Deadline: 27-NOV-2020 (Friday), 3:00pm (late submission will NOT be accepted)

Points to note:

- Different books may have slightly different descriptions of concepts, notations, and terminologies. To ensure fair assessment and uniformity in marking, you **must** follow the convention used in the lecture slides or our textbook (Database System Concepts). Other conventions will not be accepted.
- Students are expected to generalize the concepts they have learnt during the lecture in order to finish the assignment.
- You must show the steps clearly. The marker will not give you marks if s/he cannot understand your work.
- This is an individual assignment. You must work on your own. Check http://www6.cityu.edu.hk/ah/plagiarism.htm for "The Problem of Plagiarism".
- Submit the file to Canvas on or before the deadline**.**
- The file type must be either .docx file or .pdf file.
- Use your student ID(s) to name the file, such as 5xxxxxxx.docx or 5xxxxxxx.pdf.

1. Transactions [50%]

A. [30%]

There are two bank accounts A and B. Consider the following two transactions. The first transaction transfers 10% of the money in account A into account B. The second transaction accesses the balance of account A and account B and then calculates the sum. Therefore, the two transactions can be written as:

$$T_1: read(A); A := A * 0.9; write(A); read(B); B := B + A * 0.1; write(B);$$

$$T_2: read(A); read(B); write(sum);$$

Assume that the two transactions preserve database consistency in isolation.

(a) [12%]

(i) Demonstrate that no matter how much the balance in the two bank accounts initially, both serial schedules, i.e., (T_1, T_2) and (T_2, T_1) are equivalent by showing the final balance in both accounts and the calculated sum in T_2.

(ii) Give a concurrent schedule that is equivalent to a serial schedule, i.e., giving the same final balance in both accounts and the calculated sum in T_2.

(iii) Give a concurrent schedule that results in an inconsistent state, and show the final balance in both accounts and the calculated sum in T_2.

(b) [12%]
The two transactions can be simplified as follows by showing read and write instructions only.

$T_1: r_1(A); w_1(A); r_1(B); w_1(B);$
$T_2: r_2(A); r_2(B); w_2(sum);$

(i) Give all possible schedules (including serial schedule) of the two transactions that are ***conflict-equivalent*** to the serial schedule (T_1, T_2).
(ii) Give all possible schedules (including serial schedule) of the two transactions that are ***conflict-equivalent*** to the serial schedule (T_2, T_1).

(c) [6%]
(i) If a schedule of the two transactions is ***conflict-equivalent*** to the serial schedule (T_1, T_2), is it also necessarily ***equivalent*** (giving the same final balance in both accounts and the calculated sum in T_2) to the serial schedule (T_1, T_2)? If not, give an example.
(ii) If a schedule of the two transactions is ***equivalent*** (giving the same final balance in both accounts and the calculated sum in T_2) to the serial schedule (T_1, T_2), is it also necessarily ***conflict-equivalent*** to the serial schedule (T_1, T_2)? If not, give an example.

B. [20%]
Consider the following two transactions.

$T_1: w_1(A); w_1(B); r_1(C); c_1;$
$T_2: w_2(A); r_2(B); w_2(S); c_2;$

where *w*, *r*, and *c* denote a write, read, and commit instruction, respectively.

For each of the following combinations, state whether there exist any possible schedules of T_1 *and* T_2. If yes, list (at most) 3 schedules for each combination. All the listed schedules among the combinations, if exist, must be different.

Combination	recoverable (Y/N)	cascadeless (Y/N)	conflict serializable (Y/N)
(a)	N	N	N
(b)	N	N	Y
(c)	N	Y	N
(d)	N	Y	Y
(e)	Y	N	N
(f)	Y	N	Y
(g)	Y	Y	N
(h)	Y	Y	Y

2. Distributed Query Processing [50%]

Consider the following relations of a database of a university.

STU(SNO, SNAME, DEP)
VOL(SNO, ANO, DUTY)
ACT(ANO, ANAME, DEP)

For each student, there is a student number (SNO), name(SNAME), and department where the student is from (DEP). Some students are involved in voluntary work for one or more academic activity (activities) with specified duty (DUTY). For each academic activity, there is an activity number (ANO), an activity name (ANAME), and the department that organized the activity. Students can volunteer at the activities organized by the departments they come from and other departments.

The three relations STU, VOL, and ACT are stored at Site 1, Site 2, and Site 3, respectively and the following information is available.

- number of records in STU = 20,000
- number of records in VOL = 10,000
- number of records in ACT = 1,200
- percentage of students participating in voluntary work: 40%
- size of SNO field = 8 bytes
- size of SNAME field = 30 bytes
- size of DEP field = 30 bytes
- size of ANO field = 6 bytes
- size of DUTY field = 40 bytes
- size of ANAME field = 20 bytes

The following SQL query submitted at Site 3 (and therefore the query result is needed at Site 3) retrieves the names of all the students involved in voluntary work for activities organized by the departments they come from, the names of the activities, and their duties.

```
SELECT    STU.SNAME, ACT.ANAME, VOL.DUTY
FROM      STU, VOL, ACT
WHERE     STU.SNO=VOL.SNO     AND
          VOL.ANO=ACT.ANO     AND
          STU.DEP=ACT.DEP
```

(a) Suppose the optimization criterion is to minimize the data transmission cost in at most **four** data transfers between different sites. Based on the available information, for each of the following estimated number of output tuples, suggest a strategy for executing this query and find the data transmission cost in number of bytes transferred in each step and compute the total data transmission cost at the end.
(i) 7,000
(ii) 6,000
(iii) 2,000

(b) If there is no limitation on number of data transfers, is it possible to further reduce the data transmission cost. If yes, revise your strategies and re-compute the data transmission costs in part (a).