THE HONG KONG POLYTECHNIC UNIVERSITY

DEPARTMENT OF COMPUTING

EXAMINATION

Subject : COMP5112 Data Structures and Database Systems

Session : 2023 / 2024 Semester II

Date : 9 May 2024 Time : 19:00 – 21:00

Time Allowed: 2 Hours

Subject Examiner(s): Dr HUA Wency

This question paper has a total of 5 pages.
(Some pages may be intentionally omitted.)

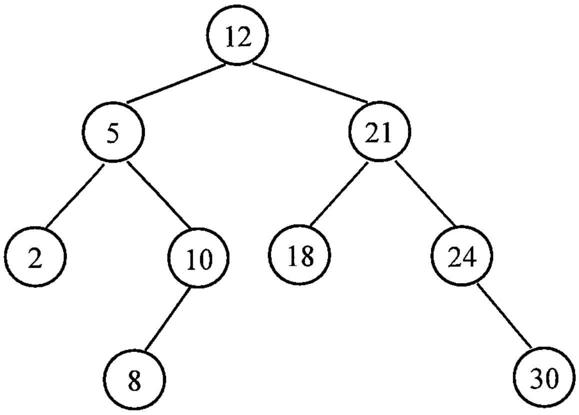
Instructions to Candidates:

- This is a closed-book exam – no written materials permitted.
- You are allowed to use a calculator.
- Answer ALL questions in the answer book provided.
- All the answers must be written in English.
- Write down your name and student ID in the answer book provided.

Do not turn over the page until you are told to do so!

Question 2. Data Structures **[15 marks]**

[1] Consider the following AVL tree. Show the running steps of inserting node 27 and rebalancing the AVL tree after the insertion.



[2] Write an algorithm (pseudocode) to merge two unsorted arrays *A* and *B* into a new array *C* that is sorted in descending order by using max-heaps. For example, given arrays *A* = | 4 | 9 | 1 | 7 | and *B* = | 3 | 8 | 5 |, the new array C = | 9 | 8 | 7 | 5 | 4 | 3 | 1 |.

Question 3. Sorting Algorithms **[10 marks]**

Suppose we need to sort array *A* with 6 numbers in ascending order by the insertion-sort algorithm.

[1] Suggest a best-case input for the array *A*. Calculate the number of comparisons needed.

[2] Suggest a worst-case input for the array *A*. Calculate the number of comparisons needed.

[3] Show the running steps of the insertion-sort algorithm (i.e., the updated array after each swap) for the following array *A*.

	0	1	2	3	4	5
A	85	91	72	67	80	88

Question 4. SQL and Relational Algebra **[10 marks]**

Consider the following relations in a *Company* database:

```
Employee (ENO, EName, Email, Salary)
Department (DNO, DName, Budget, City)
Job (JNO, JName)
Work (ENO, DNO, JNO, Duration)
```

The *ENO*, *DNO*, and *JNO* attributes in the *Work* relation are foreign keys referencing *Employee*, *Department*, and *Job*, respectively.

[1] Write each of the following queries in SQL:

(a) Get the maximum and minimum salary for each department.

(b) Get the names of employees who have worked in more than one departments.

(c) Get the names of departments that have no sufficient budget to pay for employee salary.

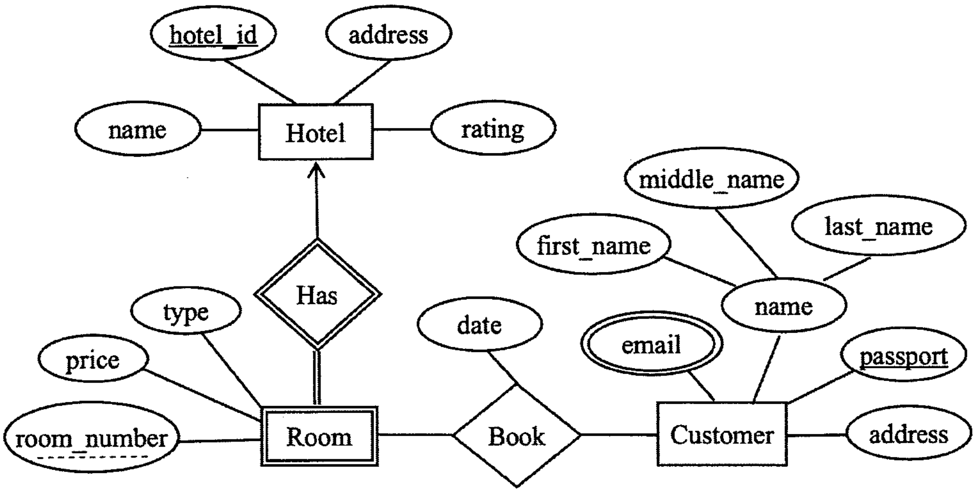
[2] Write each of the following queries in relational algebra:

(a) Get the names of employees who have worked in the department with DNO = 'D1'.

(b) Get the names of employees who have not worked for the job with JNO = 'J1'.

Question 5. ERD, Table Design, and Normalization **[20 marks]**

[1] Convert the following ER diagram into a set of relation schemas. Specify the primary key(s) and foreign key(s) for each relation.



[2] Consider the relation R = (A, B, C, D, E) and the set of functional dependencies F = {A → BC, BCD → E, B → D, A → D, E → A}.

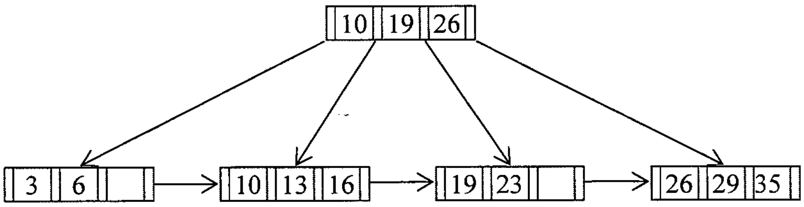
(a) Identify the candidate key(s) for R.

(b) Compute the canonical cover for F.

(c) Conduct 3NF decomposition for R which is lossless-join and dependency-preserving.

Question 6. Data Storage and Indexing **[10 marks]**

[1] Consider the following B$^+$-tree, where the tree order = 4 (i.e., maximum number of pointers in each tree node). Show the updated B$^+$-tree after inserting the key "18".



[2] Consider the following *Employee* relation. Build a hash index based on the *EID* attribute. The hash function is H(*EID*) = SUM(*EID*) mod 4, where SUM(*EID*) adds up all the numbers in the *EID*, e.g., SUM(321321321) = 3+2+1+3+2+1+3+2+1 = 18. Assume that each bucket can store 3 keys, and bucket overflow is handled by overflow chaining.

EID	Name	Salary	DeptNo
554433221	Doug Gilbert	80000	1
543216789	Joyce Pan	70000	1
333445555	Frank Wong	75000	2
987654321	Jennifer Wallace	54000	2
123456789	John Smith	56000	3
666884444	Ramesh Narayan	66000	3
453453453	Joyce Huang	60000	3
888665555	James Borg	83000	3
999887777	Alicia Zelaya	92000	4
123675134	Ahmad Jabbar	85000	4
987987989	Helen Fei	60000	4

Question 7. Query Processing and Optimization **[15 marks]**

Consider the following relations in a *University* database:

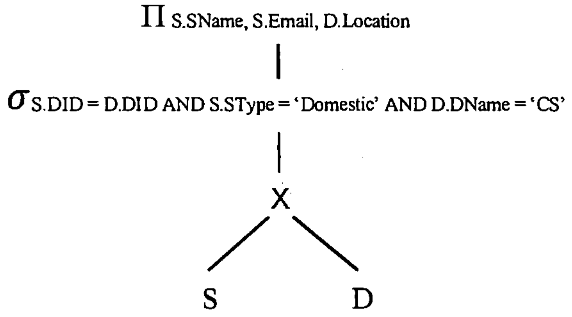
```
Student (SID, SName, SType, Email, DID)
Department (DID, DName, Location, Budget)
```

The *DID* attribute in the *Student* relation is a foreign key referencing the *Department* relation. *DName* is a unique attribute in the *Department* relation.

[1] Given the following SQL query:

```
SELECT S.SName, S.Email, D.Location
FROM Student S, Department D
WHERE S.DID = D.DID AND S.SType = 'Domestic' AND D.DName = 'CS';
```

The initial query tree is illustrated as follows:


Π S.SName, S.Email, D.Location
σ S.DID = D.DID AND S.SType = 'Domestic' AND D.DName = 'CS'
X
S
D

Show the most efficient query tree after applying all the five steps of query tree optimization.

[2] Suppose the *Student* relation contains 50000 tuples, with each disk block stores 10 tuples; the *Department* relation contains 200 tuples, with each disk block stores 20 tuples. Both relations are stored as sequential files. *Student* is ordered by *SID* and *Department* is ordered by *DID*. The memory size is 40 blocks. What is the cost of *Student* ⋈ *Department* (natural join) using the merge-join algorithm? Show your calculation process and result.

END