Solutions to Exercise 3: Transactions

Part 1: Deadlock

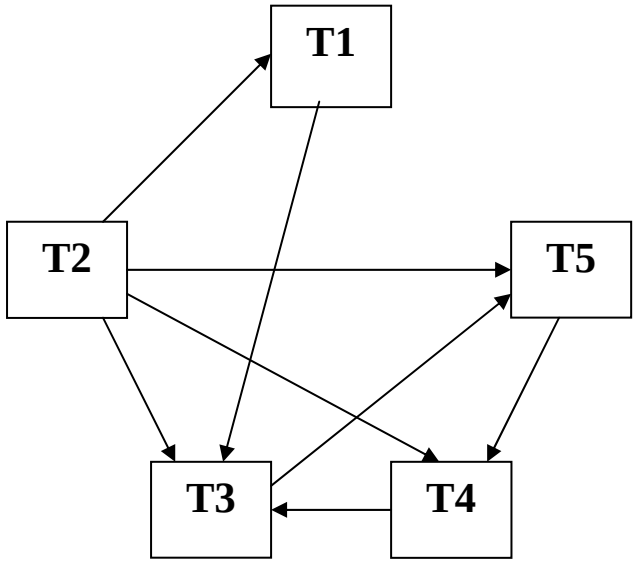
The case of two admins: the process of admin *A* acquires a lock on the table *account* and the process of admin *B* acquires a lock on the table *tx* and pause for 10 seconds. When *A* and *B* awake both of them require a lock that is held by the other one – a deadlock situation occurs. Our database system *PointBase* can wait up to 60 seconds (default *PointBase* parameter of *locks.timeout*) for a lock to be acquired before rolling back. The first admin that started the transaction will get timeout after 60 seconds and will rollback, whereas the second one will acquire the lock for the table and will execute successfully.

Part 2: Schedule Serialization

2.1

T1	T2	T3	T4	T5	Conflicts
$R_1(a)$					$W_3(a)$
			$R_4(a)$		$W_3(a)$
	$W_2(b)$				$W_3(b),W_5(b)$
				$R_5(c)$	$W_4(c)$
		$W_3(a)$			
		$W_3(b)$			$W_5(b)$
	$R_2(c)$				$W_4(c)$
				$W_5(b)$	
	$W_2(d)$				$R_1(d)$
			$W_4(c)$		
$R_1(d)$					

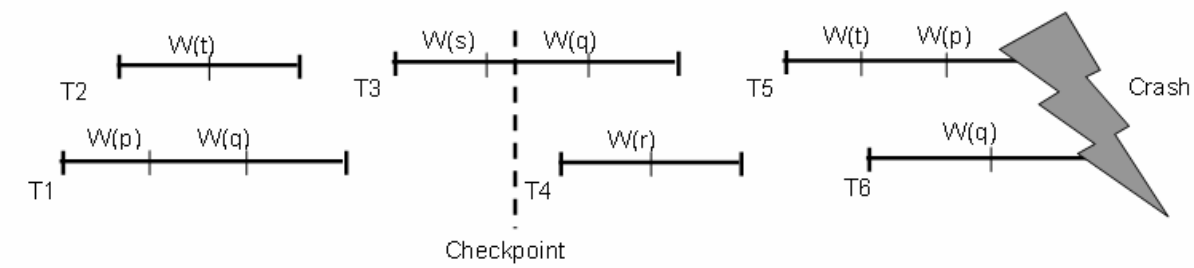
2.2



2.3

Cycle: T5 -> T4 -> T3 -> T5. Thus, the schedule is not conflict-serializable.

Part 3: Crash and Recovery



Operation	DB Buffer [page(LSN)]	Database [page(LSN)]	LogBuffer [LSN (page/transaction)]	Log File [LSN (page/transaction)]
BOT(T1)			100(BOT/T1)	
BOT(T2)			101(BOT/T2)	
W1(p)	p(102)		102(p/T1)	
W2(t)	t(103)		103(t/T2)	
W1(q)	q(104)		104(q/T1)	
EOT(T2)			105(EOT/T2)	100,101,102,103,104,105
EOT(T1)			106(EOT/T1)	106
Flush(t)		t(103)		
BOT(T3)			107(BOT/T3)	
W3(s)	s(108)		108(s/T3)	
Checkpoint		p(102),q(104),s(108)	CP ActiveTrans:{T3}	107, 108
BOT(T4)			109(BOT/T4)	
W3(q)	q(110)		110(q/T3)	
W4(r)	r(111)		111(r/T4)	
EOT(T3)			112(EOT/T3)	109,110,111,112
EOT(T4)			113(EOT/T4)	113
BOT(T5)			114(BOT/T5)	
W5(t)	t(115)		115(t/T5)	
Flush(t)		t(115)		114,115
BOT(T6)			116(BOT/T6)	
W5(p)	p(117)		117(p/T5)	
W6(q)	q(118)		118(q/T6)	
System error				

Basically, we continue with the same policy as taken from the beginning of the process. At the checkpoint, all changed buffer pages are written into the database. Log file is written sequentially (monotonically increasing LSN).

Winners: T3, T4 (Committed before crash)
Loosers: T5, T6 (did not commit before crash)
Redo Phase: q(110), r(111)
Undo Phase: t(115)