

(2)

Carolin Brandt

WS 2017/18
Mo 10-14 Uhr

Elite-Studiengang SWT

Datenbanksysteme

Dozent: Prof. Alfons Kemper
 E-Mail: kemper@in.tum.de
 Telefon: +49 89 289-17254
 Büro: 2.11.54
 Bürozeiten: Di 13 Uhr

(50)

(1) In rel. Algebra soll auf der Relation

Zehnkampf: {[name, disziplin, höheweitezeit, punkte]}

ermittelt werden, welche Sportler in den jeweiligen Disziplinen am besten waren.

(2) im Tupelkalkül sollen die Sportler ermittelt werden, die in allen (in der Relation bekannten) Disziplinen angetreten sind.-----
Sie können die Abkürzungen ZK, N, D, HWZ, P verwenden.

(1) ~~$\exists z_k \text{ name} \exists k \in \text{Zehnkampf} \wedge \forall z_{k2} \in \text{Zehnkampf} : (z_k.\text{disziplin} = z_{k2}.\text{disziplin} \Rightarrow z_{k2}.\text{punkte} \leq z_k.\text{punkte})$~~

$\exists z_k \text{ name} \exists k \in \text{Zehnkampf} \wedge \forall z_{k2} \in \text{Zehnkampf} : (z_k.\text{disziplin} = z_{k2}.\text{disziplin} \wedge z_{k2}.\text{punkte} \leq z_k.\text{punkte}) \quad \text{falls } z_k \text{ höchste Punktzahl}$ ✓

(2) $\{[n] \mid \exists D, H, P ([N, D, H, P] \in \text{Zehnkampf}) \wedge \forall D (\exists H, P ([N, D, H, P] \in \text{Zehnkampf}))\}$

↑ fehlende Bereichsberechnung

Domänenkalkül

Elite-Studiengang SWT

Datenbanksysteme

Dozent: Prof. Alfons Kemper
E-Mail: kemper@in.tum.de
Telefon: +49 89 289-17254
Büro: 2.11.54
Bürozeiten: Di 13 Uhr

Quiz

80

Erweitern Sie die Uni-Datenbank um die Relation

StudienPlan: {[Semester, VorlesungsNr]}

1. Bestimmen Sie in rel. Algebra die Studenten, die alle für ihr Semester vorgesehenen Vorlesungen hören.
2. Bestimmen sie in SQL die Studenten, die mindestens eine Vorlesung hören, die nicht für ihr Semester vorgesehen ist.

A.
 $\Pi_{s. MatrNr, s. VorlNr} ((\text{Studenten} \times \text{Studienplan}) \text{sp} \rightarrow \text{hören}) \rightarrow \text{AlleBravestudenten}$
 $\text{sp. VorlNr} = \text{h. VorlNr}$
 $s. MatrNr = h. MatrNr$
 $s. VorlNr = h. VorlNr$

PflichtBravestudenten \rightarrow AlleBravestudenten

① $\Pi_{s. MatrNr, sp. VorlNr} (\text{Studenten} \times \text{Studienplan} \text{sp}) = \text{PflichtVL}$
 $s. Sem = sp. Sem$

$\Pi_{\text{MatrNr}, \text{VorlNr}} (\text{hören}) = \text{PflichtVL} \rightarrow \text{BravestudentenMatrNr}$ ✓
~~MatrNr, MatrNr, VorlNr~~
~~Bravestudenten~~
~~MatrNr~~

② select s. MatrNr, s. Name
from studenten s join studienplan sp on s. Semester = sp. Semester
where exists (select distinct s. MatrNr, s. Name, s. Semester
from studenten s join studienplan sp
on s. Semester = sp. Semester
group by s. Semester
having count(*) > 1)

Elite-Studiengang SWT

Datenbanksysteme

Dozent: Prof. Alfons Kemper
E-Mail: kemper@in.tum.de
Telefon: +49 89 289-17254
Büro: 2.11.54
Bürozeiten: Di 13 Uhr

① with count-va as (select count(*) v.nachfolger
(count, vL) from voraussetzen v
group by v.nachfolger)

Quiz select max(count), vL
from count-va
group by vL
Ermitteln Sie in SQL

gilt konzeptuell nicht.
max \Rightarrow implizit "group by \emptyset "
also kann man kein Attribut vL select'en

1. welche Vorlesung hat die meisten direkten Voraussetzungen
2. In welchem Semester kann man "Der Wiener Kreis" frühestens hören (weil man vorher die direkten und indirekten Voraussetzungen hören muss)

② with recursive vorausgesetzt (voraus, vorlur) as (
(select * v.vorgaenger, v.nachfolger, 1 as count
from voraussetzen)
union all
(select nochvorher.vorgaenger, vL.vorlur, count + 1 as count
from voraussetzen nochvorher join vorausgesetzt vL
on nochvorher.nachfolger = vL.voraus))

)
~~select~~
~~max(voraus(count, vorlur)) as t~~

~~select max(count), vorlur~~
Select max(vg.count)
from vorausgesetzt vg join vorlesungen vL
on vg.vorlur = v.vorlur
where v.title = "Der Wiener Kreis" ✓

(5)

Carolin Brandst

WS 2017/18

Mo 10-14 Uhr

75

Elite-Studiengang SWT

Datenbanksysteme

Dozent: Prof. Alfons Kemper
 E-Mail: kemper@in.tum.de
 Telefon: +49 89 289-17254
 Büro: 2.11.54
 Bürozeiten: Di 13 Uhr

Quiz

(1) Sie erhalten (zumindest wenn Sie diese Aufgabe richtig lösen;-) einen einmaligen Bonus. Das heißt, ein (und nur ein) schwächste **erzieltes** Quizergebnis wird für jede/n Studierende/n durch die **maximal erzielbare** Punktzahl ersetzt.
 Schreiben Sie den SQL Befehl für die Sicht QuizBonus.

(2) Formulieren Sie (auf Deutsch oder Englisch) die Fragestellung der interessantesten bzw. schwierigsten der 100 SQL Aufgaben, die Sie diese Woche gelöst haben.

(3) Die Relation Quiz ist nicht normalisiert. Warum nicht? Welche funktionale Abhängigkeit „ist schuld“ daran? Geben Sie den Schlüssel von Quiz an.

(2) Für eine Verkaufs DB, bei der siele Items einen Preisnachlass hatten soll berechnet werden, was ohne den Nachlass zusätzlich eingenommen worden wäre. Die Musterlösung war halb so lang wie mein versuch!!

(3) Das sollten wir doch noch gar nicht vorbereiten XD

$$\begin{aligned} q_e &\rightarrow m \\ q_{in} &\rightarrow \epsilon \end{aligned}$$

Das hat Thomas Neumann ja wohl schon unterrichtet...

[q, n] ist der Schlüssel der Relation, aber es gibt Spalten die nur von einer Teilmenge des Schlüssels abhängen ($q \rightarrow m$), Technik ist Quiz nicht in 3NF

with worst_erg as
 select ~~q.name, q.erzielt~~, min(~~q.erzielt~~)
 from Quiz
 group by name;

worst_qno as
 select qno, ~~q.name~~ distinct name

from Quiz q join worst_erg w
 on q.name = ~~w.co.name~~

where ~~q.erzielt < w.erzielt~~
) q.erzielt - q.max
 max ~~= co.co~~

| Qno | Name | Max | Erzielt |
|-----|---------|-----|---------|
| 1 | Meier | 80 | 60 |
| 1 | Meier | 100 | 88 |
| 2 | Schmidt | 80 | 60 |
| 2 | Schmidt | 100 | 80 |

| Qno | Name | Max | Erzielt |
|-----|---------|-----|---------|
| 1 | Meier | 80 | 80 |
| 2 | Meier | 100 | 88 |
| 1 | Schmidt | 80 | 80 |
| 2 | Schmidt | 100 | 80 |

(1) create view QuizBonus as (

select q.qno, q.name, q.max,
 (case when

then q.max

else q.erzielt) as erzielt

from Quiz q join worst_qno w

on q.name = w.name

) gut modifiziert! Wenn es jetzt auch noch etwas besser lieber wäre...

versuch!!

Elite-Studiengang SWT

Datenbanksysteme

Quiz

Überprüfen Sie mittels einer SQL-Abfrage ob in der Relations-Ausprägung R mit dem Schema $R = \{A, B, C\}$ die MVD

$A \rightarrow\!\!\! \rightarrow B$

✓ As finde alle Passenden Bs $B_s \times C_s$
zugehörigen

eingehalten ist.

Tausche dann alle Bs durch &
küche ab gesuchtes Tupel (mit C)
noch drin ist
originelle Idee

select *
from R r-aussen
where not exists (

select r-innen.A, alleBs.B, r-innen.C
from R r-innen join (select r-i2.A, r-i2.B
from R r-i2
where r-i2.A = r-aussen.A
r-innen.A = alleBs.A)

select *
from R r-a2
where not exists (

wie soll man das
"passen" können?

Zusatzfragen

(Allgemeinbildung eines DB'lers)

Wer erfand ...

... das relationale Modell

✓ ... den B-Baum

... den Rot/Schwarz-Baum

... das Transaktionskonzept

Wer gründete die erfolgreichste Datenbank-Firma

Wer gründete Facebook

Nennen Sie mindestens zwei der vier SAP Gründer (wer von ihnen hat das Garchinger Planetarium finanziert?)

Wie heißt SAP's Hauptspeicher-Datenbanksystem
HANA

Welche DB-Firma hat SAP übernommen?

Was ist MariaDB und wem gehört MySQL?

Welche drei Datenbanken gewannen den Turing-Award?

)
class R-ausser

alternativ:
gruppieren nach A
 $\text{count}(B) = \text{count}(C) = \text{count}(\text{distinct } C) * \text{count}(\text{distinct } B)$

Elite-Studiengang SWT

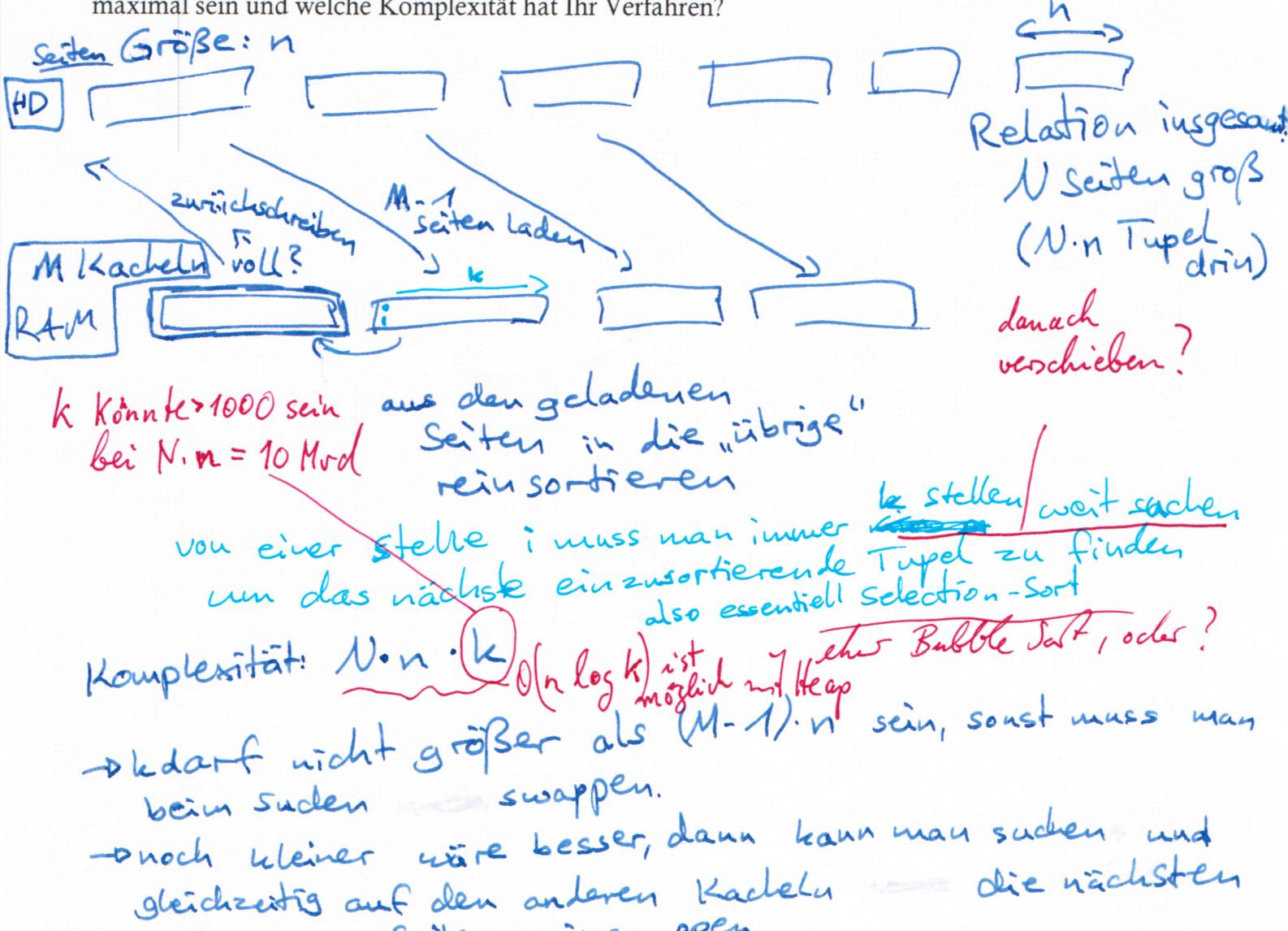
Datenbanksysteme

Dozent: Prof. Alfons Kemper
 E-Mail: kemper@in.tum.de
 Telefon: +49 89 289-17254
 Büro: 2.11.54
 Bürozeiten: Di 13 Uhr

Quiz

75

Konzipieren Sie ein externes (also für über den Hauptspeicher hinausgehende Datengrößen) Sortierverfahren, das besonders effizient ist bei fast sortierten Relationen, bei denen ein Tupel nie mehr als k Positionen "verrutscht" ist. Wie groß darf k bei Ihrem Verfahren maximal sein und welche Komplexität hat Ihr Verfahren?



⑧

⑧

Datenbanksysteme

Name: Carolin BrandtMatrikelnummer: 03660883**Aufgabe 1**

Bestimmen Sie mir Hilfe von Dynamischer Programmierung die optimale Joinreihenfolge für das folgende Problem:

$$\begin{aligned} |R_1| &= 1000 \quad |R_2| = 1000 \quad |R_3| = 100 \quad |R_4| = 10 \\ \text{sel}_{R_1 \bowtie R_2} &= 0.1 \quad \text{sel}_{R_2 \bowtie R_3} = 0.01 \quad \text{sel}_{R_3 \bowtie R_4} = 0.5 \quad \text{sel}_{R_4 \bowtie R_1} = 0.01 \\ (\text{alle anderen } \text{sel}_{R_i \bowtie R_j}) &= 1 \end{aligned}$$

$$C(T) = \begin{cases} 0 & \text{if } T \text{ is a relation } R_i \\ |T| + C(T_1) + C(T_2) & \text{if } T = T_1 \bowtie T_2 \end{cases}$$

| | |
|-------------------|---|
| R_1 | 10^3 |
| R_2 | 10^3 |
| R_3 | 10^2 |
| R_4 | 10 |
| $R_1 \bowtie R_2$ | $10^3 \cdot 10^3 \cdot 0.1 + 0 + 0 = 10^6 \cdot 0.1 = 10^5$ |
| $R_2 \bowtie R_3$ | $10^5 \cdot 0.01 = 10^3$ |
| $R_3 \bowtie R_4$ | $10^2 \cdot 0.5 = 500$ |
| $R_2 \bowtie R_4$ | 10^4 |
| $R_3 \bowtie R_1$ | 10^5 |
| $R_4 \bowtie R_1$ | $10^4 \cdot 0.01 = 10^2$ |



$$\begin{array}{l|l} R_4 \bowtie R_2 & 10^2 \cdot 10^3 + 10^2 \cdot 10^3 = 10^5 + 100 + 1000 = 10,100 \\ \hline R_4 \bowtie R_3 & 10^2 \cdot 10^2 + 10^2 + 10^2 = 10^4 + 2 \cdot 10^2 = 10,200 \end{array}$$

(2, 3, 4)?

$((R_1 \bowtie R_4) \bowtie R_3) \bowtie R_2$ nur left deep? ↴

Aufgabe 2

Geben für jedes $M \in \{\text{NestedLoop}, \text{Hash}, \text{SortMerge}\}$ ein Beispiel der Form $T_1 \bowtie_{a=b}^M T_2$ an, so dass der gewählte Joinalgorithmus optimal ist.

Nested Loop: (T_1, T_2 unsortiert + kein Index in T_1 auf a)
und keiner in T_2 auf b
a hat immer den selben Wert in T_1
und b hat immer den selben Wert in T_2
oft

Hash: Eine der Relationen T_1 (oder T_2) ist so klein,
dass eine Hashtabelle über a (oder b) in
den RAM passt. + keine Indexstruktur existiert
über a (oder b) + $T_1 \& T_2$ unsortiert

SortMerge: Beide Relationen $T_1 \& T_2$ sind bereits
sortiert (nach a bzw. b)



Elite-Studiengang SWT

Datenbanksysteme

Dozent: Prof. Alfons Kemper
 E-Mail: kemper@in.tum.de
 Telefon: +49 89 289-17254
 Büro: 2.11.54
 Bürozeiten: Di 13 Uhr

Quiz zur Serialisierbarkeit

(A) Geben Sie Beispiele von Historien (Schedules), die von einem strengen 2PL-Scheduler mit Deadlock-Erkennung mittels eines Wartegraphen akzeptiert würden aber **nicht**

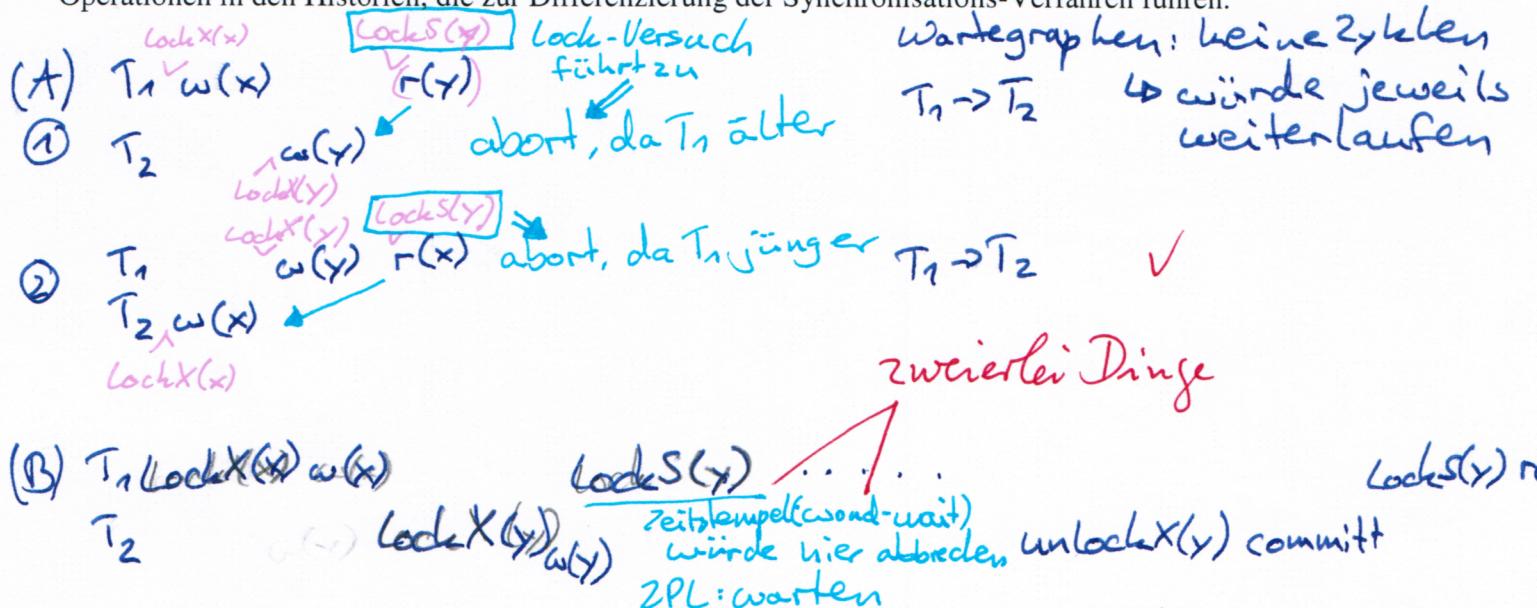
1. bei der wound-wait Deadlockvermeidung bzw
2. bei der wait-die Deadlockvermeidung

„durchkommen“.

(B) Geben Sie ein Beispiel einer Historie, die beim strengen 2PL aber nicht bei der Zeitstempel-basierten Synchronisation entstehen kann.

(C) Geben Sie ein Beispiel einer Inkonsistenz, die bei gängigen Versionsverwaltungssystemen wie git (oder perforce, subversion, cvs, etc.) entstehen kann, weil diese Systeme auf snapshot isolation basieren. Warum wird in diesen Systemen keine „harte“ Serialisierbarkeit verwendet? Welches Versionsverwaltungssystem verwenden Sie persönlich für Ihre Projekte? Wie verhält es sich bei nicht möglicher Validierung gemäß der Snapshot Isolation? Was genau wird überprüft bei der Validierung?

Begründen Sie jeweils die Konstruktion Ihrer Beispiel-Historien und markieren Sie die entsprechenden Operationen in den Historien, die zur Differenzierung der Synchronisations-Verfahren führen.



(C) keine harte Serialisierbarkeit: „Reads“ auf Codateien kann man meistens nicht rauskriegen, weil so was nicht dokumentiert ist
 + würde zu viele Konflikte verursachen

git ist das beste!

Erzeugt Merge-Konflikte, prüft auf Dateizeilenbasis
 (zwei Zeilen, die selbe Zeile geändert haben)

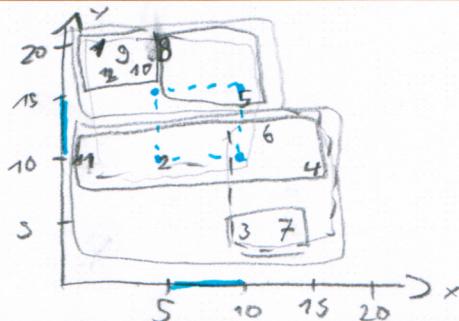
Für die Zeilen verändert wurden

Elite-Studiengang SWT

Datenbanksysteme

Dozent: Prof. Alfons Kemper
E-Mail: kemper@in.tum.de
Telefon: +49 89 289-17254
Büro: 2.11.54
Bürozeiten: Di 13 Uhr

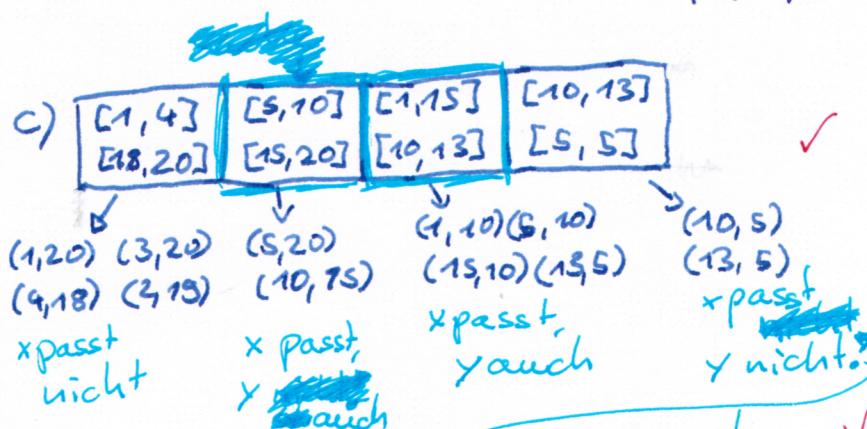
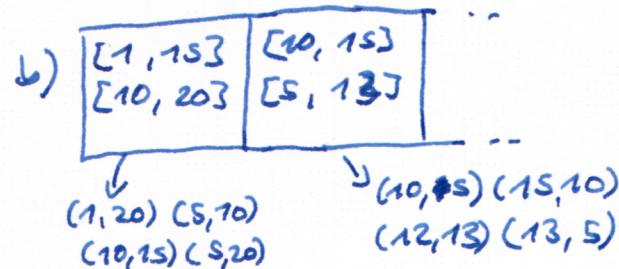
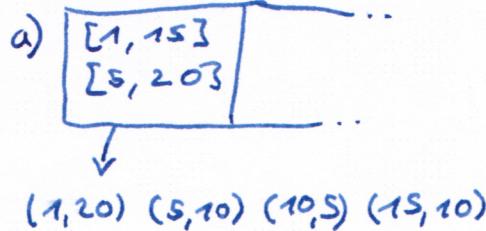
Quiz



100

Zeigen Sie den Aufbau eines R-Baums mit Knotenkapazität 4 mit folgenden Suchschlüsseln, die in der Gegebenen Reihenfolge einzufügen sind:

$$\times_{(1,20)(5,10)(10,5)(15,10)(10,15)(12,13)(13,5)(5,20)(3,20)(4,18)(1,10)(2,19)}$$



3. $(1, 10)$ passt nicht
 $(5, 10)$ passt, y auch $\rightarrow \checkmark$

② $(5, 20) \times$ passt, y_{auch}
 $(10, 15) \times$ passt, $y_{\text{auch}} \rightarrow \checkmark$

Carous Dronow

90

① Threshold

Zwergenlos: Phase 1

Threshold 10.160 & 1. Threshold

~~Seat Leon~~
~~Audi A1~~

Seat One

Audi

Mini

Citroen

TS

Mercedes

10.160

12.680

15.288

16.079

16.967

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.960

25.9

Elite-Studiengang SWT

Datenbanksysteme

Dozent: Prof. Alfons Kemper
 E-Mail: kemper@in.tum.de
 Telefon: +49 89 289-17254
 Büro: 2.11.54
 Bürozeiten: Di 13 Uhr

Quiz

Zeigen Sie, dass ROWA und Majority/Consensus Spezialisierungen des Quorum/Consensus-Verfahrens sind, gemäß folgender Teilaufgaben:

16.17 Zeigen Sie („grob“), dass bei der write-all/read-any Methode zur Synchronisation bei replizierten Daten zwar nur serialisierbare Schedules erzeugt werden – unter der Voraussetzung, dass das strenge 2PL-Protokoll angewendet wird. Aber die Deadlock-„Gefahr“ nimmt dramatisch zu... warum?

16.18 Zeigen Sie, dass die write-all/read-any Methode zur Synchronisation replizierter Daten einen Spezialfall der Quorum-Consensus-Methode darstellt.

- Wie werden Stimmen zugeordnet, um write-all/read-any zu simulieren? • Wie müssen die Quoren Qw und Qr vergeben werden?

16.19 Einen weiteren Spezialfall des Quorum-Consensus-Verfahrens stellt das Majority-Consensus-Protokoll dar. Wie der Name andeutet, müssen Transaktionen sowohl für Lese- als auch für Schreiboperationen die Mehrzahl der Stimmen einsammeln. Zeigen Sie die Konfigurierung des Quorum-Consensus-Verfahrens für die Simulation dieses Majority-Consensus-Protokolls.

| | | |
|--|--|---|
| 16.17 $Q_w(t) = w(t) = \# \text{ Repliken}$ $Q_r(t) = 1$ | 16.18 Werte $\text{Gewichte sind pro Replik je } 1$ | Werte $\text{write Stimmen gehen zu Q}_w$ $\text{Lese-Stimmen gehen zu Q}_r$ |
|--|--|---|

- Deadlock-Gefahr einer Transaction nimmt zu, da man auf alle Repliken ein X-lock braucht um schreiben zu können.

2 update auf derselbe Datei können schon "deadlocken"

$$16.19$$

Majority-Consensus-Protokoll

$$Q_w(t) = \frac{w(t)}{2} + 1 \text{ geradig?}$$

$$Q_r(t) = \frac{w(t)}{2} - Q_w(t)$$

damit immer wenig eine aktuelle Kopie beim Lesen der Daten

- Immer serialisierbar, da bei X-locks auf alle Repliken das System so ist wie ein zentralisiertes DB MS (✓)