Himanshu Dixit
B11
21103262

Date
Page

Database System & Web (15311CI312)

Assignment 2

Q.1. $R = \{A, B, C, D, E\}$

$A \rightarrow BC$

$CD \rightarrow E$

$B \rightarrow D$

$E \rightarrow A$

$(A+) \longrightarrow \{ABCDE\}$

$(CD+) \longrightarrow \{CDEAB\}$

$(BE+) \longrightarrow \{BEACD\}$

A, E, CD are the candidate keys

Q.2. $R = \{A, B, C, D, E\}$

$R_1 = \{A, B, C\}$

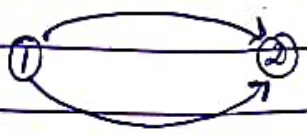
$R_2 = \{A, D, E\}$

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

Decomposition is lossless-join.

Q.3. (i)

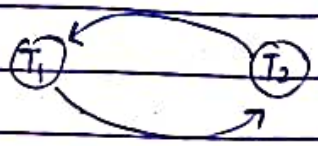
T_1	T_2
R(A)	
w(A)	
	R(A)
	w(A)
R(B)	
w(B)	
	R(B)
	w(B)



conflict serializable.

(iii)

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



not conflict serializable.

Q.4

```
T1 : read (A);
     read (B);
     if A=0 then B:=B+1;
     write (B);
T2 : read (B);
     read (A);
     if B=0 then A:=A+1;
     write (A);
```

(a) There are two possible executions : T_1T_2 & T_2T_1

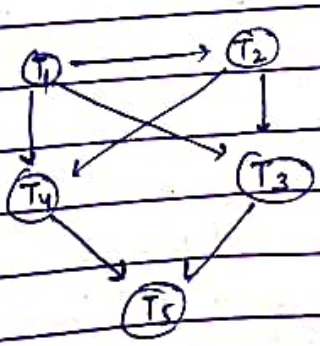
Case 1 :

	A	B	
initially	0	0	
after T_1	0	1	consistency met :
after T_2	0	1	A = 0 V B = 0 (T V F = T)

case 2 :

	A	B	
initially	0	0	consistency met :
after T_2	1	0	A = 0 V B = 0 (F V T = T)
after T_1	1	0	

(b)



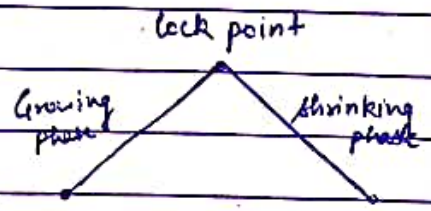
Any interleaving of T_1 & T_2 result in a non-serializable schedule

T_1	T_2
r(A)	
	r(B)
	r(A)
r(B)	
if A = 0 then	
B = B + 1	
	if B = 0 then
	A = A + 1
	w(A)
w(B)	

(C) There is no parallel execution resulting in a serializable schedule.

Q.5. Two-Phase Locking Protocol (2PL) :

* Each transaction in a schedule will have two phase
* Growing phase : Transaction can only obtain locks but not release any locks
* Shrinking phase : Transaction can only release locks but not obtain any locks.



* 2PL ensures serializability but some problems are there cascading roll backs, dead locks, irrecoverable schedule.

① Static 2PL :-

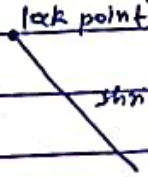
* Transactions obtain all the locks they need before the transactions begin.
* A transaction will not even obtain any locks if it cannot obtain all the locks it needs in its initial request.

* Shrinking phase works as usual, transaction can unlock any data item at any given point.

→ Independent from deadlock

→ possibility of irrecoverable schedules & cascading rollbacks (existence of dirty read).

→ ensures serializability



(2) Rigorous 2PL :-

* improvement over 2PL to ensures recoverability & cascadless schedules.

* To remove the problem of irrecorability & cascading rollbacks there should not be dirty read. All the locks must be held untill transaction commits. i.e no shrinking phase.

→ Ensure conflict serializability, view serializability, recoverability & cascadeless ness.

→ Suffer from deadlock & inefficiency.

(3) Strict 2PL :-

* Improvement over rigorous 2PL.

* in the shrinking phase, unlocking of shared locks can be done but not exclusive locks

* properties are same as rigorous 2PL but better concurrency provided by strict 2PL.