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 2024/25 — 26.2.2025

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.
- 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). **MRU** (most recently used) is used as the replacement strategy. Consider the following page access pattern of the pages A,B,C,D,E:

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

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 "*".

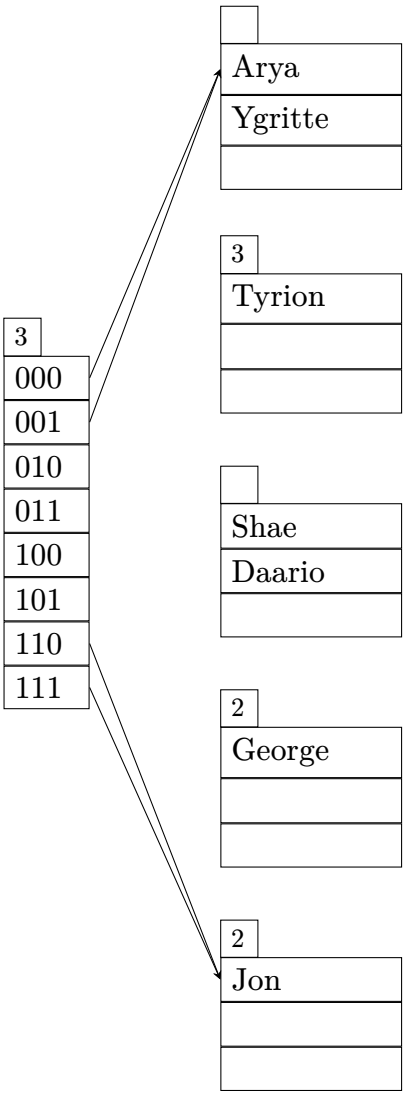
buffer page 1	E												
buffer page 2		A	*										
buffer page 3				D									
buffer page 4													

Exercise 2 - *Extendible Hashing*. 1 Point

The hash function $h(x)$ returns the binary values as shown in the table. **Fill in the missing values and components (global/local depth, pointer).**

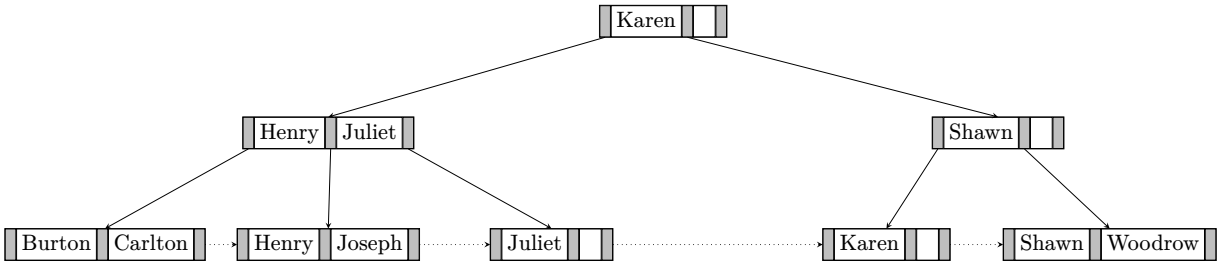
Hash table:

Name (x)	$h(x)$
Arya	0010
Cersei	0100
Daario	0110
George	1000
Jon	1100
Melisandre	1110
Robb	1011
Shae	0111
Tyrion	0101
Ygritte	0001



Exercise 3 - *B^+-Tree Insertion*. 1 Point

Given the following B^+-tree with $m = 3$. Draw the B^+-tree after inserting the key **Isaac**.



Exercise 4 - *Indexes with Composite Search Keys*. 1 Point

Consider an ordered composite index on the attributes *(LanguageFamily, Language, NumSpeaker)* in that order. State whether the index structure can be used to efficiently answer the following query predicates. Explain your answers briefly.

1. **WHERE** Language="Spanish" **AND** LanguageFamily="Romance"
2. **WHERE** Language="Mandarin" **AND** 10^7<NumSpeaker<10^9
3. **WHERE** LanguageFamily="Germanic" **AND** Language="English" **AND** 10^6<NumSpeaker<10^7
4. **WHERE** LanguageFamily="Slavic"

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. The relations are stored on $b_R = 250$ and $b_S = 2000$ consecutive blocks. There is a **sparse** B^+-tree index on $S.B$, where each node in the B^+-tree stores 20 keys. No index on $R.A$ is available. 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 $R[A]$. There is a **sparse $\mathbf{B^+}$-tree index** of attribute $R.A$ What is the **most efficient strategy** for processing **range queries** of the following type?

$$\sigma_{a<A<b}(R)$$

Describe **all necessary steps**.

Exercise 8 - *Join Size Estimation*. 1 Point

Consider the following 3 relations $R[A, B, C]$, $S[A, D, E]$, $T[D, E, F]$ 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, $V(S, A) = 50$, $V(S, D) = 200$, $V(S, E) = 300$
- $|T[D, E, F]| = 2000$ tuples, $V(T, D) = 200$, $V(T, E) = 400$, $V(T, F) = 600$

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

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

Exercise 9 — 1 Point

Consider the following schedule. State whether the schedule is

1. conflict serializable,
2. recoverable, and
3. cascadeless.

Explain your answer!

T1:	T2:	T3:
write(A)		
	read(D)	
		write(A)
		write(D)
write(D)		
commit		
	write(C)	
		read(A)
		commit
	commit	

Exercise 10 — 1 Point

Can the following schedule be the output of a **strict two-phase locking** scheduler? If yes, add all required lock/unlock instructions. Otherwise, explain why.

```
T1:         T2:         T3:
──────────────────────────────
            read(A)
            read(B)

                        read(A)
                        write(A)
            write(B)
            COMMIT

                        read(A)
                        read(B)
read(B)
read(A)
                        COMMIT
COMMIT
──────────────────────────────
```