Data Management – exam of 1/7/2015

Problem 1 Consider the following schedule

$$S = r_1(Y)\, r_1(X)\, w_2(X)\, r_3(W)\, w_4(W)\, w_1(W)\, w_2(Y)\, w_2(W)\, w_3(Z)\, w_4(Z)\, w_3(T)$$

1. Tell whether S is accepted by the 2PL scheduler with exclusive and shared locks. If the answer is yes, then show the 2PL 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 view-serializable. If the answer is yes, illustrate a serial schedule which is view-equivalent to S. If the answer is no, then explain the answer.
3. Answer all the following questions, motivating the answers: (*i*) Is S recoverable? (*ii*) Is S ACR? (*iii*) Is S strict? (*iv*) Is S rigorous?

Problem 2 A schedule S is said to be *semi-view-serializable* if there exists a serial schedule S' on the same transactions of S such that S and S' have the same "read-from" relation.

1. Prove or disprove that every semi-view-serializable schedule is conflict serializable.
2. Prove or disprove that every conflict serializable schedule is semi-view-serializable.

Problem 3
Describe what a clustering, secondary, non-unique, dense sorted index (without duplicates in the index) is. Describe a picture of an example of such index, and illustrate the algorithm that, given a value K, find all data records with the value K in the search key.

Problem 4
Consider the relation `Flight(code,date,company,departure,arrival)` that stores information about 1.000.000 flights in the last 20 years, with the code of the flight, the date, the air company, the airport of departure and the airport of arrival. We assume that all fields in every record have the same size of 20 bytes, independently of the field type, and the size of each page is 1000 bytes. There is a sparse B^+-tree index on `Flight` with search key $\langle$`date`$\rangle$, using alternative 2. Consider the query that, given a month of an year, asks for the code and the air company of all the flights with the date falling in that month. Tell which algorithm you would use for executing the query, and how many page accesses such algorithm needs for computing the answer (a uniform distribution of flights among the various months can be assumed).

Problem 5
Describe in detail the one-pass algorithm for computing the natural join between $R(X, Y)$ and $S(Y, Z)$, where Y denotes the attributes that R and S have in common, in the case where R and S are not sorted, and there is no index defined on them. Which are the conditions under which such algorithm can be applied?