Example Transaction and Concurrency Questions

Purpose: to provide some example questions related to transaction management and concurrency control

Note: you can find answers to these questions on the course Moodle site, under "Selected solutions". I highly recommend that you try to answer them without looking at those answers first, and then compare your results to those posted.

1. We have discussed how a DBMS may support (or enforce) entity integrity, domain integrity, referential integrity, and transaction integrity.

   Give the most appropriate of these four types of integrity in answer to each of the following:

   (a) This is being supported (to at least some degree) in the Oracle DBMS by refusal to allow the insertion of an attribute value that is not the given type for that attribute (based on the type declaration for that attribute).

       (that is, this kind of integrity is being enforced when the Oracle DBMS will not permit the insertion of a row that includes a date value for a column declared to be of type integer)

   (b) This is being supported when a DBMS does not permit the insertion of a row with the same primary key as an already-existing row in that table.

   (c) This is being supported when a DBMS does not permit the insertion of a row with a foreign key value for which there is not a corresponding primary key value in the parent table.

   (d) Oracle SQL supports this with its COMMIT and ROLLBACK statements.

   (e) If a DBMS permitted you to insert duplicate rows within a table, it would not be supporting this.

   (f) One way in which Oracle DBMS supports this is with its "on delete restrict" default for foreign keys --- that is, in Oracle, if you attempt to delete a parent row with a primary key matching children rows with that value for their foreign key, you are not permitted to do so.

   (g) When a DBMS has automatic backup and recovery capabilities, it could be said to be providing support for this (in terms of the desired property of durability).

   (h) Oracle is providing additional support for this with the check clauses that one may use in a create table statement to restrict the allowed values in a particular column, which it then uses to only permit insertions of rows which have values for that column that satisfy the check clause.

2. Consider the five main database transaction properties, making up the acronym ACIDS.

   Give the most appropriate of these five properties in answer to each of the following:
(a) Which of these means that (at least in effect, if not in reality) the data used during the execution of a transaction cannot be used by a second transaction until the first one is completed -- or, at least conceptually, that the partial effects of incomplete transactions should not be visible to other transactions?

(b) Two of these are aided with the use of transaction logs. Which two, and for each, why?

(c) Which of these means that the concurrent execution of transactions is equivalent to the case where the transactions executed serially in some arbitrary order?

(d) Which of these stipulates that the database is expected to maintain its consistent state between transactions?

(e) Which of these -- really tied into the very definition of a transaction! -- requires that either all operations/parts/steps of a transaction be completed, or be as if that transaction had never begun?

(f) Dirty reads, nonrepeatable reads, and phantom reads are all examples of problems that can occur if which two of these properties are not ensured?

(g) Which of these is satisfied if performing a transaction does not violate database integrity -- if a transaction transforms a database from one state with all database constraints satisfied to another state still with all database constraints satisfied?

(h) Locking, by itself, is one approach to achieve which of these?

(i) ...but locking needs additional protocols, such as two-phased locking, to achieve which other of these?

3. Assume that you have a transaction log, in which all transaction operations since the last complete database backup are recorded in chronological order. A failure occurs at time X, and recovery is begun by starting with the last complete database state. In this system, recovery is done using rollforward, as discussed in lecture.

(a) In looking at the transaction log during recovery, it is discovered that transaction T1 had been rolled back/aborted before time X. Will its logged actions be re-done as part of the recovery process?

(b) In looking at the transaction log during recovery, it is discovered that transaction T2 had been committed before time X. Will its logged actions be re-done as part of the recovery process?

(c) In looking at the transaction log during recovery, it is discovered that transaction T3 had not yet been committed before time X. Will its logged actions be re-done as part of the recovery process?

Assume that A, B, C, D, and E are data items, and that T1, T2, T3, T4, T5, and T6 are transactions in all of the following problems.
4. Assume that your DBMS is using **binary locks** for concurrency control. Currently, \( T_1 \) has a binary lock on data item \( A \), and \( T_2 \) has a binary lock on data item \( C \). All other data items are unlocked.

   (a) \( T_3 \) requests a binary lock on \( B \). Will \( T_3 \) obtain the lock at this point, or will \( T_3 \) have to wait?

   (b) \( T_4 \) requests a binary lock on \( C \). Will \( T_4 \) obtain the lock at this point, or will \( T_4 \) have to wait?

5. Assume that your DBMS is using **shared/exclusive locks** for concurrency control. Currently \( T_1 \) has an exclusive/write lock on data item \( A \), and \( T_2 \) has a shared/read lock on data item \( C \). All other data items are currently unlocked.

   (a) Consider the operations Read and Write. Which of these can \( T_1 \) do to \( A \), given its lock  (read, write, neither read nor write, or both read and write)?

   (b) Consider the operations Read and Write. Which of these can \( T_2 \) do to \( C \), given its lock  (read, write, neither read nor write, or both read and write)?

   (c) \( T_3 \) requests an **exclusive lock** on \( D \). Will \( T_3 \) obtain the lock at this point, or will \( T_3 \) have to wait?

   (d) \( T_4 \) requests a **shared lock** on \( C \). Will \( T_4 \) obtain the lock at this point, or will \( T_4 \) have to wait?

   (e) \( T_5 \) requests an **exclusive lock** on \( C \). Will \( T_5 \) obtain the lock at this point, or will \( T_5 \) have to wait?

   (f) \( T_6 \) requests a **shared lock** on \( B \). Will \( T_6 \) obtain the lock at this point, or will \( T_6 \) have to wait?

6. Transaction \( T_1 \) has an exclusive lock on data item \( A \), and has requested, and is waiting for, a shared lock on data item \( B \). But, \( T_2 \) has an exclusive lock on data item \( B \), and has requested, and is waiting for, an exclusive lock on data item \( A \). What is this situation an illustration of?

7. Consider **timestamps**. Please consider the following questions **independently** of one another.

   (a) If the DBMS were to **not** permit one of these transactions to perform a desired action, what would it subsequently need to do to that transaction?

   (b) Transaction \( T_1 \), with timestamp 18, wishes to write a data item \( A \), and the DBMS discovers that this data item was last **read** by a transaction with timestamp 8 (\( R\text{-ts}(A) = 8 \)), and last **written** by a transaction with timestamp 7 (\( W\text{-ts}(A) = 7 \)). Will the DBMS permit \( T_1 \) to perform this action? Give \( R\text{-ts}(A) \) and \( W\text{-ts}(A) \) after this. (There should be three parts in your answer, note: will it be permitted, what's \( R\text{-ts}(A) \) afterwards, and what's \( W\text{-ts}(A) \) afterwards...)

   (c) Transaction \( T_2 \), with timestamp 19, wishes to write a data item \( B \), and the DBMS discovers that this data item was last **read** by a transaction with timestamp 22 (\( R\text{-ts}(B) = 22 \)), and last **written** by a transaction with timestamp 21 (\( W\text{-ts}(B) = 21 \)). Will the DBMS permit \( T_2 \) to perform this action? Give the \( R\text{-ts}(B) \) and the \( W\text{-ts}(B) \) after this.
(d) Transaction T3, with timestamp 13, wishes to read a data item C, and the DBMS discovered that this data item was last read by a transaction with timestamp 17 (R-ts(C) = 17), and last written by a transaction with timestamp 12 (W-ts(C) = 12). Will the DBMS permit T3 to perform this action? Give R-ts(C) and W-ts(C) after this.

(e) Transaction T4, with timestamp 24, wishes to read a data item D, and the DBMS discovers that this data item was last read by a transaction with timestamp 12 (R-ts(D) = 12), and last written by a transaction with timestamp 29 (W-ts(D) = 29). Will the DBMS permit T4 to perform this action? Give R-ts(D) and W-ts(D) after this.

(f) Transaction T5, with timestamp 18, wishes to write a data item E, and the DBMS discovers that this data item was last read by a transaction with timestamp 20 (R-ts(E) = 20), and last written by a transaction with timestamp 16 (W-ts(E) = 16). Will the DBMS permit T5 to perform this action? Give R-ts(E) and W-ts(E) after this.