

1 Concurrency

Hieronder zijn twee schedules gegeven.

- (i) Stel voor elk van de schedules de complete precedentiegraaf op. Geef aan of deze schedules serialiseerbaar zijn of niet. Geef zo mogelijk de equivalente seriële schedules.
- (ii) Geef aan welke transacties gedrag vertonen dat niet getolereerd wordt door een 2PL-scheduler.

<i>S1</i>					<i>S2</i>				
T1	T2	T3	T4	T5	T1	T2	T3	T4	T5
w(z)			r(z)		r(z)			w(z)	
	w(x)					r(x)			
r(x)		r(z)			w(x)		w(z)		
	r(y)					w(y)			
r(z)				r(x)	w(z)				w(x)
			w(y)					r(y)	
				w(y)					r(y)

ZOZ

2 recovery

We beschouwen nonquiescent recovery met UNDO-logging. Hieronder vind je een log met before images.

```
<START T1>
<T1, A, 5>
<START T2>
<T2, B, 10>
<COMMIT T1>
<START CKPT (T2)>
<T2, D, 15>
<START T3>
<T3, E, 10>
<START T4>
<T4, F, 25>

<END CKPT>
<COMMIT T2>
<COMMIT T4>
<T3, G, 30>
```

- (i) Stel dat een crash optreedt direct na `<T4, F, 25>` maar voor `<END CKPT>`. Welk gedeelte van de log file wordt gescand? Welke operaties worden undone?
- (ii) Stel dat een crash optreedt direct na `<T3, G, 30>`. Welk gedeelte van de log file wordt gescand? Welke operaties worden undone?

We gaan dezelfde log beschouwen, maar nu met REDO-logging en after images.

- (iii) Stel dat een crash optreedt direct na `<T4, F, 25>` maar voor `<END CKPT>`. Welk gedeelte van de log file wordt gescand? Welke operaties worden redone?
- (iv) Stel dat een crash optreedt direct na `<T3, G, 30>`. Welk gedeelte van de log file wordt gescand? Welke operaties worden redone?