

Distributed Data Management – Exam

Summer Term 2021

Matriculation Number: _____

| | | | | | | | | | | | |
|---|---|---|---|----|---|---|---|---|---|----|----|
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | Σ |
| | | | | | | | | | | | |
| 1 | 6 | 6 | 4 | 13 | 6 | 8 | 5 | 7 | 9 | 5 | 70 |

Important Rules:

- The exam must be solved within **180 minutes** (13:00 – 16:00).
- Fill in your matriculation number (Matrikelnummer) above and **on every page**.
- Answers can be given in English or German.
- Any usage of external resources (such as scripts, prepared pages, electronic devices, or books) is not permitted.
- Make all **calculations transparent and reproducible!**
- Use the **free space under each task for your answers**. If you need more space, continue on the **back of the page**. Use the extra pages at the end of the exam only if necessary. **Provide a pointer to an extra page** if it should be considered for grading. The main purpose of the extra pages is for drafting.
- Please write clearly. **Do not use the color red or pencils.**
- If you have any questions, raise your hand.
- The exam consists of 18 pages including cover page and extra pages.
- For any multiple choice question, more or fewer than one answer might be correct.
- Good luck!

Task 0: Matriculation Number

Fill in your matriculation number (Matrikelnummer) on every page including the cover page and the extra pages (even if you don't use them).

Hint: Do it now!

1 point

Task 1: Distributed Systems

1. Which of the following statements about distributed systems are true? Tick all true statements. **4 points**

- A distributed system is a group of independent compute nodes that communicate and collaborate to solve a common task.
- A reliable distributed system is one that expects faults and errors to happen, but prevents that these events escalate into failures.
- If a distributed system uses data parallelism, it needs to replicate data to different nodes.
- A distributed system that uses quorum-based leaderless replication has to compromise on CAP's "C" if failures occur, but it is therefore 100% "A" and "P".

2. Alice was given the homework to write an algorithm that should achieve a speedup of at least 4 on 5 nodes. She wrote a 95% parallel solution and hands the homework in hoping that it will be sufficient. Considering Amdahl's law and assuming the problem size did not change with the distribution (and ignoring communication costs), what is the speedup that Alice will actually get? **2 points**

Task 2: Encoding

1. Encoding is the process of changing the representation of digital information, which is needed for remote communication. In the lecture, we discussed four data model layers: *representation*, *physical*, *conceptual*, and *logical*. Bring the layers in the correct order and assign each of the following concepts to the layer it belongs to: *XML documents*, *byte sequences*, *code objects*, *magnetic fields*, *hash maps*, *relational tables*, and *pulses of light*. **3 points**


2. Variable-length integers are a central trick used in different binary encoding formats. Consider the following variable-length integer bit sequence. What bit sequence does it represent? **2 points**

byte 1:

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 0 | 0 | 1 | 0 |
|---|---|---|---|---|---|---|---|

 byte 2:

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 |
|---|---|---|---|---|---|---|---|



Integer in little endianness:

3. The binary encoding format Avro uses two types of schemata: the **readers schema** and the **writers schema**. When reading data, Avro performs on-the-fly schema mapping. By what criterion are the attributes in the two schemata matched? **1 points**

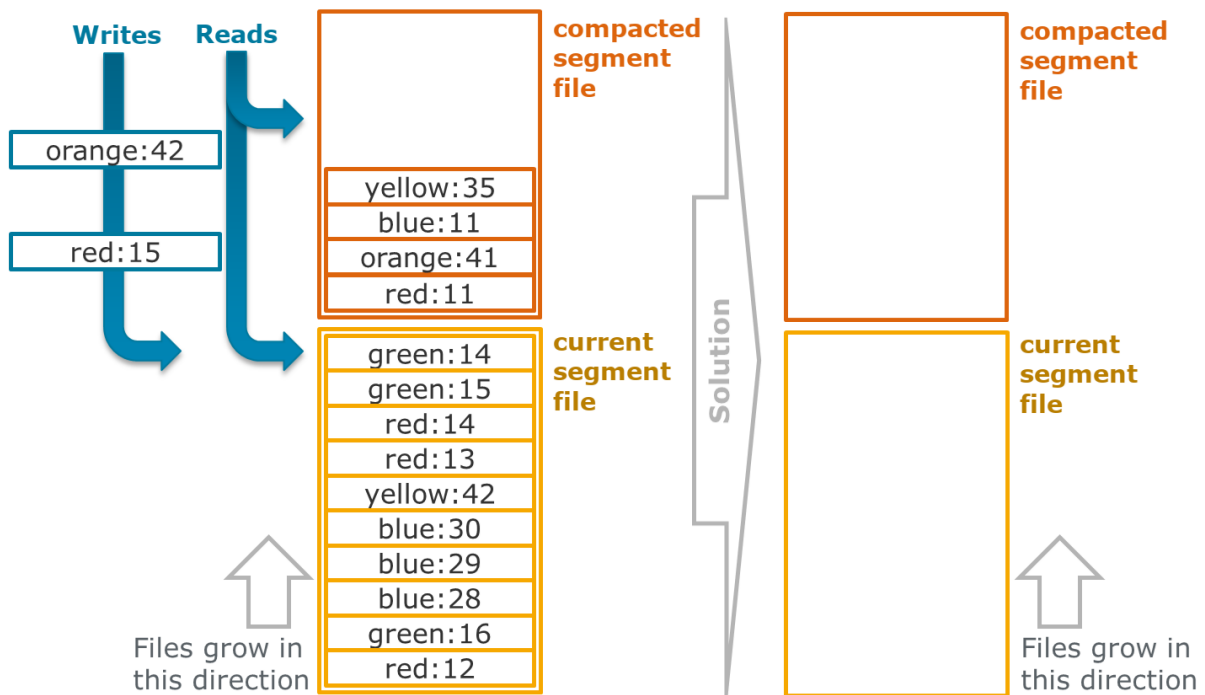
Task 4: Actor Programming

1. The Actor model uses *actors* as the universal primitive of concurrent programming. What three properties define an actor and how do actors communicate? **2 points**
2. Which of the following statements on the actor model and actor programming are *true*? Tick the true ones. **3 points**
 - An *actor system* in Akka organizes all created actor objects in a single hierarchy; for execution, the ActorSystem schedules these actor objects dynamically on threads.
 - The *proxy pattern* helps an actor to, for instance, externalize functionality, handle resource intensive actions, or protect itself from other actors.
 - The *cameo pattern* serves to encapsulate waiting behavior in a proxy that collects partial results from (potentially multiple) other actors.
3. A reoccurring problem in actor programming is that actors overload other actors with too many messages so that their mailboxes overflow. Given a **Producer** actor that needs to send lots of work messages to a **Consumer** actor, how can this problem be avoided? **1 points**
4. Another reoccurring problem in actor programming is that actors become unresponsive if they need to work on long running tasks. How can a **Worker** actor stay responsive and avoid extensively long thread-blocking even if it needs to work on a long task? Assume that the task cannot be broken down and/or offloaded to other actors. **2 points**

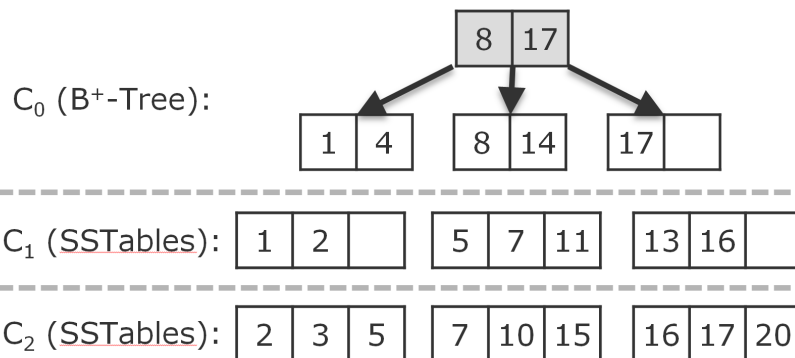
5. Shutting a distributed system down is a challenge, because some entity in each system needs to figure out that all work has been done. The *reaper pattern* in Akka serves exactly this purpose. Use the space on this sheet to implement a **Reaper** actor that watches (`this.context().watch(<ActorRef>)`) the actors of its ActorSystem and terminates (`this.context().system().terminate()`) the system when all actors are gone. **5 points**

Task 5: Storage and Retrieval

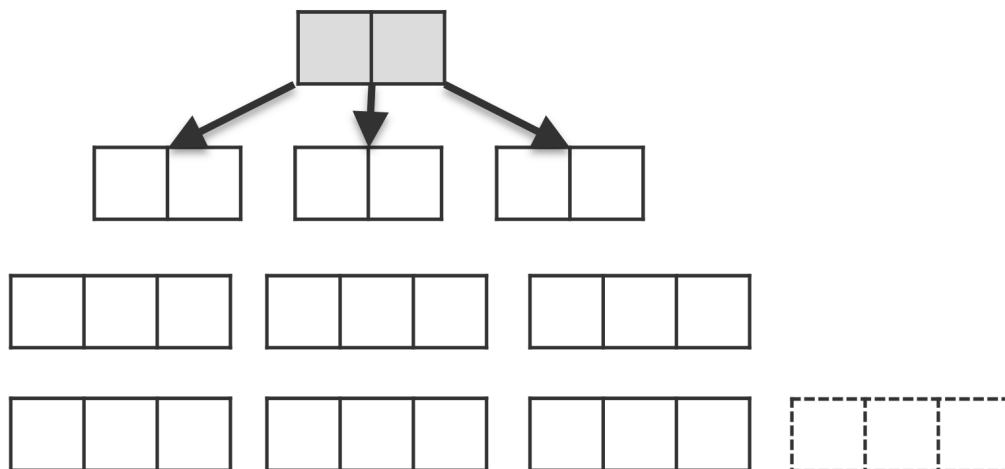
1. Consider the following *current segment file* and *compacted segment file* as well as the two entries that are currently in the process of being written. Because the current segment file is full, the two new entries do not fit into the current segment any more and we need to trigger a merge-and-compact operation. Write down where all the entries end up after this operation by listing them in the correct order in the correct files. Use the two files on the right for your solution. **3 points**



2. The following figure depicts an LSM tree instances with three levels, which are C_0 , C_1 , and C_2 . In the LSM tree, we show only the keys and not their values. Assume that all values have the same size, so that a B⁺-tree leaf can hold exactly up to two key-value pairs and each SSTable can hold exactly up to three key-value pairs. The maximum depth of the B⁺-tree shall be two, which means that the depicted tree cannot grow any further; the number of SSTables in C_1 is fix, but we can, if necessary, add SSTables in C_2 .



We now insert a new element with key **9** into the LSM tree. Write the result of this insert operation into the template below. **3 points**



Task 6: Replication and Partitioning

1. Bob operates a leaderless replicated, distributed database on a cluster with 256 nodes. The database uses quorum reads and writes to ensure consistency and a gossip protocol to ensure that all updates will eventually spread to all nodes in the cluster. We assume that the gossip protocol has a perfect spreading technique, i.e., newly written data is always gossiped to nodes that do not already know it.

What read (r) and write (w) values does Bob need to define for the database's quorum $q(r,w)$ if writes should return as fast as possible (i.e., possibly small w) and every successfully written value should reach all cluster nodes in not more than 3 rounds of (perfect) gossiping? **4 points**

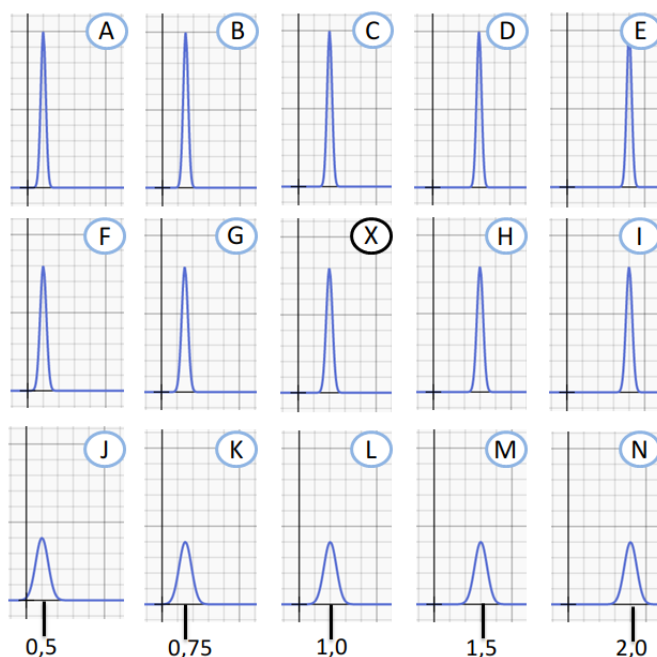
2. Consider a quorum $q(r,w)$ with $q(100,101)$ that was used in a leaderless replicated database system on a cluster with 200 nodes. Answer the four questions in the following matrix w.r.t. that quorum and briefly explain your answer: **4 points**

| | ... adding a node to the cluster? | ... removing a node from the cluster? |
|--|--|--|
| Do reads still work consistently when ... | | |
| Do writes still work consistently when ... | | |

Task 7: Distributed Systems

- The *Network Time Protocol* (NTP) can be used to calculate the time offset between a time server's reference clock and any client's local clock. Assume the round-trip time message has recorded the server times 16:42:00 and 16:42:02 and the client times 16:43:08 and 16:44:12 in the time format hh:mm:ss. What time offset would the NTP protocol calculate on the client w.r.t. this server? **2 points**

- The ϕ accrual failure detector method uses the probabilistic density curve over the heartbeats of a process to calculate a continuous suspicion value $\phi(t)$. We now focus on this probabilistic density estimation. Assume a distribution as shown in chart (X) below. We now accelerate the heartbeats from 1 beat/sec to 2 beat/sec. What curve shape can we expect when the sampling window contains (for a short while) both 1 and 2 beat/sec heartbeats and what will the curve look like with only 2 beat/sec heartbeats? Enter the according chart labels in the solution sequence next to the charts! **3 points**



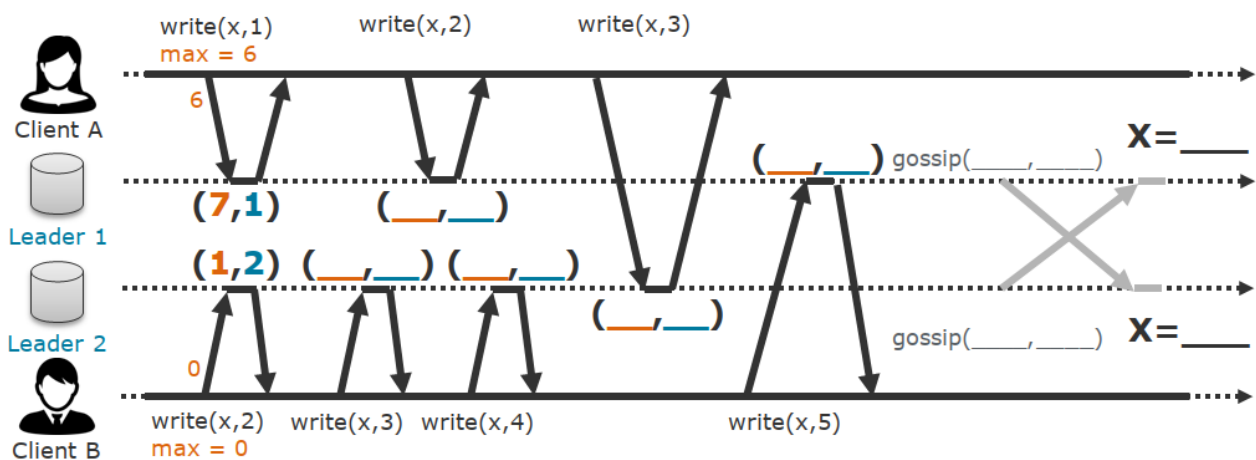
Solution:

X → _____ → _____

Task 8: Consistency and Consensus

- Linearizability is a consistency guarantee of eventual consistent databases stating that a read operation should always return the most recent value of an object although replicas might have older values. What three techniques are needed to make a field (or the entire database) linearizable in a leaderless replicated setup? Name the three techniques and very briefly note down, what each technique does (one or two sentences per technique). **3 points**

- Given a database management system that implements *Lamport Timestamps* for causal write ordering and a gossip protocol for eventual consistency. A Lamport Timestamp is a pair (c, i) with a write counter c and a node identifier i . Every write operation is associated with such a timestamp. A gossip call $gossip(f, v)$ sends the value v for field f to another node; (f, v) implicitly carries the current Lamport Timestamp (c, i) . Add the Lamport Timestamps and gossip values in the following example. What is the final value of field x on leader 1 and on leader 2? **4 points**



Task 9: Batch Processing

Suppose you are given three datasets: A `students` dataset that contains general information about students, a `courses` dataset that describes available courses for the students, and an `enrollment` dataset, which is basically a join table between `students` and their `courses`. The following code snippets read the three datasets into Spark Datasets:

```
val students = spark
  .read
  .option("quote", "\"")
  .option("delimiter", ",")
  .csv(s"data/students.csv")
  .toDF("ID", "Name", "Semester", "Supervisor")
  .as[(String, String, String, String)]
```

```
val enrollments = spark
  .read
  .option("quote", "\"")
  .option("delimiter", ",")
  .csv(s"data/enrollments.csv")
  .toDF("StudentID", "CourseID", "Credits")
  .as[(String, String, String)]
```

```
val courses = spark
  .read
  .option("quote", "\"")
  .option("delimiter", ",")
  .csv(s"data/courses.csv")
  .toDF("ID", "Title", "Teacher", "Topic")
  .as[(String, String, String, String)]
```

Use the three Datasets to solve the following tasks. You may use Spark's *Dataset* and/or *DataFrame* API but no SQL! Also have a look at the *Dataset* API documentation at the end of this task. If you are not sure about how a particular interface, call, or class works, make a good guess and provide a comment on how you *think* it works.

1. For administrative purposes, we need a list of all **students** that did not finish their studies in 10 semesters (value of **Semester** > 10) with their supervisors. Because this is very sensitive information, all student names need to be anonymized via **hash()** call on their **Name** values. The result should have the schema (**HashedName**, **Supervisor**) and be copied into a local variable in the driver program. Write a transformation pipeline that solves the request. **4 points**

2. The following SQL query searches for supervisors that supervise PhD students (**Semester** < 0), but never taught a course. Translate the SQL query into a Spark transformation pipeline that writes the result to the console of the driver. **5 points**

```
(SELECT Supervisor
 FROM students
 WHERE Semester < 0)
except
(SELECT Supervisor
 FROM students, courses
 WHERE Supervisor = Teacher)
```

Typed transformations

- ▶ `def as(alias: String): Dataset[T]`
Returns a new Dataset with an alias set.

- ▶ `def distinct(): Dataset[T]`
Returns a new Dataset that contains only the unique rows from this Dataset.

- ▶ `def except(other: Dataset[T]): Dataset[T]`
Returns a new Dataset containing rows in this Dataset but not in another Dataset.

- ▶ `def filter(func: FilterFunction[T]): Dataset[T]`
(Java-specific) Returns a new Dataset that only contains elements where `func` returns `true`.

- ▶ `def filter(func: (T) => Boolean): Dataset[T]`
(Scala-specific) Returns a new Dataset that only contains elements where `func` returns `true`.

- ▶ `def flatMap[U](f: FlatMapFunction[T, U], encoder: Encoder[U]): Dataset[U]`
(Java-specific) Returns a new Dataset by first applying a function to all elements of this Dataset, and then flattening the results.

- ▶ `def flatMap[U](func: (T) => TraversableOnce[U])(implicit arg0: Encoder[U]): Dataset[U]`
(Scala-specific) Returns a new Dataset by first applying a function to all elements of this Dataset, and then flattening the results.

- ▶ `def groupByKey[K](func: MapFunction[T, K], encoder: Encoder[K]): KeyValueGroupedDataset[K, T]`
(Java-specific) Returns a `KeyValueGroupedDataset` where the data is grouped by the given key `func`.

- ▶ `def groupByKey[K](func: (T) => K)(implicit arg0: Encoder[K]): KeyValueGroupedDataset[K, T]`
(Scala-specific) Returns a `KeyValueGroupedDataset` where the data is grouped by the given key `func`.

- ▶ `def intersect(other: Dataset[T]): Dataset[T]`
Returns a new Dataset containing rows only in both this Dataset and another Dataset.

- ▶ `def joinWith[U](other: Dataset[U], condition: Column): Dataset[(T, U)]`
Using inner equi-join to join this Dataset returning a `Tuple2` for each pair where `condition` evaluates to `true`.

- ▶ `def map[U](func: MapFunction[T, U], encoder: Encoder[U]): Dataset[U]`
(Java-specific) Returns a new Dataset that contains the result of applying `func` to each element.

- ▶ `def map[U](func: (T) => U)(implicit arg0: Encoder[U]): Dataset[U]`
(Scala-specific) Returns a new Dataset that contains the result of applying `func` to each element.

- ▶ `def sort(sortExprs: Column*): Dataset[T]`
Returns a new Dataset sorted by the given expressions.

- ▶ `def sort(sortCol: String, sortCols: String*): Dataset[T]`
Returns a new Dataset sorted by the specified column, all in ascending order.

- ▶ `def union(other: Dataset[T]): Dataset[T]`
Returns a new Dataset containing union of rows in this Dataset and another Dataset.

Untyped transformations

- ▶ `def col(colName: String): Column`
Selects column based on the column name and return it as a `Column`.

Actions

- ▶ `def collect(): Array[T]`
Returns an array that contains all rows in this Dataset.

- ▶ `def show(numRows: Int, truncate: Int): Unit`
Displays the Dataset in a tabular form.

- ▶ `def foreach(f: (T) => Unit): Unit`
Applies a function `f` to all rows.

- ▶ `def reduce(func: ReduceFunction[T]): T`
(Java-specific) Reduces the elements of this Dataset using the specified binary function.

- ▶ `def reduce(func: (T, T) => T): T`
(Scala-specific) Reduces the elements of this Dataset using the specified binary function.

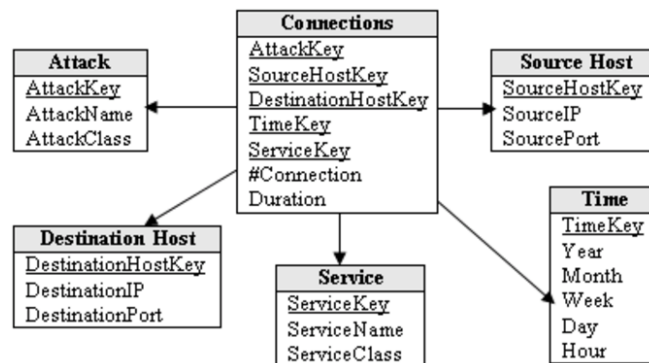
- ▶ `class KeyValueGroupedDataset[K, V] extends Serializable`

- ▶ `def mapGroups[U](f: MapGroupsFunction[K, V, U], encoder: Encoder[U]): Dataset[U]`
(Java-specific) Applies the given function to each group of data.

- ▶ `def mapGroups[U](f: (K, Iterator[V]) => U)(implicit arg0: Encoder[U]): Dataset[U]`
(Scala-specific) Applies the given function to each group of data.

Task 10: Distributed Database Management Systems

1. Given the following data warehouse *star schema*, relation cardinalities and join query, which join algorithm, i.e., normal join or star join has the smaller intermediate results? Calculate the sizes for the two algorithms and, then, state the superior algorithm in this setting! **3 points**



$((Connections \bowtie Attack) \bowtie Service) \bowtie Time$

Connections: 10 000 tuples

Attack: 100 tuples

Service: 100 tuples

Time: 100 tuples

2. Given the bitmap [1000000000010000]. Does it make sense to apply run-length encoding to this bitset before sending it over the network or not? Justify your answer! **2 points**

Extra page 1

Extra page 2