Department of Computer Science

Prof. Dr. Nikolaus Augsten
Jakob-Haringer-Str. 2
5020 Salzburg, Austria
Phone: +43 662 8044 6347
E-Mail: nikolaus.augsten@plus.ac.at

PARIS LODRON UNIVERSITÄT SALZBURG

Non-Standard Database Systems — Exam
Summer Semester 2024 — 24.07.2024

Name: _____ **Student ID:** _____

Hints

- Check whether you received all pages of the exam (9 pages).
- Write your name or your student ID on each sheet of the exam and hand in all pages.
- All answers are expected to be written on the exam sheets.
- Clearly highlight and enumerate additional pages that are used for longer answers. Match your text with the according exercise.
- Only use pencils that are permanent and non-red colored.
- Use the notation and techniques discussed in the lecture.
- Exercises with more than one solution are not graded.
- You are allowed to use one A4 sheet with your personal notes (both sides, hand written or printed).
- Exam duration: 90 minutes

Signature _____

Grading — Filled by the examiner

Exercise	1	2	3	4	5	6	7	8	Sum
Total points	2	2	2	2	2	2	2	2	16
Points reached									

Exercise 1 - *Throughput and Response Time.* 2 Points

A database system must process transactions t_1, t_2, t_3, and t_4. For each transaction, Table 1 shows the start time (when the transaction enters the system) and the execution time (the runtime of the transaction in the system when it is not interrupted).

Compute the average response time and the throughput for the following task scheduling strategies where longer transactions are given priority over shorter transactions (i.e., the waiting queue is ordered by descending execution time):

1. No running transaction is aborted.
2. Abort the running transaction when a longer transaction enters the waiting queue.

Transaction	Start Time	Execution Time
t_1	0	3
t_2	1	4
t_3	2	2
t_4	9	3

Tabelle 1: Start and execution time of transactions.

Exercise 2 - *Query Parallelism*. 2 Points

What form of query parallelism (interquery, interoperation, or intraoperation) is likely to be the most important for each of the following tasks and why? Explain your choice.

1. Increasing the throughput of a system with many small queries.
2. Increasing the throughput of a system with a few large queries, when the number of disks and processors is large.

Exercise 3 - *Commit Protocols.* 2 Points

Mark the following statements as true (**T**) or false (**F**).

Grading: Positive marking rewards each correct answer with 0.5p. Each incorrect marking is penalized with 0.25p per question. Questions without any marking are considered unanswered and will not be graded.

1. 2-Phase Commit guarantees that a transaction which executes at multiple sites must either be committed or aborted at all the sites. ☐
2. Active sites may have to wait for a failed coordinator to recover in 3-Phase Commit. ☐
3. 3-Phase Commit assumes that at most k sites can fail at a time, including the coordinator. ☐
4. A message can be received more than once using the persistent messaging protocol. ☐

Exercise 4 - *3-Phase-Commit (3PC).* 2 Points

Show by example how the 3PC protocol results in an inconsistent state if network partitioning occurs.

Exercise 5 - *Replication*. 2 Points

Give an example where lazy replication can lead to an inconsistent database state even when updates get an exclusive lock on the primary (master) copy.

Exercise 6 - *Election Algorithms*. 2 Points

A distributed system consist of four sites, S_i with $1 \leq i \leq 4$, and elects a coordinator node using the bully algorithm. Illustrate the election algorithm when site S_1 recovers from a crash and all other sites are alive. Assume that site S_4 is the coordinator.

Exercise 7 - *2-Phase Locking*. 2 Points

Execute the schedule in Figure 1 using rigorous 2-phase locking (i.e., a transaction releases its locks after the commit) with the majority-based locking protocol. Assume a distributed system with five nodes and full replication.

List the locking actions for each command in the schedule.

T_1	T_2	T_3
start		
R(A)		
		start
		R(A)
W(B)		
commit		
		W(B)
	start	
		R(B)
		commit
	W(A)	
	commit	

Abbildung 1: Transaction Schedule.

Exercise 8 - *Range-Partitioning Sort*. 2 Points

Execute range-partitioning sort on 20 values in the range $0 \ldots 99$ that are horizontally partitioned (round robin) on five processors, P_i, $0 \leq i \leq 4$, in a shared-nothing environment. The processors P'_0, P'_1, and P'_2 support the sorting.