Chair of Data Science and Engineering
Department of Informatics
Technical University of Munich

Personal sticker

S5007

Compliance to the code of conduct
I hereby assure that I solve and submit this exam myself under my own name by only using the allowed tools listed below.

Signature or full name if no pen input available

Foundations in Data Engineering

Exam:	IN2326 / Retake	**Date:**	Monday 29th June, 2020
Examiner:	Prof. Dr. Thomas Neumann	**Time:**	11:15 – 12:45

Working instructions

- This exam consists of **12 pages** with a total of **7 problems**.
  Please make sure now that you received a complete copy of the exam.
- The total amount of achievable credits in this exam is 90 credits.
- Detaching pages from the exam is prohibited.
- Allowed resources:
  - any lecture or exercise material (this exam is open-book)
  - a **calculator**
  - a **dictionary** English ↔ native language
- **Please document the solution approach for your answers.** Give a reason for each answer unless explicitly stated otherwise in the respective subproblem.
- Do not write with red or green colors.

Left room from _____ to _____ / Early submission at _____

Problem 1 Estimations (10 credits)

0 1 2 3 4

a) The company NiceBikes Inc. wants to set up a server farm for to keep all their production records in a data lake. They estimate their overall data volume to be 15 PB.
Occasionally, an employee of NiceBikes Inc. searches for specific events in their production data. There are no index data structures in this data lake, so the requests will result in a complete scan of the data.
Estimate how many hard drives and servers are required so that each scan query can finish within 16 minutes.

0 1 2 3 4 5 6

b) A key-value store uses a radix tree to index the stored data. With the current data the radix tree has 6 levels. Each level uses one Byte from the key, thus a node of one level contains 256 pointers (of size 8 Bytes) to nodes of the next level.
Overall, the radix tree contains 10^6 entries and the whole index structure fits into main memory.
The workload on the radix tree consists of 10^5 key containment checks. Each containment check takes key of 6 Bytes and traverses the index structure to check whether there is a value for the given key. The value is not accessed.
Estimate how long a single containment check will take on average in this workload. Assume that the checked keys are uniformly distributed and every checked key is contained in the index.

Problem 2 Hash Tables (10 credits)

Regular hash-table implementations store keys and values in the hash table. For example, when inserting `"birthday" -> "4.8.1961"` the hash-table stores `("birthday", "4.8.1961")` as entry somewhere in the table.
An alternative idea is to store only a 128-bit hash of the key in the table instead of the key. Thus, instead of storing `("birthday", "4.8.1961")` the table would store `(3457230467245..., "4.8.1961")`, where 3457230467245... is the 128-bit hash of "birthday". According to the proponents of this idea, keeping only the hash and discarding the original value is o.k., because a hash collision is highly improbable for practical applications.

a) Give two advantages of the alternative idea and give two arguments against it.

0 1 2 3 4

b) For practical applications, how big is the risk to get a wrong answer from the hash-table due to a collision of 128-bit hashes?
Give an estimate for the risk (e.g., by providing an upper bound for the collision risk with a hash table of 10^{12} entries).

0 1 2 3 4 5 6

Problem 3 SQL I (22 credits)

The Allgau windpark has hundrets of wind turbines. For each turbine they log the produced power at regular intervals. The logs are kept in the table **output**:

Turbine ID	*Timestamp of measurement*	*power produced since last measurement (in kJ)*
id	timestamp	output
23	1593174178	100000
3	1593174178	20000
23	1593174179	100000
...	...	...

They also log how much energy they must invest to keep each turbine operational in the table **costs**:

Turbine ID	*Timestamp of measurement*	*power used since last measurement (in kJ)*
id	timestamp	invested_power
23	1493474178	1000000
3	1528672094	4500
...	...	...

Give SQL queries to answer the following questions on the given tables.

0 1 2 3 4

a) Find the turbine with the most power output. In case of a tie list all turbines with most power output.

0 1 2 3 4

b) Find the most profitable turbine (consider the invested power).

c) Find the break-even point of each turbine. That is, find the first point in time of each turbine when the produced energy exceeds the invested energy.

0 1 2 3 4 5 6

d) Identify the 5 last periods of low yield for each turbine. A log entry classifies as low yield when the power output is below 250 kJ. A period of low yield are one or more consecutive low yield log entries (consecutive by timestamp).

0 1 2 3 4 5 6 7 8

Page empty

Problem 4 SQL II (12 credits)

The table **exchange** contains many currency exchange rates:

From	To	Rate
EUR	GBP	0.91
GBP	USD	1.24
USD	EUR	0.89
...	...	...

Give SQL queries to answer the following questions on the given table.

0 1 2 3 4

a) Which currencies exist for which you can exchange money into the currency, but not exchange from the currency to any other?

0 1 2 3 4 5 6 7 8

b) Are there any cycles of money exchanges through which you can generate a profit by trading through the cycle? Make sure that your query terminates.

Problem 5 Map-Reduce (14 credits)

You are given a data set *S* of shipping container travel information. Every row denotes that a specific container either arrived at or left a port. These columns are available:

- ContainerID (Id of shipping container)
- Port (Name of the port)
- RecordType (Departure or Arrival)
- Timestamp (Time of departure or arrival)

Write Map-Reduce pseudo code to answer the following questions.

a) Find decommissioned containers (not used in the last year).

0 1 2 3 4

b) Find containers that have been to all ports.

0 1 2 3 4 5

0 1 2 3 4 5

c) Find ports that send out more containers than they receive.

Problem 6 Message Passing (14 credits)

YouOk? is a social network service where users can update their status and check on the status of their friends. Overall, users can perform one of two actions:

1. View a list of status updates of their friends from the last 2 hours.
2. Update their own status.

As often observed in social networks, there is a power-law distribution of how many friends each user has. Most users have a few up to a moderate amount of friends. A few users, however, have many friends, up to hundreds of thousands.
Processing the status updates poses some challenges for *YouOk?*. Their service is so popular that they must use multiple servers to handle the workload. Furthermore, their current implementation is inefficient. It keeps a **list of status** updates for every user. Then, whenever a user updates their status, it updates the status list of all friends with the new status. Unfortunately, this causes a lot of random access on the status lists (which is resource intensive). What makes the situation worse is that most of the status updates are not even read by the friends because the update is only valid for 2 hours and most users are only active once per day.

a) Design a data partitioning scheme to organize users' friend information, status, and friend status list for all users over multiple servers. Take care that for the user actions 1) and 2) the amount of communication to other servers is minimized. Shortly explain why your scheme minimizes communication.

0 1 2 3 4

b) Devise an algorithm that makes the processing of status updates more resource efficient.
Assume that sending a status update from a user to their friends costs a fixed amount x of compute resources. Similarly, retrieving the status update of a user's friend costs the same amount x.
Your processing scheme should improve resource consumption that is caused by

- random access when sending updates to friends,
- and wasted effort for updates that are never read.

However, make sure that when a user wants to see a list of friends' status, this must be delivered reasonably quickly. After all, users hate to wait. **Hint:** Therefore, fetching all the friend's status updates when the list is requested takes too long.

0 1 2 3 4 5 6 7 8 9 10

Problem 7 RDF (8 credits)

0 1 2 3 4 5

a) Give RDF triples for the following facts:

- Mountainbikes have gear shifters, brakes, and handle bars.
- Mountainbikes are bikes.
- A bike is a vehicle.
- Other vehicles are boats and skateboards.

Make sure that for any concept your triples always use the same identifier (component of a triple).

0 1 2 3

b) Write a Sparql query that lists all vehicles. Also list relations recursively.

Additional space for solutions–clearly mark the (sub)problem your answers are related to and strike out invalid solutions.