Data Management – exam of 13/07/2022

Problem 1

In a schedule S on transactions $\{T_1, \ldots, T_n\}$ we say that two transactions T_i, T_j share the element X of the database if there exist actions $\alpha(X)$ in T_i and $\beta(X)$ in T_j such that α is either r_i or w_i and β is either r_j or w_j. Moreover, S is called "chary" if (i) no transaction in S uses the same element twice, and (ii) for every $T_i, T_j \in \{T_1, \ldots, T_n\}$, T_i and T_j share at most one element. Prove or disprove the following claims:

1.1 Every chary schedule is view-serializable.

1.2 Every chary schedule on two transactions is conflict-serializable.

1.3 Every chary schedule on two transactions is a 2PL schedule with exclusive and shared locks.

Problem 2

Let S be the schedule: $r_1(Z)\, w_3(Y)\, w_3(V)\, r_1(Y)\, r_2(V)\, w_2(Y)\, w_3(X)\, r_2(X)\, r_2(Z)\, r_3(Z)\, w_4(Z)\, w_4(X)\, w_2(X)$

2.1 Tell whether S is accepted by the 2PL scheduler with exclusive and shared locks. If the answer is yes, then specify the 2PL schedule obtained from S by adding suitable lock and unlock commands. If the answer is no, then explain the answer.

2.2 Tell whether S is view-serializable. If the answer is yes, then illustrate a serial schedule which is view-equivalent to S. If the answer is no, then explain the answer.

2.3 Answer all the following questions, motivating the answers: (i) Is S recoverable? (ii) Is S ACR? (iii) Is S strict?

Problem 3

Let R(<u>A</u>,B,C), S(<u>A</u>,D,E), T(A,B,C) be three tables (where T is a bag) and let τ indicate the ternary operator such that $\tau(\mathsf{R,S,T}) = \delta(\mathsf{T} \cup_b \pi_{\mathsf{A,B,C}}(\mathsf{R} \bowtie \mathsf{S}))$, where δ denotes duplicate elimination, $\cup_b$ denotes bag union and $\bowtie$ denotes natural join.

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

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

3.3 Tell what does it change if all the tables have A as key and are stored as sorted file with search key A.

Problem 4

Consider the relations Flight(<u>code</u>,company,type) with 1.000 pages and 10.000 tuples, and Ticket(<u>number</u>,code,company,type) with 2.000 pages and an associated index on Ticket with search key ⟨company,type⟩, for which we know that the cost of retrieving the records with a given value of attribute company is 3 page accesses. Assume a buffer with 50 frames, and consider the two queries shown below.

Query Q_1:	Query Q_2:

`select code, company from Flight`
`except all` *–– not removing duplicates*
select code, company from Ticket
	`select company, type from Flight`
	`except all` *–– not removing duplicates*
	`select company, type from Ticket`

where "except all" denotes bag difference. For both queries Q_1 and Q_2, tell (i) whether it is possible to process the query by using a block-nested loop algorithm, and (ii) whether it is possible to process the query by using an index-based algorithm. In all four cases, if the answer is positive, then describe the algorithm and tell which is its cost in terms of number of page accesses. If the answer is negative, then motivate the answer in detail.

Problem 5 (A.Y. 2021/22)

Describe in detail the notion of "star schema" in data warehousing and illustrate the difference between such a notion and the notion of "snowflake schema".

Problem 5 (A.Y. before 2021/22)

Consider the relations R(A,B,C) (with 1.500 pages and with one duplicate, in the average, for each tuple) and S(<u>D</u>,E) (with 7.000 pages), and the query `select distinct B,E from R, S where A <> 3`.
Assume that the buffer contains 260 frames and show the logical query plan associated to the query, as well as the logical query plan and the physical query plan you would choose for executing the query as efficiently as possible. Also, tell which is the cost (in terms of number of page accesses) of executing the query according to the chosen physical query plan.

Problem 1

In a schedule S on transactions $\{T_1, \ldots, T_n\}$ we say that two transactions T_i, T_j share the element X of the database if there exist actions $\alpha(X)$ in T_i and $\beta(X)$ in T_j such that α is either r_i or w_i and β is either r_j or w_j. Moreover, S is called "chary" if (i) no transaction in S uses the same element twice, and (ii) for every $T_i, T_j \in \{T_1, \ldots, T_n\}$, T_i and T_j share at most one element. Prove or disprove the following claims:

1.1 Every chary schedule is view-serializable.

1.2 Every chary schedule on two transactions is conflict-serializable.

1.3 Every chary schedule on two transactions is a 2PL schedule with exclusive and shared locks.

1) T_1: $w_1(x)$ $r_1(y)$

T_2: $w_2(z)$ $r_2(x)$

T_3: $w_3(z)$ $w_3(y)$

S: $w_1(x)$ $w_3(y)$ $r_2(x)$ $w_2(z)$ $r_1(y)$ $w_3(z)$

$T_3 \to T_1$

READ-FROM: $\langle r_2(x), w_1(x)\rangle$, $\langle r_1(y), w_3(y)\rangle$

$T_2 \to T_3$

FINAL-WRITE: $w_1(x)$, $w_3(y)$, $w_3(z)$

$T_1 \to T_2$

S IS VIEW-SER IF THERE EXISTS A SCHEDULE S' SERIAL SUCH THAT

READ-FROM (S) = READ-FROM (S')

FINAL-WRITE (S) = FINAL-WRITE (S')

2) IF TWO TRANSACTIONS SHARE ONLY ONE ELEMENT, MEANS THAT THERE IS ONLY ONE CONFLICT IN A DIRECTION.
ALSO BECAUSE WE CAN'T USE THE SAME VARIABLE TWICE.
THEREFORE THE PRECEDENCE GRAPH IS ACYCLIC → CONFLICT-SER

3) T_1: $w_1(x)$ $r_1(y)$

T_2: $w_2(x)$

S: $w_1(x)$ $w_2(x)$ $r_1(y)$

S': $xl_1(x)$ $w_1(x)$ $sl_1(y)$ $u_1(x)$ $xl_2(x)$ $w_2(x)$ $u_2(x)$ $r_1(y)$ $u_1(y)$

SINCE ONLY ONE VARIABLE IS SHARED, EVEN IF WE HAVE $w_1(x)$ $w_2(x)$, WE CAN ANTICIPATE ALL LOCKS IN A BEFORE $w_2(x)$, AND UNLOCK $u_1(x)$

✓

Problem 2

Let S be the schedule: $r_1(Z)\, w_3(Y)\, w_3(V)\, r_1(Y)\, r_2(V)\, w_2(Y)\, w_3(X)\, r_2(X)\, r_2(Z)\, r_3(Z)\, w_4(Z)\, w_4(X)\, w_2(X)$

2.1 Tell whether S is accepted by the 2PL scheduler with exclusive and shared locks. If the answer is yes, then specify the 2PL schedule obtained from S by adding suitable lock and unlock commands. If the answer is no, then explain the answer.

2.2 Tell whether S is view-serializable. If the answer is yes, then illustrate a serial schedule which is view-equivalent to S. If the answer is no, then explain the answer.

2.3 Answer all the following questions, motivating the answers: (*i*) Is S recoverable? (*ii*) Is S ACR? (*iii*) Is S strict?

1) $r_1(Z)\, w_3(Y)\, w_3(V)\, r_1(Y)\, r_2(V)\, w_2(Y)\, w_3(X)\, r_2(X)\, r_2(Z)\, r_3(Z)\, w_4(Z)\, w_4(X)\, w_2(X)$

S': $sl_1(z)$ $r_1(z)$ $xl_3(y)$ $w_3(y)$ $xl_3(v)$ $w_3(v)$ $xl_3(x)$ $sl_3(z)$ $u_3(v)$
$sl_1(y)$ $r_1(y)$ $u_3(v)$ $sl_2(v)$ $r_2(v)$ $u_1(y)$ $xl_2(y)$ $w_2(y)$ $w_3(x)$ $u_3(x)$
$sl_2(x)$ $r_2(x)$ $sl_2(z)$ $r_2(z)$ $r_3(z)$ $u_1(z)$...

IT'S NOT 2PL BECAUSE WE HAVE $r_2(x)$ .. $w_4(x)$... $w_2(x)$, SO WE CANNOT ANTICIPATE LOCKS

2) $r_1(Z)\, w_3(Y)\, w_3(V)\, r_1(Y)\, r_2(V)\, w_2(Y)\, w_3(X)\, r_2(X)\, r_2(Z)\, r_3(Z)\, w_4(Z)\, w_4(X)\, w_2(X)$

READ-FROM: $\langle r_1(y), w_3(y)\rangle$, $\langle r_2(v), w_3(v)\rangle$, $\langle r_2(x), w_3(x)\rangle$

$T_3 \rightarrow T_1$ $\quad T_3 \rightarrow T_2$

FINAL-WRITE: $w_2(x)$, $w_4(z)$, $w_2(y)$, $w_3(v)$

S IS VIEW-SER IF THERE EXISTS A SCHEDULE S' SERIAL SUCH THAT

READ-FROM (S) = READ-FROM (S')
FINAL-WRITE (S) = FINAL-WRITE (S')

T_2 T_4 CHANGE FINAL WRITE FOR X

T_4 T_2 CHANGE READ FROM FOR Z

NOT VIEW

3) $r_1(Z)\, w_3(Y)\, w_3(V)\, r_1(Y)\, r_2(V)\, w_2(Y)\, w_3(X)\, r_2(X)\, r_2(Z)\, r_3(Z)\, w_4(Z)\, w_4(X)\, w_2(X)$

READ-FROM: $\langle r_1(y), w_3(y)\rangle$, $\langle r_2(v), w_3(v)\rangle$, $\langle r_2(x), w_3(x)\rangle$

c_3 BEFORE c_1, c_3 BEFORE c_2 RECOVERABLE

NOT ACR BECAUSE SOME T READ BEFORE COMMITS

NOT ACR $\Rightarrow$ NOT STRICT

Problem 3

Let R($\underline{\text{A}}$,B,C), S($\underline{\text{A}}$,D,E), T(A,B,C) be three tables (where T is a bag) and let τ indicate the ternary operator such that $\tau(\text{R},\text{S},\text{T}) = \delta(\text{T} \cup_b \pi_{\text{A,B,C}}(\text{R} \bowtie \text{S}))$, where δ denotes duplicate elimination, $\cup_b$ denotes bag union and $\bowtie$ denotes natural join.

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

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

3.3 Tell what does it change if all the tables have A as key and are stored as sorted file with search key A.

1) ~~WE LOAD ALL RELATIONS IN THE BUFFER B(T) + B(R) + B(S). WE PERFORM THE NATURAL JOIN AND WE SAVED THE RESULTS IN A FRAME, WHICH WE USE TO UNION THE RESULTS WITH T. THEN WE PERFORM THE DUPLICATE ELIMINATION~~

2)

Problem 4

Consider the relations Flight(<u>code</u>,company,type) with 1.000 pages and 10.000 tuples, and Ticket(<u>number</u>,code,company,type) with 2.000 pages and an associated index on Ticket with search key ⟨company,type⟩, for which we know that the cost of retrieving the records with a given value of attribute company is 3 page accesses. Assume a buffer with 50 frames, and consider the two queries shown below.

Query Q_1:	Query Q_2:

`select code, company from Flight`
`except all` -- *not removing duplicates*
`select code, company from Ticket`
	`select company, type from Flight`
	`except all` -- *not removing duplicates*
	`select company, type from Ticket`

where "except all" denotes bag difference. For both queries Q_1 and Q_2, tell (i) whether it is possible to process the query by using a block-nested loop algorithm, and (ii) whether it is possible to process the query by using an index-based algorithm. In all four cases, if the answer is positive, then describe the algorithm and tell which is its cost in terms of number of page accesses. If the answer is negative, then motivate the answer in detail.

1) WE PUT IN 48 FRAMES FLIGHT, 1 FOR TICKET, 1 FOR OUTPUT

1000 / 48 = 21 RUNS COST = B(FLIGHT) + 21 · B(TICKET) = 43 000

Q_1 ✓ Q_2 ✗

2)