Deadline: 23-OCT-2020 (Friday), 3:00pm (late submission will NOT be accepted)

Points to note:

- Different books may have slightly different descriptions of concepts, estimation, algorithms 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 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.

Query Processing and Optimization

Searching and Protecting Morpheus

You probably know the story of the Matrix Trilogy, the hero Neo and his tutor Morpheus. But do you know that once Morpheus was nobody but a wretch who lost in the dream too? You, a fighter of Zion, receive the guidance from the prophet and decide to free Morpheus from the machine. The prophet offered you an ambiguous description to locate the probable targets: Morpheus' name is not known now. He is a man aged between 20 and 30, who had turned to doctors of dermatology or endocrinology for hair-losing problem. He got an E2 type medical care plan in his insurance. Before launching the rescue, you can hack into Matrix's database once to search for the target, but no optimizer is provided and the memory you get is limited.

Consider the following database schema.

Citizen (citizen-id, age, gender, …)
Hospital-record (timestamp, c-id, i-id, department, …)
Insurance (insurance-id, p-no, …)
Medical-care-plan (plan-no, type, …)

The attributes *c-id* and *i-id* in *Hospital-record* are foreign keys referencing *citizen-id* in *Citizen* and *insurance-id* in *Insurance*, respectively. The attribute *p-no* in *Insurance* is a foreign key referencing *plan-no* in *Medical-care-plan*.

Suppose ONLY the following information and statistics are available and assume that all distributions are uniform and independent of each other.

Relation	*Citizen*	*Hospital-record*	*Insurance*	*Medical-care-plan*
Number of records	6,000,000	18,000,000	9,000,000	600
Blocking factor	75	100	75	60

- *Citizen, Hospital-record, Insurance* and *Medical-care-plan* are sequential files stored in sorted order of their primary key.

- Records do not span across blocks.
- Proportion of dermatology department records: 1/59
- Proportion of endocrinology department records: 1/60

- Range of *age*: from 0 to 100
- Medical care plan type: A1, A2, B1, B2, C1, C2, D1, D2, E1, E2

Consider the following SQL query that retrieves all information about male citizens aged between 20 and 30 and has a E2 medical care plan and hospital record(s) on either dermatology or endocrinology.

```
select *
from Citizen c, Insurance i, Hospital-record h, Medical-care-plan m
where c.citizen-id = h.c-id  and i.insurance-id = h.i-id and m.plan-no = i.p-no
   and c.age <= 30 and c.age >= 20 and c.gender = 'male' and m.type = 'E2'
   and (h.department = 'dermatology' or h.department = 'endocrinology');
```

(a) Write an unoptimized relational-algebra expression of the above SQL query.

(b) Estimate the number of records the query returns. Explain your estimation.

(c) Given the following configuration parameters:

- Number of memory blocks available: 60
- Average block transfer time (t_T): 0.1 ms
- Average disk seek time (t_S): 4 ms

If only **left-deep join orders** and **materialization** are considered, suggest the best evaluation plan for the query. Particularly, you need to do the followings.

(i) Transform the unoptimized relational-algebra expression in Part (a) it into an equivalent optimized relational-algebra expression for producing your suggested evaluation plan. Show all the steps and state clearly the rule number of the equivalence rule you used in each step.

(ii) Draw the fully annotated evaluation plan (including exactly what algorithm is used for each operation).

(iii) Indicate in the evaluation plan the memory allocation (number of memory blocks allocated to each input and output (b_b), but the total cannot exceed the available memory at one time).

(iv) Find the **cost of each operation** and the **total cost of the whole plan.** You need to show clearly the **number of block transfers** and the **number of disk seeks** that are used to compute the total cost in the calculations.

(v) State clearly any justifiable assumptions you have made in your estimation.

(d) If **any** join orders can be considered, is it possible to have a better evaluation plan for the query? If yes, you need to follow the steps in Part (c) to explain your answer.

(e) If **pipelining** is allowed between any **one** pair of join operations, is it possible to further improve the evaluation plan for the query in Part (d)? If yes, you need to follow the steps in Part (c) to explain your answer.