Prof. Dr. T. Härder
Fachbereich Informatik
Arbeitsgruppe Datenbanken und Informationssysteme
Universität Kaiserslautern

Übungsblatt 6 – Lösungsvorschläge

für die freiwillige Übung

Unterlagen zur Vorlesung: „wwwdvs.informatik.uni-kl.de/courses/DBSREAL/“

Aufgabe 1: Darstellung von Sätzen - Extremlösung

Es soll eine Speicherungsstruktur für eine Satzmenge mit ≥ 200 Feldern/Satz entwikkelt werden, von denen ggf. viele NULL-Werte besitzen können. Da der Software-Hersteller seine Anwendungssoftware weltweit vertreibt und den Kunden eine freie Wahl aus k ≤ 6 relationalen DBMS gestattet, muß die Struktur in allen DBMS gleichermaßen abbildbar sein. Jeder Satz oder mindestens jedes Satzfragment muß in einer Seite speicherbar sein, d.h. $S_l \leq L_S - L_{SK}$. Außerdem sollen sich die Felder indexieren lassen. Beliebige dynamische Erweiterungen mit neuen Feldern sind wünschenswert.

Als Beispiel wählen wir die Tabelle Pers mit den herkömmlichen Spalten (Attributen) Pnr, Vorname, Name, Beruf, Alter usw. Auf die einzelnen Felder wird mit unterschiedlicher Häufigkeit zugegriffen. Auf der DB-Schemaebene kann Pers natürlich vertikal partitioniert werden. Dann entstehen aber mehrere Tabellen, die bei DB-Operationen explizit angesprochen werden müssen.

a) Es werde eine Tabellenstruktur für die Sätze von Pers gewählt. Welche Speicherungsoptionen sind sinnvoll, wenn auch darauf geachtet werden muß, daß ein Satz (Satzfragment) immer in eine Seite paßt?

   Einführung von Satzfragmenten, d. h. Aufspaltung der gespeicherten Sätze, und deren Kettung auf Feldebene; Zuordnung der Felder nach Zugriffshäufigkeiten zu den Satzfragmenten (Primary, Secondary, ...); wird möglicherweise nicht von allen k DBMS unterstützt.

b) Wenn eine Tabellenstruktur gewählt wurde, wie kann man dann auf die Programmierer zugreifen, die älter als 50 Jahre sind und den Vornamen Xaver haben?

```
Select   *
From     Pers
Where    Vorname = ‘Xaver’
And      Beruf = ‘Programmierer’
And      Alter > 50
```

c) Welche Systemunterstützung ist möglich, um diesen Anfragetyp so schnell wie möglich auswerten zu können?

Indexdefinition für Vorname, Beruf und Alter

d) Da Erweiterbarkeit, NULL-Werte und Abbildbarkeit auf k DBMS ein großes Problem bleiben, untersuchen Sie die extreme Form einer Speicherungsstruktur mit AOW (Attribut von Objekt ist Wert), wobei in einer AOW-Tabelle prinzipiell nur solchermaßen aufgebaute Tripel gespeichert werden. Die O-Spalte enthalte systemweite eindeutige OIDs als Werte.
Wie sieht die Speicherungsstruktur für einen Satz dieser Tabelle aus?

A	O	W
Vorname	XYZ0815	Xaver

e) Formulieren Sie die vereinfachte obige Anfrage auf der AOW-Tabelle, wobei nur das Prädikat (Vorname = 'Xaver') verwendet wird. Die Ausgabe kann zunächst als Sub-Tabelle von AOW spezifiziert werden.

```
Select    T1.A, T1.O, T1.W
From      AOW   T1
Where     T1.O IN (
   Select      T2.O
   From       AOW T2
   Where      T2.A = 'Vorname'
   And        T2.W = 'Xaver'
   And        T2.O = T1.O
)
```

oder

```
Select    *
From      AOW   T1
Where     T1.O IN (
   Select      T2.O
   From       AOW T2
   Where      T2.A = 'Vorname'
   And        T2.W = 'Xaver'
)
```

oder

```
Select    T1.A, T1.O, T1.W
From      AOW   T1, AOW T2,
Where     T2.A = 'Vorname'
   And       T2.W = 'Xaver'
   And       T2.O = T1.O
```

oder

```
Select    T1.A, T1.O, T1.W
From      AOW   T1
Where     Exists (
   Select      *
   From        AOW T2
   Where       T2.A = 'Vorname'
   And         T2.W = 'Xaver'
   And         T2.O = T1.O
)
```

In der Ergebnis-Tabelle stehen die Tripel, die zum selben Xaver-Satz (von potentiell mehreren Xaver-Sätzen) gehören, noch untereinander. Mit einem OID-Join müßten diese Werte zu langen Sätzen (≥ 200 Feldern) verknüpft werden.

f) Wie lautet die Anfrage auf AOW mit (Vorname = 'Xaver' And Beruf = 'Programmierer' AndAlter > 50), die das äquivalente Ergebnis zur Anfrage auf Pers erzielt? Da das in SQL sehr aufwendig ist, schränken Sie die Ausgabe auf die explizit aufgelisteten Attribute Pnr, Vorname, Name, Beruf und Alter ein.

```
Select    P.W, Vor.W, Nam.W, Ber.W, Alt.W, ...
From      AOW P,
          AOW Vor,
          AOW Nam,
          AOW Ber,
          AOW Alt,
          . . .
Where     P.O = Vor.O      And P.A = 'Pnr'       And Vor.A = 'Vorname'
  And     P.O = Nam.O                            And Nam.A = 'Name'
  And     P.O = Ber.O                            And Ber.A = 'Beruf'
  And     P.O = Alt.=O              And Alt.A = 'Alter'
          . . .
  And     Vor.W = 'Xaver'
  And     Ber.W = 'Programmierer'
  And     Alter.W > 50
```

g) Wie wird dynamische Erweiterbarkeit gelöst?

Dynamische Erweiterbarkeit (Schemaevolution) ist eingebaut. Es ist also nur ein Insert mit dem entsprechenden AOW-Satz erforderlich und dies kann für jeden Pers-Satz zum Modifikationszeitpunkt erfolgen.

h) Die bisherige Darstellung der W-Spalte war stark vereinfacht, da ja Werte verschiedenen Typs in dieser Spalte auftreten. Das wird u. a. auch zum Problem, wenn man auf den Werten (vom gleichen Typ) Indexstrukturen einführen will. Wie kann dieses Problem gelöst werden?

Die Darstellung von W wird künstlich auf W1 bis W5 erweitert, wobei den Wi die Typen Integer, Float, Money, Varchar und Own zugewiesen werden (in der Realisierung des erwähnten Software-Herstellers). In einem Satz ist nur der zutreffende Wert belegt. Die restlichen Wi enthalten NULL-Werte.

In der A-Spalte werden nicht die vollen Attributnamen gespeichert, sondern systeminterne Kürzel, also beispielsweise

A	O	W1	W2	W3	W4	W5
A15	XYZ0815	NULL	NULL	NULL	Xaver	NULL

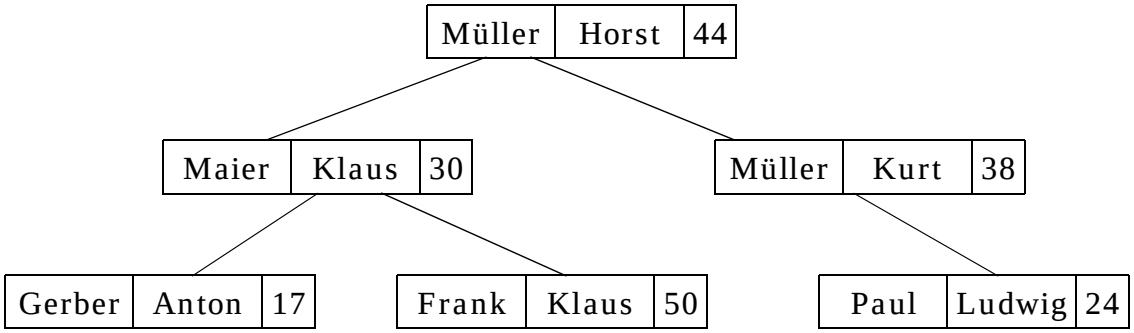
Jetzt sind die Wi-Spalten homogen und es können Indexstrukturen darauf definiert werden.

Diese Technik wird bei einem großen Software-Hersteller angewendet, um Generalisierungshierarchien (mit multipler Vererbung) in relationalen DBMS zu repräsentieren. Dabei kommen sehr viele Attribute vor, wobei häufig dynamische Erweiterungen anfallen. Die Struktur der Generalisierungshierarchie ist in den (sehr langen) Werten der OIDs codiert.

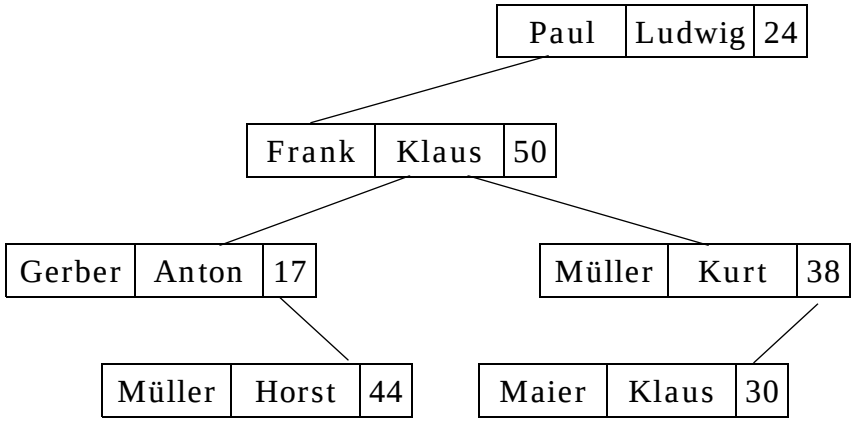
Aufgabe 2: k-d-Bauma)

a) Speichern Sie folgende Tupel in einem 3-d-Baum (k-d-Baum mit k = 3) mit den Schlüsselteilen „Nachname“, „Vorname“ und „Alter“ ab.

	Nachname	Vorname	Alter
1.	Müller	Horst	44
2.	Maier	Klaus	30
3.	Müller	Kurt	38
4.	Gerber	Anton	17
5.	Frank	Klaus	50
6.	Paul	Ludwig	24



b) Speichern Sie zum Vergleich die Tupel in umgekehrter Reihenfolge (6 -> 1) ab.



c) Suchen Sie in den sich aus a) und b) ergebenden Bäumen Tupel, deren Schlüssel die folgenden Werte haben:

1. Müller
2. Gerber, Anton

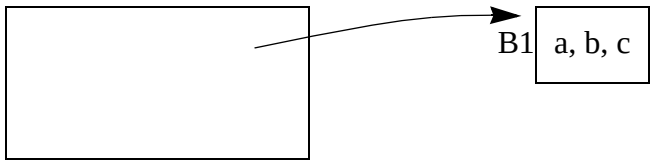
Aufgabe 3: Grid-File

Machen Sie sich die dynamischen Reorganisationen im GRID-File bei Einfügungen und Löschungen anhand des folgenden Beispiels klar. Ein Bucket kann jeweils 3 Sätze aufnehmen, das GRID-File ist anfangs leer. Von der Relation AUTO (KFZ-NR, MARKE, FARBE) sollen die beiden Attribute MARKE und FARBE mit der GRID-File-Technik abgespeichert werden.

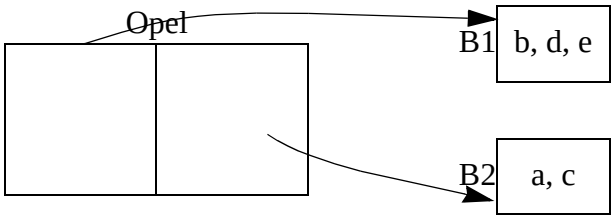
- Zeichnen Sie alle wesentlichen Strukturen (*Directory*, Suchraum, *Buckets*) nach jeder der unten angegebenen Einfügungen.
- Löschen Sie danach die ersten 4 Einträge und zeichnen Sie die obigen Strukturen auf.

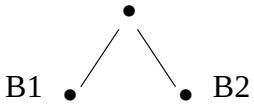
	KFZ-NR,	MARKE,	FARBE
a)	KL-PP 1	OPEL	GELB
b)	PS-A17	FORD	BLAU
c)	KL-CX 33	OPEL	BLAU
d)	KIB-AM 13	BMW	ROT
e)	SB-F16	AUDI	GRÜN
f)	KL-DZ 12	ALFA	SCHWARZ
g)	ZW-AL 43	VW	WEISS
h)	HOM-C 1	FIAT	VIOLETT
i)	SLS-AF 47	AUDI	AZUR
j)	KL-DM 31	ALFA	BEIGE
k)	VW-FS 40	SAAB	BRAUN

Eintragung a, b, c

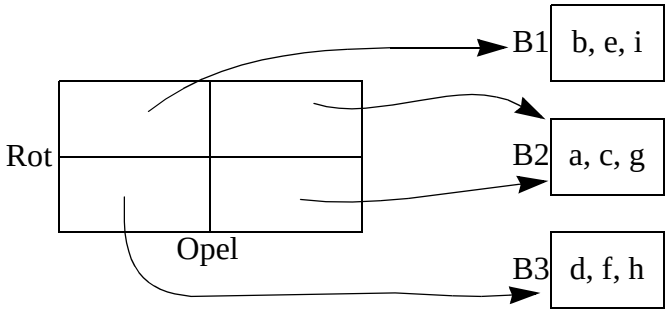


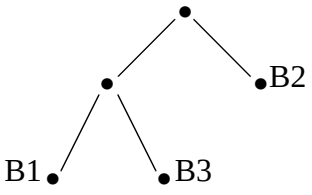
Eintragung von d führt zu Split bei „>= Opel“,
Eintragung e



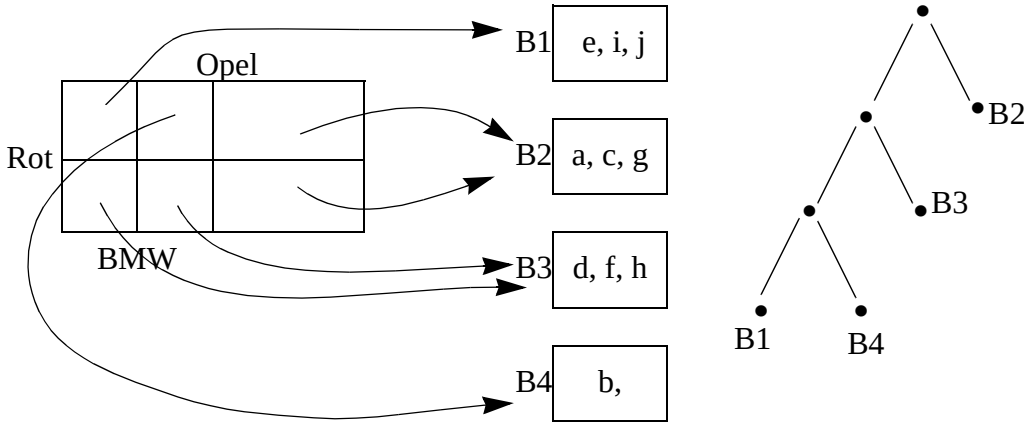


Eintragung f führt zu Split bei „>= Rot“,
Eintragung g, h, i

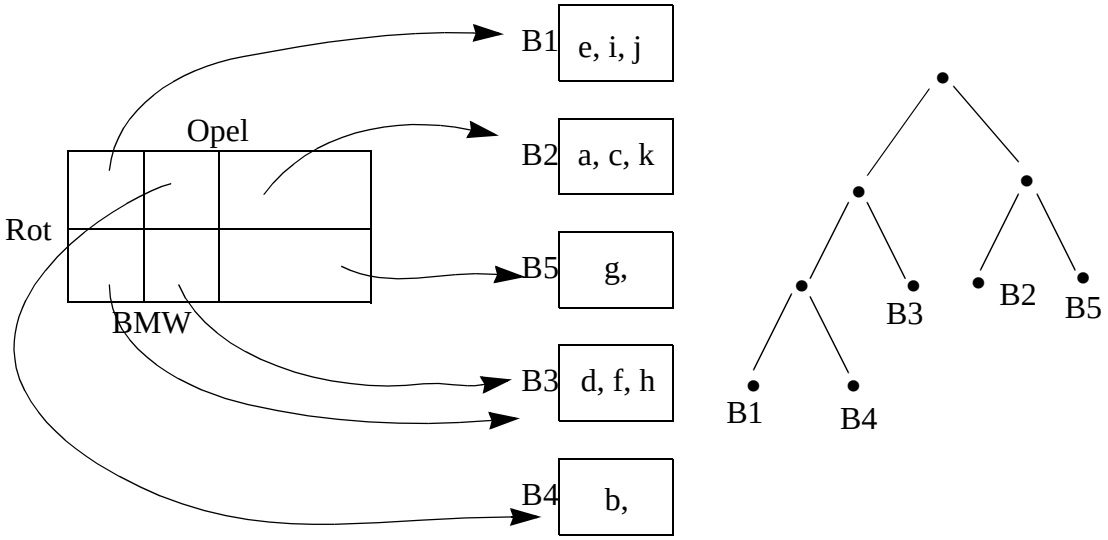




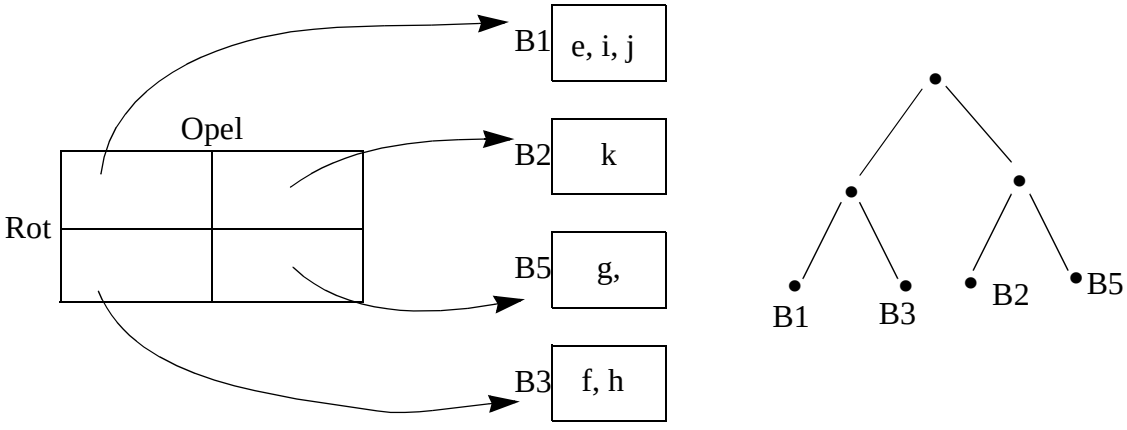
Eintragung j führt zu Split bei „>= BMW“,



Eintragung von k führt zu Split bei „>= Rot“



Nach dem Löschen der ersten 4 Einträge sieht die Struktur wie folgt aus:
B2 und B5 werden nicht verschmolzen, da bei einer erneuten Einfügung dieselbe Aufteilung wieder durchgeführt werden müßte. B2 und B5 werden nur dann verschmolzen, wenn gleichzeitig auch B1 und B3 verschmolzen werden kann.



Nur verschmelzen, wenn:
1. die Buckets Zwillinge im Baum sind und
2. ein Bucket leer ist oder eine Trennlinie in einer Dimension verschwindet(, was aber sehr schwer zu überprüfen ist).