Data Management – AA 2016/17 – exam of 12/01/2017

Problem 1
We have to sort a relation `R` with 375 pages using the multipass (or, k-way) merge sort, and initially we have 200 free frames in the buffer. However, the system is currently very busy, and every time a run is written in secondary storage during the execution of the algorithm, after such writing the number of free frames in the buffer is halved. Describe in detail what happens during the execution of the multipass merge sort algorithm in this situation, and tell how many pages are accessed during such execution.

Problem 2
Assume that relation `Customer(firstName,lastName,yearOfBirth,salary)` (where `lastName` is a key), has 650.000 tuples, each attribute and each pointer in the system occupies 100 Bytes, each page has space for 4.000 Bytes, the last names of customers are equally distributed on the first letter over the 26 letters of the alphabet, and the most frequent query on `Customer` asks for all customers whose last name falls into a given range.

2.1 Tell which method would you use to store the relation.

2.2 Tell which algorithm would you use to answer the following query Q

```
select *
from Customer
where lastName >= 'Rabad' and lastName <= 'Suza'
order by salary
```

2.3 Assuming that 80 free buffer frames are available, tell which is the cost of executing the algorithm you have defined for item 2.2, in terms of the number of page accesses.

Problem 3
Let `S(A,B,C)` be a relation with 40.000.000 tuples in 200.000 pages, and let `V(C,D)` be a relation stored in a heap file of 10.000 pages with primary key `C`, and with an associated hash-based index with search key `C`, requiring 2 page accesses for searching for the tuple with a given value of `C`. We have to compute the natural join between `S(A,B,C)` and `V(C,D)`. If M is the number of free frames in the buffer, tell if there are values for M such that executing the block nested-loop join algorithm with such number of free frames is more efficient (with respect to the number of page accesses) than executing the index-based join algorithm with the same number of free frames. Explain your answer in detail.

Problem 4
Consider the relation `CAR(code,owner,type,year)`, storing information about cars, each one with its type, its owner, and the year of its construction. `CAR` has 500.000 tuples stored in a heap file, where each page contains 50 tuples. Consider the aggregate query Q that, for each owner o, computes the number of cars owned by o, and assume that we have a good hash function on `owner` that distributes the tuples of `CAR` uniformly, and that we have 101 free buffer frames.

4.1 Which algorithm would you use for computing Q?

4.2 Describe in detail the algorithm chosen.

4.3 Tell which is the cost of executing the algorithm in terms of number of page accesses.

Problem 5
Consider the following schedule

$$S = r_1(x)\, w_3(x)\, w_3(z)\, c_3\, w_2(x)\, w_2(y)\, c_2\, r_4(x)\, w_4(z)\, w_1(y).$$

5.1 Tell whether S is a 2PL schedule or not, explaining the answer in detail.

5.2 Tell whether S is view-serializable or not, explaining the answer in detail.

5.3 Describe the behavior of the timestamp-based scheduler when processing the schedule S, assuming that, initially, rts(α)=wts(α)=wts-c(α)=cb(α)=`true` for each element α of the database, and assuming that the subscript of each action denotes the timestamp of the transaction executing such action.

5.4 Tell whether S is strict or not, explaining the answer in detail.