Question A: Algorithms
(20 points)

Each question below describes a specific use-case. Based only on the factors given, decide which of the following three data layout strategies would be ideal for each use-case. Assume that each layout strategy is being used with a hard disk drive (i.e., spinning platter, not SSD). For each use-case mark which option is best and explain why.

Option 1 (Record-Oriented LSM Tree)

The table is stored in a single *Log-Structured Merge Tree* with a max of 1 sorted run per level. Each entry in the LSM Tree is an *entire record*. Records are stored in a *fixed-width format*, and laid out on disk contiguously (i.e., *without paging*) in each level. A cache of the first two levels is maintained in memory. A fence pointer table (i.e., a B+ tree with a single directory node) is maintained for each level.

Option 2 (Simple Column Store)

The table is broken up by column; *Each column is stored in two separate files*: One sorted by record-id, and the other sorted by field value. Records (i.e., record-id/field value pairs) are stored *fixed-width* for columns with compatible types. For other types, records are *separated by delimiters*.

Option 3 (B+Tree)

The table is stored in a single *B+Tree index*. Each entry in the B+Tree is an entire record. Leaf pages (records) and directory pages are *4k chunks of the data file*. Each leaf page has a *footer* pointing to the position of each record on the page; each record is stored *field-delimited*.

Part 1. A table has 5 string and integer columns. New records are rarely inserted or deleted. The workload consists almost exclusively of queries for a key attribute and updates to non-key attributes of records identified by a key. **(5 points)**.

Pick Option 3 B+Trees lack the read/write amplification of a LSM tree, and do not require the expensive re-organization that would be required in response to each update in an on-disk column store.

Part 2. A table with 1000 integer columns and is re-generated from existing data each night. During the day, the workload is purely read only, and queries frequently select on or project out 3-10 columns at a time. **(5 points)**.

Pick Option 2 The number of columns makes this an ideal choice for a column store, especially given that data updates are infrequent and batched.

Part 3. A table has 30 columns including both variable length strings and integers. The workload is approximately 50% reads indexed by a single key attribute, and 50% inserts/deletes. **(5 points)**.

Pick Option 3 Nominally, this would be an ideal workload for LSM trees, but the specific configuration listed in option 1 requires fixed-size records, which are incompatible with strings.

Part 4. A table has 30 columns including both integers and floats. The workload is approximately 50% reads indexed by a single key attribute, and 50% inserts/deletes. **(5 points)**.

Pick Option 1 Unlike the previous question, here all columns are fixed-size types.

Question B: Concurrency Control
(25 points)

For each of the following schedules, identify whether the schedule ...

1. ... could have been created by 2-Phase Locking (with shared/exclusive locks)
2. ... could have been created by Snapshot Isolation (as described in class). Assume that transaction ordering is assigned in terms of commit order and that the write phase is instantaneous.
3. ... could have been created by Timestamp Concurrency Control (as described in class). Assume that transaction ordering is assigned based on the transaction ID (i.e., T1 happens before T2 happens before T3).
4. ... is Conflict Serializable
5. ... is View Serializable

Circle all schemes that could have generated the schedule, as well as every form of serializability that the schedule conforms to.

Part 1. (5 points).

T1	T2	T3
W(B)		
	R(B)	
		R(B)
R(A)		
	R(A)	
		W(A)
COMMIT		
	COMMIT	
		COMMIT

2-Phase Locking ✓

Snapshot Isolation ✓

Timestamp CC ✓

Conflict Serializable ✓

View Serializable ✓

Part 2. (5 points).

T1	T2	T3
W(B)		
	R(B)	
		R(B)
R(A)		
	W(A)	
		W(A)
COMMIT		
	COMMIT	
		COMMIT

2-Phase Locking

Snapshot Isolation ✓

Timestamp CC ✓

Conflict Serializable ✓

View Serializable ✓

Part 3. (5 points).

T1	T2	T3
		R(A)
	R(A)	
		W(A)
R(A)		
		COMMIT
W(B)		
	W(B)	
	W(A)	
	COMMIT	
COMMIT		

2-Phase Locking

Snapshot Isolation

Timestamp CC ✓

Conflict Serializable

View Serializable ✓

Part 4. (5 points).

T1	T2	T3
R(C)		
W(A)		
	R(B)	
	W(A)	
	W(B)	
R(B)		
		R(C)
		W(A)
	COMMIT	
COMMIT		
		COMMIT

2-Phase Locking

Snapshot Isolation

Timestamp CC

Conflict Serializable

View Serializable ✓

Part 5. (5 points).

T1	T2
R(A)	
W(A)	
	R(A)
	W(B)
W(B)	
	COMMIT
COMMIT	

2-Phase Locking

Snapshot Isolation

Timestamp CC

Conflict Serializable

View Serializable

Question C: LOGGING
(25 points)

Consider the following log

LogEntry	Transaction	Operation	PreviousEntry
		... 102 entries...	
103	T3	Write A [73 → 26]	-1
104	T2	Write B [33 → 50]	-1
105	T3	Write C [78 → 78]	0
106	T4	Write B [50 → 48]	-1
107		START CHECKPOINT	
108	T6	Write C [78 → 67]	-1
109		END CHECKPOINT	
		Earliest Dirty Log Entry: 2	
		T4(LastLog: 116; State: commit)	
		T5(LastLog: n/a; State: run)	
		T6(LastLog: 108; State: run)	
110	T3	Write C [67 → 81]	2
111	T2	COMMIT	1
112	T3	ABORT	7
113	T4	Write B [48 → 47]	3
114	T2	END	8
115	T3	END	9
116	T4	COMMIT	10
		Crash!	

... and the following state of objects A, B, C, D on disk

A = 26 B = 50 C = 78 D = 23

Part 1. Show the state of the buffer manager and the transaction table after the **ANALYZE** phase of ARIES recovery **(5 points)**.

Buffer:
A = 26 B = 50 C = 78 D = 23
Transaction Table:
T4(LastLog: 116; State: commit) T5(LastLog: n/a; State: run) T6(LastLog: 108; State: run)

Part 2. Show the state of the buffer manager and the transaction table after the **REDO** phase of ARIES recovery **(10 points)**.

Buffer:

A = 26 B = 47 C = 81 D = 23

Transaction Table:

T4(LastLog: 116; State: commit) T5(LastLog: n/a; State: run) T6(LastLog: 108; State: run)

Part 3. Show the state of the buffer manager and the transaction table after the **UNDO** phase of ARIES recovery **(10 points)**.

Buffer:

A = 26 B = 47 C = 81 D = 23

Transaction Table:

T4(LastLog: 116; State: commit)

Question D: Materialized Views
(10 points)

For each of the following queries $Q(R, S, T)$, and using *bag* relational algebra, compute delta query with respect to insertions into table R. That is, construct a query $\Delta Q(R, \Delta R, S, T)$ such that

$$Q(R, S, T) \uplus \Delta Q(R, \Delta R, S, T) \quad \equiv \quad Q(R \uplus \Delta R, S, T)$$

Use the following three relations: $R(A, B)$ $S(B, C)$ $T(C, D)$

Part 1. (3 points). $Q(R, S, T) = \pi_A\big(\sigma_{D=1}(R \bowtie S \bowtie T)\big)$

$\Delta Q(R, \Delta R, S, T) = \pi_A\big(\sigma_{D=1}(\Delta R \bowtie S \bowtie T)\big)$

Part 2. (3 points). $Q(R, S, T) = \pi_B(R) \uplus \pi_B(S)$

$\Delta Q(R, \Delta R, S, T) = \pi_B(\Delta R)$

Part 3. (2 points). $Q(R, S, T) = \big(\pi_{B \leftarrow B, C \leftarrow A \times 2}(R)\big) - S$ (note the *set-relational* difference operator)

$\Delta Q(R, \Delta R, S, T) = \pi_{B \leftarrow B, C \leftarrow A \times 2}(\Delta R) - S$

Part 4. (2 points). $Q(R, S, T) = R \bowtie \big(\pi_{B \leftarrow B, C \leftarrow A \times 2}(R)\big)$

$$\begin{aligned} \Delta Q(R, \Delta R, S, T) \quad &= \quad R \bowtie \pi_{B \leftarrow B, C \leftarrow A \times 2}(\Delta R) \\ &\uplus \quad \Delta R \bowtie \pi_{B \leftarrow B, C \leftarrow A \times 2}(R) \\ &\uplus \quad \Delta R \bowtie \pi_{B \leftarrow B, C \leftarrow A \times 2}(\Delta R) \end{aligned}$$

Question E: Datalog
(10 points)

Each problem provides a set of datalog rules. Translate these into equivalent *set-relational* algebra expressions.

Use the following three relations: $R(A, B)$ $S(B, C)$ $T(C, D)$

Part 1. (4 points).
$Q(A) :- R(A, B), R(B, C), S(B, C), [\![C > 2]\!]$

$$\pi_A\big(\sigma_{C>2}(R \bowtie (\pi_{B\leftarrow A, C\leftarrow B}R) \bowtie S)\big)$$

Part 2. (3 points).
$Q(B) :- R(1, B)$
$Q(B) :- R(2, B)$
$Q(B) :- S(B, C), [\![C > 2]\!]$

$$\pi_B\big(\sigma_{(A=1)\vee(A=2)}(R)\big) \cup \pi_B\big(\sigma_{C>2}(S)\big)$$

Part 3. (3 points).
$Q(C) :- M(C), T(C, D), [\![D > 2]\!]$
$M(B) :- R(1, B)$
$M(B) :- S(B, 2)$

$$\pi_C\left(\sigma_{D>2}\left(\left(\pi_B\left(\sigma_{A=1}(R)\right) \cup \pi_B\left(\sigma_{C=2}(S)\right)\right) \bowtie T\right)\right)$$

Question F: PROVENANCE
(10 points)

Consider the following three tables

R	**A**	**B**
R_0	1	1
R_1	2	2
R_2	2	0
R_3	0	2
R_4	1	2

S	**B**	**C**
S_0	2	0
S_1	1	0
S_2	2	0
S_3	1	1
S_4	2	2

T	**C**	**D**
T_0	0	0
T_1	1	2
T_2	0	2
T_3	1	0
T_4	1	2

Both questions pertain to tuple $\langle A : 1, D : 0\rangle$ in the output of query $\pi_{A,D}(R \bowtie S \bowtie T)$

Part 1. Provide one witness for the indicated result tuple. **(5 points)**.

Witnesses of the expression include: $\{\ R_0,\ S_1,\ T_0\ \}$; $\{\ R_0,\ S_3,\ T_3\ \}$; $\{\ R_4,\ S_0,\ T_0\ \}$; $\{\ R_4,\ S_2,\ T_0\ \}$

Part 2. Using row identifiers (e.g., R_1) as annotations, provide the how provenance as a semiring polynomial for the indicated result tuple. **(5 points)**.

$$(R_0 \otimes S_1 \otimes T_0) \oplus (R_0 \otimes S_3 \otimes T_3) \oplus (R_4 \otimes S_0 \otimes T_0) \oplus (R_4 \otimes S_2 \otimes T_0)$$

(factorized representations are acceptable as well.)