CS W186 - Spring 2024
Exam Prep Section 8
Transactions & Concurrency

0) Transactions & Concurrency Warmup

1. Suppose a transaction T1 wants to scan a table R and update a few of its tuples. What kinds of locks should T1 have on R, the pages of R, and the updated tuples?

2. Is an S lock compatible with an IX lock?

3. Consider a table which contains two pages with three tuples each, with Page 1 containing Tuples 1, 2, and 3, and Page 2 containing Tuples 4, 5, and 6.

    i. Given that a transaction T1 has an IX lock on the table, an IX lock on Page 1, and an X lock on Tuple 1, which locks could be granted to a second transaction T2 for Tuple 2?

    ii. Given that a transaction T1 has an IS lock on the table and an S lock on Page 1, what locks could be granted to a second transaction T2 for Page 1?

1) Transactions & Concurrency

In this question, we will explore the key topics of transactions and concurrency: serializability, types of locks, two-phase locking, and deadlocks.

We will do this by actually running multiple transactions at the same time on a database, and seeing what happens. You may find it helpful (but not necessary) to draw some graphs:

- For the lock type questions, you may wish to draw a graph representing the whole database and which resources are being locked.
- For serializability questions, you may wish to draw a graph with a node for each transaction, and arrows if there are *conflicts* between transactions.
- For deadlock questions, you may wish to draw a graph with a node for each transaction, and arrows if a transaction is *waiting* for a lock held by another transaction.

We will use a database with tables A, B, C, ... and table A holds rows A1, A2, A3, ... and so on.

Consider the following sequence of operations:

Txn 1: IX-Lock(Database)	(1)
Txn 1: IX-Lock(Table A)	(2)
Txn 1: X-Lock(Row A1)	(3)
Txn 1: Write(Row A1)	(4)
Txn 1: Unlock(Row A1)	(5)
Txn 1: S-Lock(Row A2)	(6)
Txn 2: IX-Lock(Database)	(7)
Txn 2: IX-Lock(Table A)	(8)
Txn 2: X-Lock(Row A1)	(9)
Txn 2: Write(Row A1)	(10)
Txn 2: S-Lock(Row A2)	(11)
Txn 2: Read(Row A2)	(12)

1. Is Transaction 1 doing Two-Phase Locking so far?

2. Is Transaction 1 doing Strict Two-Phase Locking?

3. Is this schedule conflict-serializable so far? If not, what is the cycle?

4. Is this schedule serial so far?

Continuing with all operations so far:

Txn 2: SIX-Lock(Table B) (13)
Txn 2: X-Lock(Row B1) (14)
Txn 2: Write(Row B1) (15)
Txn 2: Unlock(Row B1) (16)
Txn 2: Unlock(Table B) (17)
Txn 1: SIX-Lock(Table B) (18)
Txn 1: X-Lock(Row B1) (19)
Txn 1: Read(Row B1) (20)

5. Is Transaction 2 doing Two-Phase Locking so far?

6. Is Transaction 2 doing Strict Two-Phase Locking?

7. Is this schedule conflict-serializable so far? If not, what is the cycle?

8. Suppose we start a new transaction, Transaction 3. What kind of locks can Transaction 3 acquire on the whole database?

9. Given the above answers, what kind of locks can Transaction 3 acquire on Table A?

10. Given the above answers, what kind of locks can Transaction 3 acquire on Row A3?

11. Given the above answers, what kind of locks can Transaction 3 acquire on Table B?

12. Given the above answers, what kind of locks can Transaction 3 acquire on Row B2?

13. What kind of locks can Transaction 1 acquire on Row B2?

Txn 2: IX-Lock(Table B) (21)
Txn 3: IX-Lock(Database) (22)
Txn 3: X-Lock(Table C) (23)

Txn 3: IX-Lock(Table A) (24)
Txn 3: X-Lock(Row A1) (25)
Txn 1: IX-Lock(Table C) (26)

14. We have now entered a deadlock. What is the waits-for cycle between the transactions?

15. We can end this deadlock by aborting the youngest transaction. Which transaction do we abort?

Alternatively, we could have avoided this deadlock in the first place by using *wound-wait* or *wait-die*. Recall from lecture that these methods cause transactions to sometimes abort, according to a priority order, when they try to acquire locks.

Let's say that the priority of the transaction is its number (Txn 1 is highest priority).

16. If we were using *wound-wait*, what is the first operation in this sequence that would cause a trans action to get aborted, and which transaction gets aborted?

17. If we were using *wait-die*, what is the first operation in this sequence that would cause a transaction to get aborted, and which transaction gets aborted?

2) Miscellaneous Transactions & Concurrency

Consider a database with objects X and Y and two transactions. Transaction 1 reads X and Y and then writes X and Y. Transaction 2 reads and writes X then reads and writes Y.

1. Create a schedule for these transactions that is not serializable. Explain why your schedule is not serializable.

2. Would your schedule be allowed under strict two-phase locking? Why or why not?

Now consider the following schedule.

	1	2	3	4	5	6	7	8
T1				X(B)	X(A)			S(D)
T2		X(D)	S(E)					
T3	X(E)						S(B)	
T4						X(A)		

3. Draw the waits-for graph for this schedule.

4. Are there any transactions involved in deadlock?

5. Next, assume that T1 priority > T2 > T3 > T4. You are the database administrator and one of your users purchases an exclusive plan to ensure that their transaction, Transaction 2, runs to completion. Assuming the same schedule, what deadlock avoidance policy would you choose to make sure Transaction 2 commits?