Student ID: 33049246

Student Name: Xiaowen Zhou

Unit Code: FIT 2094

Applied Class No: Applied 01, Melbourne Wed 18:00

(a)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **TIME** | **TRANS** | **ACTION** | **A** | **B** | **C** | **D** |
| **0** | **T1** | **UPDATE A** | X(T1) |  |  |  |
| **1** | **T1** | **UPDATE B** |  | X(T1) |  |  |
| **2** | **T2** | **READ C** |  |  | S(T2) |  |
| **3** | **T2** | **READ D** |  |  |  | S(T2) |
| **4** | **T3** | **UPDATE A** | X(T3) |  |  |  |
| **5** | **T2** | **UPDATE C** |  |  | X(T2) |  |
| **6** | **T1** | **ROLLBACK** |  |  |  |  |
| **7** | **T3** | **UPDATE C** |  |  | X(T3) |  |
| **8** | **T2** | **UPDATE B** |  | X(T2) |  |  |
| **9** | **T2** | **UPDATE A** | X(T2) |  |  |  |

* Does a deadlock exist in this transaction sequence?

I think no deadlock exists in this transaction sequence.

* Explain why you came to this conclusion.

FOLLOW STEP BY STEP:

In Time 0:

Transaction 1 requests an exclusive lock on item A (X(A)): X(T1)

--Lock on A.

In Time 1:

Transaction 1 requests an exclusive lock on item B (X(B)): X(T1)

--Lock on B.

In Time 2:

Transaction 2 requests a shared lock on item C (S(C)): S(T2)

--Lock on C.

In Time 3:

Transaction 2 requests a shared lock on item D (S(D)): S(T2)

--Lock on D.

In Time 4:

Transaction 3 requests an exclusive lock on item A (X(A)): X(T3)

--Lock on A.

Transaction 3 releases the exclusive lock on item A acquired earlier by Transaction 1 from Time 0

--No lock on A.

Transaction 3 requests an exclusive lock on item A again (X(A)): X(T3)

--Lock on A.

In Time 5:

Transaction 2 requests an exclusive lock on item C (X(C)): X(T2)

--Lock on C.

In Time 6:

Transaction 1 rolls back the transaction

--Without any locks present.

In Time 7:

Transaction 3 requests an exclusive lock on item C (X(C)): X(T3)

--Lock on C.

In Time 8:

Transaction 2 requests an exclusive lock on item B (X(B)): X(T2)

--Lock on B.

In Time 9:

Transaction 2 requests an exclusive lock on item A (X(A)): X(T2)

--Lock on A.

So no deadlock exists in this transaction sequence as no circular dependency is evident.

(b)

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **TRL ID** | **TRX NUM** | **PREV PTR** | **NEXT PTR** | **OPERATION** | **TABLE** | **ROW ID** | **ATTRIBUTE** | **BEFORE VALUE** | **AFTER VALUE** |
| 101 | 601 | Null | 102 | START | \*\*\*\*Start Transaction |  |  |  |  |
| 102 | 601 | 101 | 103 | UPDATE | PRODUCT | ABC | PROD\_QQH | 1205 | 1206 |
| 103 | 601 | 102 | 104 | UPDATE | PART | A | PART\_QQH | 567 | 556 |
| 104 | 601 | 103 | 105 | UPDATE | PART | B | PART\_QQH | 98 | 97 |
| 105 | 601 | 104 | 106 | UPDATE | PART | C | PART\_QQH | 549 | 548 |
| 106 | 601 | 105 | Null | COMMIT | \*\*\*\*End of Transaction |  |  |  |  |

…. add extra rows as needed