NATIONAL UNIVERSITY OF SINGAPORE

CS3223 – DATABASE SYSTEMS IMPLEMENTATION

(Semester 2: AY2021/22)

Time Allowed: 120 minutes

INSTRUCTIONS TO STUDENTS

1. Please write your **Student Number** only. **DO NOT** write your name.
2. This assessment paper consists of **TWO** Sections and comprises **TWELVE (12)** printed pages. Section **A** has **THREE (3)** questions, and Section **B** has **SEVENTEEN (17)** questions. The maximum score for this paper is **THIRTY** (30).
3. Answer **ALL** questions. Write your answers within the **space provided** in this booklet.
4. This is an **OPEN BOOK** assessment.
5. You are allowed to use a calculator. You are allowed to bring your laptop/iPad, and use it to refer to your notes/books. You are **NOT ALLOWED** to have network access during the assessment.

Student Number:												

EXAMINER'S USE ONLY	
Questions	**Marks**
Section A (13 marks)	
Q1 (4 marks)	
Q2 (5 marks)	
Q3 (4 marks)	
Section B (17 marks)	
Total (30 marks)	

Section A (13 marks)

1. **(4 marks)** (Serializability)
   A. If 2PL ensures conflict-serializability, why do we need strict 2PL?

   B. Consider the following execution schedule. Use the polygraph to test for view serializability. Show the polygraph. Is the schedule view serializable? Explain your answer.

$$R_3(C), W_4(B), W_4(A), W_1(B), R_2(B), R_3(D), R_2(A), W_1(A)$$

2. (5 marks) Consider the following statistics for three relations R, S, U, W**.**

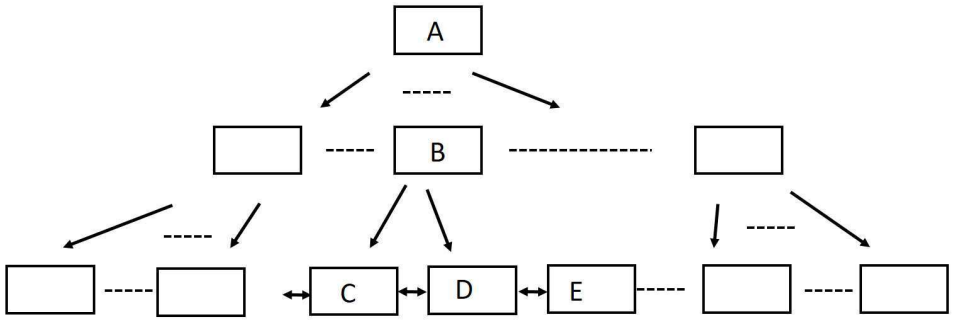
R(a, b, c)	S(a, b, d)	U(a, c, d)	W(b, c, d)
T(R) = 1000	T(S) = 2000	T(U) = 6000	T(W) = 5000
V(R, a) = 1000	V(S, a) = 50	V(U, a) = 30	
V(R, b) = 20	V(S, b) = 50		V(W, b) = 40
V(R, c) = 10		V(U, c) = 30	V(W, c) = 50
	V(S, d) = 200	V(U, d) = 50	V(W, d) = 40

Moreover, suppose that all attributes are of the same size, and that each page can contain 10 tuples of R. Records do not span across pages. Unless otherwise stated, only distinct attributes should be included in a join output, for example, the join output of R and S should contain 4 attributes (a, b, c, d). We also made the standard assumptions that we have introduced in our lecture, e.g., preservation of values, uniform distribution, independence, etc. Consider the following query plan: (((R JOIN S) JOIN U) JOIN W) where the join predicates are on all common attributes, e.g., the join between R and S will be on R.a=S.a and R.b=S.b.

A. What is the size (in number of tuples) of the query? If the number of tuples is a fraction, you should round up (e.g., if the answer is 329.5 tuples, then you should put it as 330 tuples).

B. What is the minimum buffer size required to run the query such that each table is only read once? You can use any method (except index-based methods since there are no indexes on the tables). Moreover, intermediate results are not written out to disk – so, you may store them in memory, or intermediate results can be pipelined to the next operator. For simplicity, we can IGNORE the input buffers for each table. For example, if we have 1300 buffers, then we can load S, U and W into memory. Then we can scan R to join with S (using nested loops join), and the intermediate results are pipelined to join with U (nested loops join), and the intermediate results are pipelined to join with W (nested loops join). This is just an example – it may/may not be the minimum number of buffer size required. Note that, except for the first join (R JOIN S), you MUST follow the JOIN ORDER specified according to the plan. SPECIFY the join method used.

3. **(4 marks)** Consider the concurrency control mechanism for B^+-tree (as taught in class). Suppose we have a 3 level tree as shown in the figure below. Now, transaction T1 wants to insert an entry to the B+-tree, and the path it takes is A, B, D (i.e., entry to be inserted into node D). Assume node B is full and all other nodes are 50% full. Let there be only one kind of locks, namely exclusive locks. Let Li(X) denote node X is locked by transaction Ti, and Ui(X) denote node X is unlocked by transaction Ti. Mi(X) means node X is modified by transaction Ti.



A. Write down the sequence of actions on the tree by transaction T1. For example, if you think that the sequence is T1 locks A, T1 locks B, T1 locks D, T1 unlocks A, T1 unlocks B, T1 unlocks D, then write: L1(A), L1(B), L1(D), U1(A), U1(B), U1(D). Note that this sample sequence is just an example – the actions may or may not follow the protocol that we learn.

B. Now, suppose we have another transaction T2 that issues a range search and traverses the path A, B, C and D (i.e., reading entries from leaf nodes C, and D). Moreover, suppose T1 starts at time 0, and each lock action (locking or releasing a lock) takes 1 time unit, and each modification action takes 5 time units. In addition, a transaction can only perform its next action after its previous action has successfully completed. Using the same example above, T1 locks A at time 0 and locks B at time 1 and so on. Here we further introduce a new action Wi(X) which means transaction Ti is waiting to hold a lock on node X. Now suppose transaction T2 arrives at time 1. If there is a deadlock (fill in D), you may assume wound-wait deadlock resolution scheme is used. If a transaction aborts (use Ai if Ti aborts), all locks are released in one time unit, and the transaction will restart immediately after that. When a lock is released, it becomes available and can be accessed by other transactions that need it. Use the following table to list the sequence of actions by the two transactions (the first 2 actions have been listed for you; the number of actions may not use up all 24 time units):

Time	Actions
0	**L1(A)**
1	**L1(B) W2(A)**
2	
3	
4	
5	
6	
7	
8	
9	
10	
11	
12	
13	
14	
15	
16	
17	
18	
19	
20	
21	
22	
23	
24	