Department of Computer and Information Science
Database and Information Systems
Prof. Dr. Michael Grossniklaus

Universität
Konstanz

Winter Semester 2017/2018

INF-20210

Database System Architecture and Implementation

Exam, February 19, 2018, 14:00–16:00

Your Student Number

Read the following instructions *before* you start to write your answers.

- The examination consists of 4 questions each of which is worth a total of **25 points**.
- Answers can be given in **German** or **English**.
- You should start the answer to each question on **a new page**.
- Do **not** write in the margins.
- Your answers should be written in **blue** or **black ink** and not in pencil. Any parts of your answer that you do not want to be included as part of the final answer should be **clearly scored through** with your pen.
- With the exception of a calculator, **no electronic devices** (laptops, phones, music players, etc.) can be used during the exam.

Do not write below this line

1	2	3	4	Total	Exam	Course	Final

1 Disk, File, and Buffer Management

(a) **Hard Disks.** Consider a disk with a sector size of 512 bytes, 16,383 tracks per surface, 63 sectors per track, 4 platters, 8 heads, and an average seek time of 8.5 ms. The track-to-track seek time is 0.5 ms. (7 points)

(i) What is the capacity of a track in bytes?

. . . .

(ii) What is the capacity of each surface?

. . . .

(iii) What is the capacity of the disk?

. . . .

(iv) How many cylinders does the disk have?

. . . .

(v) Which of the following are valid block sizes?

☐ 128 bytes
☐ 8,192 bytes
☐ 1,893 bytes
☐ 3,584 bytes

(vi) If the disk rotates at 15,000 rpm (revolutions per minute), what is the maximum rotational delay?

. . . .

(vii) If one track of data can be transferred per revolution, what is the transfer rate?

. . . .

(b) **RAID Systems.**

(i) What are characteristics of RAID levels 1+0, 3, and 6? (3 points)

. . . .

. . . .

. . . .

. . . .

. . . .

(ii) RAID systems guarantee high data stability by storing so-called *parity bits*. Let us assume a RAID 4 system with four data disks and one parity disk. Which bits are stored on the parity disk for the following bit sequences? (2 points)

Data Disk 1:	`00000000`
Data Disk 2:	`10010010`
Data Disk 3:	`10101010`
Data Disk 4:	`00110011`
Parity Disk:	. . . .

(iii) Assume we have 10 disks with 980 GB each. How much disk space is available if RAID 5 is used? (2 points)

☐ 8.2 TB
☐ 9.8 TB
☐ 10.0 TB

(iv) Suppose that the mean time to failure of the disks used in the above RAID system is 5 million hours. What is the overall mean time to failure of the RAID system? (2 points)

. . . .

. . . .

(c) **Buffer Management.**

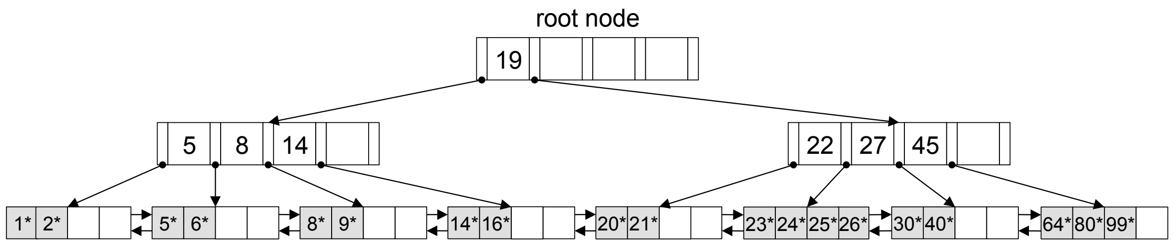
(i) Assume the existence of 4 buffer pages (1-4) and 6 data blocks (A-F). Initially, all buffer pages are free. Which pages are chosen as victims in **each step** by the **LRU** (Least Recently Used) algorithm, when loading the following sequence of blocks? (5 points)

A F B D F E B D C B C D E F A C E C

(ii) Sketch an example of a **worst case** for the LRU buffer replacement policy. Which alternative policy would you recommend, and why? (4 points)

2 Hash and Tree-based Indexes

(a) **B$^+$ Tree.** Given below is a B$^+$ tree with order $m = 2$. For each of the following operations, sketch the resulting tree. Details of the original trees can be omitted as long as all changed nodes are shown. Each operation is to be performed on the **original tree**. (9 points)



(i) Entry **22*** is inserted and siblings are checked for redistribution.

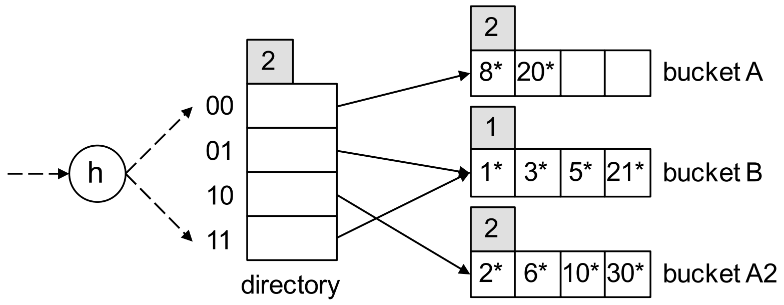
(ii) Entry **22*** is inserted without checking siblings for redistribution.

(iii) Entry **5*** is deleted with checking siblings for redistribution.

Answer

(b) **Extendible Hashing.** Given below is an extendible hashing index with global depth $d = 2$. For each of the following two operation sequences, sketch the resulting index. Each operation sequence is to be performed on the **original index**.

(8 points)

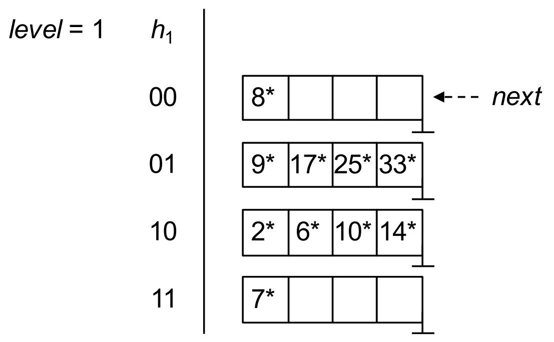


(i) Entries **7*** (`0b111`) and **15*** (`0b1111`) are inserted.

(ii) Entries **8*** (`0b1000`) and **20*** (`0b10100`) are deleted.

Answer

(c) **Linear Hashing.** Given below is a linear hashing index. Assume that the index splits whenever an overflow page is created. For each of the following two operation sequences, sketch the resulting index. Each operation sequence is to be performed on the **original index**. (8 points)



(i) Entries **41*** (`0b101001`) and **18*** (`0b10010`) are inserted (in this order).

(ii) Entries **7*** (`0b111`) and **8*** (`0b1000`) are deleted (in this order).

Answer

3 Query Evaluation

(a) **Sort Operator.** Assume that a query evaluator has to sort a relation that is stored in a heap file consisting of 2,000,000 pages with 125 records each. The buffer pool consists of 1500 pages. Calculate the I/O costs for the following evaluation strategies. (15 points)

(i) two-way external sort

(ii) general external sort (without refinements such as blocked I/O, double buffering, and replacement sort)

(iii) B^+ tree index (fan-out 1000) with the search key being the same as the desired sort key

- Variant 1 index entries: $\langle k, \langle \dots, \mathtt{A} = k, \dots \rangle\rangle$
- Variant 2 index entries: $\langle k, rid \rangle$ (unclustered)

Note: You can compute the worst-case cost for the unclustered index.

Answer

(b) **Join Algorithms.** Suppose a database contains a relation **R** and a relation **S**. There exists a key-foreign key relationship between **S**.*k* (key) and **R**.*fk* (foreign key). Relation **R** has 15,000 pages with 250 tuples each, whereas relation **S** has 9,000 pages with 150 tuples each. The query processor has 3,000 buffer pages available. Assuming 3 ms access time per page (I/O cost) and 10 μs to construct an output tuple (CPU cost), compute the total time required to evaluate

$$\mathbf{R} \bowtie_{k=fk} \mathbf{S}$$

for the following three join algorithms. For each join, you may choose which of the two relations is the outer and which is the inner relation. (10 points)

(i) nested-loops join

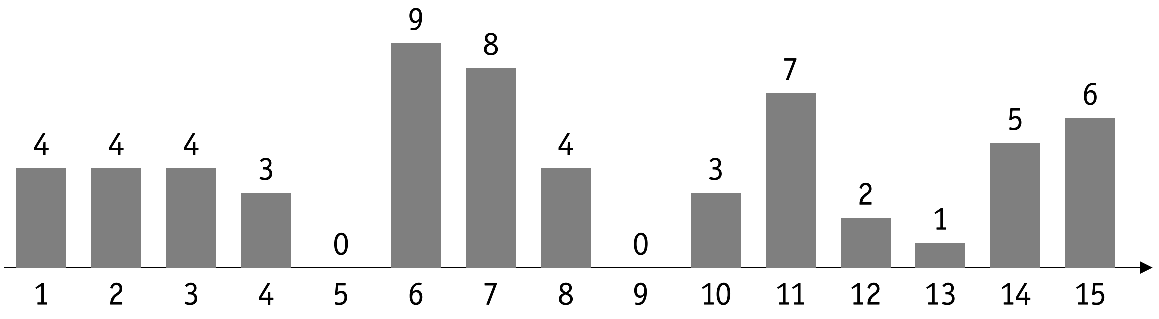
(ii) block nested-loops join (that uses an in-memory hash-table on the outer input)

(iii) index nested-loops join (assuming a clustered hash index on **S**.*k*)

Answer

4 Query Optimization

(a) **Histograms.** Assume a column **R**.*a* of SQL type `INTEGER` (domain $\{\ldots, -2, -1, 0, 1, 2, \ldots\}$) with the following actual non-uniform value distribution in a relation **R**. (10 points)



(i) Create an **equi-width histogram** for **R**.*a* with width 5.

(ii) Create an **equi-depth histogram** for **R**.*a* with depth 12.

(iii) Estimate the result cardinality of the following queries based on these two types of histograms. Compare them to the estimated cardinality under the **uniformity assumption** as well as to the **actual value**.

- `SELECT * FROM R WHERE a = 10`
- `SELECT * FROM R WHERE a < 10`
- `SELECT * FROM R WHERE a BETWEEN 10 AND 15`

Answer ______________________________

(b) **Heuristic and Cost-based Optimization.** Consider the following schema of an online auction platform, which is used to represent information about users, items, and bids. (15 points)

User($\underline{\textit{uid}\text{: integer}}$, *name*: char(50), *price_limit*: float)

Item($\underline{\textit{iid}\text{: integer}}$, *description*: char(500), *closing_time*: date, *start_price*: float)

Bid($\underline{\textit{uid}\text{: integer, }\textit{iid}\text{: integer, }\textit{bid\_time}\text{: date}}$, *amount*: float)

Now also consider the following query.

```
SELECT U.name
  FROM User U, Bid B, Item I
 WHERE U.uid = B.uid AND B.iid = I.iid AND
       I.closing_time > NOW() AND I.start_price <= 10.0 AND
       U.price_limit >= B.amount
```

(i) Draw a **relational algebra tree** for the above query.

(ii) List all **join orders** in which pairs of relations can be joined to compute the query result. Limit this set of join orders to those that a typical relational query optimizer would consider, i.e., only left-deep plans that do not require the computation of cross-products.

(iii) Based on your relational algebra tree, identify one **query evaluation plan** that reflects the order of operations, the access methods, and physical operators, which a decent query optimizer would choose. Your query evaluation plan does **not** have to be the optimal plan.

(iv) In your query evaluation plan, identify both an example of **heuristic optimization** and of **cost-based optimization**.

(v) Rewrite the nested SQL query below to **canonical form**.

```
SELECT I.iid, I.start_price FROM Item I
 WHERE I.closing_time > NOW() AND
       I.uid IN (SELECT U.uid FROM User U
                  WHERE U.price_limit >= I.start_price) AND
       I.uid IN (SELECT B.uid FROM Bid B WHERE B.iid = I.iid)
```

Answer

Answer ______________________________