

Group A

Please fill in your name and registration number (Matrikelnr.) **immediately**.

Exam 1 ON		SOLUTION KEY		04.10.2019
		Advanced Database Systems (184.780)		GROUP A
Matrikelnr.	Last Name		First Name	

Duration: 75 minutes. Provide the solutions on the designated pages. **Good Luck!**

Question 1:

(4)

1. Consider the natural join $e \bowtie R$, where R is a table of the database and e is itself a relational algebra expression yielding some relation. Then, by appropriate choice of inner and outer relation, it is always possible to evaluate $e \bowtie R$ by a block nested loop join without materializing the result of the evaluation of e .
2. Consider relations $R(AB)$ and $S(BC)$ and suppose that the natural join $R \bowtie S$ is evaluated by a hash join. Then it is guaranteed (for any contents of the relations R and S) that this computation can be done in linear time, i.e., in time $O(\|R\| + \|S\|)$, where we write $\|T\|$ to denote the size (= number of bits) of a table T .
3. Consider again relations $R(AB)$ and $S(BC)$. Then the following equivalence of relational algebra expressions holds: $\pi_{AC}[R \bowtie S] = \pi_{AC}[(S \bowtie R) \times (R \bowtie S)]$.
4. Consider once more relations $R(AB)$ and $S(BC)$. Then the following equivalence of relational algebra expressions holds: $\sigma_{A=1 \wedge C=2}(R \bowtie S) = \sigma_{A=1}(R) \bowtie \sigma_{C=2}(S)$.

Question 2:

Suppose that the database of a zoo contains the following relations:

Animals(Id, Species, Origin, DateOfBirth, Price) (short a),

Zookeepers(Id, Name, Age, EntryDate) (short z) and

Care(Id, AId, ZId, Begin, End) (short c),

where Care.AId and Care.ZId are foreign keys into the animals and zookeeper relations, respectively. Further, suppose that the following query has to be processed:

```
select a.Origin, z.Name, c.Begin, c.End
```

```
from Animals a, Zookeepers z, Care c
```

```
where a.Species = 'tiger' and z.Age < 25 and c.AId = a.Id and c.ZId = z.Id
```

i.e., we are interested in information on young zoo keepers how have taken care of a tiger.

For the size of the relations, assume that Animals has 10,000 tuples, Zookeepers has 40 tuples and Care has 20,000 tuples. Moreover, assume that the predicate “ $z.Age < 25$ ” has selectivity 0.15 and the predicate “ $z.Species = 'tiger'$ ” has selectivity 0.001.

(a) In the first box below, draw the query tree of the canonical translation. To save space, you may use the abbreviations a , z , and c , for the relations Animals, Zookeeper, and Care, respectively. You may also abbreviate any attribute name to a unique prefix (e.g., $a.O$ for Animal.Origin).

(b) In the second box below, draw the query tree of the optimized Relational Algebra expression. For the optimization, apply the following rules:

- Push selections as deep as possible,
- project out attributes as soon as they are not needed anymore,
- replace cross products by joins,
- choose the join order in such a way that the first join is expected to yield the smallest possible result (applying the uniformity assumption to the distribution of attribute values).

(a) Canonical Translation:

(b) Optimized Expression:

Question 3:

(3)

Consider a magnetic disk with the following characteristics:

- Average seek time: 7ms
- Track-to-track seek time: 0.5ms
- Transfer time (for 1 block): 2ms
- Rotational speed: 6000 rpm
- Block size: 12 000 byte

Assume a file **payments** with 200 000 fixed-length records stored on the disk in an unspanned organization.

a) Your application requires that your DBMS is able to read the whole **payments** file in a fixed time limit. Calculate the maximal size of a single record of **payments** such that the table can be read in at most 7 seconds. *You can make no assumptions on the layout of its blocks on the disk.*

b) You've switched the storage of your DBMS to a SSD with the following characteristics:

- Transfer rate 128 MB/s ($128 \cdot 10^6$ bytes/s)
- Data is read in 64kB ($64 \cdot 10^3$ bytes) blocks

Furthermore, you have configured your DBMS such that the **payments** file is stored on consecutive blocks. Calculate the maximal size of a single record of **payments** such that the whole table can be read in at most 1 second.

Question 4:

(2)

a) Your DBMS maintains equi-depth histograms to support selectivity calculation for query planning. In particular, for the column `credits` of `integers` of a table `students` with 1200 rows, the following 5 values divide the column values into 6 groups of equal size: 8, 14, 72, 80, 210. Furthermore, the maximum value present in the column `credits` is known to be 300. **Assume that the dividing values in the histogram are always part of the left bucket, i.e., the value 8 is in the first bucket, 14 in the second, and so on.**

Estimate the selectivity for the following two predicates as accurately as possible. Assume that values are evenly distributed inside the buckets.

i) `credits > 270`

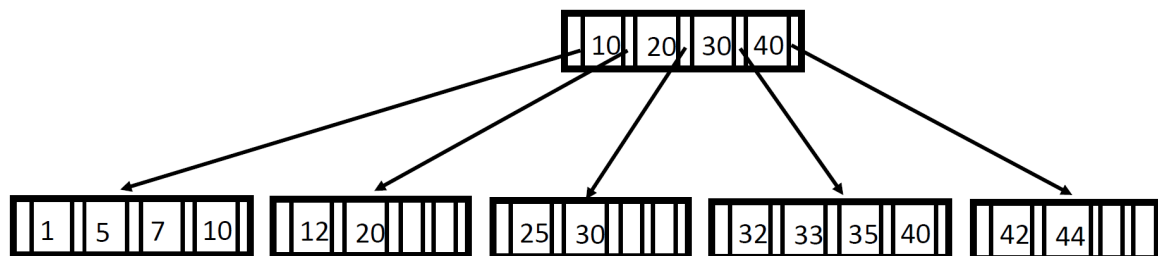
ii) `credits ≤ 74`

Question 5:

(3)

Consider the following B^+ -tree index, which is used to organize integer key values.

B^+ -tree T_0 :



Assume the following characteristics for this B^+ -tree index:

- In all nodes of the B^+ -tree except for the root, there are ≥ 2 and ≤ 4 key values. The root may have ≥ 1 and ≤ 4 key values.
- For the navigation in the B^+ -tree, suppose that a node in the B^+ -tree contains the sequence K_{i-1}, P_i, K_i of key value K_{i-1} , pointer P_i , and key value K_i . Then P_i points to a subtree where all key values v satisfy the condition $K_{i-1} < v \leq K_i$.
- To keep things simple, we ignore horizontal pointers.

a) (2 credits)

Suppose that, in the B^+ -tree T_0 , the key value 8 is inserted. Display the resulting B^+ -tree T_a . Make sure that the above assumption on the number of key values in each node still holds.

B^+ -tree T_a :

b) (1 credit)

Suppose that, in the B^+ -tree T_0 , the key value 25 is deleted. Display the resulting B^+ -tree index T_b . Again, make sure that the above assumption on the number of key values in each node still holds.

B^+ -tree T_b :

Question 6:

(4)

A friend of yours is writing their own database management system. You are curious about how they handle large queries with many joins and ask how their system determines the order in which the joins are computed **for optimal performance**. Your friend tells you that the order doesn't matter because joins are associative and commutative.

Do you agree or disagree? If you agree give a supporting argument. If you disagree provide an example that illustrates why your friend is wrong.

Overall: 20 points