Exam INFO-H-417 Database systems architecture

January 2022

Mahmoud SAKR

Name:

ULB Student ID:

Q1 (25)				Q2 (25)			Q3 (15)		Q4 (25)	

Important notes

- Please make sure to sign your name with the monitor twice: when you enter the exam room, and before you leave it. Without these two signatures, you will be considered absent.
- Please make sure to write your name and student ID on this page. On every other paper, including the draft, make sure to write your student ID. A paper without your ID number is considered a cheat source. Your ID number on every paper also guarantees identifying your answers in case some staples break.
- You are allotted a maximum of **3 hours** to complete this exam.
- The exam is closed book. You are hence not allowed to use any notes/books/PC/etc. during the exam.
- Draft papers are provided by the monitor upon request. **Draft paper will not be corrected!**
- You should answer each question in the foreseen space after the question. Should this space prove to be insufficient you are allowed to use the back of the page as well.
- Mobile phones cannot be used during the whole exam duration.

STUDENT ID:

Question 1. (25 pts)

Consider the following relations for a database that keeps track of student enrollement in courses. Also consider that the relations are bags that allow duplicates:

- student<id, name, age, deptid> storing 100000 tuples
- department<id, name> storing 600 tuples
- course<id, name, semester> strong 3000 tuples
- enrollment<student_id, course_id, year, grade> storing 4000000 tuples

(a) For the following query:

List id and name of all students who didn't take courses in the first semester of 2021

give two RA expressions for this query (for example, by applying equivalence rules), one of which is optimized in terms of performance, and briefly discuss the estimated performance of the two expressions

(b) Write an RA expression to express this query:

List all department id and name of the students who attend the Database course
Make sure not to show duplicate results

(c) Normalize/flatten the following query and give its equivalent RA expression

```
SELECT course.name
FROM course
WHERE course.id in (
      SELECT enrollment.course_id
      FROM student, enrollment
      WHERE student.id = enrollment.student_id AND
      student.name LIKE '%son');
```

STUDENT ID:

(d) Suppose relation R<A,B,C, D> has the tuples
(1,2,3,4)
(1,2,3,5)
(3,2,1,0)

Using **bag** projection and theta-join, how many tuples appear in the result of:

$$\pi_{A,B}(R) \bowtie_{R.B<S.B} \rho_S(\pi_{B,C}(R))$$

Using **set** projection and theta-join, how many tuples appear in the result?

Question 2. (25 pts)

(a) Discuss by giving atleast two reasons why left deep join trees are used, for example in System R, as a heuristic for solving the join ordering optimization problem

(b) Which of the following will be stored in the system catalogue of PostgreSQL. Choose zero to all, and give one line discussion on every point explain why you chose or didn't choose it:

1. Materialized views

STUDENT ID:

2. Attribute statistics for the optimizer

3. The optimization rules

4. User tables, such as course, student, enrollment in question 1

5. The SQL user functions

6. The B+tree indexes

7. User defined type

8. The database schema

(c) State wether each of the following points are true about images in System R. Give one line discussion on every point:

1. One may define multiple images on the same relation

2. One may define multiple images on the same attribute

3. Image may be declared as unique

4. Defining a clustered image may lead to changing the ordering of the stored tuples in the physical medium

5. One may define multiple clustered images on the same relation

6. One may define multiple unique images on the same relation

7. If available, it is always more efficient to access a relation via an image

Question 3. (15 pts)

(a) Recall that in a B+tree:

- one tree node is stored in one disk block. In this excercise consider that the block size is 4096 B
- one tree node stores p pointers and p-1 keys

Consider a B+tree index defined over an int16 attribute. Consider also that a tree/disk pointer is 8 bytes. If the relation that contains the indexed attribute has four million tuples, how many disk accessess would be required to answer a query *key= const*, where *const* is a literal value?

STUDENT ID:

(b) Discuss the truth of each of the following statements.

1. The leaf level of a B+tree index is always a dense index

2. The root level of a B+tree index is always a dense index

3. Inserting duplicate keys can result in a non-balanced B+tree

4. A B+tree index may require a bigger storage size than its relation

5. A B+tree index on a string/text attribute cannot be used to answer queries in the form of *key LIKE pattern*

Question 4. (25 pts)

(a) Give a short answer on the following.

1. What is an operator class in PostgreSQL? Give its importance.
2. How is it possible for PostgreSQL to process user types, without knowing them in advance?
3. How is a materialized view different from a table? How does a materialized view affect query performance.
4. How does tuple insertion happen over a distirbuted table using hash-distribution?
5. What is the importance of replicated tables in distributed databases? Why not distribute all tables?

STUDENT ID: ___________

STUDENT ID:

(b) Consider the following relations in a MobilityDB database that keeps track of the STIB bus, tram, and metro trips in Brussels:

- trip<tripId, vehicleId, lineId, variant, destination, gpsTrack> storing 10^8 tuples
- line<id, name_fr, name_nl, variant, startStop, terminalStop> storing 200 tuples

The underlined attributes are primary keys. Consider that the gpsTrack attribute is of type tgeompoint, which is the MobilityDB type for representing a spatiotemporal trajectory. Consider also that MobilityDB has among others two functions

- length(tgeompoint)→ float : the total length of a trajectory in meters
- dwithin(tgeompoint, tgeompoint, float) → boolean : reads as distance within. It returns true if the two spatiotemporal trajectories in the first two arguments ever came closer than the distance value given in the third parameter in meters.

Assume that we are often interested in doing the following query:

```
SELECT t.lineId, t.variant, SUM(length(t.gpsTrack))
FROM trip AS t, line AS l
WHERE  t.lineId=l.id AND t.variant=l.variant
GROUP BY t.lineId, variant
```

In a distributed DBMS that supports only hash distribution:

1. (a) Desribe a strategy to distribute this schema to quickly answer the query above
2. (b) Using your answer of (a), illustrate an optimized strategy to distribute the query above (best in multi-relational algebra)
3. (c) Illustrate a strategy to distribute the following query, which finds the pairs of trips that crossed or travelled close to each other, within 50 meters. You may use the answer of (a), or suggest another distribution strategy.

```
SELECT t1.tripId, t2.tripId
FROM trip AS t1, trip AS t2
WHERE
      t1.tripId < t2.tripId AND
      dwithin(t1.gpsTrack, t2.gpsTrack, 50)
```

STUDENT ID: __________