
2		st2=200		
3		r2(X)		
4			st3=300	
5				st4=400
6	r1(Y)			
7				r4(Z)
8			w3(X)	
9			w3(Y)	
10				w4(Z)
11		w2(X)		
12	w1(Y)			
13			r3(Z)	

Asumimos los bits de commit para cada elemento inicialmente en true.

Sin usar multiversión

st1, st2, r2(X), st3, st4, r1(Y), r4(Z), w3(X), w3(Y), w4(Z), w2(X), w1(Y), r3(Z)

Tiempo	T1	T2	T3	T4	X=2, Y=1, Z=4 RT(X)=200, RT(Y)=100, RT(Z)=400 WT(X)= 200, WT(Y)=100 , WT(Z)= 400
1	st1=100				Se lanza T1 con timestamp=100
2		st2=200			Se lanza T2 con timestamp=200
3		r2(X)			Ts (T2) >= WT(X) (200>=0) y como Ts (T2) >= RT(X) (200>=0) => RT(X)=200
4			st3=300		Se lanza T3 con timestamp=300
5				st4=400	Se lanza T4 con timestamp=400
6	r1(Y)				Idem tiempo 3 => (100 >= 0) => RT(Y)=100
7				r4(Z)	Idem tiempo 3 => (400 >=0) => RT(Z)=400
8			w3(X)		Ts (T3) >= RT(X) (300>=200) y Ts (T3) >= WT (X) (300>=0) => X:= 3, WT(X)= 300, C(X)=false
9			w3(Y)		Idem tiempo 8 => Y:=30, WT(Y)=300, C(Y)=false
10				w4(Z)	Idem tiempo 8 => Z:= 4, WT(Z)= 400, C(Z)=true
11		w2(X)			Ts (T2) >= RT(X) (200>=200) pero Ts (T2) < WT (X) (200<300) => Se demora la escritura hasta que T3 finalice o aborte
12	w1(Y)				Ts (T1) >= RT(Y) (100>=100) pero Ts (T1) < WT (Y) (100<300) => Se demora la escritura hasta que T3 finalice o aborte
13			r3(Z)		TS(T3) < WT(Z) (300<400) =>

Aborta T3, se produce un Rollback

Al abortar T3, las escrituras en el tiempo 11 y 12

que estaban demoradas continuan y T1 y T2 terminan normalmente

También termina T4

Valores finales: X=2, Y=1, Z=4

Con multiversión

Tpo	T1	T2	T3	T4	X0	Y0	Z0	X300	Y300	Z400	X200	Y100	
1	st1=100												Se lanza T1
2		st2=200											Se lanza T2
3		r2(X)			lee X0								
4			st3=300										Se lanza T3
5				st4=400									Se lanza T4
6	r1(Y)					lee Y0							
7				r4(Z)			lee Z0						
8			w3(X)					Create X:=3					
9			w3(Y)						Create Y:=30				
10				w4(Z)						Create Z:=4			
11		w2(X)									Create X:=2		
12	w1(Y)											Create Y:=1	
13			r3(Z)				lee Z0						

Lee la version Z0 y T3 termina exitosamente

Por lo tanto las escrituras en el tiempo 11 y 12 se pueden realizar, creando nuevas versiones para X y para Y

Se crean X200 e Y100

T1 y T2 terminan exitosamente

Valores finales: X=3, Y=30, Z=4 en la última versión de cada elemento

Enunciado

Dada la siguiente Historia (para el planificador basado en validación):

$$H = R_1(A,B); R_2(B,F); V_2; V_1; R_3(B,D); W_2(D); V_3; W_1(A,C); W_3(D,E)$$

a. Indique qué ocurre en cada momento de validación y, en caso de no validar, cuál es problema preciso que presenta.

b. Realice un cambio en la Historia. Como resultado de ese cambio, todas las validaciones deben ser exitosas. Justifique.

- Verificar que $\boldsymbol{R_T \cap W_U}$ es vacío para cualquier transacción $\boldsymbol{U}$ validada previamente y que no finalizo antes de que $\boldsymbol{T}$ comenzara.
- Verificar $\boldsymbol{W_T \cap W_U}$ es vacío para cualquier transacción $\boldsymbol{U}$ validada previamente y que no finalizo antes de que $\boldsymbol{T}$ sea validada.

Resolución - Item a.

- Se valida T_2, como ninguna transacción fue validada entonces T_2 es la primer transacción y se valida exitosamente.
- Se valida T_1: Pasa que: $VAL = \{T_2\}$ y $END = \emptyset$ ademas $END(T_2) > START(T_1)$ debo comprobar que $RS(T_1) \cap WS(T_2) = \emptyset$. Como $RS(T_1) = \{A, B\}$ y $WS(T_2) = \{D\} \rightarrow RS(T_1) \cap WS(T_2) = \emptyset$ Por lo cual esta parte se valida.

  También pasa que $END(T_2) > VAL(T_1)$ debo entonces verificar que $WS(T_1) \cap WS(T_2) = \emptyset$ como $WS(T_1) = \{A, C\}$ y $WS(T_2) = \{D\} \rightarrow WS(T_1) \cap WS(T_2) = \emptyset$ por lo cual T_1 es validada.
- Se valida T_3. Pasa que: $VAL = \{T_2, T_1\}$ y $END = \{T_2\}$ ademas $END(T_2) > START(T_3)$ y $END(T_1) > START(T_3)$ Debo verificar:
  - $RS(T_3) \cap WS(T_2) = \emptyset$
  - $RS(T_3) \cap WS(T_1) = \emptyset$

Resolución - Item a.

- Cómo $WS(T_2) = \{D\}$, $WS(T_1) = \{A, C\}$ y $RS(T_3) = \{B, D\} \rightarrow RS(T_3) \cap WS(T_2) = \{D\}$ por lo cual no es $\emptyset$.
  Entonces: **NO VALIDA**.

  Con eso basta para que se haga el rollback de T_3. Notar que no importa que $RS(T_3) \cap WS(T_1) = \emptyset$

  También pasa que $END(T_1) > VAL(T_3)$ pero al haber fallado en la validación no haría falta comprobar si pasa que $WS(T_1) \cap WS(T_3) = \emptyset$.

El problema es que T_2 escribe un ítem después de que una transacción T_3, que es posterior de acuerdo al orden serial equivalente, lee ese ítem.

Resolución - Item b.

- La validación falló en $RS(T_3) \cap WS(T_2) = \emptyset$ que debía comprobarse porque $END(T_2) > START(T_3)$. Si invertimos el orden de $R_3(B, D); W_2(D)$ a $W_2(D); R_3(B, D)$ entonces $END(T_2) < START(T_3)$ y no haría falta comprobar $RS(T_3) \cap WS(T_2) = \emptyset$. Por lo cual eliminamos el problema que hizo que la validación fallara.

  Con ese cambio todavía pasa que $END(T_1) > VAL(T_3)$ por lo que deberíamos comprobar:

  $WS(T_1) \cap WS(T_3) = \emptyset$

  Como $WS(T_1) = \{A, C\}$ y $WS(T_3) = \{D, E\} \rightarrow WS(T_1) \cap WS(T_3) = \emptyset$

  La Historia cambiada queda:

$$H = R_1(A, B); R_2(B, F); V_2; V_1; W_2(D); R_3(B, D); V_3; W_1(A, C); W_3(D, E)$$