Midterm Solution

2023 Spring

1a. Explain ACID (8%)

- ***Atomicity***
  - All operations in a transaction either succeed (transaction commits) or fail (transaction rollback) together
- ***Consistency***
  - After/before each transaction (which commits or rollback), your data do not violate any rule you have set
- ***Isolation***
  - Multiple transactions can run concurrently, but cannot interfere with each other
- ***Durability***
  - Once a transaction commits, any change it made lives in DB permanently (unless overridden by other transactions)

1b. Explain Phantom Read (4%)

Ans:

Phantom read occurs within a transaction when the same query produces different sets of rows at different times as a result of another transaction’s insert or update.

2. Why does S2PL require a transaction to hold all blocks until it completes? (7%)

Ans:

In S2PL, if a write operation of tx1 precedes a conflicting transaction of tx2 (either read or write), tx1 will commit before that conflicting operation of tx2 (No cascading abort).

3. What is the difference between the purposes of using E-R Model and Relational Model when designing your data model? (7%)

Ans:

E-R Model is better at performing the relationship with entities.

Relational Model is more closely related to the real implementation of our database.

4. What are the benefits of executing transactions concurrently (by interleaving operations) in a DBMS? Please describe at least one benefit that also works on a single-core machine. (7%)

Ans:

Interleaving operations increases throughput by pipelining CPU and IO.

5. Do we need a buffer pool for the log blocks? Why or why not? (7%)

Ans:

No, one buffer is enough.

When appending log, all worker threads pin the tail block of the log file.

For backward read (recovery), the recovery thread pins the block that is being read.

6. What is the difference between a lock and a latch in a DBMS? (5%)

	Locks	*Latches*
Separate …	User transactions	Threads
Protect …	Database contents	In-memory data structures
During …	Entire transactions	Critical sections
Modes …	Shared, exclusive, update, intention, escrow, schema, etc.	Read, writes, (perhaps) update
Deadlock …	Detection & resolution	Avoidance
… by …	Analysis of the waits-for graph, timeout, transaction abort, partial rollback, lock de-escalation	Coding discipline, "lock leveling"
Kept in …	Lock manager's hash table	Protected data structure

7. Consider the following steps, called write-ahead-logging (WAL), for writing a record into disk in a DBMS.
1) Write log to log buffer
2) Write data to buffer
3) Flush log buffer into disk
4) Flush data buffer into disk
Is it valid to swap steps 3 and 4? Why or why not? (10%)

Ans:

No. If the system crashes during step when data is being flushed, there is no guarantee that the log is flushed (possibly non-recoverable).

Consider two tables R and S,
B(R) is the number of block accesses required to load table R into main memory.
B(S) is the number of block accesses required to load table S into main memory.
Table R has **m** records.
Table S has **n** records.

8a. How many block accesses are required to run the following join algorithm? (10%)
(Express your answer in terms of **B(R)**, **B(S)**, **m**, and **n**)

```
for record r of R:
     for record s of S:
          yield if r and s match
```

Ans:

B(R) + m * B(S)

B(R) + B(S) assuming the buffer pool is big enough.

Consider two tables R and S,
B(R) is the number of block accesses required to load table R into main memory.
B(S) is the number of block accesses required to load table S into main memory.
Table R has **m** records.
Table S has **n** records.

8b. How many block accesses are required to run the following join algorithm? (10%)
(Express your answer in terms of **B(R)**, **B(S)**, **m**, and **n**)

```
for block BR of R:
     for block BS of S:
          for record r of BR:
               for record s of BS:
                    yield if r and s match
```

Ans:

B(R) + B(R) * B(S)

B(R) + B(S) assuming the buffer pool is big enough.

9a. Is this schedule serializable? If yes, what is the equivalent serial order? If not, why? (5%)

Transaction 1	Transaction 2
Read A	
Write A	
	Read A
Write B	
	Write B
	Commit
Commit	

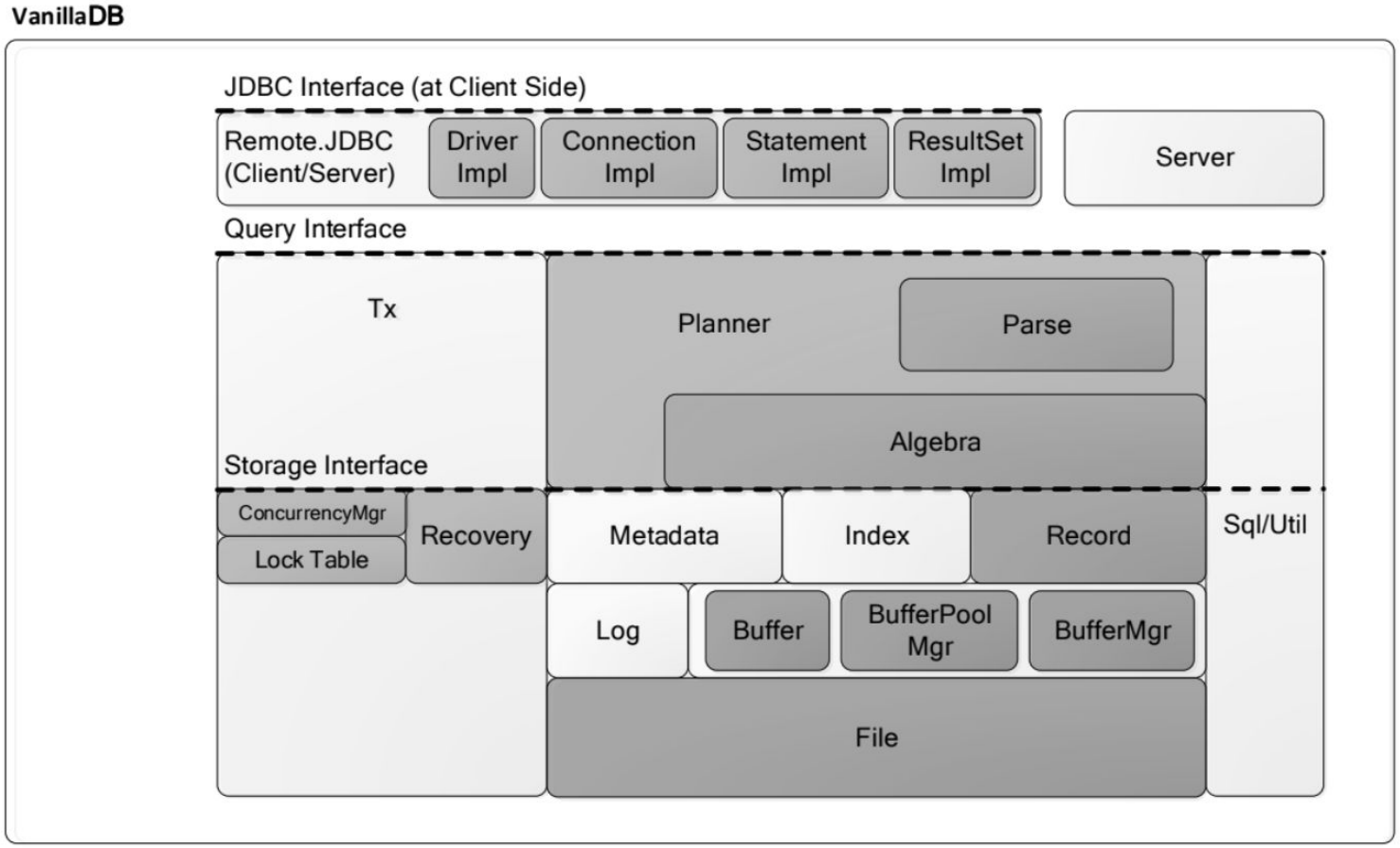
Ans:

Yes. The result is the same as serially executing transaction 1 followed by transaction 2.

9b. Is this schedule recoverable? Why? (5%)

Transaction 1	Transaction 2
Read A	
Write A	
	Read A
Write B	
	Write B
	Commit
Commit	

Ans:
No. If transaction 1 intends to abort after transaction 2 commits, there is no way to rollback
transaction 1 without affecting the states made by transaction 2.



10. Among the shaded components, which are thread-safe? (thread-safe: safe for sharing between threads) (10%)

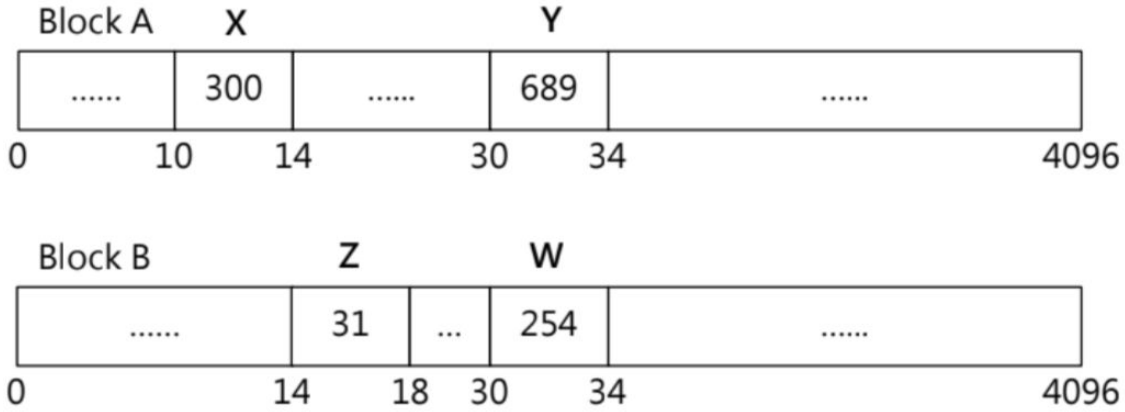
Ans:

DriverImpl, LockTable, Buffer, BufferPoolMgr, File (2% each)

(-1% for each wrong answer)

11. Suppose a DBMS shuts down due to power outage. Consider the following log records on disk:

<Tx.11, Start>
<Tx.12, Start>
<Tx.11, Block B, offset 14, 11, 15>
<Tx.12, Block A, offset 10, 200, 300>
<Tx.13, Start>
<Tx.11, Commit>
<Tx.13, Block A, offset 30, 92, 689>
<Tx.12, Block B, offset 14, 15, 31>
<Checkpoint, 12, 13>
<Tx.14, Start>
<Tx.12, Block A, offset 10, 300, 400>
<Tx.12, Block B, offset 30, 254, 361>
<Tx.12, Commit>
<Tx.14, Block B, offset 14, 31, 52>
<Tx.13, Block A, offset 10, 400, 700>
<Tx.13, Abort>
<Tx.15, Start>
<Tx.15, Block B, offset 30, 361, 74>
<Tx.15, Commit>



Use the undo-redo algorithm to recover the system state, what are the values of X, Y, Z, and M after recovery? (15%)

X: 400, Y:92,

Z: 31, W: 74 (5% per answer)