Masinde Muliro University of Science & Technology

School of Computing & Informatics

Dept: Computer Science

BCS 314: Database System II **SEM I 2024/2025** **CAT I**

Instruction: Attempt All Questions

a) Consider a database with objects *X* and *Y* and assume that there are two transactions *T*1 and *T*2. Transaction *T*1 reads objects *X* and *Y* and then writes object *X*. Transaction *T*2 reads objects *X* and *Y* and then writes objects *X* and *Y* .

i. Give an example schedule with actions of transactions *T*1 and *T*2 on objects *X* and *Y* that results in a write-read conflict.

ii. Give an example schedule with actions of transactions *T*1 and *T*2 on objects *X* and *Y* that results in a read-write conflict.

iii. Give an example schedule with actions of transactions *T*1 and *T*2 on objects *X* and *Y* that results in a write-write conflict.

b) Justify the following statement: Concurrent execution of transactions is more important when data must be fetched from (slow) disk or when transactions are long, and is less important when data is in memory and transactions are very short.

c) When is it useful to have replication or fragmentation of data? Explain your answer.

d) Database-system implementers have paid much more attention to the ACID properties than have file-system implementers. Why might this be the case?

e) In your view might a distributed database designed for a local-area network differ from one designed for a wide-area network? Justify your answer

f) Consider the following two transactions:

***T*1**: read(*A*);	***T*2**: read(*B*);
read(*B*);	read(*A*);
if *A* = 0 **then** *B* := *B* + 1;	**if** *B* = 0 **then** *A* := *A* + 1;
write(*B*).	write(*A*).

Let the consistency requirement be A = 0 ∨ B = 0, with A = B = 0 the initial values.

i. Show that every serial execution involving these two transactions preserves the consistency of the database.

ii. Show a concurrent execution of ***T*1** and ***T*2*** that produces a nonserializable schedule.

iii. Is there a concurrent execution of ***T*1*** and ***T*2*** that produces a serializable schedule?

g) For each of the following protocols, describe aspects of practical applications that would lead you to suggest using the protocol, and aspects that would suggest not using the protocol:

i. Two-phase locking

ii. Timestamp ordering

iii. Validation

h) The following table shows the schedule for three transactions updating values A, B, C.

Time	Transactions		
	T1	**T2**	**T3**
t0		read_item (A)	
t1		read_item (B)	
t2		read_item (B)	
t3			read_item (B)
t4			read_item (A)
t5	read_item (C)		
t6	write_item (C)		
t7			write_item (B)
t8			write_item (A)
t9		read_item (C)	
t10	read_item (B)		
t11	write_item (B)		
t12	abort	write_item (C)	
t13		commit	commit

i) Explain why the above schedule could result in update errors in a database without concurrency control.

j) Locking could be used to allow concurrent execution of the three transactions t1, t2 and t3.

i. Explain how locking works, and how it interacts with the transaction mechanism.

ii. Demonstrate the effect of locking on the schedule shown above.