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

Databases II — Exam
Winter Semester 2025/26 — 24.04.2026

Name: _____ **Student ID:** _____

Hints

- Check whether you received all pages of the exam (11 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.
- **Multiple-Choice**: Wrong answers result in point deductions!
- 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	9	10	Sum
Total points	1	1	1	1	1	1	1	1	1	1	10
Points reached											

Exercise 1 - *Buffer Management*. 1 Point

Consider a system with a (initially empty) buffer of size 4 (i.e., the buffer can store 4 pages). **LRU** (least recently used) is used as the replacement strategy. Consider the following page access pattern of the pages A,B,C,D,E:

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

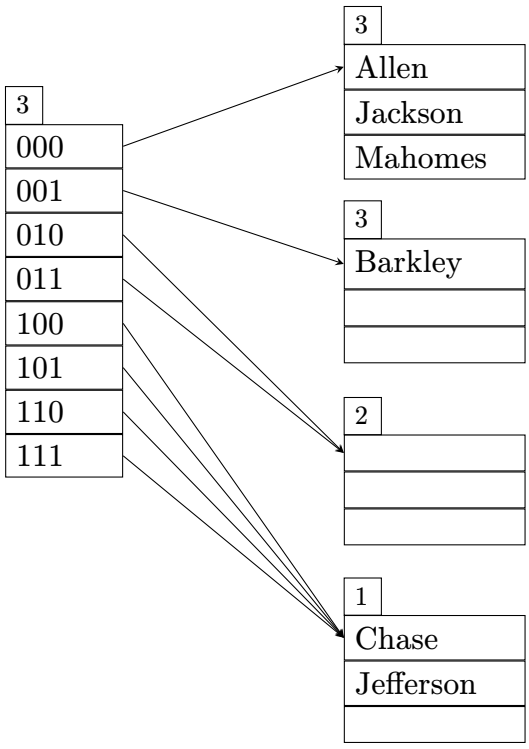
What is the number of disk accesses? Which pages are stored in the buffer after the sequence of page accesses? Complete the following table. Buffer hits are marked with "*".

buffer page 1	A												
buffer page 2		B											
buffer page 3			D	*									
buffer page 4													

Exercise 2 - *Extendible Hashing*. 1 Point

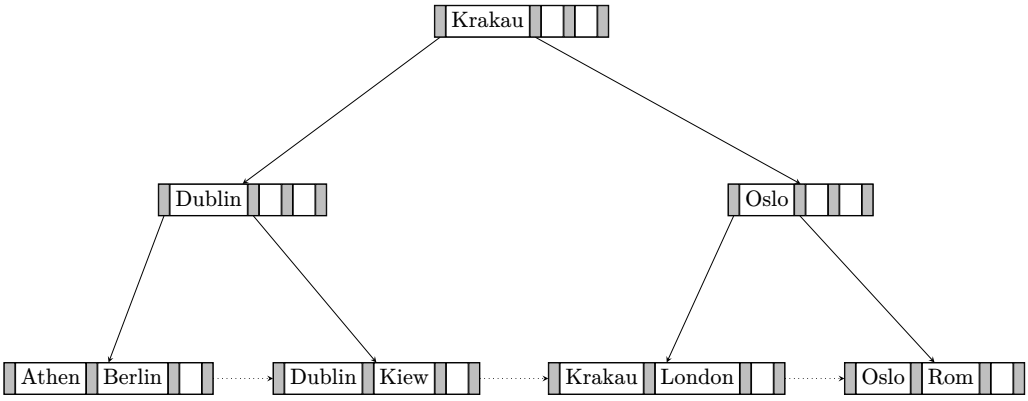
The hash function $h(x)$ returns the binary values as shown in the table. The tuple **Jackson** should be deleted from the hash container. A bucket in the hash container stores up to 3 tuples. The container should be as small as possible. **Illustrate the resulting hash container**.

x	$h(x)$
Allen	00001
Barkley	00101
Chase	10010
Henry	01101
Jackson	00010
Jefferson	10101
Mahomes	00011



Exercise 3 - *B⁺-Tree Deletion*. 1 Point

Given the following B⁺-tree with $m = 4$. Draw the B⁺-tree after deleting the key **Krakau**.



Exercise 4 - *Duplicates and Indexes.* 1 Point

Consider the following (unsorted) relation $R[name]$ with duplicate entries.

a) Draw a **single-level, non-clustered index** on attribute *name* and resolve duplicates using **buckets.** *Hint:* Although there are duplicates in the relation, there must not be duplicate search keys in the index.

b) Duplicates can also be resolved using **tuple identifiers (TID).** Explain what a **TID** is, and **how to use TIDs** to avoid duplicate search keys in an index.

$R[name]$:

Noah
George
Noah
Ivy
Harry
Noah
George
Lily
Ivy
Florence
Noah
George

Exercise 5 - *Bitmap Index Scan.* 1 Point

Given relation $R[A, B]$ with the following properties:

- $|R| = 10^6$ tuples are stored on $b_R = 10^3$ blocks,
- non-clustering B^+-tree index on attribute B, $m = 1024$, and maximum height (i.e., each node is minimally occupied),
- duplicates are resolved via tuple identifiers (TID),
- the value 2500 on attribute B is stored 10^3 times in total and distributed among 100 data blocks.

The following query should be answered:

$$\sigma_{B=2500}(R)$$

Determine the number of **data block accesses** in the **worst case**

a. **without using** a bitmap index scan **(0.5 points)**,
b. **by using** a bitmap index scan **(0.5 points)**.

Exercise 6 - *Join Algorithms.* 1 Point

Which join algorithm (**Merge Join**, **Index Nested Loop Join**) provides the minimal costs in the given scenario? State the **costs** for both **algorithms**.

Compute the **natural join** between relations $R[A, B]$ and $S[B, C]$ with $|R| = 1000$ and $|S| = 12000$ tuples. $R.A$ and $S.B$ are key attributes. The relations are stored on $b_R = 250$ and $b_S = 2000$ consecutive blocks, respectively. There is a **sparse** B^+-tree index on $S.B$, where each node in the B^+-tree stores 20 keys. Assume that sorting b blocks requires $\lceil b \cdot \log_2(b) \rceil$ block accesses.

Exercise 7 - *Efficient Query Processing*. 1 Point

Consider a relation $R[A, B, C]$ with the following properties:

- $|R| = 2.000.000$ tuples.
- Each block of data holds 400 tuples.
- Attribute A holds integer values in the range $[1; 2.000.000]$.
- Attribute B holds integer values in the range $[400.001; 500.000]$.
- Attribute C holds integer values in the range $[100.001; 1.000.000]$.
- The following indexes exist:
  - Sparse B$^+$-tree index on attribute A, $m = 2^8 = 256$, minimal height,
  - dense B$^+$-tree index on attribute B, $m = 2^8 = 256$, minimal height,
  - dense B$^+$-tree index on attribute C, $m = 2^8 = 256$, minimal height.

The following query is to be processed:

$$\sigma_{A>1.800.000 \wedge B=450.000}(R)$$

State the used **strategy (0.5 points)** and calculate the **number of block accesses (0.5 points)** to process the query **in the most efficient way** (accessing 1 node in the B$^+$-tree corresponds to 1 block access).

Exercise 8 - *Join Size Estimation*. 1 Point

Consider the following 3 relations $R[A, B, C]$, $S[A, D, E]$, $T[C, D, F]$ and their properties:

- $|R[A, B, C]| = 1000$ tuples, $V(R, A) = 100$, $V(R, B) = 200$, $V(R, C) = 300$
- $|S[A, D, E]| = 4000$ tuples, $V(S, A) = 50$, $V(S, D) = 200$, $V(S, E) = 300$
- $|T[C, D, F]| = 2000$ tuples, $V(T, C) = 100$, $V(T, D) = 200$, $V(T, F) = 600$

Attribute values are assumed to be distributed uniformly and independently. Estimate the size of the following query. $(\sigma_{A=100 \vee A=200}(R) \neq \emptyset)$.

$$(\sigma_{A=100 \vee A=200}(R)) \bowtie S \bowtie T$$

Exercise 9 — 1 Point

Consider the following schedule.

(a) Which transaction must abort to trigger a **cascading rollback** of all other transactions?

(b) Show the commit order (by inserting the commit commands into the schedule) such that the schedule is **cascadeless**.

T1:	T2:	T3:	T4:
	read(B)		
read(D)			
	write(A)		
			read(A)
			write(C)
		read(E)	
read(C)			
write(B)			
		read(B)	

Exercise 10 — 1 Point

Consider the following schedule and the **two-phase locking** scheduler.

(a) Assume that all transactions want to issue a `write` operation on item `A` in the following order: `T3`, `T2`, `T1`. Complete the schedule by inserting lock requests (e.g., `R:lock-S(A)`) and granted locks (e.g., `G:lock-S(A)`) for all remaining `read` and `write` operations. Denote in the schedule if a transaction has to wait since another transaction holds an incompatible lock.

(b) Draw the wait-for graph. Does the schedule result in a deadlock? Explain your answer.

```
T1:          T2:          T3:
_____________________________________
R:lock-S(A)
G:lock-S(A)
read(A)
_____________________________________


             read(A)
_____________________________________


_____________________________________


_____________________________________


_____________________________________
```