Data Management – exam of 02/09/2013

Problem 1 Consider the following schedule

$$S = r_1(A)\, r_1(C)\, w_2(A)\, r_2(B)\, r_1(B)\, w_3(A)\, w_3(B).$$

1. Tell whether S is accepted by the 2PL scheduler with exclusive and shared locks. If the answer is yes, then show the schedule obtained from S by adding suitable lock and unlock commands. If the answer is no, then explain the answer.
2. Tell whether S is conflict-serializable. If the answer is yes, then show a serial schedule that is conflict-equivalent to S. If the answer is no, then explain the answer.

Problem 2 Illustrate the definition of the "update loss anomaly", and prove or disprove the following statements:

1. Every schedule that is view serializable does not suffer from the update loss anomaly.
2. Every schedule that follows the 2PL protocol (with both shared and exclusive locks) does not suffer from the update loss anomaly.
3. Any schedule that is in ACR (i.e., that avoids cascading rollback) does not suffer from the update loss anomaly.

Problem 3 Assume that a system failure occurs when the log is as follows (where B means "begin", C means "commit", A means "abort", and CK means "checkpoint"):
$B(T_1), I(T1, o_1, a_1), B(T_2), U(T_2, o_1, b_1, a_2), CK(T_1, T_2), C(T_2), B(T_3), D(T_1, o_2, b_2), I(T_3, o_3, a_3),$
$CK(T_1, T_3), B(T_4), I(T_4, o_4, a_5), C(T_3), D(T_4, o_5, b_5), A(T_4), I(T_1, o_6, a_6), B(T_5), U(T_5, o_6, b_6, a_7),$
$D(T_5, o_6, b_8), I(T_1, 0_6, a_9).$
Illustrate all the actions performed by the recovery subsystem when the failure occurs, assuming that a mixed effect strategy is adopted.

Problem 4 Consider the relations `R(A,B,C)` and `Q(D,E,F,G)`, and consider the join of the two relations on the condition

$$(\texttt{B} = \texttt{D} \ \text{ and } \ \texttt{C} = \texttt{E})$$

We know that `R` has 600 pages, `Q` has 5000 pages, the buffer has 152 pages free, and there is a tree-based index on `Q` with search key `D,E,F` that allows a cost of 3 page accesses for the equality search query. Answer the following questions:

1. Is the "nested loop" algorithm applicable for computing the join? If yes, which is the cost of the join in this case? Motivate the answer.
2. Is the "block nested loop" algorithm applicable for computing the join? If yes, which is the cost of the join in this case? Motivate the answer.
3. Is the "index nested loop" algorithm applicable for computing the join? If yes, which is the cost of the join in this case? Motivate the answer.

Problem 5 Consider the relation `Author(code,name,age)`, that stores the data about the authors of books, and the relation `WrittenBy(bookcode,authorcode)`, that tells which are the authors of the various books. We know that the number of books is 44.000.000, that each book has an average of 3 authors, that each value of each attribute, as well as each page address in our DBMS require 10 bytes, and that the size of every page in our DBMS is 410 bytes. We also know that we have a hash-based index on the attribute `code` of the relation `Author`, and a B$^+$-tree unclustered index using alternative 3 on the attribute `bookcode` of the relation `WrittenBy`. Consider the query that, given a particular book b, retrieves the age of all the authors of b, and tell how many page accesses are needed in order to answer such query.