Data Management – AA 2014/15 – exam of 30/1/2015

Problem 1
Let `R` and `S` be two relations (bags of tuples), stored in 100.000 pages, and 50.000 pages, respectively. Suppose our buffer has 400 frames available. Describe in detail (step by step) the algorithm you would use to compute the "bag intersection" of `R` and `S`, and tell which is the cost of the algorithm in terms of page accesses (ignoring, as usual, the cost of producing the output).

Problem 2
Let O be a total order on the transactions $T_1, T_2, \ldots, T_n$, and let $T_h <_O T_k$ denote the fact that T_h comes before T_k according to O. Let `P` be the concurrency strategy defined for schedules on transactions $T_1, T_2, \ldots, T_n$ as follows: a schedule S is accepted by `P` if and only if (i) no action $w_j(x)$ in S precedes an action $r_i(x)$ in S with $T_i <_O T_j$, and (ii) no action $w_j(x)$ in S precedes an action $w_i(x)$ in S with $T_i <_O T_j$. Prove or disprove the following claim: every schedule accepted by `P` is view serializable.

Problem 3
Consider the following schedule S, where we assume that the timestamp of each transaction T_i is i:

$$r_4(y)\, w_1(z)\, r_2(y)\, w_3(x)\, w_1(y)\, r_1(x)\, r_3(z)\, c_3\, w_5(z)\, c_2\, w_5(y)\, c_5\, w_4(z)\, c_1\, r_4(x)\, c_4$$

You are asked to:

- Illustrate the various steps carried out by the timestamp-based scheduler when analyzing the above schedule.
- Tell whether the above schedule is conflict serializable.
- Tell whether the above schedule is in 2PL with shared and exclusive locks.
- Tell whether the above schedule is recoverable.

Problem 4
(i) Discuss the methods for storing and managing non-clustering sorted index with duplicates. (ii) Suppose we have a relation `R` with 100.000 tuples, where `R` has 5 attributes (called `A`,`B`,`C`,`D` and `E`), each attribute is of size 10 bytes, and for each value of `A`, `R` contains 2 tuples with that value. Assuming that the size of each page is 1000 bytes, tell which is the number of page accesses needed for searching for all tuples of `R` with value V in attribute `A` in the following two scenarios:

1. `R` has no index, and is sorted on `A`;
2. for `R` there is a non-clustering sorted index with duplicates on attribute `A`.

Problem 5
Consider the relation `Engineer(code,department,lastname,age,role)`, which stores information about engineers, where `code,department` form the key. The most frequent queries posed to such relation are:

- Produce the sorted list of code, department, and last name of all engineers, without duplicates;
- Given a code and a department, find the age and the role of the engineer with the given code and the given department.

We know that relation `Engineer` has 14.000.000 tuples, the size of every memory page is 11.200 B, and the size of each field (relation attribute, or pointer) is 20 B.

Tell which method would you choose for representing the relation in secondary storage, taking into account that your goal is to execute the above queries efficiently. Also, for each query, tell which algorithm would you choose for executing the query, and how many page accesses would be needed for computing the answer to the query, given the chosen algorithm.