Datenbanksysteme Klausurangabe WiSe 2012/13 – Prof. Böhm

[Sebastian Groß]

Aufgabe 1

(a) Was versteht man unter der Anomalie LOST UPDATE, die im Mehrbenutzerbetrieb auftreten kann? Geben Sie ein Beispiel an, in dem ein LOST UPDATE auftritt.

- Anomalie, bei der dasselbe Datum X von zwei TAs geschrieben wird, und die erste Änderung überschrieben wird. (Informelle Begründung auch okay)
- Beispiel $S= (r1(x), w2(x) , w1(x))$

(b) Gegeben sei das Relationenschema R=(<u>A</u>, B, C, D). Über die Relation sei nur bekannt, dass A ein Schlüssel ist, und es keine weiteren SKen gibt. Über weitere FDs sei nichts bekannt, es kann aber weitere FDs geben. Nehmen Sie an, dass R die erste NF erfüllt.

Welche NF erfüllt R noch? Welche möglicherweise nicht?

Mindestens 2NF auf jeden Fall erfüllt, da Schlüssel nicht zusammengesetzt.

3 NF möglicherweise verletzt, da z.B. die FD : B → C gelten kann

(Über BCNF und 4NF stand nix in der Musterlösung und hat auch Böhm nichts zu gesagt! Man musste also nur bis zur dritten NF begründen…)

(c)

Gegeben sei R(A,B,C) und S=(C, D,E). R enthalte 50 Tupel und S enthalte 10 Tupel.

A1:

SELECT *

FROM R,S ;

A2:

SELECT *

FROM R NATURAL JOIN S;

Wieviele Ergebnistupel liefert dieAnfrage A1?:

500

Wieviele Attribute enthält die Ergebnisrelation von Anfrage A1?:

6

Wieviele Ergebnistupel liefert die Anfrage A1 MINDESTENS:

0

(Erklärungsbeispiel: Bei R sind unter C als Attributwerte nur Einsen und bei S als Attributewerte nur Zweien → Es gibt keinen Join-Partner => 0 Tupel)

Wieviele Ergebnistupel liefert die Anfrage A1 HÖCHSTENS:

500

(Erklärungsbeispiel: Sowohl bei R als auch bei S stehen unter C als Attributwerte überall Einsen → Natural Join entspricht dann dem Kreuzprodukt („Jeder mit Jedem“)

Wieviele Attribute enthält die Ergebnisrelation von Anfrage A2?:

5

Begründung: Natural Join eliminiert identische Spalten! (Spalte C wäre doppelt)

d) Was versteht man unter dem „Cursor-Konzept“, und weshalb wird es eingesetzt?

Bei der Integration von Anfragen in Programmiersprachen wird beim Cursor-Konzept statt der Ergebnismenge ein Cursor übergeben, der auf die Position des aktuellen Tupels zeigt. Dadurch wird die Ergebnismenge tupelweise an die Host-Sprache übergeben (1 Pkt).

Das Konzept wird eingesetzt, da

- die Host-Sprache die Datenstruktur „Tabelle/Menge“ evtl. nicht kennt (Impedance Mismatch)
- die Übertragung der kompletten Ergebnismenge kapazitäts- und zeitintensiv und oft unnötig ist

e) Welche maximale Höhe kann ein B-Baum der Ordnung 2, der n Schlüssel enthält, annnehmen, wenn gilt:

n=16 Schlüssel=Elemente → max. Höhe ist 2

n= 17 Elemente → max. Höhe ist 3

Herleitung mit Formel:

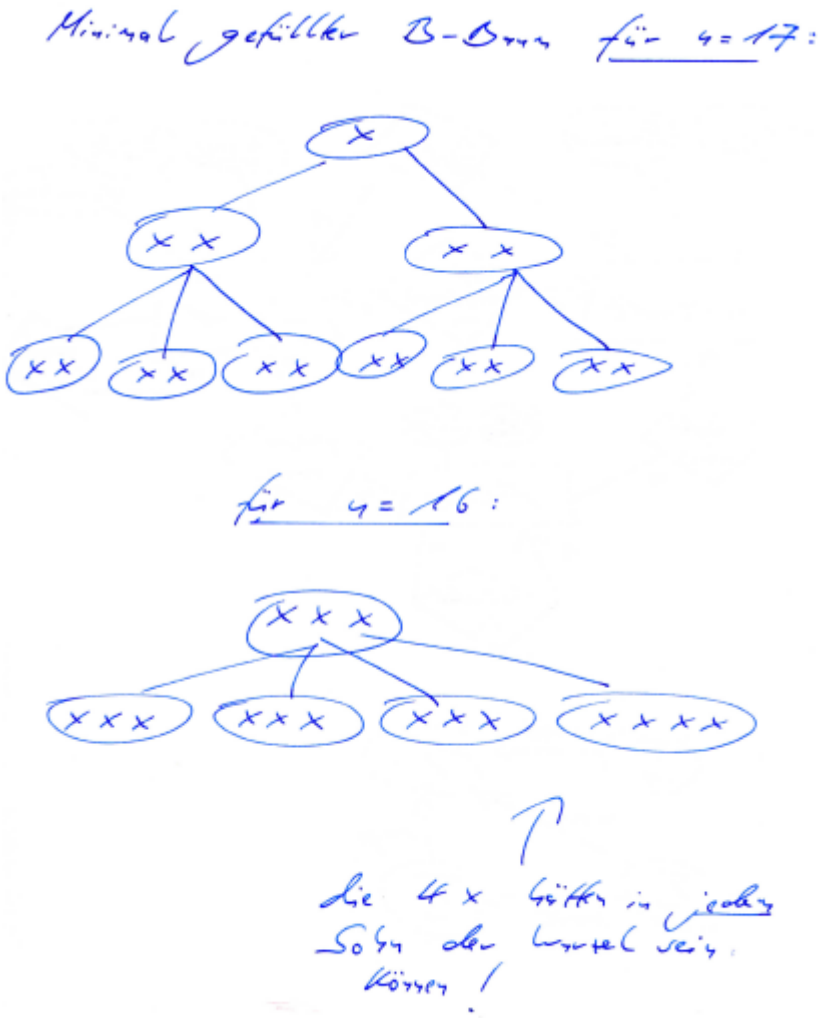
$$h(n) \leq \left\lfloor log_{m+1} \left(\frac{n+1}{2}\right)\right\rfloor + 1$$

(die nach unten gesetzte Klammer ist die floor-function, m ist die Ordnung und n die Anzahl der Elemente / Schlüssel)

Oder Herleitung mit MINIMAL gefülltem Baum: Wiederholung – Definition von B-Baum

Definition: B-Baum der Ordnung *m*
(Bayer und McCreight (1972))

(1) Jeder Knoten enthält höchstens $2m$ Schlüssel.
(2) Jeder Knoten außer der Wurzel enthält mindestens m Schlüssel.
(3) Die Wurzel enthält mindestens einen Schlüssel.
(4) Ein Knoten mit k Schlüsseln hat genau k+1 Söhne.
(5) Alle Blätter befinden sich auf demselben Level.



Aufgabe 2 (Relationales Modell)

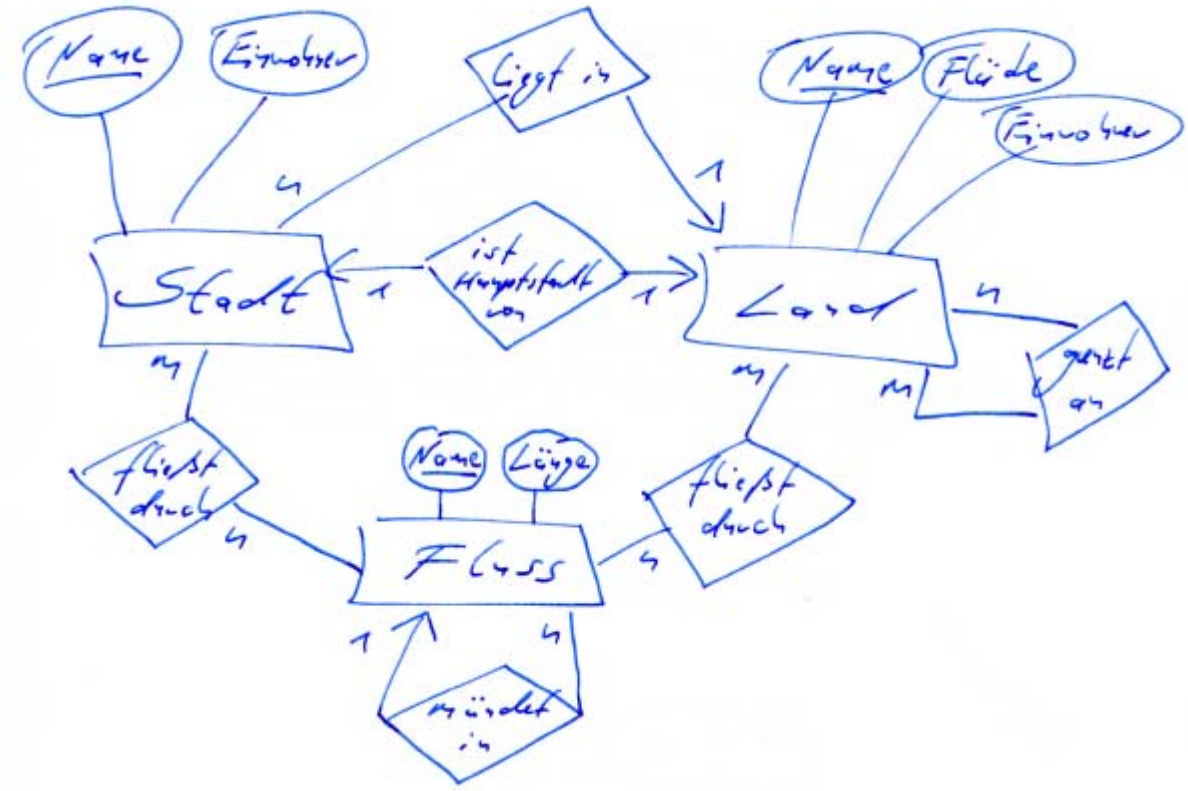
Stadt, Land, Fluss mal anders: Die relationale Datenbank eines geographischen Systems beinhaltet Städte, Länder und Flüsse mit folgenden Einzelheiten:

Eine Stadt wid durch ihren Namen und das Land gekennzeichnet, in dem sie liegt. Außerdem hat jede Stadt eine Einwohnerzahl.

Ein Land hat einen eindeutigen Namen, eine Fläche und ebenfalls eine Einwohnerzahl. Ein Land kann an mehrere Länder angrenzen. In einem Land liegt mindestens eine Stadt, wobei jede Stadt nur in genau einem Land liegt. Außerdem hat jedes Land eine Hauptstadt.

Ein Fluss wird durch seinen Namen bestimmt und hat eine gewisse Länge. Jeder Fluss kann durch mehrere Städte und auch durch mehrere Länder fließen und kann am Ende in einen anderen Fluss münden. Eine Stadt kann an mehreren Flüssen liegen und durch ein Land können mehrere Flüsse fließen.

Erstellen Sie ein ER-Diagramm für obige Datenbank. Markieren die die Funktionalitäten der Relationships und unterstreichen sie den Primärschlüssel jeder Entität.



Aufgabe 3 (Relationale Algebra)

In der Relation besucht werden Studenten mit ihrem Vor- und Nachnamen, zusammen mit den Vorlesungen die sie besuchen gespeichert. Die Relation Vorlesung beschriebt eine Menge von Vorlesungen mit ihrer Teilnehmerzahl. Die DB hat folgenden Zustand:

(Attributnamen in **bold face**)

VORLESUNG

Lehrveranstaltung	**Teilnehmerzahl**
Einf in Program	500
DBS I	300
Software	100

BESUCHT

Vorname	**Nachname**	**Lehrveranstaltung**
Michael	Meier	Einf in Program
Peer	Schmidt	Einf in Program
Andreas	Meier	DBS I
Michael	Meier	DBSI
Peer	Schmidt	DBS I
Nina	Schubert	DBS I
Michael	Meier	Software
Nina	Schubert	Software
Peer	Schmidt	Software

Geben Sie die Ergebnisrelation folgender Anfragen in relationaler Algebra an:

(a) $besucht \div \pi_{Lehrveanstaltung}(Vorlesung)$

(b) $\pi_{Nachname,Lehrveranstaltung}(besucht) \div \pi_{Lehrveanstaltung}(Vorlesung)$

(c) $\sigma_{Nachname='Schubert'}(besucht) \bowtie Vorlesung$

(a)

BESUCHT[1]

Vorname	Nachname
Peer	Schmidt
Michael	Meier

(b)

BESUCHT

Nachname
Schmidt
Meier

(c)

BESUCHT

Vorname	Nachname	Lehrveranstaltung	Teilnehmerzahl
Nina	Schubert	DBS I	100
Nina	Schubert	Software	300

[1] Anscheinend muss man einen Relationennamen angeben.

Aufgabe 4 (Anfragen in SQL, relationale Algebra, Kalküle)

Gegeben seien die vier Relationen:[2]

Dozent (DNR, DVorname, DNachname, Dtitel)

Vorlesung (VNR, VTitel, Klausurtermin, Dozent)

Student (Matrikelnummer, SVorname, SNachname, Semesterzahl)

besucht (Student, Vorlesung)

a) SQL-DDL für besucht Relation. Achte auf Fremdschlüsselbeziehungen!

CREATE TABLE besucht (

Student INTEGER references Student(Matrikelnummer),

Vorlesung INTEGER references Vorlesung(VNR)

primary key (Student, Vorlesung)

);

b) Erstellen Sie eine Liste bestehend aus Vornamen und Nachname aller Studenten, die im dritten Semester sind und an einer Vorlesung von Peter Krüger teilnehmen. Achten Sie darauf, dass die Ausgabe nach folgenden Kriterien absteigend sortiert ist: 1. Nachname, 2. bei gleichem Nachnamen nach dem Vornamen

[2] Auf Klausur war noch eine Beispielausprägung, aber die ist ja sowieso unwichtig für die Anfrage.

SELECT SVorname, SNachname

FROM Student s, besucht b, Vorlesung v, Dozent d

WHERE a.Matrikelnummer=b.Student AND b.Vorlesung = v.VNR AND

v.Dozent = d.DNR AND

s.Semesterzahl = 3 AND

d.DVorname = 'Peter'

AND d.DNachname = 'Krueger'

ORDER BY SNachname DESC, SVorname DESC ;

c) Bestimmen Sie die Matrikelnummer aller Studenten, die am 29.01.2014 keine Klausur schreiben

SELECT b.Matrikelnummer

FROM besucht b

WHERE NOT EXISTS (Select * FROM Student s, Vorlesung v

WHERE B.Matrikelnummer= s.Student AND b.Vorlesung = v.VNR AND v.Klausurtermin = '29.01.2014');

d) Bestimmen Sie für alle Vorlesungen, die im Durchschnitt bis zum 6. Semester gehört werden, die VNR, den Vorlesungstitel, Vor- und Nachnamen des Dozenten und die Anzahl der mit der zugehörigen Klausur kollidierenden Klausurtermine aller anderen Vorlesungen. Vorlesungen ohne Kollision oder ohne Teilnehmer sollen nicht in der Auflistung erscheinen.

a. Möglichkeit mit VIEW
b. andere Möglichkeit mit Self-Join (Musterlösung):

```
SELECT V.VNR, Vtitel, DVorname, DNachname, AnzahlKollisionen
FROM Vorlesung V, Dozent, besucht, Student,
( Select V1.VN, count(V2.VNR) AS AnzahlKollisionen
FROM Vorlesung V1, Vorlesung V2
WHERE V1.Klausurtermin = V2.Klausurtermin AND V1.VNR <> V2.VN
GROUP BY V1.VNR)  Sub
WHERE
V.VNR = Vorlesung AND Dozent = DNR AND
Student = Matrikelnummer
AND V.VNR = Sub.VNR
GROUP BY V.VNR
Having avg(Semesterzahl) <= 6 ;
```

(e) Formulieren Sie folgende Anfrage in relationaler Algebra.

Bestimmen Sie die Matrikelnummer, Name und Vorname aller Studenten, die sich mindestens im elften Semester befinden und (immer noch) die VL DBS I besuchen. Sie können an passender Stelle auch den Join-Operator benutzen.

Lösung klar.

Formulieren Sie folgende Anfrage in TUPELkalkül:

(f)

Geben Sie die Namen der Vorlesungen an, die Fritz Müller bei Herrn Huber besucht.

Lösung ebenfalls klar.

Aufgabe 5 (Normalisierung)

Gegeben Sei die Relation R(A,B,C,D,E,F)

sowie die Menge der zugehörigen nicht-trivialen FDs

F =

{B,C → D

A → B, C, F

C → F

C → D,E

D → E }

(a) Begründen Sie warum A der einzige Schlüsselkandidat (SK) ist.

A ist eindeutig, da B, C und F direkt von A abhängig sind, und D und E indirekt von A abhängig sind. Da A nur aus einem Element besteht ist es auch minimal und damit Schlüsselkandidat. Es gibt keine weiteren SKen, da A nur von sich selbt abhängig ist.

(b) Bringen Sie das Relationenschema R mit Hilfe des Synthesealgorithmus in die 3NF. Führen Sie jeden Schritt des Algorithmus durch und kennzeichnen Sie Stellen, an denen nichts zu tun ist deutlich.

Kanonische Überdeckung F_C:

C → D,F

A → B,C

D → E

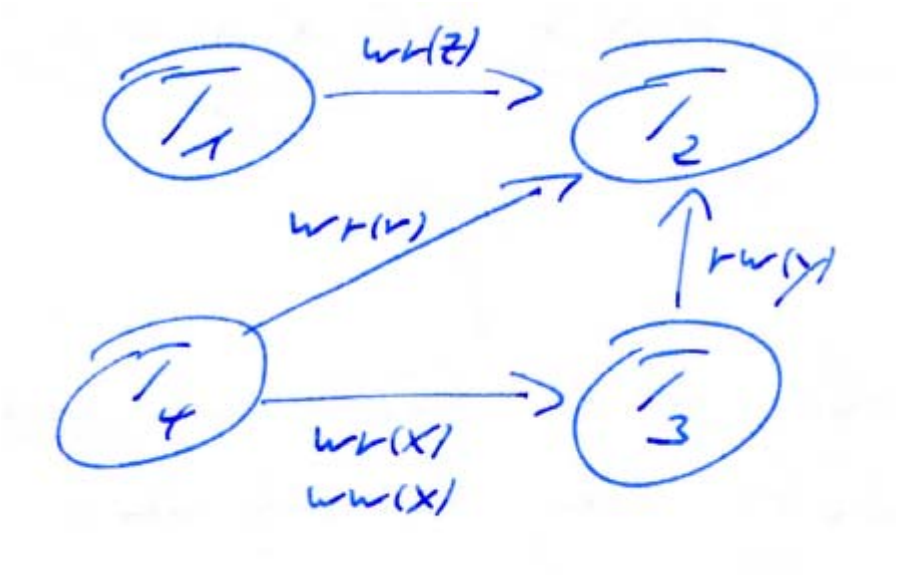
Erzeugen der Schemata ist klar! Keine Rekonstruktion des SK und keine Relation streichen! FERTIG!

Aufgabe 6 (Transaktionen / Serialisierbarkeit von Schedules)

S= (r2(y), r4(v), r1(z), w1(z), r3(y), w4(x), w4(v), r2(z), r2(v), w2(y), r3(x), w3(x))

(a) Stellen Sie S als Serialisierbarkeitsgraph dar.
(b) Ist der Schedule aus Teilaufgabe (a) serialisierbar? Falls ja, geben Sie einen möglichen seriellen Schedule an. Falls der Schedule nicht serialisierbar ist, begründen Sie ihre Antwort.

(a)



(b) Der Schedule ist serialisierbar, da azyklischer Abhängigkeitsgraph. Folgende äquivalente, serielle Schedules wären also möglich:

T1 → T4 → T3 → T2

T4 → T3 → T1 → T2

T4 → T1 → T3 → T2