Q1

HW 4 COMS363

1. (25 points) A transaction is an execution of a user program seen by a DBMS. An application developer organizes a sequence of SQL statements in a transaction such that if one SQL statement in the same transaction fails, all SQL statements in the same transaction must not succeed in updating the database. A SQL statement is implemented by a DBMS as a read action, a write action, or a sequence of read or write actions. Your answers to these questions are to be of no less details than those given in WK8_PracticeProblems_Solution.pdf. Consider the following actions taken by transaction T_1 on database objects X and Y: R(X), W(X), R(Y), W(Y)

a. (7 points)

Give an example of another transaction T_2 that, if run concurrently to transaction T_1 without concurrency control, does not interfere with T_1. Provide reasons to support your answer.

T_2 =

```
R(X)
R(Y)
R(X)
```

This does not interfere with T_1 when run concurrently because it has no writes, and you can have many computers reading data, but only one computer can be writing the data at a time.

Since T_2 can only read data, that means that whenever T_1 does finish a transaction, the transaction has not been interfered with.

b. (9 points)

Give an example of another transaction T_3 that, if run concurrently to transaction T_1 without some form of concurrency control, could create a write-read conflict (or dirty read)

with T_1. Specify in your schedule where the conflict occurs. Show how the Strict Two Phase Locking protocol can prevent the interference.

T_3 =

```
R(X)
W(X)
```

This will give a dirty read because if T_1 and T_3 both read X in at the same exact time, and then after that both will try to write X to the database.

This can cause problems because both T_1 and T_3 will overwrite each other based on whichever transaction completes first.

An example of this is if T_3 reads X, adds 1 to it, and then writes X, and T_1 does the exact same thing, both transactions will take place, but one will overwrite the other one, making one meaningless

Strict Two Phase Locking protocol can prevent this interference, because it will only release the read and write lock once a transaction has ended. Meaning that when T_3 tries to do anything while T_1 is still in progress, the computer will stop it from happening.

c. (9 points)

Give an example of another transaction T_4 that, if run concurrently to transaction T_1 without some form of concurrency control, could create read-write conflict (or unrepeatable read) with T_1. Specify in your schedule where the conflict occurs. Show how the Strict Two Phase Locking protocol can prevent the interference.

T_4 =

```
R(X)
-- do some work
R(X)
```

This could result in an unrepeatable read because in between T_4's reads of X, where it is doing some work, T_1 could finish and change X to something else, and the second read of X could be completely different from the first time.

Strict Two Phase Locking protocol can prevent the interference because there is no way for T_1 to be able to update X while T_4 is doing work.