IN3020&4020 – Database Systems Exam, 2024 (with solutions)

30 May 2024, 9:00–13:00pm (Inspera)

Thresholds

A:	73–100%
B:	62–72%
C:	50–61%
D:	38–49%
E:	27–37%
F:	0–26%

Marking guidelines

For each part of each question:

- minor mistake: 6% for 8% questions, 7% for 10% questions
- major mistake: 3% for 8% questions, 4% for 10% questions
- understood the question: 2% for all questions

Intermediate results (-1%, -3%) are also possible

Note: several exam questions allow for many correct answers, in which cases the solutions below give only one possibility.

Question 1 of 12 (8%)

Consider the following relations:

Table1:

A	B	C
NULL	1	1
NULL	1	2
1	*NULL*	3
1	*NULL*	4

Table2:

D	E	F
NULL	1	1
NULL	1	2
1	*NULL*	3
1	*NULL*	4

Evaluate the following (bag) RA expression over these relations:

$$\gamma_{A,\mathsf{COUNT(B)}\to C1,\mathsf{COUNT_DISTINCT}(B)\to C2,\mathsf{COUNT}(C)\to C3,\mathsf{COUNT_DISTINCT}(C)\to C4}(\delta(\mathsf{Table1} \ltimes \mathsf{Table2}))$$

Note: you do not need to give any intermediate steps, only the final result is enough.

Solution

Expr:

A	$C1$	$C2$	$C3$	$C4$
NULL	2	1	2	2
1	0	0	2	2

Question 2 of 12 (8%)

Consider relation schema $\mathsf{R}[A]$. Write a relational algebra expression that outputs a relation with one attribute B of type string that contains one tuple $\langle$'Empty'$\rangle$ if the input R is empty and one tuple $\langle$'NonEmpty'$\rangle$ otherwise.

Note: in this and the following questions you may introduce and then use reasonable sequences of symbols for RA operations; for example, you can first say "I use X| for right semi-join below" and then use X| instead of $\rtimes$.

Solution

One possibility is

$$\pi_{\text{'Empty'}\to\mathsf{B}}(\sigma_{\mathsf{C}=0}(\gamma_{\mathsf{Count(D)}\to\mathsf{C}}(\pi^{b}_{1\to\mathsf{D}}(\mathsf{R})))) \cup \delta(\pi_{\text{'NonEmpty'}\to\mathsf{B}}(\sigma_{\mathsf{C}!=0}(\gamma_{\mathsf{Count(D)}\to\mathsf{C}}(\pi^{b}_{1\to\mathsf{D}}(\mathsf{R}))))))$$

Note that the inner projections in both arguments of the union are needed to correctly treat *NULL*; if a solution is correct except that it does not treat *NULL* properly, it is a minor-minor mistake (i.e., 7%).

Question 3 of 12 (8%)

Consider relations with schemas $\mathsf{Table1}[A, B]$ and $\mathsf{Table2}[C, D]$. Compile the following SQL query to a (logical) query plan:

```
SELECT A FROM Table1 WHERE
        NOT EXISTS (SELECT * FROM Table2 WHERE A = C);
```

Note: it is only necessary to write any query plan that does the same as the query; in particular, you do not need to write any intermediate steps of compilation and you do not need to follow any specific compilation algorithm.

Solution

For example,

$$\pi_{\mathsf{A}}(\mathsf{Table1}) \setminus \pi_{\mathsf{C}\to\mathsf{A}}(\mathsf{Table2})$$

Question 4 of 12 (8%)

Recall that, in our cost function, we estimated the number $Tups(\sigma_{Cond}(\mathsf{R}))$ of tuples, for a relation R with an attribute A and a condition $Cond$ of the form $A < c$ with a constant c, in two ways: $Tups(\mathsf{R})/3$ and $Tups(\mathsf{R})/2$. Explain (in at most a few sentences) the rationale under both options.

Solution

The rationale for the second option is that, without any further information, we can expect that this condition is satisfied by approximately a half of tuples in the table; in other words, our estimation is the expectation (in the probability theory sense) of the real value. The first option also takes into account an empirical observation that users that write such conditions often have some knowledge about the data they query and tend to write conditions more selective that the one selecting a half.

Question 5 of 12 (8%)

Consider a relation with only one attribute and assume that, when stored, the relation is ordered (by the only attribute). In this case, a key-value pair for an index of any type takes more space than the tuple it refers to. Can it still be beneficial to use an index of some type rather than binary search over the relation itself? (Here, 'beneficial' means that the number of block reads is smaller on average.) Explain your answer (try to be as concise and to the point as possible).

Solution

Yes, it is still possible. For example, when we use sparse primary index, the number of blocks of the table itself is

$$\lceil Tups * KeySize/BlockSize \rceil,$$

while the number of blocks in the index is

$$\lceil (\lceil Tups * KeySize/BlockSize \rceil * (KeySize + PointerSize))/BlockSize \rceil$$

where $Tups$, $KeySize$, $BlockSize$, and $PointerSize$ are the numbers of tuples in the relation, the size of the attribute, the size of a block, and the size of a pointer. So, when $Tups$ is big enough (and a block can accommodate several key-pointer pairs), the binary search over the index and accessing the data block is quicker than the search over the table itself, because diving over $BlockSize$ compensate both $PointerSize$ summand and one extra block access to the data).

Note 1: Similar argument may be given for sparse clustering index (of both variants).

Note 2: The formulas above are not necessary for a full mark, only the intuition given in word is enough.

Question 6 of 12 (10%)

Recall that we have a rule of thumb that pushing selection through natural join (if possible) usually gives a smaller logical plan cost. Give an example when it is not the case—that is, give a condition *Cond* and relations R1 and R2 such that

- the plan $\sigma_{Cond}(\mathsf{R1} \bowtie \mathsf{R2})$ is equivalent to the plan $\sigma_{Cond}(\mathsf{R1}) \bowtie \mathsf{R2}$;
- the cost (as introduced in the lectures) of the first plan is less than the cost of the second plan.

Solution

For example, we may have

R1:

A	B
1	1
2	1
3	1

and

R2:

A
1

such that $Size(A) = Size(B) = 1$, and $B = 1$ as *Cond*. Then we have

$$Cost(\sigma_{B=1}(\mathsf{R1} \bowtie \mathsf{R2})) = Cost(\mathsf{R1} \bowtie \mathsf{R2}) = (TupSize(\mathsf{R1} \bowtie \mathsf{R2})) * Tups(\mathsf{R1} \bowtie \mathsf{R2}) = 2 * (3 * 1/3) = 2,$$

but

$$Cost(\sigma_{B=1}(\mathsf{R1}) \bowtie \mathsf{R2}) = Cost(\sigma_{B=1}(\mathsf{R1})) = TupSize(\mathsf{R1}) * Tups(\sigma_{B=1}(\mathsf{R1})) = 2 * 3/1 = 6.$$

Question 7 of 12 (8%)

Consider the following four schedules:

$$S_1 : \mathtt{r}_1(X), \mathtt{w}_1(X), \mathtt{c}_1, \mathtt{r}_2(X), \mathtt{w}_2(Y), \mathtt{w}_2(X), \mathtt{c}_2$$
$$S_2 : \mathtt{r}_1(X), \mathtt{w}_1(X), \mathtt{r}_2(X), \mathtt{w}_2(Y), \mathtt{w}_3(Y), \mathtt{w}_2(X), \mathtt{c}_2, \mathtt{c}_1, \mathtt{c}_3$$
$$S_3 : \mathtt{r}_1(X), \mathtt{w}_1(X), \mathtt{c}_1, \mathtt{r}_2(X), \mathtt{w}_2(Y), \mathtt{w}_2(X), \mathtt{c}_2, \mathtt{w}_3(Y), \mathtt{c}_3$$
$$S_4 : \mathtt{r}_1(X), \mathtt{r}_2(X), \mathtt{w}_1(X), \mathtt{c}_1, \mathtt{w}_3(Y), \mathtt{w}_2(Y), \mathtt{c}_3, \mathtt{w}_2(X), \mathtt{c}_2$$

Which pairs of these schedules are conflict equivalent? Explain your answer (try to be as concise and concrete as possible).

Solution

Schedules S_2 and S_3 are the only pair of conflict equivalent schedules. They are conflict equivalent since all conflicting pairs come in the same order.

Schedule S_1 is not a schedule of the same transactions as the other three, and is therefore not conflict equivalent to any of these. For schedules S_2 and S_4 we have, for example, the conflict $w_1(X)$ and $r_2(X)$ that have differing order. The same holds for the pair S_3 and S_4.

Question 8 of 12 (8%)

Consider the following two transactions:

$$T_1 : \mathtt{r}(X), \mathtt{w}(Y), \mathtt{r}(Z), \mathtt{w}(Z)$$
$$T_2 : \mathtt{r}(X), \mathtt{w}(Z)$$

and the following partial schedule (with T_2 requesting a read-lock on X as the last operation):

$$\mathtt{b}_1, \mathtt{rl}_1(X), \mathtt{r}_1(X), \mathtt{wl}_1(Y), \mathtt{b}_2, \mathtt{rl}_2(X)$$

Assume that we are using the two-phase locking protocol with the no-waiting deadlock prevention protocol. Write down how the rest of the schedule (i.e., so that the result is a complete schedule) can be.

Solution

One possibility is

$$\ldots, \mathtt{r}_2(X),$$
$$\mathtt{w}_1(Y), \mathtt{wl}_1(Z), \mathtt{r}_1(Z), \mathtt{w}_1(Z), \mathtt{u}_1(X), \mathtt{u}_1(Y), \mathtt{u}_1(Z), \mathtt{c}_1,$$
$$\mathtt{wl}_2(Z), \mathtt{w}_2(Z), \mathtt{u}_2(X), \mathtt{u}_2(Z), \mathtt{c}_2$$

However, a solution such as

$$\ldots, \mathtt{a}_2,$$
$$\mathtt{w}_1(Y), \mathtt{wl}_1(Z), \mathtt{r}_1(Z), \mathtt{w}_1(Z), \mathtt{u}_1(X), \mathtt{u}_1(Y), \mathtt{u}_1(Z), \mathtt{c}_1,$$
$$\mathtt{b}_2, \mathtt{rl}_2(X), \mathtt{r}_2(X), \mathtt{wl}_2(Z), \mathtt{w}_2(Z), \mathtt{u}_2(X), \mathtt{u}_2(Z), \mathtt{c}_2$$

would also be accepted due to ambiguity of lecture slides.

Question 9 of 12 (8%)

Assume that we have a database D with the following levels of item granularity:

1. Database
2. Table
3. Record

In other words, let the root of the granularity tree be the entire database, then let this root have one child per table in the database, and finally let each table have one child per record.

Furthermore, assume that database D has tables Table1 and Table2; that Table1 has records R1 and R2; and that Table2 has records P1, P2, and P3.

Make a schedule for the following transactions that uses the two-phase locking protocol with multiple granularity locking with the granularity described above:

$$T_1 : \mathtt{r}(\mathsf{R1}), \mathtt{w}(\mathsf{P2})$$
$$T_2 : \mathtt{r}(\mathsf{R2}), \mathtt{w}(\mathsf{R2})$$

Solution

One possibility is

$\mathtt{b}_1, \mathtt{is}_1(D), \mathtt{is}_1(\mathsf{Table1}), \mathtt{rl}_1(\mathsf{R1}), \mathtt{r}_1(\mathsf{R1}), \mathtt{ix}_1(D), \mathtt{ix}_1(\mathsf{Table2}), \mathtt{wl}_1(\mathsf{P2}), \mathtt{w}_1(\mathsf{P2}),$
$\mathtt{u}_1(\mathsf{R1}), \mathtt{u}_1(\mathsf{Table1}), \mathtt{u}_1(\mathsf{P2}), \mathtt{u}_1(\mathsf{Table2}), \mathtt{u}_1(D), \mathtt{c}_1,$
$\mathtt{b}_2, \mathtt{ix}_2(D), \mathtt{ix}_2(\mathsf{Table2}), \mathtt{wl}_2(\mathsf{R2}), \mathtt{r}_2(\mathsf{R2}), \mathtt{w}_2(\mathsf{R2}), \mathtt{u}_2(\mathsf{R2}), \mathtt{u}_2(\mathsf{Table1}), \mathtt{u}_2(D), \mathtt{c}_2$

Question 10 of 12 (8%)

Consider the following transactions:

$$T_1 : \mathtt{r}(X), \mathtt{w}(X)$$
$$T_2 : \mathtt{r}(X), \mathtt{w}(Y), \mathtt{w}(Z)$$
$$T_3 : \mathtt{r}(Y), \mathtt{w}(Z)$$

Write down a complete schedule for these transactions that is recoverable but not cascadeless, or explain why this is not possible.

Solution

One possibility is

$$\mathtt{r}_1(X), \mathtt{w}_1(X), \mathtt{r}_2(X), \mathtt{w}_2(Y), \mathtt{w}_2(Z), \mathtt{r}_3(Y), \mathtt{w}_3(Z), \mathtt{c}_1, \mathtt{c}_2, \mathtt{c}_3$$

Question 11 of 12 (8%)

Assume that we are using the immediate update protocol. Consider a database where item X has value 7 and item Y has value 3. Write down what the log will contain after the following (partial) schedule has been executed starting on this database:

$$\mathtt{b}_1, \mathtt{r}_1(X), \mathtt{eval}_1(X{:=}X * 2),$$
$$\mathtt{b}_2, \mathtt{r}_2(Y), \mathtt{eval}_2(X{:=}Y + 1), \mathtt{w}_2(X), \mathtt{w}_1(X), \mathtt{c}_1, \mathtt{b}_3, \mathtt{r}_3(Y)$$

In this schedule, $\mathtt{eval}_1$ and $\mathtt{eval}_2$ are (local) update operations that are evaluated within transaction T_1 and transaction T_2, respectively.

Solution

Since we use the immediate update protocol, we need both the UNDO and REDO entries in the log—that is, both the old value and the new value for each write. Therefore, the log may be as follows.

Lsn	Transaction	Operation	Item	Old value	New value
1	1	begin			
2	2	begin			
3	2	write	X	7	4
4	1	write	X	4	14
5	1	commit			
6	3	begin			

Question 12 of 12 (10%)

Consider the following statements in natural language:

- 'UiO is an organisation and a university, and it was established in 1811.'
- 'Carl is a person and he worked for UiO from 2017 to 2020.'

(a) Represent these statements in RDF. You can invent your own namespaces and IRIs (and you can use the rdf-prefix without defining it in your answer).

(b) Represent these statements in Neo4J (i.e., make CREATE-statements that adds the representations of the statements to a database).

Solution

(a) In RDF the statements can be encoded as the following:

```
@prefix : <http://example.org> .

:uio rdf:type :Organisation, :University ;
     :established 1811 .

:carl rdf:type :Person ;
      :workedFor [ :organisation :uio ;
                   :from 2017 ;
                   :to 2020 ].
```

(b) In Neo4J's language Cypher the statements can be encoded as the following:

```
CREATE (uio:Organisation:University {established:1811})
CREATE (carl:Person)-[:WorkedFor {from:2017, to:2020}]->(uio)
```