Yash Agarwal
21104039
B-15

1) R (A B C D E)

$A \rightarrow BC$
$CD \rightarrow E$
$B \rightarrow D$
$E \rightarrow A$

$A^+ = BCDEA$ ✓
$B^+ = DB$
$C^+ = C$
$D^+ = D$
$E^+ = ABCDE$ ✓

Number of candidate keys = 4
A, E, BC, DC

$BC^+ = BDCEA$ ✓
$BD^+ = DB$
$DC^+ = EABCD$ ✓

2) R:(A B C D E)
FD: { $A \rightarrow BC$, $CD \rightarrow E$, $B \rightarrow D$, $E \rightarrow A$ }

R_1 (A B C)
R_2 (A D E)

	A	B	C	D	E
R_1	α_A	α_B	α_C	β_{1D}	β_{1E}
R_2	α_A	~~β_{2B}~~	~~β_{2C}~~	α_D	α_E

$A \rightarrow BC$

	A	B	C	D	E
R_1	α_A	α_B	α_C	β_{1D}	β_{1E}
R_2	α_A	α_B	α_C	α_D	α_E

⇒ Lossless decomposition.

2) a)

T_1	T_2
R(A)	
W(A)	
	R(A)
	W(A)
R(B)	
W(B)	
	R(B)
	W(B)

$T_1 \rightarrow T_2$

No cycle
Hence the schedule is
Conflict serializable

b)

T_1	T_2
	R(A)
R(A)	
	W(A)
	R(B)
W(A)	

$T_1 \leftarrow T_2$

cycle
Hence it is not conflict serializable

for VS

S' $T_1 \rightarrow T_2$

T_1	T_2
R(A)	
W(A)	
	R(A)
	W(A)
	R(B)

	S	S'
Initial read A :	T_2	T_1 ✗

S''

T_1	T_2
	R(A)
	W(A)
	R(B)
R(A)	
W(A)	

		S	S''
Initial Read	A	T_2	T_2
	B	T_2	T_2
Initial write	A	T_2	T_2
Intermediate read	A	$T_1 \neq T_1$	

Not View serializability

Hence it is not a serializable schedule.

YASH AGARWAL
2110403
B15

T_1	T_2
R(A)	R(B)
R(B)	R(A)
if A=0 then B=B+1	if B=0 then A=A+1
W(B)	W(A)

let $A = 0 = B$.

a) $T_1 \rightarrow T_2$

T_1	T_2
R(A)	
R(B)	
if A=0 then B=B+1	
W(B)	
C	
	R(B)
	R(A)
	if B=0 then A=A+1
	W(A)
	C

$A = 0$
$B = \not{0}\ 1$

Consistency is preserered

$T_2 \rightarrow T_1$

T_1	T_2
	R(B)
	R(A)
	if B=0 then A=A+1
	W(A)
	C
R(A)	
R(B)	
if A=0 then B=B+1	
W(B)	
C	

$A = \not{0}\ 1$
$B = 0$

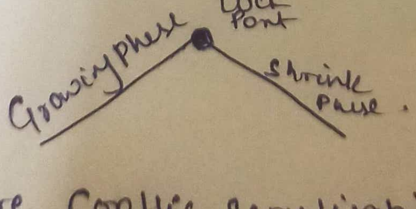
Consistency is preserved.

b) Execution of T_1 & T_2 non serializable Because they forms a cycle $T_1 \rightarrow T_2$ ↺

T_1	T_2
R(A)	
R(B)	
if A=0 then B=B+1	
W(B)	
	R(B)
	R(A)
	if B=0 then A=A+1
	W(A)
	C
C	

c) There is no parallel execution resulting in a serializable schedule, then from a we know that a is serializable schedule results in A=0 ∨ B=0. Suppose we start with read(A). Then when the schedule ends no matter when we run the steps of T_2, B=1. Now suppose we start executing T_2 prior to T_1, then T_2 read(B) will B=0 so when T_2 completes, A=1 Thus B=1 ∧ A=1 ¬(A=0 ∨ B=0) similar for starting with T_2 read(A)

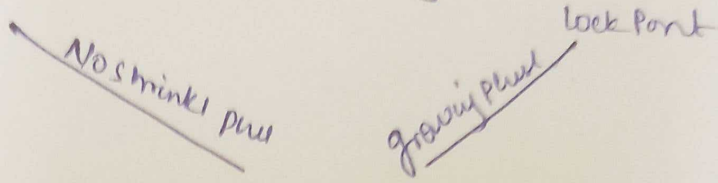
Q4 2 PL is a extension of lock based protocol
it has 2 phases
a) Shrinking phase: In this process, only unlocking is allowed.
b) Growing phase: In this phase, transaction only perform locking unlocking is not allowed



This ensure Conflict serializability and View serializability
But irrecoverable & Cascading schedules are possible and Deadlock may occur.
Variant of 2PL
(1) static / conservative 2PL: In this technique, a Transaction first acquires all the lock which it requires on every data item.
In this no growing phase is there — lock point — Shrinking
→ Prevents Deadlock but irrecoverable
& cascading schedules are possible

regrous 2PL

In this variant of 2PL there is no shrinking phase, ie the uncocking of data item is done only when Transaction is completed

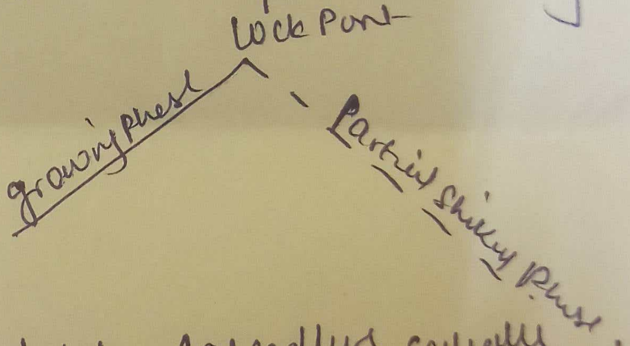


It ensure conflic serializability, View serializability, recoverable & cascadless schedule.

But Deadlock may occur.

Strict 2PL

It is innance form of reguorus 2PL in this, unlocking is done only of shared lock hence it has partial shrinking phase



In ensure CS, VS, recoverabilit, cascadless schedule.

But deadlock may occur.