CompSci 516: Database Systems

Midterm

Fall 2018

This booklet has 12 pages (including the cover page and 2 blank pages at the end).
You can use the reverse sides as additional space for writing your answers.

INSTRUCTIONS

1. No external help (books, notes, laptops, tablets, phones, etc.) or collaboration is allowed.
2. You have **75 minutes** to answer questions that add up to **100 points**. i.e. you have about 7.5 mins for 10 points, and about 15 mins for 20 points.
3. Some questions may need more time, some questions less. Please budget your time accordingly.
4. If you cannot solve a problem fully, write partial solution for partial credit. **Even if explanations are not sought in a question, if you cannot fully solve a problem, writing down your thought process may lead to some partial credit.**
5. Do not spend too much time on a problem that you find difficult to solve - move on to other problems.
6. **The problems are organized in no particular order, easier problems may appear later.**

****Write Your Name Here** (1 bonus point):**

First Name:

Last Name:

All the best!

DO NOT WRITE BELOW THIS LINE

Problem 1	/ 20	Problem 2	/ 20	Problem 3	/ 10
Problem 4	/ 18	Problem 5	/ 32	bonus point	/ 1
Total		/ 100			

Q1. (20 = 10 + 10 pts) RA, RC

Consider the following tables storing information about an international music competition with different events like guitar, violin, piano etc. (keys are underlined).

- `E(eid, event)` – stores event information.
- `A(aid, aname, country)` – stores artist information – name and country of origin.
- `P(aid, eid, rank)`
  Also `aid` and `eid` are foreign keys referring to `A` and `E` respectively. One artist can participate in multiple events.

Q1a: (10 pts) RC

Write an RC expression (TRC or first order logic form) to find names of all the artists (`aname`) who got rank = 1 in all events they participated .

(Hence if a rank-1 holder participated in only one event, s/he is going to appear in the solution.)

$E(\underline{eid}, event)$	$A(\underline{aid}, aname, country)$	$P(\underline{aid, eid}, rank)$

Q1b: (10 pts) RA

(**same query in RA**) Write an RA expression (or logical query plan tree) to find names of all the artists (`aname`) who got rank = 1 in all events they participated.

$E(\underline{eid}, event)$	$A(\underline{aid}, aname, country)$	$P(\underline{aid, eid}, rank)$

Q2: (20 = 10 + 10 pts) SQL

Q2a: (10 pts) SQL for query in Q1

(**same query from Q1 in SQL**) Write a SQL query to find names of all the artists (`aname`) who got rank = 1 in all events they participated.

$E(\underline{eid}, event)$	$A(\underline{aid}, aname, country)$	$P(\underline{aid, eid}, rank)$

Q2b: (10 pts) More SQL

Write an SQL query **without using nested sub-queries or without using a WITH clause** that outputs the names of the events `event` where all participating artists (not necessarily rank = 1 holders) are from the same country.

Note: there can be ****exactly one**** `SELECT` clause in your solution.

$E(\underline{eid}, event)$	$A(\underline{aid}, aname, country)$	$P(\underline{aid, eid}, rank)$

Q3. (10 = 1 * 10 pts) Indexing

Given the same schema as in Q1 (repeated above) and the following query, specify (write yes or no – NO explanations are needed) whether each of the following choices of indexes can speed up the query (for some data distribution), **assuming it is the only index that is available.**

```
SELECT A.aname
FROM A, P, E
WHERE A.aid = P.aid AND P.eid = E.eid
       AND event = 'guitar'
       AND rank > 3
```

1. B+-tree index on `P(rank)`. **Yes/No:**
2. Hash index on `P(rank)`. **Yes/No:**
3. B+-tree index on `A(aid, aname)` **Yes/No:**
4. B+-tree index on `A(aname, aid)` **Yes/No:**
5. Hash index on `E(eid)` **Yes/No:**
6. Hash index on `E(eid, event)` **Yes/No:**
7. Hash index on `E(event, eid)` **Yes/No:**
8. Hash index on `A(aid, country)` **Yes/No:**
9. Hash index on `A(country, aid)` **Yes/No:**
10. B+-tree index on `P(aid, rank)` **Yes/No:**

Q4. (18 pts) Query Evaluation

Consider the following two relations from Q1 with the stated assumptions:

- `A(aid, aname, country)`: no. of tuples $T_A = 20,000$; no. of tuples/page $n_A = 200$; no. of pages $N_A = 100$.
- `P(aid, eid, rank)`: no. of tuples $T_P = 5000$; no. of tuples/page $n_P = 100$; no. of pages $N_P = 50$.
- Assume that the no. of buffer pages available is $B = 12$.
- Assume on average 20 artists participate in each event.
- Assume all index pages are in memory.
- Ignore page boundaries.

Consider the following query

```
SELECT *
FROM A, P
WHERE A.aid = P.aid
```

Consider three alternatives for the join:

- **option 1:** Block-oriented nested-loop join with `A` as outer.
- **option 2:** Sort-merge join.
- **option 3:** Index nested loop join with `P` as outer.

For the three scenarios below, for all three options, write the cost (in terms of I/O, assuming initially all relations are on disk, ignore final write). If an option does not apply for a scenario, **write "N/A"**.

No explanations are necessary. But you can show your calculations in the boxes or on reverse side of this page, which we may consider for partial credit.

Scenario	cost: option 1	cost: option 2	cost: option 3
(1) Clustered hash index on `A(aid)`			
(2) Both relations are sorted on `aid`			
(3) Clustered B+-index on `A(aid)` and `P` is sorted on aid			

Q5. (32 pts) Short Q/A

Q5a. (2 * 14 = 28 pts) Are the following statements True or False?

<u>**No explanations are needed.**</u>

1. Given two relations $R(A,B)$ and $S(C,D)$ without any nulls, the following equality holds. (**True/False**):

$$R - \Pi_{AB}[R \bowtie_{B=C} S] = \Pi_{AB}[R \bowtie_{B \neq C} S]$$

2. A relation R (set semantic) has at least one superkey (**True/False**):

3. Given relations R and S with 100 and 10 pages on disk respectively, the cost of best possible join algorithm can be as low as 100 (**True/False**):

4. Suppose relations R and S have the same schema and p is a predicate over this schema. The relational algebra expressions $\sigma_p(R-S)$ and $\sigma_p R - \sigma_p S$ are equivalent (i.e., their answers always agree with each other regardless of the predicate p and the contents of R and S). (**True/False**):

5. Suppose relations R and S have the same schema and L is a subset of attributes. The relational algebra expressions $\pi_L(R-S)$ and $\pi_L R - \pi_L S$ are equivalent (**True/False**):

6. Suppose relations R, S and T have the same schema. The relational algebra expressions $(R \bowtie T) - (S \bowtie T)$ and $(R-S) \bowtie T$ are equivalent (**True/False**):

7. Consider relation $R(A,B,C,D)$. Suppose we know $\{A,B\}$ is a key of R (and R may or may not have other keys). Then the FD $A \rightarrow B$ cannot hold in R. (**True/False**):

8. Consider relation $R(A,B,C,D)$. Suppose we know $\{A,B\}$ is a key of R (and R may or may not have other keys). Then the FD $C \rightarrow AB$ cannot hold in R. (**True/False**):

9. Consider the database schema below:

```
create table R(A integer not null PRIMARY KEY, B integer not null);
create table S(C integer not null PRIMARY KEY,
A integer not null REFERENCES R(A));
```

Regardless of the database instance, the number of distinct $S.A$ values must be no greater than the number of distinct $R.A$ values.

(**True/False**):

10. In the above question, regardless of the database instance, the number of distinct $S.C$ values must be no greater than the number of distinct $R.A$ values.

(**True/False**):

11. The following two SQL queries are equivalent over a schema $Users(\underline{uid}, pop, date)$:
(i) `SELECT * FROM Users WHERE pop > 0.5 AND pop < 0.9;`
(ii) `(SELECT * FROM Users WHERE pop > 0.5) INTERSECT ALL (SELECT * FROM Users WHERE pop < 0.9);`

(Recall that INTERSECT ALL or UNION ALL preserves duplicates.)

(**True/False**):

12. The following two SQL queries are equivalent over a schema $Users(\underline{uid}, pop, age)$:
:
(i) `SELECT * FROM Users WHERE age < 6 OR pop > 0.9;`
(ii) `(SELECT * FROM Users WHERE age < 6) UNION ALL (SELECT * FROM Users WHERE pop > 0.9);`

(**True/False**):

13. Consider the natural join between two tables $R(A,B)$ and $S(B,C)$, where R is sorted by A and S is sorted by B . The output of a sort merge join between R and S on B will be naturally sorted by A.

    (**True/False**):

14. Consider the natural join between two tables $R(A,B)$ and $S(B,C)$, where R is sorted by A and S is sorted by B . Further, there is an index on $S.B$.The output of a index-nested loop join between R and S on B will be naturally sorted by A.

    (**True/False**):

Q5b. (4 pts)

Consider the following relation R:

A	**B**
3	null
10	null

Consider the query

```
SELECT A
FROM R
WHERE B >= 7 OR B < 7
```

What is the output of the above query?

Briefly explain your answer in 1-2 sentences.

Blank additional page to be used for rough calculations. DO NOT write solutions here.

Blank additional page to be used for rough calculations. DO NOT write solutions here.