# COMP90050 Semester 1, 2023

## Sample Exam Questions

**Question 1**: Are solid state drives becoming faster than hard disk drives in general?
If you think so rationalize why you think that is the case, if not explain your rationale for that as well.

**Answer:**
Yes, solid state drives are generally getting faster than classical hard disks. The reason is that SSDs do not have any moving parts, which helps with the elimination of seek/rotational latency.

**Question 2**: Which of the following RAID configurations has the highest disk space utilization when all disks have the same capacity? The utilization is the ratio of the size of data, excluding mirroring and parity, to the total capacity. Your answer needs to show explanations with calculations. (1) RAID 0 with 3 disks, (2) RAID 1 with 2 disks, (3) RAID 3 with 3 disks, (4) RAID 4 with 3 disks.

**Answer:**
RAID 0 stores contiguous blocks of data of a file across different disks. There is no mirroring or parity in RAID 0. Therefore, its utilization is 1.
RAID 1 uses mirroring, where each block of data is copied to another disk. Therefore, its utilization is 1/2.
RAID 3 uses a dedicated parity disk. Therefore, the utilization of a RAID 3 with 3 disks is (3-1)/3=2/3.
RAID 4 also uses a dedicated parity disk. The utilization of a RAID 4 with 3 disks is (3-1)/3=2/3.
Among the four configurations, the one with RAID 0 (the first configuration) has the highest utilization.

**Question 3**: Is the duration of locks usually shorter in optimistic concurrency control than in two-phase locking? Briefly explain your answer.

**Answer:**
Yes, the duration of locks is usually shorter in optimistic concurrency control than in two-phase locking. In two-phase locking, all lock operations must precede object accesses. For achieving serializability, data objects are normally locked earlier in a transaction then unlocked towards the end of the transaction in two-phase locking with a locking and an unlocking phase. Optimistic case only needs to take locks when it is time to commit. It assumes that during concurrent execution, most of the time, transactions do not create conflicts with each other. Therefore, the duration of locks in optimistic case can be significantly shorter than in two-phase locking.

**Question 4**: A failfast system has 7 devices. Assume 4 out of the 7 devices are unavailable but there are still 2 agreeing devices. Can the system continue to operate in this scenario and why?

**Answer:**
Failfast requires that a majority of the available modules have agreeing output. The system can continue to operate because a majority of the available devices, i.e., 2 out of the 3 available devices have agreeing output.

**Question 5**: Compare ACID and BASE properties; what do they aim each; what are their differences, and advantages or disadvantages if a DBMS designer follows one or the other set. Briefly explain.

**Answer:**
ACID properties prioritize consistency over availability. That is the classical view, i.e. the consistency of data over even a networked environment is primal. The CAP theorem shows that it is impossible to guarantee strict Consistency and Availability at all times i.e., especially if you want to tolerate network partitions. This is a problem as companies do not want their systems to be unavailable. Thus research resulted in databases with relaxed properties than ACID properties. New databases, e.g., NoSQL, apply the BASE properties at times which are:

Basically Available: the system guarantees availability

Soft-State: the state of the system may change over time

Eventual Consistency: the system will eventually become consistent

Basically, such loose consistency leads to availability which is easily implementable and efficient and the reverse is also true where strict consistency is harder to implement and inefficient. But note that nothing is free, not all parts of a system can tolerate loose consistency as well, such as checkout systems at online shopping where strict consistency is expected.

**Question 6**: Relation T1 has 2,000 records stored in 50 blocks that can be read consecutively. Relation T2 has 500 records stored in 20 blocks that can be read consecutively. For a SQL query with a join operation, the query optimizer chooses to use block nested-loop join. Should the outer relation be T1 or T2 based on the cost in the worst case? Your answer needs to show the computation steps.

**Answer:**
If T1 is the outer relation: The number of block transfers is 50*20+50=1050. The number of seeks is 2*50=100.
If T2 is the outer relation: The number of block transfers is 20*50+20=1020. The number of seeks is 2*20=40.
Based on these numbers, the outer relation should be T2 as it can lead to lower costs.

**Question 7**: A history H of transactions is <(T1,R,O1), (T3,W,O5), (T3,W,O1), (T2,R,O5), (T2,W,O2), (T5,R,O4), (T1,R,O2),(T5,R,O3)>. What are the dependency relations in this history? Is this history equal to a serial execution, briefly explain? Is it equal to a serial execution if the second item in the history was changed to (T3, R, O5), resulting in a history <(T1,R,O1), (T3,R,O5), (T3,W,O1), (T2,R,O5), (T2,W,O2), (T5,R,O4), (T1,R,O2), (T5,R,O3)> ? Your answer needs to show explanations.

**Answer:**
In the first case, dependency relations are DEP(H) = {< T1, O1, T3>, <T3, O5, T2>, <T2, O2, T1>}. Serial execution equivalence is not possible. This is because there exists a wormhole transaction (e.g., T1 is before and after T3).

If (T3, W, O5) was changed to (T3, R, O5), the dependency relation would be DEP(H) = {< T1, O1, T3>, <T2, O2, T1>}, which removes the wormhole transaction and makes it possible to be equal to a serial execution.

**Question 8**: Assume memory access time is M, cache access time is C and hit ratio is H. In scenario 1, M=1000C and H= 30%. In scenario 2, M=10C and H=90%. Which scenario has the best effective access time EA? Answer needs to show the computation steps.

**Answer:**
In scenario 1, EA=H*C+(1-H)*M=0.3C+0.7M=0.3C+700C=700.3C
In scenario 2, EA=H*C+(1-H)*M=0.9C+0.1M=0.9C+C=1.9C
Scenario 2 has the best effective access time.

**Question 9**: What is the difference between two-phase locking and strict two-phase locking and where would it matter? Discuss a problem that the strict version addresses.

**Answer:**
In two-phase locking, all the locks are acquired in one phase and the locks are released in another phase before the commit time. In strict two-phase locking, the locks are not released before the commit, i.e., the locks are released at commit. Strict two-phase locking can lead to lower concurrency and lower efficiency compared to two-phase locking because locks are hold for a longer time. However, strict two-phase locking can help with preventing cascading aborts because it ensures that transactions only read values that were committed.

**Question 10**: A database system makes checkpoints while logging transaction operations. The system crashes and needs to be recovered. Which transactions should be redone/undone/ignored given the following log records? What are the values of A and B after recovery?

<T0 start>

<T0, A, 1000, 1500>

<T1 start>

<T1, B, 2000, 2100>

<T1 commit>

<T2 start>

<T2, B, 2100, 2200>

<checkpoint>

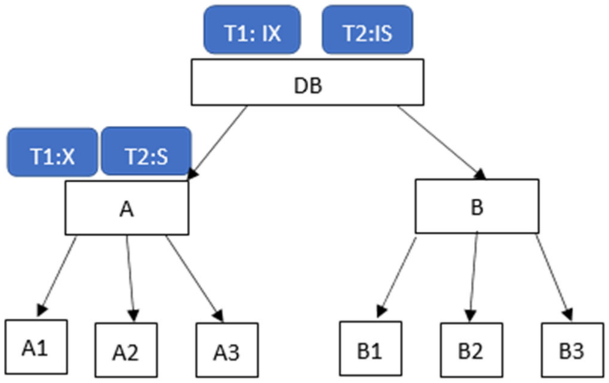
<T2 commit>

<T0, A, 1500, 1600>

System crash at this point

**Answer:**
T0 should be undone. T1 should be ignored. T2 should be redone. After recovery, A is 1000 and B is 2200.

**Question 11**: Two transactions T1 and T2 start to run at the same time. T1 needs to modify object A1 and T2 needs to read A3. The following diagram shows a plan of granular locks for the two transactions. What would be the problem with this plan? How to rectify the problem?



**Answer:**

The problem is that the two transactions need to get incompatible locks at the same node. Specifically, T1's X lock is not compatible with T2's S lock at node A. Consequently, one of the transactions must be delayed until the other transaction releases the lock on A. To rectify this problem, T1 can get IX lock on A and T2 can get IS lock on A. Then T1 can get X lock on A1, which the transaction needs to modify. T2 can get S lock on A3, which the transaction needs to read. By doing so, the two transactions can run in parallel; leads to higher concurrency.

**Question 12:** From the table below, please create the Cross Tabulation (crosstab) **by** *Type* **and** *Specs*.

| **Product ID** | **Type** | **Brand** | **Specs** |
|---|---|---|---|
| 1 | laptop | apple | low |
| 2 | desktop | dell | medium |
| 3 | server | hp | high |
| 4 | laptop | lenovo | medium |
| 5 | desktop | hp | high |
| 6 | desktop | dell | low |
| 7 | laptop | hp | high |
| 8 | desktop | apple | medium |
| 9 | server | dell | high |
| 10 | laptop | hp | low |

**Answer:**

*Brand = All*

| | | **Specs** | | | |
|---|---|---|---|---|---|
| | | low | medium | high | **Total** |
| **Type** | laptop | 2 | 1 | 1 | 4 |
| | desktop | 1 | 2 | 1 | 4 |
| | server | 0 | 0 | 2 | 2 |
| | **Total** | 3 | 3 | 4 | 10 |