SWE3003 Introduction to Database Systems - Final Fall 2024

Student ID	Name

	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Total
For Instructor/TA only,											

Academic Honor Pledge

I affirm that I will not at any time be involved with cheating or plagiarism while enrolled as a student at Sungkyunkwan University.
I understand that violation of this code will result in penalties as severe as indefinite suspension from the university.

Your signature: _____

1. [Query Optimization (20 pts)] Suppose you have a database table *Student* that has 10,000 records. Each 4 KB disk page holds 10 records without free space, i.e., 1,000 disk pages are used for the database table. Suppose your DBMS does not use the buffer cache. For each of the following database table format, describe how many disk pages need to be accessed **on average** for a given query.

(a) Select from Heap file:

```
SELECT *
FROM Student
WHERE ID = '1234';
```

answer:
500 pages on average.

(b) Insert into Sequential file sorted by ID:

```
INSERT INTO Student
VALUES (ID='1234', name='John Doe', department='Comp.Sci.', gpa=0.0);
```

answer:
For search: Log(1,000) = approximately 10 pages
For shift: 500 pages
Therefore, 510 pages on average.

(c) Update using B+tree file (key=ID) of height 4:

```
UPDATE student
SET gpa = 4.0
WHERE ID = '1234';
```

answer:
4 pages for the B+ tree traversal

(d) Select the first 100 tuples using B+tree file (key=ID) of height 4:

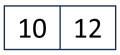
```
SELECT name
FROM student
ORDER BY ID ASC
LIMIT 100;
```

answer:
4 pages for the B+ tree traversal. Then, 9 sibling pages Total: 13 pages. (14 pages → give 3 points)

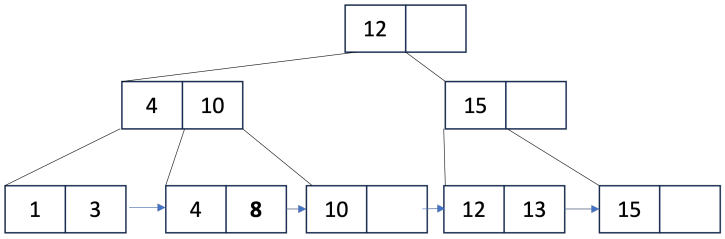
2. [B+tree (10 pts)] For the following questions about B+trees, show the tree after each insert or delete. To maintain consistency in answers, please follow the following rules:

- You should split nodes whenever there is an overflow due to insertion; that is, do not use redistribute, i.e., do not borrow from or migrate to sibling nodes.
- When splitting a leaf node due to insertion overflow, keep half (rounded up) in the left node and half (rounded down) in the right, i.e., left node has 2 keys and right node has 1 key.

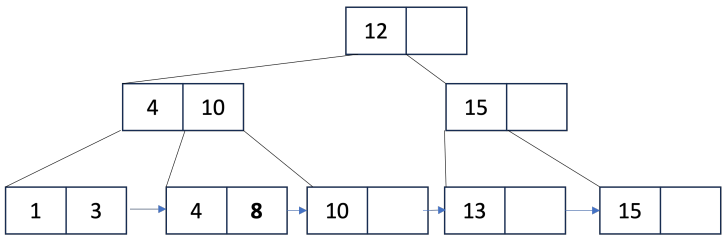
(a) Show the result of inserting 15, 4, 1, 3, 13, and 8 into the following B+tree (degree 3, i.e., 2 keys and 3 child nodes).



answer:



(b) Show the result of deleting 12.
answer:



3. [External Merge Sort (10pts)] Suppose there is a table: R with 800 pages. What is the cost of external merge sort for this table? Assume the computer uses 10 pages (9 pages for input, 1 page for output) of RAM for this operation. Also, assume that the last pass does not store the outputs in disks. How many page accesses occur at each pass? For unnecessary pass, write 0.

1st pass: 1600

2nd pass: 1600

3rd pass: 800

4th pass: 0

5th pass: 0

6th pass: 0

4. [Hash Join (10pts)] Suppose there are two tables: P and Q. Table P has 300 pages and 35 records per page. Table Q has 100 pages and 50 records per page.

(a) What is the I/O cost of hash join? Assume that the tuples are uniformly distributed enough that recursive partitioning is not necessary. However, the memory is not sufficient to hold the entire input table.
partitioning: 2 x (300 + 100)
build and probe: 300 + 100
= 1,200

(b) What is the I/O cost of hash join if the memory is sufficiently large to hold Table Q.
build and probe: 300 + 100 = 400

5. [Serializability (30pts)] Consider the following schedules where time increases from top to bottom.

time	T1	T2
1	R(C)	
2	R(B)	
3		R(A)
4		W(A)
5	W(B)	
6		R(C)
7		W(C)
8	commit	
9		W(B)
10		commit

Table 1: Schedule S1

(a) S1 is recoverable (T/F): T

(b) S1 is cascade-less (T/F): T

(c) S1 is conflict-serializable (T/F): T

(d) S1 can be generated by 2PL (T/F): T

(e) S1 can be generated by Strict 2PL (T/F): T

time	T1	T2	T3
1		R(C)	
2		W(A)	
3	W(A)		
4		W(B)	
5		Commit	
6			R(B)
7	Commit		
8			Commit

Table 2: Schedule S2

(a) S2 is recoverable (T/F): T

(b) S2 is cascade-less (T/F): T

(c) S2 is conflict-serializable (T/F): T

(d) S2 can be generated by 2PL (T/F): T

(e) S2 can be generated by Strict 2PL (T/F): F

time	T1	T2	T3
1		R(A)	
2	R(B)		
3		W(A)	
4		R(B)	
5			R(A)
6	W(B)		
7	Commit		
8			W(A)
9			Commit
10		W(B)	
11		Commit	

Table 3: Schedule S3

(a) S3 is recoverable (T/F): F

(b) S3 is cascade-less (T/F): F

(c) S3 is conflict-serializable (T/F): F

(d) S3 can be generated by 2PL (T/F): F

(e) S3 can be generated by Strict 2PL (T/F): F

6. [Timestamp-ordering (10pts)] Consider the following schedule.

(a) Using the timestamp-ordering protocol, fill in the following table that contains the values of all read-timestamp (RTS) and write-timestamp (WTS) after each operation. Initially, the write and read timestamps of data `A`, `B`, and `C` are all 0, and the timestamps of `T1`, `T2`, and `T3` are 1, 2, and 3, respectively.
To make grading easier, DO NOT fill in cells whose values do not change.

Time	T1	T2	T3	RTS(A)	WTS(A)	RTS(B)	WTS(B)	RTS(C)	WTS(C)
1	start			0	0	0	0	0	0
2		start							
3			start						
4	R(A)			1					
5	W(A)				1				
6		R(B)				2			
7		W(B)					2		
8			R(C)					3	
9			W(C)						3
10		W(A)			2				
11	R(A)								
12			R(B)			3			
13			W(B)				3		
14		W(C)							
15			W(A)		3				

(b) If any transaction is aborted, explain when and why it is aborted.

answer:
T1 aborts at time 11 because TS(T1) < W-TS(A)
T2 aborts at time 14 because TS(T2) < W-TS(C) and also TS(T2) < R-TS(C)

7. [Recovery (10pts)] The following table shows a log file after a crash occurred. The DBMS employs the *immediate database modification*, which allows updates of an uncommitted transaction to be made to the buffer, or the disk itself, before the transaction commits

LSN	LOGS
1	<START T1>
2	<T1, X, 0, 1>
3	<START T2>
4	<T1, Y, 0, 2>
5	<T2, X, 1, 3>
6	<CHECKPOINT (T1,T2)>
7	<T2, Y, 2, 4>
8	<START T3>
9	<T3, Z, 0, 10>
10	<COMMIT T1>
11	<T2, X, 3, 5>
12	<T3, Y, 4, 6>
*C*R*A*S*H*	

(a) Which log entries need to be redone, and which need to be undone? List the LSNs in the order they should be executed.
answer:

redo: 7, 8, 9, 10, 11, 12
undo: 12, 11, 9, 7, 5, 3

(b) What is the value of X, Y, and Z at the end of the recovery?

X: 1

Y: 2

Z: 0

8. [Skip Lists and LSM Trees (10 pts)] For each of the following statements, indicate whether it is TRUE or FALSE. You will get 2 points for each correct answer, -2 points for each incorrect answer, and 0 point for each answer left blank or both answers marked.

		T	F
(a)	A skip list guarantees $O(logn)$ search time in the worst case. ..... F	☐	☐
(b)	In a skip list with n keys, the expected number of nodes at level k is $n/2^k$. T	☐	☐
(c)	An LSM tree is designed to handle write intensive workloads efficiently. T	☐	☐
(d)	In LSM trees, sequential I/O patterns are leveraged to optimize both writes and reads. ..... F	☐	☐
(e)	In LSM trees, reads are always faster than writes. ..... F	☐	☐

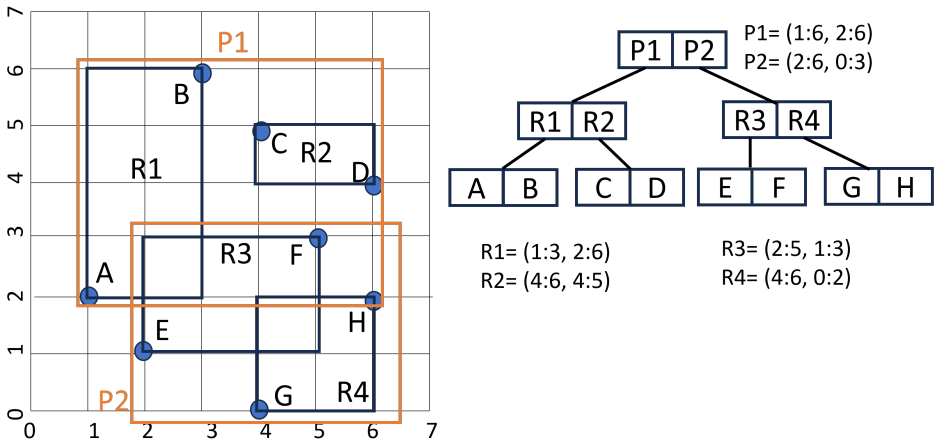
9. [MapReduce (10 pts)] Consider the following MapReduce program written in pseudo code.

```
def map(key, value):
    // key:  relation name
    // value: tuple from relation
    relation_name = key;
    if relation_name is R:
        emit(value.attribute[2], (relation_name, value))
    else :
        primary_key = value.attribute[0]
        emit(primary_key, (relation_name, value))  // pass intermediate outputs to reducers

def reduce(key, values):
    // values: list of tuples
    values1 =  values2 = []

    for each (relation_name, value) in values:
        if relation_name is R:
            values1.append(value)
        else :
            values2.append(value)

    for r in values1:
        for s in values2:
            emit(key, (r, s))  // generate output tuples
```

(a) What relational operator in a DBMS does this parallelize? Specifically, what query processing algorithm does it parallelize?:
hash join

(b) What is the number of workers performing the reduce function dependent on?
number of unique keys (in this case, the join keys) that are emitted by the map function.

10. [Spatial Indexing (10 pts)] Consider the following 2-dimensional point data and R-tree.



(a) Point query: List the rectangles and points that are visited and compared in the process of finding point F, e.g., P1 → R1 → ...

answer:
P1 → R1 → R2 → P2 → R3 → E → F → R4

(b) Nearest neighbor query: List the rectangles and points that are visited and compared in the process of finding the point nearest to F, e.g., P1 → R1 → ...

answer:
P1 → R1 → A ($\sqrt{17}$) → B ($\sqrt{15}$) → R2 → C ($\sqrt{5}$) → D (1) → P2 → R3 → E → F (0)

R4, G, and H must not be visited since the distance is 0 at this point. The distance numbers are not required in the answer.

List the rectangles or points only if they are used to compute the overlap or distance. When performing tree traversal, it is explored in a depth-first manner and in left-to-right order.