Fachbereich für Computerwissenschaften

Prof. Dr. Nikolaus Augsten
Jakob-Haringer-Str. 2
5020 Salzburg, Austria
Telefon: +43 662 8044 6347
E-Mail: nikolaus.augsten@sbg.ac.at

PARIS LODRON UNIVERSITÄT SALZBURG

Datenbanken II — Prüfung
Wintersemester 2022/2023 — 31.01.2023

Name: _____ **Matrikelnummer:** _____

Hinweise

- Bitte überprüfen Sie die Vollständigkeit des Prüfungsbogens (11 nummerierte Seiten).
- Schreiben Sie Ihren Namen und Ihre Matrikelnummer auf jedes Blatt des Prüfungsbogens und geben Sie alle Blätter ab.
- Grundsätzlich sollten Sie alle Antworten auf den Prüfungsbogen schreiben.
- Sollten Sie mehr Platz für eine Antwort benötigen, bitte einen klaren Verweis neben die Frage auf die Seitennummer des zusätzlichen Blattes setzen.
- Keinen Bleistift verwenden. Keinen roten Stift verwenden.
- Verwenden Sie die Notation und die Lösungsansätze, die während der Vorlesung besprochen wurden.
- Aufgaben mit mehr als einer Lösung werden nicht bewertet.
- Als Unterlage ist ein beliebig (auch beidseitig) beschriftetes A4-Blatt erlaubt.
- Zeit für die Prüfung: 90 Minuten

Unterschrift _____

Korrekturabschnitt — Bitte frei lassen

Aufgabe	1	2	3	4	5	6	7	8	9	10	Summe
Maximale Punkte	1	1	1	1	1	1	1	1	1	1	10
Erreichte Punkte											

Aufgabe 1 - *Slotted Page*. 1 Punkt

Gegeben sei eine Slotted Page mit folgenden Eigenschaften:

- Größe: $2^{11} = 2048$ Bytes,
- Adressierungstyp: **Byte-Adressierung** (es kann jedes Byte adressiert werden)

In dieser Slotted Page werden **3 Tupel** O, P, Q gespeichert:

- d_1: $|O| = 32$ Bytes
- d_2: $|P| = 64$ Bytes
- d_3: $|Q| = 128$ Bytes

Ergänzen Sie die Slotted Page um die **fehlenden Werte/Adressen** (numerische Werte erwartet, Pfeile reichen nicht aus), wobei p_i und g_i sich auf den jeweiligen Datensatz d_i beziehen. **(0.125 Punkte pro Wert/Adresse)**

a	f	g_3	p_3	g_2	p_2	g_1	p_1	$\ldots$	d_1	d_2	d_3
									O	P	Q

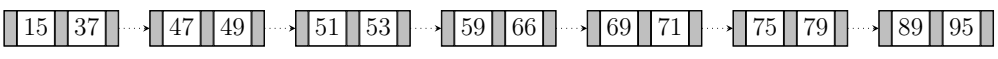
Aufgabe 2 - *Indexstrukturen.* 1 Punkt

Zeichnen Sie für die folgende Tabelle einen **3-stufigen Sekundärindex** auf dem Attribut `Tier`. Die **erste Indexstufe** soll **dense** und die **zweite und dritte Indexstufe** sollen **sparse** sein. In einen **Indexblock** können **3 Einträge** gespeichert werden.

`Tier`	`Hab.`
Vogel	Luft
Leopard	Land
Pfau	Land
Katze	Land
Bär	Land
Affe	Land
Koala	Land
Orca	Wasser
Delfin	Wasser
Wal	Wasser

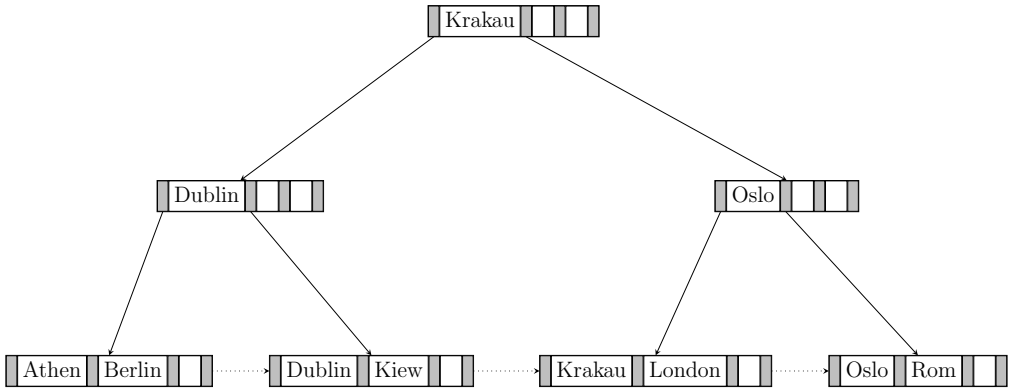
Aufgabe 3 - *B^+-Baum-Konstruktion.* 1 Punkt

Gegeben sind die Blätter eines B^+-Baumes ($m = 3$). Konstruieren Sie die **darüberliegenden Ebenen** (d.h. die inneren Knoten) so, dass ein **gültiger B^+-Baum mit minimaler Höhe** entsteht.



Aufgabe 4 - *B⁺-Baum-Löschen.* 1 Punkt

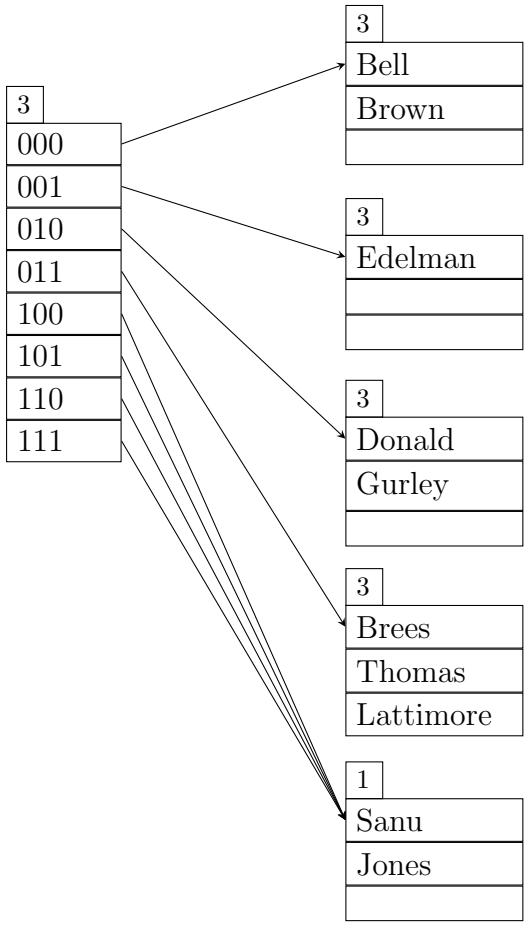
Gegeben ist ein B$^+$-Baum mit $m = 4$. Zeichnen Sie den B$^+$-Baum, der nach dem Löschen von **Krakau** entsteht.



Aufgabe 5 - *Erweiterbares Hashing*. 1 Punkt

Die Hashfunktion $h(x)$ liefert die in der Tabelle angegebenen Binärwerte. Es sollen die Tupel *Edelman* und *Jones* aus dem gegebenen Hashcontainer gelöscht werden. Ein Bucket im Hashcontainer kann bis zu 3 Tupel speichern. Das Verzeichnis soll so klein wie möglich gehalten werden. **Illustrieren Sie den resultierenden Hashcontainer**.

x	$h(x)$
Bell	0000
Brees	0110
Brown	0001
Donald	0100
Edelman	0011
Gurley	0101
Jones	1110
Lattimore	0111
Sanu	1100
Thomas	0110



Aufgabe 6 - *Join-Algorithmen*. 1 Punkt

Gegeben seien zwei Relationen mit folgenden Eigenschaften:

- $R[A]$ mit Werten $1, 2, 3, 4, 5, 6, 7, 8, 9, 11$,
- $S[A]$ mit Werten $2, 5, 6, 8, 9, 11, 13, 15, 16, 18, 19, 20$,
- die Größe des Puffer beträgt $M = 4$ Blöcke,
- jeder Block fasst 1 Tupel.

Für die Durchführung eines Hash Joins soll $h(x) = x \mod 3$ als Hashfunktion verwendet werden. Erklären Sie, ob und warum diese Hashfunktion für die Durchführung eines Hash Joins auf die Attribute $R[A]$ und $S[A]$ unter den gegebenen Umständen verwendet werden kann.

Aufgabe 7 - *Join-Algorithmen*. 1 Punkt

Welcher Join Algorithmus (**Hash Join**, **Index Nested Loop Join**) generiert die minimalen Kosten für das folgende Szenario? Geben Sie in der Lösung die **Algorithmen und** die **dazugehörigen Kosten** an.

Berechnen Sie einen Natural Join zwischen zwei Relationen $R[A, B]$ und $S[B, C]$, wobei $|R| = 1000$ Tupel und $|S| = 12000$ Tupel. Die Relationen sind auf $b_R = 250$ bzw. $b_S = 2000$ hintereinander liegenden Blöcken gespeichert. Der Buffer hat Platz für $M = 21$ Blöcke. Es existiert ein *sparse* B^+-Baum Index auf $S.B$, wobei jeder Knoten im B^+-Baum 20 Schlüssel speichern kann.

Aufgabe 8 - *Effiziente Anfragebearbeitung.* 1 Punkt

Gegeben ist die Relation $R[A, B]$. Auf $R.A$ existiert ein sparse B^+-Baum Index und auf $R.B$ existiert ein dense Hash-Index. Die Werte für Attribut B sind eindeutig. Was ist die **effizienteste Strategie** um Anfragen von folgendem Typ zu beantworten?

$$\sigma_{A<a \,\vee\, B=b}(R)$$

Geben Sie **alle notwendigen Schritte** an.

Aufgabe 9 - *Join Kardinalität.* 1 Punkt

Gegeben seien 3 Relationen $R[A, B, C]$, $S[B, C, D, F]$ und $T[A, B, D, E, F]$ mit folgenden Eigenschaften:

- $|R| = 5000$ Tupel, $V(R, A) = 30$, $V(R, B) = 60$, $V(R, C) = 50$
- $|S| = 4000$ Tupel, $V(S, B) = 20$, $V(S, C) = 30$, $V(S, D) = 20$, $V(S, F) = 10$
- $|T| = 3000$ Tupel, $V(T, A) = 50$, $V(T, B) = 10$, $V(T, D) = 40$, $V(T, E) = 100$, $V(T, F) = 20$

Schätzen Sie $|R \bowtie S \bowtie T|$ ab.

Aufgabe 10 - *Anfrageoptimierung*. 1 Punkt

Betrachte die folgenden Relationen und eine SQL-Anfrage.

(B)oats(bid, name, color)
(S)ailors(sid, name, rating, age)
(R)eservations(bid, sid, day)

```
SELECT DISTINCT B.name
  FROM Boats B, Sailors S, Reservations R
 WHERE S.rating > 8
   AND B.color = 'blue'
   AND B.bid = R.bid
   AND S.sid = R.sid;
```

a. Zeichnen Sie die **algebraische Normalform als Operatorbaum** für die gegebene SQL-Anfrage. **(0.5 Punkte)**
b. Wenden Sie **heuristische Optimierung** an, um den **Operatorbaum zu optimieren**. **(0.5 Punkte)**