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 2021/2022 — 21.01.2022

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

In die folgende Tabelle werden Werte eingefügt:

```
CREATE TABLE bands (
  bid INTEGER,
  bname VARCHAR(20)
);
```

Das Speichern eines `INTEGER` benötigt 4 Byte. Strings, die als `VARCHAR` gespeichert werden, benötigen ein Byte pro Zeichen und zusätzlich ein Byte zur Terminierung. Beispielsweise benötigt die Zeichenfolge `DBMS` 5 Bytes zur Speicherung. Die Tupel werden in eine Slotted Page mit folgenden Eigenschaften eingefügt:

- Größe: $2^{13} = 8192$ Bytes,
- Adressierungstyp: **Byte-Adressierung** (es kann jedes Byte adressiert werden)

Die folgenden Operationen werden in dieser Reihenfolge durchgeführt:

```
INSERT INTO books VALUES (123, 'Queen');                -- Tupel A
INSERT INTO books VALUES (336, 'Guns n Roses');         -- Tupel B
INSERT INTO books VALUES (52, 'The Beatles');           -- Tupel C
```

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

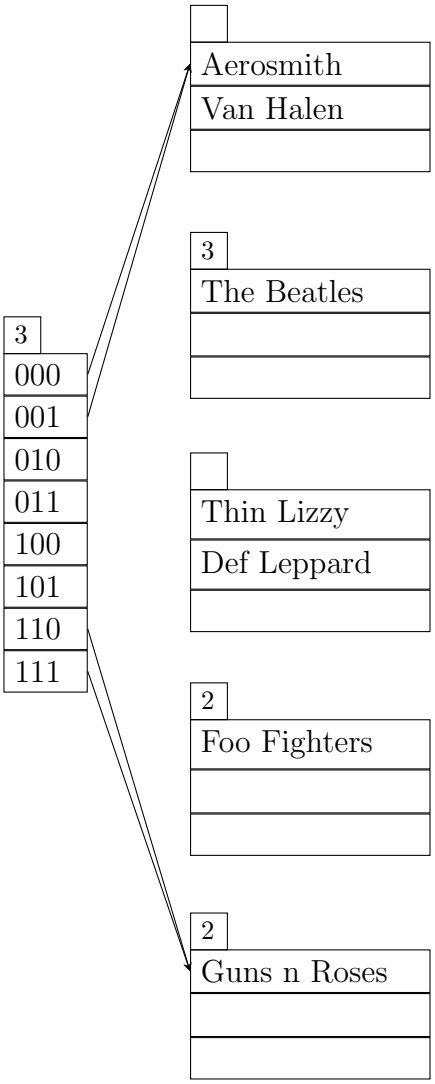
a	f	g_1	p_1	g_2	p_2	g_3	p_3	$\ldots$	d_3	d_2	d_1
									C	B	A

Aufgabe 2 - *Erweiterbares Hashing*. 1 Punkt

Die Hashfunktion h liefert die in der Tabelle angegebenen Binärwerte. **Ergänzen Sie die fehlenden Werte/Komponenten (Werte, lokale Tiefen und Pointer).**

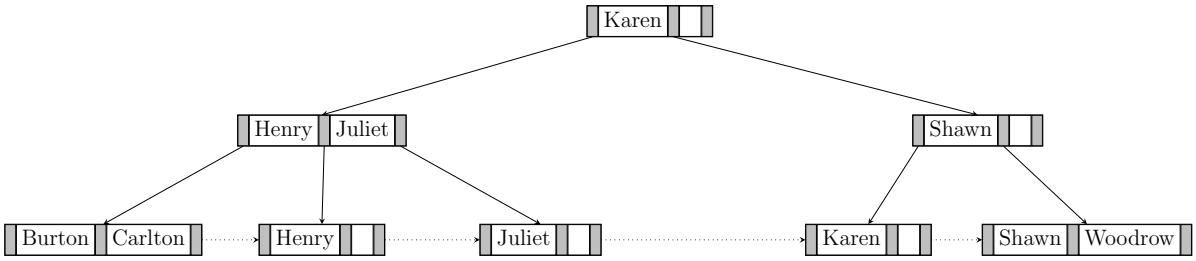
Hash-Tabelle:

Name (x)	$h(x)$ (binär)
Aerosmith	0010
Bon Jovi	0100
Def Leppard	0110
Foo Fighters	1000
Guns n Roses	1100
Metallica	1110
Rush	1011
Thin Lizzy	0111
The Beatles	0101
Van Halen	0001



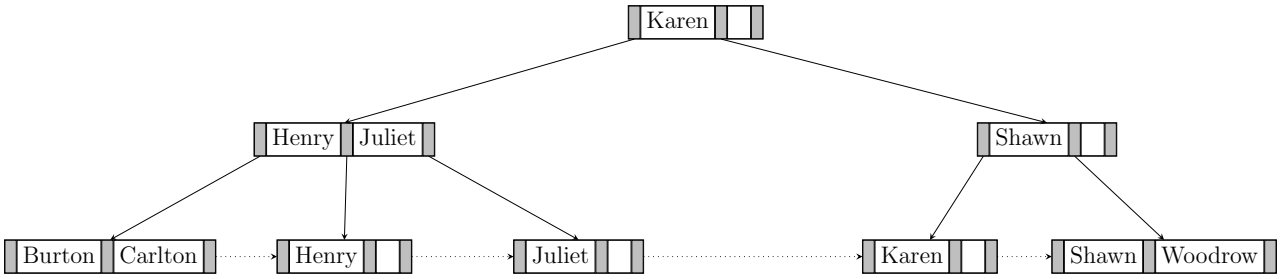
Aufgabe 3 - *B⁺-Baum-Einfügen*. 1 Punkt

Gegeben ist ein B$^+$-Baum mit $m = 3$. Zeichnen Sie den B$^+$-Baum der nach dem Einfügen von **Dexter** entsteht.



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

Gegeben ist ein B$^+$-Baum mit $m = 3$. Zeichnen Sie den B$^+$-Baum der nach dem Löschen von **Karen** entsteht.



Aufgabe 5 - *Indexstrukturen.* 1 Punkt

Zeichnen Sie für die folgende Tabelle einen **3-stufigen Sekundärindex** auf dem Attribut `Player`. Die **innere Indexstufe** soll **dense** und die **äußeren beiden Indexstufen** sollen **sparse** sein. Ein **Datenblock** kann **2 Tupel** speichern. In einen **Indexblock** können **3 Einträge** gespeichert werden.

`Player`	`Team`
Bell	Steelers
Brees	Saints
Brown	Steelers
Donald	Rams
Edelman	Patriots
Fournette	Jaguars
Gurley	Rams
Houston	Chiefs
Jones	Falcons
Kuechly	Panthers
Lattimore	Saints
Leonard	Colts
Mack	Bears
Michel	Colts
Sanu	Falcons
Thomas	Saints

Aufgabe 6 - *Join-Algorithmen.* 1 Punkt

Die Relation $R[\underline{A, B}, C]$ hat $n_R = 3000$ Tupel auf $b_R = 500$ Blöcken, die Relation $S[\underline{B}, D]$ hat $n_S = 2000$ Tupel auf $b_S = 500$ Blöcken. Es soll ein natürlicher Join mittels Block Nested Loop Joins durchgeführt werden.

Berechnen Sie die Anzahl an Blockzugriffen basierend auf unterschiedlichen Parametern:

- Es steht ein Puffer der Größe $M = 2$ zur Verfügung.

- Es steht ein Puffer der Größe $M = 501$ zur Verfügung.

- Es wird der Zick-Zack-Modus mit $M = 501$ und $k = 200$ verwendet.

Aufgabe 7 - *Effiziente Anfragebearbeitung*. 1 Punkt

Gegeben ist die Relation $R[A, B, C, D]$. Auf Attribut C existiert ein **dense B^+ Baum Index**. Werte von Attribut C sind gleichverteilt. Was ist die **effizienteste Strategie** um **Bereichsanfragen** von folgendem Typ zu beantworten?

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

Annahme:
$b - a \ll max(R.C) - min(R.C)$, d.h. der abgefragte Wertebereich ist wesentlich kleiner als der gesamte Wertebereich von Attribut C.

Geben Sie **alle notwendigen Schritte** an.

Aufgabe 8 - *Join-Algorithmen*. 1 Punkt

Gegeben seien zwei Relationen mit folgenden Eigenschaften:

- $R[A]$ mit Werten $1, 2, 3, 4, 5, 6, 8, 9, 12$,
- $S[A]$ mit Werten $2, 5, 6, 8, 9, 11, 13, 15, 16, 18, 19, 21$,
- 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. Kann diese Hashfunktion für die Durchführung eines Hash Joins auf die Attribute $R[A]$ und $S[A]$ unter den gegebenen Umständen verwendet werden? Erklären Sie, warum bzw. warum nicht.

Aufgabe 9 - *Join Kardinalität.* 1 Punkt

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

- $|R| = 2500$ Tupel, $V(R, A) = 50$, $V(R, B) = 10$, $V(R, C) = 20$
- $|S| = 500$ Tupel, $V(S, B) = 50$, $V(S, C) = 40$, $V(S, D) = 10$, $V(S, F) = 100$
- $|T| = 4000$ Tupel, $V(T, A) = 25$, $V(T, D) = 50$, $V(T, E) = 200$, $V(T, F) = 20$

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

Aufgabe 10 - *Anfrageoptimierung*. 1 Punkt

Gegeben seien die folgenden 3 Relationen $R[A, B, C]$, $S[A, B, V]$ und $T[A, B, X]$:

- $|R| = 10^3$ Tupel, $V(R, A) = 250$, $V(R, B) = 10$, $V(R, C) = 1.000$.
- $|S| = 10^4$ Tupel, $V(S, A) = 200$, $V(S, B) = 250$, $V(S, V) = 2.000$.
- $|T| = 10^5$ Tupel, $V(T, A) = 2.500$, $V(T, B) = 1.000$, $V(T, X) = 3.000$.

Weiters sei die folgende SQL-Anfrage gegeben:

```
SELECT  R.A, T.X
FROM    R, S, T
WHERE   R.A = S.A
        AND R.B = S.B
        AND S.A = T.A
        AND S.B = T.B
```

1. Zeichnen Sie die **algebraische Normalform als Operatorbaum** für die gegebene SQL-Anfrage. **(0.5 Punkte)**
2. Wenden Sie **heuristische Optimierung** an, um den **Operatorbaum zu optimieren**. Im resultierenden Operatorbaum soll auch die **Join-Reihenfolge optimal** sein, d.h. es soll zuerst der Join mit dem kleinsten Zwischenergebnis durchgeführt werden. **(0.5 Punkte)**