Q-2

Question No: 02

This is a subjective question, hence you have to write your answer in the Text-Field given below.

Consider the following classes of schedules: serializable, conflict-serializable, view-serializable, recoverable, avoids-cascading-aborts, and strict. For each of the following schedules, state which of the above classes it belongs to. If you cannot decide whether a schedule belongs in a certain class based on the listed actions, explain briefly. The actions are listed in the order they are scheduled, and prefixed with the transaction name. If a commit or abort is not shown, the schedule is incomplete; assume that abort/commit must follow all the listed actions:

(A) T1:R(X), T1:R(Y), T1:W(X), T2:R(Y), T3:W(Y), T1:W(X), T2:R(Y) **[1]**
(B) T1:R(X), T2:W(X), T1:W(X), T2:Abort, T1:Commit **[1]**
(C) T1:W(X), T2:R(X), T1:W(X), T2:Abort, T1:Commit **[1]**
(D) T1:W(X), T2:R(X), T1:W(X), T2:Commit, T1:Commit **[1]**
(E) T1:W(X), T2:R(X), T1:W(X), T2:Commit, T1:Abort **[1]**
(F) T2: R(X), T3:W(X), T3:Commit, T1:W(Y), T1:Commit, T2:R(Y), T2:W(Z), T2:Commit **[2]**
(G) R(X), T2:W(X), T2:Commit, T1:W(X), T1:Commit, T3:R(X), T3:Commit **[1]**
(H) T1:R(X), T2:W(X), T1:W(X), T3:R(X), T1:Commit, T2:Commit, T3:Commit **[1]**

A-2

Exercise 17.2 Consider the following classes of schedules: *serializable*, *conflict-serializable*, *view-serializable*, *recoverable*, *avoids-cascading-aborts*, and *strict*. For each of the following schedules, state which of the preceding classes it belongs to. If you cannot decide whether a schedule belongs in a certain class based on the listed actions, explain briefly.

The actions are listed in the order they are scheduled and prefixed with the transaction name. If a commit or abort is not shown, the schedule is incomplete; assume that abort or commit must follow all the listed actions.

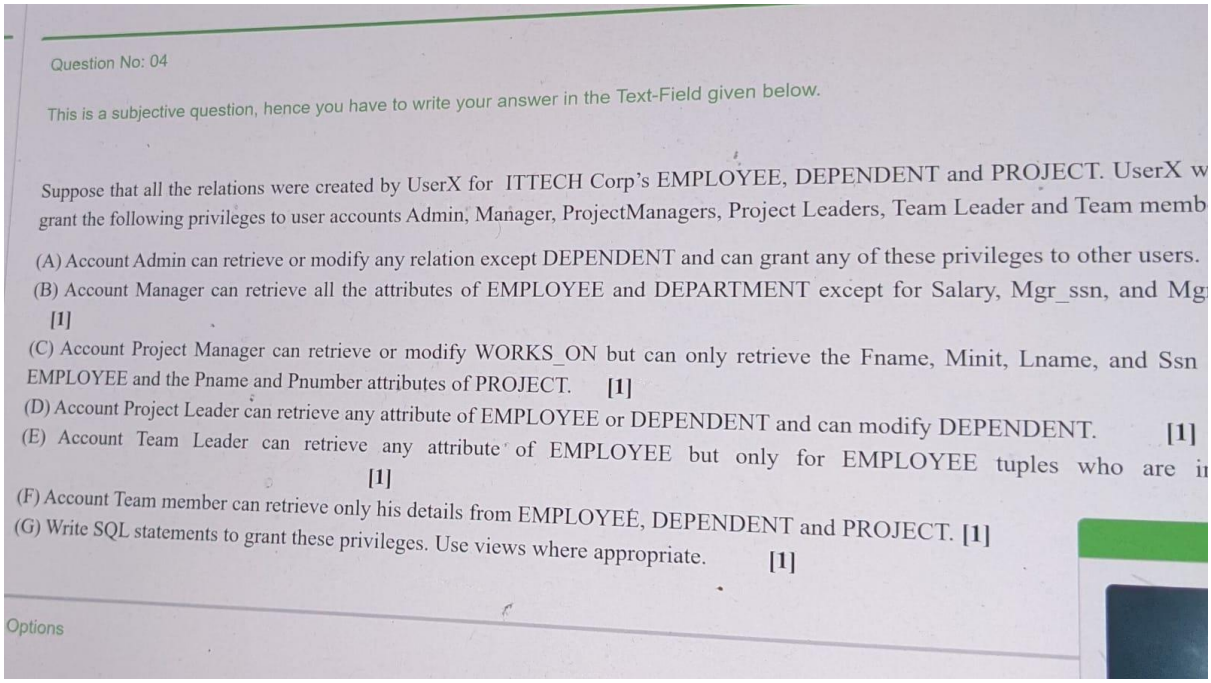
1. T1: R(X), T2: R(X), T1: W(X), T2: W(X)
2. T1: W(X), T2: R(Y), T1: R(Y), T2: R(X)
3. T1: R(X), T2: R(Y), T3: W(X), T2: R(X), T1: R(Y)
4. T1: R(X), T1: R(Y), T1: W(X), T2: R(Y), T3: W(Y), T1: W(X), T2: R(Y)
5. T1: R(X), T2: W(X), T1: W(X), T2: Abort, T1: Commit
6. T1: R(X), T2: W(X), T1: W(X), T2: Commit, T1: Commit
7. T1: W(X), T2: R(X), T1: W(X), T2: Abort, T1: Commit
8. T1: W(X), T2: R(X), T1: W(X), T2: Commit, T1: Commit
9. T1: W(X), T2: R(X), T1: W(X), T2: Commit, T1: Abort
10. T2: R(X), T3: W(X), T3: Commit, T1: W(Y), T1: Commit, T2: R(Y), T2: W(Z), T2: Commit
11. T1: R(X), T2: W(X), T2: Commit, T1: W(X), T1: Commit, T3: R(X), T3: Commit
12. T1: R(X), T2: W(X), T1: W(X), T3: R(X), T1: Commit, T2: Commit, T3: Commit

Answer:

1. Serizability (or view) cannot be decided but NOT conflict serizability. It is recoverable and avoid cascading aborts; NOT strict
2. It is serializable, conflict-serializable, and view-serializable regardless which action (commit or abort) follows It is NOT avoid cascading aborts, NOT strict; We can not decide whether it's recoverable or not, since the abort/commit sequence of these two transactions are not specified.
3. It is the same with 2.
4. Serizability (or view) cannot be decided but NOT conflict serizability. It is NOT avoid cascading aborts, NOT strict; We can not decide whether it's recoverable or not, since the abort/commit sequence of these transactions are not specified.
5. It is serializable, conflict-serializable, and view-serializable; It is recoverable and avoid cascading aborts; it is NOT strict.
6. It is NOT serializable, NOT view-serializable, NOT conflict-serializable; it is recoverable and avoid cascading aborts; It is NOT strict.
7. It belongs to all classes
8. It is serializable, NOT view-serializable, NOT conflict-serializable; It is NOT recoverable, therefore NOT avoid cascading aborts, NOT strict.
9. It is serializable, view-serializable, and conflict-serializable; It is NOT recoverable, therefore NOT avoid cascading aborts, NOT strict.
10. It belongs to all above classes.
11. It is NOT serializable and NOT view-serializable, NOT conflict-serializable; it is recoverable, avoid cascading aborts and strict.
12. It is NOT serializable and NOT view-serializable, NOT conflict-serializable; it is recoverable, but NOT avoid cascading aborts, NOT strict.

Refer 3 to 11 from above answer

Q-4



Question No: 04

This is a subjective question, hence you have to write your answer in the Text-Field given below.

Suppose that all the relations were created by UserX for ITTECH Corp's EMPLOYEE, DEPENDENT and PROJECT. UserX w grant the following privileges to user accounts Admin, Manager, ProjectManagers, Project Leaders, Team Leader and Team memb

(A) Account Admin can retrieve or modify any relation except DEPENDENT and can grant any of these privileges to other users.

(B) Account Manager can retrieve all the attributes of EMPLOYEE and DEPARTMENT except for Salary, Mgr_ssn, and Mg [1]

(C) Account Project Manager can retrieve or modify WORKS_ON but can only retrieve the Fname, Minit, Lname, and Ssn EMPLOYEE and the Pname and Pnumber attributes of PROJECT. [1]

(D) Account Project Leader can retrieve any attribute of EMPLOYEE or DEPENDENT and can modify DEPENDENT. [1]

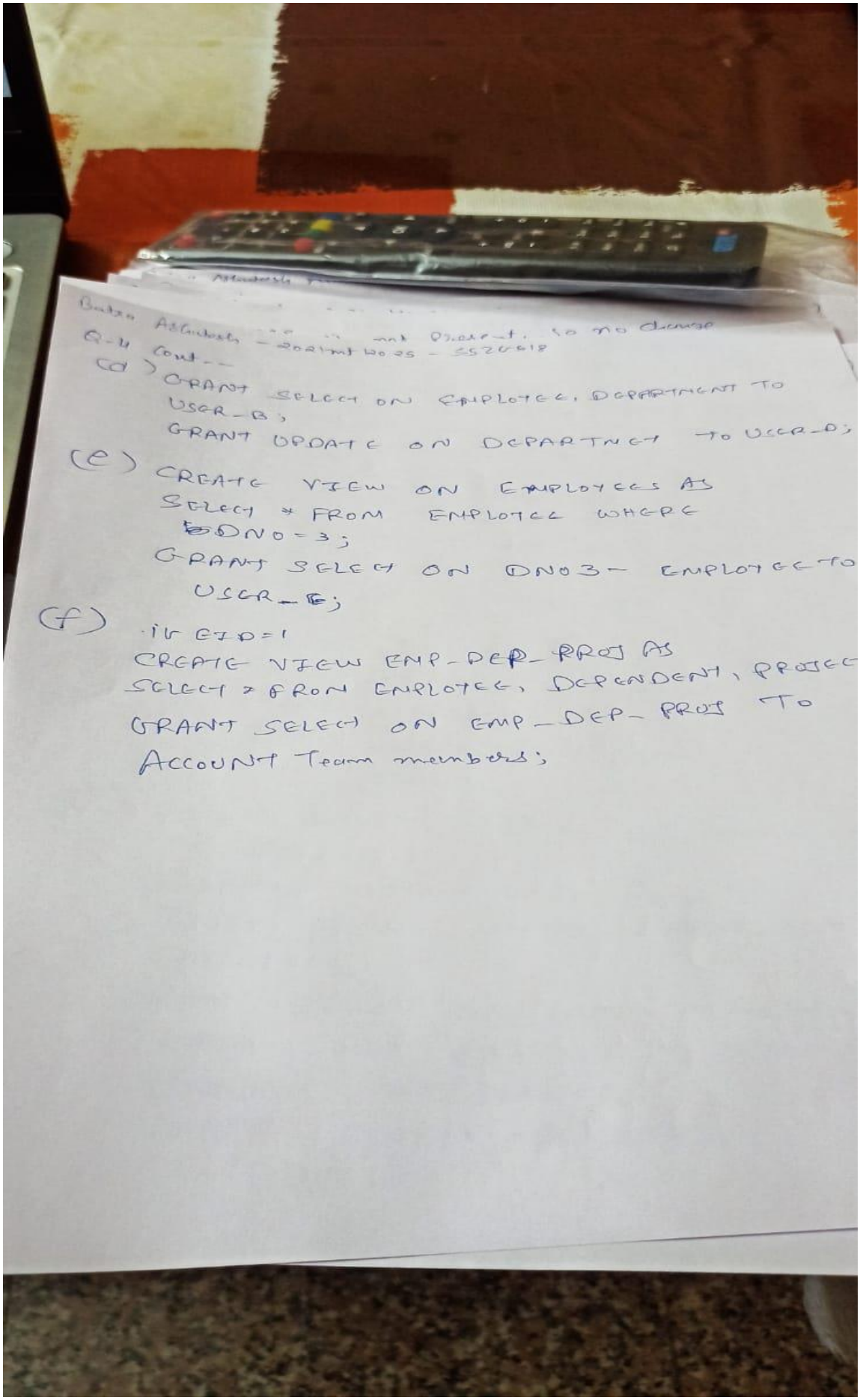
(E) Account Team Leader can retrieve any attribute of EMPLOYEE but only for EMPLOYEE tuples who are i [1]

(F) Account Team member can retrieve only his details from EMPLOYEE, DEPENDENT and PROJECT. [1]

(G) Write SQL statements to grant these privileges. Use views where appropriate. [1]

Options

A-4

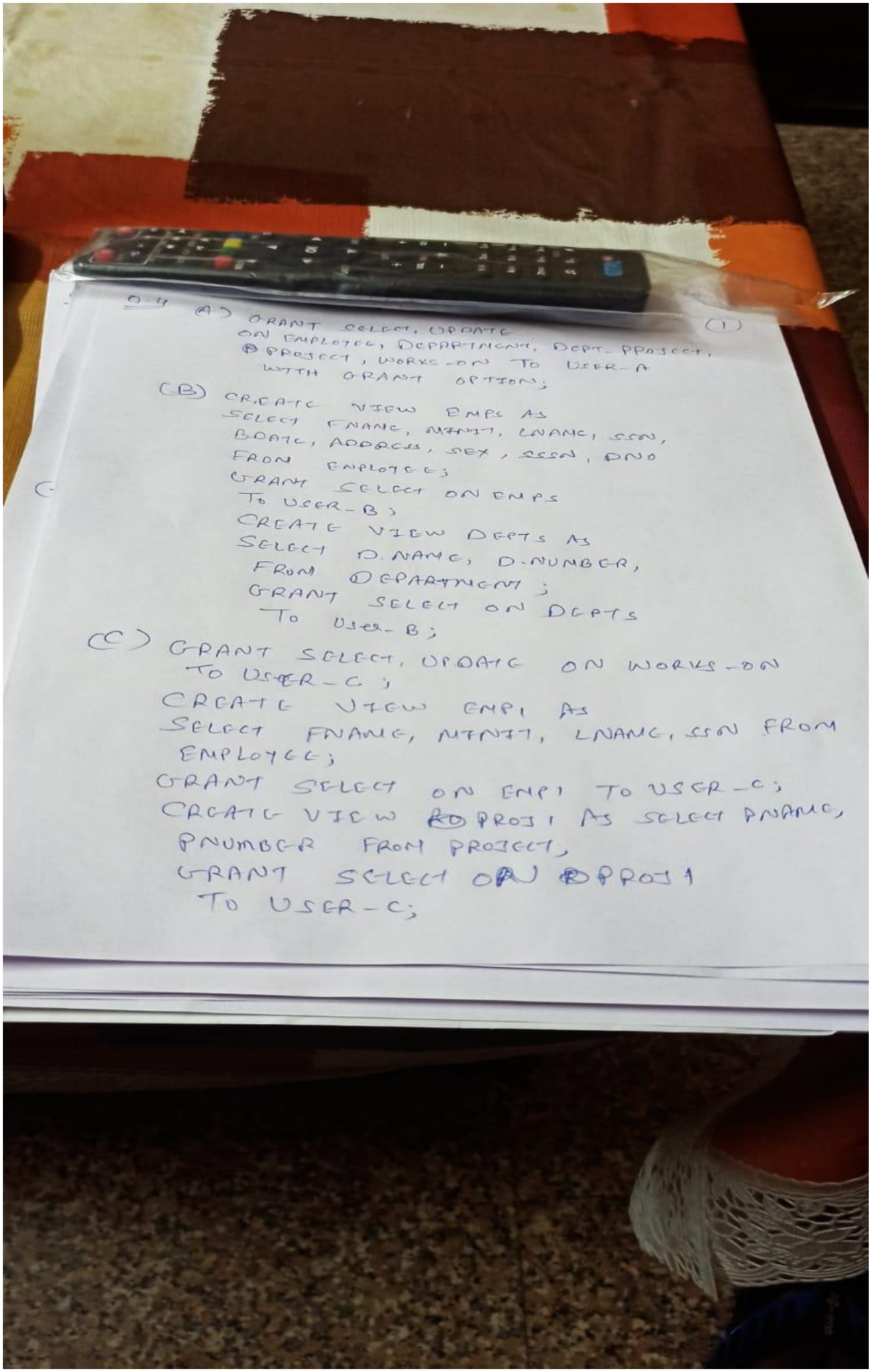


Q-4 Cont--

(d) GRANT SELECT ON EMPLOYEE, DEPARTMENT TO USER_B;
GRANT UPDATE ON DEPARTMENT TO USER_D;

(e) CREATE VIEW ON EMPLOYEES AS
SELECT * FROM EMPLOYEE WHERE
DNO = 3;
GRANT SELECT ON DNO3 - EMPLOYEE TO
USER_E;

(f) IF EID = 1
CREATE VIEW EMP_DEP_PROJ AS
SELECT * FROM EMPLOYEE, DEPENDENT, PROJECT
GRANT SELECT ON EMP_DEP_PROJ TO
ACCOUNT Team members;

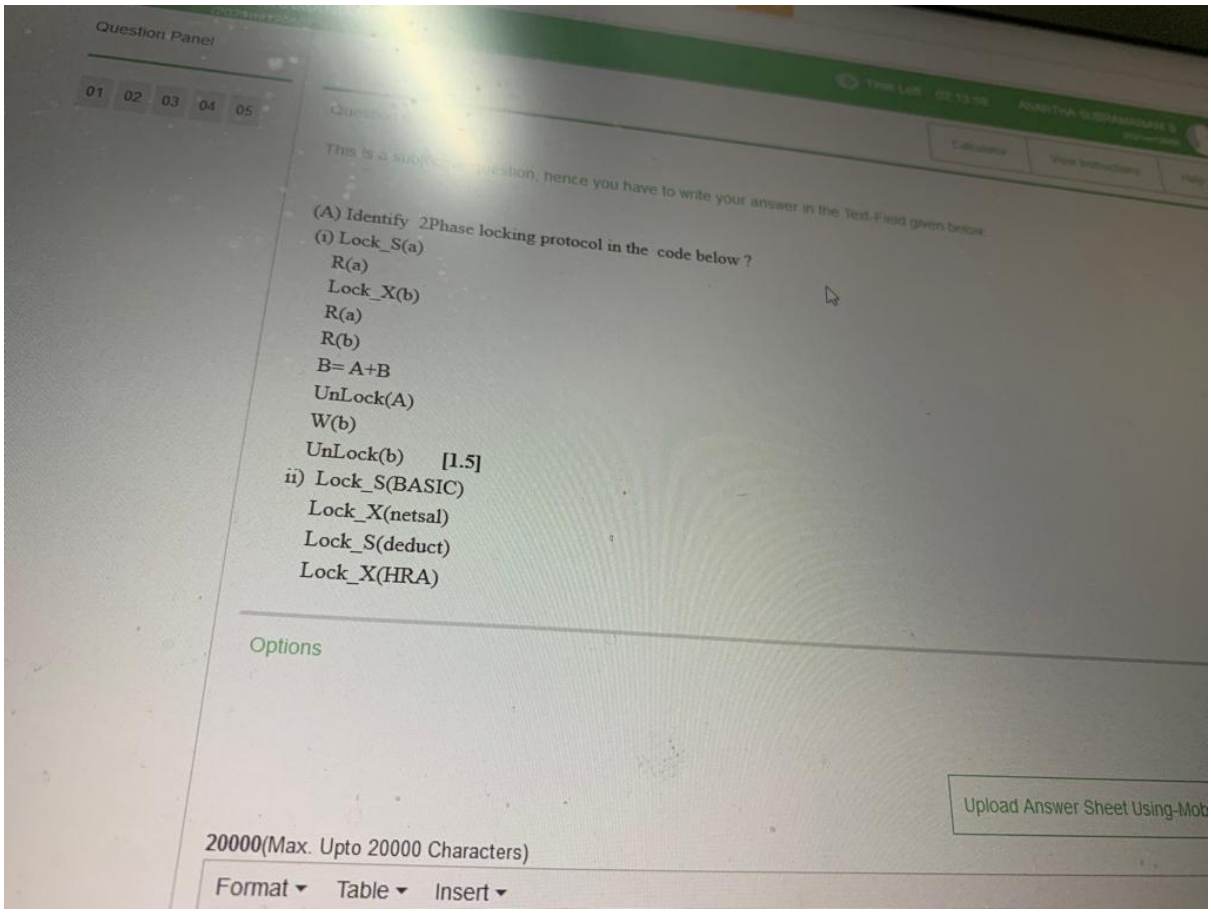


Q 4 (A) GRANT SELECT, UPDATE (1)
ON EMPLOYEE, DEPARTMENT, DEPT_PROJECT,
PROJECT, WORKS_ON TO USER_A
WITH GRANT OPTION;

(B) CREATE VIEW EMPS AS
SELECT FNAME, MINIT, LNAME, SSN,
BDATE, ADDRESS, SEX, SSSN, DNO
FROM EMPLOYEE;
GRANT SELECT ON EMPS
TO USER_B;
CREATE VIEW DEPTS AS
SELECT D.NAME, D_NUMBER,
FROM DEPARTMENT;
GRANT SELECT ON DEPTS
TO User_B;

(C) GRANT SELECT, UPDATE ON WORKS_ON
TO USER_C;
CREATE VIEW EMP1 AS
SELECT FNAME, MINIT, LNAME, SSN FROM
EMPLOYEE;
GRANT SELECT ON EMP1 TO USER_C;
CREATE VIEW PROJ1 AS SELECT PNAME,
PNUMBER FROM PROJECT,
GRANT SELECT ON PROJ1
TO USER_C;

Q-3


Question Panel
01 02 03 04 05
This is a subjective question, hence you have to write your answer in the Text-Field given below.
(A) Identify 2Phase locking protocol in the code below ?
(i) Lock_S(a)
R(a)
Lock_X(b)
R(a)
R(b)
B= A+B
UnLock(A)
W(b)
UnLock(b) [1.5]
ii) Lock_S(BASIC)
Lock_X(netsal)
Lock_S(deduct)
Lock_X(HRA)
Options
Upload Answer Sheet Using-Mob
20000(Max. Upto 20000 Characters)
Format
Table
Insert

Unlock (octal)
Unlock(HRA) [1.5]

(b) Consider the following code:

T1	T2
Lock_X(P) Read(P) P=P-100 Write(P) Lock_X(Q) Read(Q) Q=Q+100 Write(Q) UnLock(P) UnLock(Q)	Lock_S(Q) Read(Q) Lock_S(P) Read(P) Display(P+Q) UnLock(Q) UnLock(P)

Is there is any deadlock? If so how do you modify it to avoid deadlock? [3]

(c) Consider the following code

Begin Trans1

```
UPDATE Payroll SET wage = 600 WHERE empno=15;
```

Options

Upload Answer Sheet Using-Mobile

20000(Max. Upto 20000 Characters)

Format Table Insert

Formats B I

Q=Q+100
Write(Q)
UnLock(P)
UnLock(Q)
UnLock(Q)
UnLock(P)

Is there is any deadlock? If so how do you modify it to avoid deadlock? [3]

(c) Consider the following code

Begin Trans1

```
UPDATE Payroll SET wage = 600 WHERE empno=15;
UPDATE Payroll SET wage = 600 WHERE empno=17;
```

End

Begin Trans2

```
UPDATE Payroll SET wage = 100 WHERE empno=17;
UPDATE Payroll SET wage = 750 WHERE empno=15;
```

End

What would be the empno 15 ‘s and empno 17 ‘s wage be after **Trans1 and Trans2 RUNS concurrently?** **[3]**

Options

Upload Answer Sheet Usi

20000(Max. Upto 20000 Characters)

Format Table Insert

A3 –

Batra Ashutosh - 2021mt12025 - SS20518

Q-3 Cont..

(A)

i) Basic 2PL

~~Conserv~~

- Conservative ✗
- Strict 2PL ✗ No Commit.
- Rigorous ✓

ii) B.

- All locks required at start
- Exclusive lock get unlocked after commit, This is Strict 2PL and can also be Conservative 2PL

Q-3

(A) i) Basic 2PL
ii) Strict 2PL

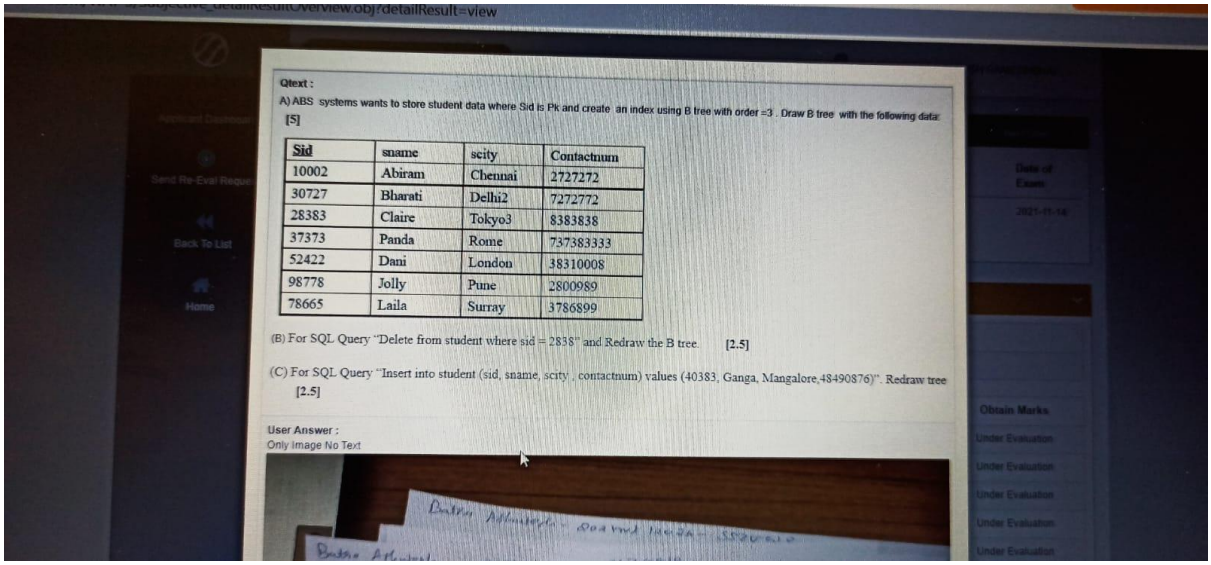
(B) It is deadlock situation
1) Yes T_1 is exclusive locking P
2) Before releasing / unlocking T_2 is Shared Locking P Lock-S(P)
3) T_2 is shared locking Q.
4) T_1 is exclusive locking Q, it is unlocked
5) T_2 can unlock Q after read, right away to remove deadlock

(C)

Time ↓	T_1	T_2
	$e_{15} = 600$	
	$c_{17} = 600$	
		$T_{17} = 100$
		$T_{15} = 750$

emp 15 = 750
emp 17 = 100 as they run T_1 T_2 concurrently

Q-1.



Qtext :

A) ABS systems wants to store student data where Sid is Pk and create an index using B tree with order =3 . Draw B tree with the following data: [5]

Sid	sname	scity	Contactnum
10002	Abiram	Chennai	2727272
30727	Bharati	Delhi2	7272772
28383	Claire	Tokyo3	8383838
37373	Panda	Rome	737383333
52422	Dani	London	38310008
98778	Jolly	Pune	2800989
78665	Laila	Surray	3786899

(B) For SQL Query "Delete from student where sid = 2838" and Redraw the B tree. [2.5]

(C) For SQL Query "Insert into student (sid, sname, scity , contactnum) values (40383, Ganga, Mangalore,48490876)". Redraw tree [2.5]

User Answer :

Only Image No Text

A1

Batra Ashutosh - 2021mt12025 - 5520518

Q-1

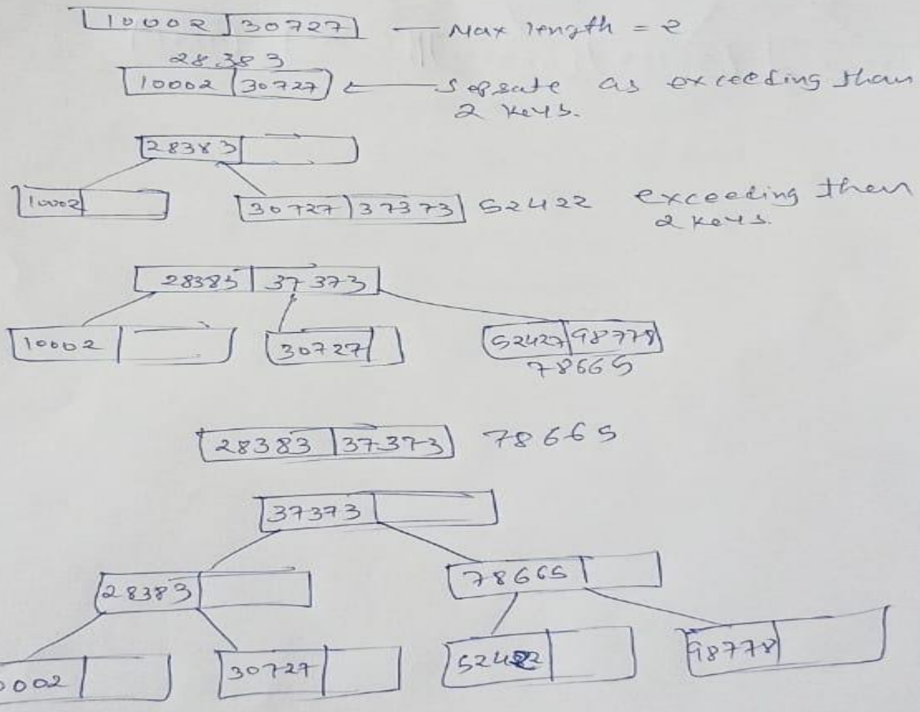
(A) Btree with order 3

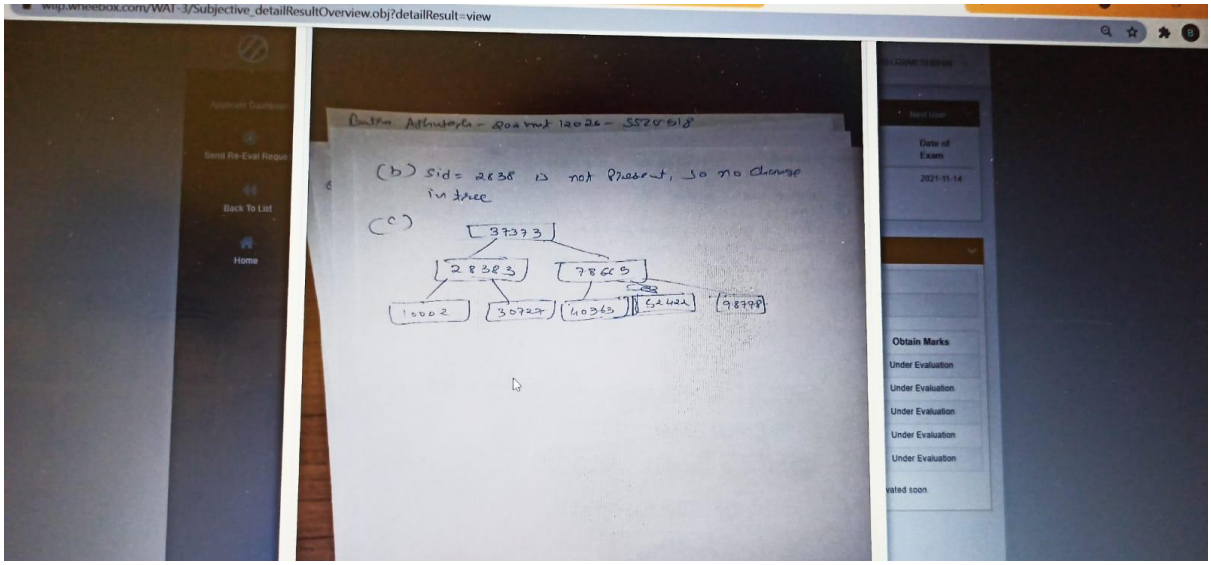
$m = 3$

Max Children = 3, min children = 2

Max key length = $m-1$ = 2.

Min key Length = $m/2 - 1 = 3/2 - 1 = 1$





Q-5

Qtext :

Given the following database schema:

EMPLOYEE								
FNAME	MINIT	LNAME	SSN	BDATE	ADDRESS	SEX	SALARY	DNO

DEPARTMENT			
DNAME	DNUMBER	MGRSSN	MGRSTARTDATE

DEPT_LOCATION	
DNUMBER	DLOCATION

PROJECT			
PNAME	PNUMBER	PLOCATION	DNUM

DEPENDENT				
ESSN	DEP-NAME	SEX	BDATE	RELATIONSHIP

For the following query, prepare the initial (canonical) query tree, then show how the query tree is optimized by the use of heuristic optimization.

"For every project located in 'Stanford', list the project number, the controlling department number, and the department manager's lastname, address and birthdate. "

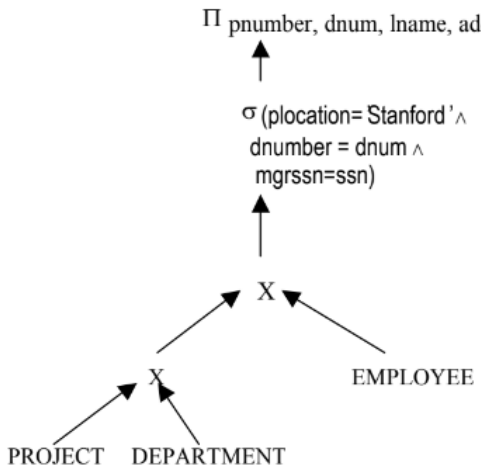
(A) Draw the query tree. **[5]**

(B) Draw the heuristic optimization tree. **[5]**

A5:

ANSWER:

a. **CANONICAL**



HEURISTIC OPTIMIZATION