Data Management – exam of 08/06/2022 (Compito A)

Problem 1

Given a schedule S, a serial schedule S_1 on the same transactions as S is said to be "begin-order preserving with respect to S" if it satisfies the following property: for every pair of transactions T_i, T_j in S, if the first action of T_i precedes the first action of T_j in S, then T_i precedes T_j in S_1. A schedule S is called *begin-order preserving conflict serializable* if there exists a serial schedule S_1 on the same transactions that is both conflict equivalent to S and begin-order preserving with respect to S.

1.1 Prove or disprove the following claim: every conflict serializable schedule is "begin-order preserving conflict serializable".

1.2 Is the problem of checking whether a schedule is "begin-order preserving conflict serializable" decidable? If the answer is negative, then motivate the answer; if the answer is positive, then exhibit an algorithm for the problem, provide evidence of the correctness of the algorithm and illustrate its computational conplexity.

Problem 2

Consider the following schedule S (where we have relaxed the condition that no transaction contains more than one occurrence of the same action):

$$B(T_1)\ r_1(D)\ r_1(A)\ w_1(A)\ B(T_2)\ r_2(A)\ w_2(A)\ B(T_3)\ r_3(D)\ w_3(D)\ r_1(D)\ c_3\ r_1(D)\ c_1\ c_2$$

where the action B means "begin transaction", the initial values of A and D are 10 and 30, respectively, and every write action increases the value of the element on which it operates by 10. Suppose that S is executed by PostgreSQL, and describe what happens when the scheduler analyzes each action (illustrating also which are the values read and written by all the "read" and "write" actions) in both the following two cases: (1) all the transactions are defined with the isolation level "read committed"; (2) all the transactions are defined with the isolation level "repeatable read".

Problem 3

Let τ indicate the ternary operator such that $\tau(R, S, T) = R \cup_s (S - T)$, where R, S and T are three relations with the same schema and without duplicates, $\cup_s$ indicates set union and $-$ indicates set difference.

3.1 Design and describe in detail a two pass algorithm that, given R, S, T, each one stored as a heap, computes $\tau(R, S, T)$.

3.2 Tell under which condition the algorithm can be used and illustrate the cost of the algorithm in terms of number of page accesses.

Problem 4

Consider the relations `Restaurant(code,citycode,seats)` with 18.000.000 tuples, and `City(citycode,region,country)` occupying 195 pages, each page with 100 tuples (keys are underlined). We assume that 150 values are in the attribute `country`, that each value (regardless of the type) requires the same number of bytes and that we have 200 frames available in the buffer. Consider the query

```
select country, sum(seats)
from Restaurant r, City c where r.citycode = c.citycode
group by country
```

and describe the algorithm you would use to execute the query, illustrating the number of page accesses required by the execution of the algorithm.

Problem 5

Consider a graph database with nodes of type `Player` with properties `name` (identifying the player) and `age`, nodes of type `Team` with properties `name` (identifying the team) and `city`, and edges of type `PlayedIn`, connecting each player with the teams they have played in. In such database, for example, one may represent the node `p1` of type `Player` with `name`:"Totti" and `age`:39 connected to node `t1` of type `Team` with `name`:"Roma" and `city`:"Roma" by means of an edge of type `PlayedIn`.

5.1 Illustrate how you would represent the above database as a schema in the relational model.

5.2 Assuming that the three most relevant queries are (1) given the name of a player, compute the teams where the player played, (2) produce the sorted list of ⟨name of player, name of team⟩ for players and the teams they have played in, and (3) compute all the names of the teams of a given city, describe the file organizations you would choose for each of relations in the relational database mentioned above and then describe the algorithms for executing the three queries on the basis of such file organizations.