Practice final exam

Due No due date **Points** 24 **Questions** 6 **Time Limit** None

Instructions

This practice exam does not count for your final grade.

Answers should only contain simple text. You do NOT need to upload any image. DO NOT click anything in the editing toolbar, e.g., changing format, uploading media, adding equation, etc. The final exam questions will be in the similar format, but there will be more questions. The final exam will have a strict time limit as well.

Have a look at the model answers at the end of the practice. Your answers in the final exam can be written in the same style.

Attempt History

	Attempt	Time	Score
LATEST	Attempt 1	less than 1 minute	0 out of 24 *

* Some questions not yet graded

Submitted Jul 30 at 23:25

Unanswered

Question 1

0 / 2 pts

Machine A has a higher cache hit ratio than Machine B. The cache access time and the memory access time of both machines are the same. Which machine has a faster effective memory access time?

Correct Answer

- Machine A
- Both machines have the same effective memory access time
- Machine B

Unanswered

Question 2

0 / 2 pts

Which one of the following RAID settings does not have the equal number of write operations on average among all the disks?

Correct Answer

- RAID 3 with 3 disks
- RAID 1 with 3 disks
- RAID 2 with 2 disks
- RAID 5 with 3 disks

Unanswered

Question 3

Not yet graded / 4 pts

In a nested transaction, a transaction PARENT has three sub-transactions A, B, C. For each of the following scenarios, answer which of these four transactions' commits can be made durable, and which ones has to be forced to rollback. Write your answer in separate lines for each scenario.

- Scenario 1: Commit by A, B, and C; but PARENT rolls back.
- Scenario 2: Commit by A, B, C, and PARENT.
- Scenario 3: Commit by A, B, and PARENT; but C rolls back.

Your Answer:

- Scenario 1: No transaction is made durable. Transactions A, B, and C are forced to rollback
- Scenario 2: All four transactions are made durable, no forced rollback.
- Scenario 3: A, B, and PARENT are made durable. C rolls back.

Unanswered

Question 4

Not yet graded / 4 pts

There are two nodes in a network that use stable storage and acknowledgment message passing for reliable communication. The stable storage of Node A contains the following record - **Received message (In6); Transmitted message(Out3); Out:3 Ack:3 In:6.** The stable storage of Node B contains the following record - **Received message (In3); Transmitted message(Out6); Out:6 Ack:6 In:3.**

Now Node B sends a new message 7 to Node A. What will be in the stable storage of A and B if the message is received correctly, including a correctly received acknowledgement?

Your Answer:

Node A: Received message (In**7**); Transmitted message(Out3); Out:3 Ack:3 In:**7**.

Node B:Received message (In3); Transmitted message(Out**7**); Out:**7**Ack:**7**In:3.

Unanswered

Question 5

Not yet graded / 4 pts

Assume a large distributed system has many servers at different locations. The network connection between the servers is not reliable. There are millions of active users who regularly use this system. Why eventual consistency is more suitable than strong consistency for this system?

Your Answer:

Immediate strong consistency is impossible in case of a network partition. Moreover, as there are many servers and users, it is non-trivial to efficiently and reliably propagate and store data at any given time with unreliable network connection. Ensuring strong consistency will require relaxing availability and having lower latency according to the CAP theorem. As there are millions of users, availability is important to provide service to all these users with minimum latency. With eventual consistency, the updates will propagate to the other nodes and partitions of the network eventually, but at the cost of this relaxed consistency, the system will be able to offer higher availability in the event of network partition.

Unanswered

Question 6

Not yet graded / 8 pts

10. The following transactions are issued in a system at the same time. Answer for both scenarios.

(i) Scenario 1: When the value of the variable some_input is 3, which of the following transactions can run concurrently from the beginning till commit (that is, all operations and locks are compatible to run concurrently with another one) and which ones need to be delayed? Please give explanation for the delayed transactions.

(ii) Scenario 2: When the value of the variable some_input is 1, which of the following transactions can run concurrently from the beginning till commit (that is, all operations and locks are compatible to run concurrently with another one) and which ones need to be delayed? Please give explanation for the delayed transactions.

A compatibility matrix is as follows –

Compatibility Mode of Granular Locks							
Current	None	IS	IX	S	SIX	U	X
Request	+I- (Next mode) + granted / - delayed						
IS	+(IS)	+(IS)	+(IX)	+(S)	+(SIX)	-(U)	-(X)
IX	+(IX)	+(IX)	+(IX)	-(S)	-(SIX)	-(U)	-(X)
S	+(S)	+(S)	-(IX)	+(S)	-(SIX)	-(U)	-(X)
SIX	+(SIX)	+(SIX)	-(IX)	-(S)	-(SIX)	-(U)	-(X)
U	+(U)	+(U)	-(IX)	+(U)	-(SIX)	-(U)	-(X)
X	+(X)	-(IS)	-(IX)	-(S)	-(SIX)	-(U)	-(X)

T1	T2	T3

Lock (S,A)
Read A
Unlock A
	Lock (IS,A)
	If(some_input == 3){
	Lock(S,A)
	Read A
	}
	Unlock A
		Lock (IX,A)
		If(some_input == 3){
		Lock (X,A)
		Write A
		}
		Unlock A

Your Answer:

Scenario 1: When some_input is 3, T1 and T2 can run concurrently as S and IS locks are compatible.

Although T2's IS lock and T3's IX lock are compatible, however for some_input==3, T3's X lock request will conflict with other's S locks. So T3 cannot run concurrently with any of the other two transactions.

Scenario 2: T1 and T2 can run concurrently as S and IS locks are compatible.

When some_input is 1, T2 and T3 will not request S or X lock. Hence T2 and T3 can run concurrently as IS and IX are compatible.

T1 and T3 cannot run concurrently as S and IX are not compatible.