



## Übung zur Vorlesung *Grundlagen: Datenbanken* im WS14/15

Harald Lang (harald.lang@in.tum.de)

<http://www-db.in.tum.de/teaching/ws1415/grundlagen/>

### Blatt Nr. 8

**Am 17.12.2014 findet anstelle der Vorlesung eine Zentralübung statt.**

### Hausaufgabe 1

Gegeben sei das folgende Schema:

- Familie: {[Opa, Oma, Vater, Mutter, Kind]}

Hierbei sei vereinfachend vorausgesetzt, dass Personen eindeutig durch ihren Vornamen identifiziert werden. Für ein Tupel [Theo, Martha, Herbert, Maria, Else] soll gelten, dass Theo und Martha entweder die Eltern von Herbert oder von Maria sind – die Großeltern werden also immer als Paar gespeichert, ohne dass ersichtlich ist, ob es die Großeltern väterlicher- oder mütterlicherseits sind. Wir gehen weiterhin davon aus, dass zu einem Kind immer beide Elternteile und beide Großeltern-Paare (also sowohl mütterlicherseits als auch väterlicherseits) bekannt sind.

1. Bestimmen Sie alle FDs und MVDs.
2. Überführen Sie die Relation in die 4. Normalform.

### Anmerkungen:

- Wir erlauben Scheidungen, wir sind Realisten.
- Wir vernachlässigen anderweitig unethische oder unpraktische Praktiken wie etwa Samenspenden, inter-generations Heiraten, künstlich kreierte Embryonen.

Bei der Bestimmung der mehrwertigen Abhängigkeiten sollte man insbesondere die Komplementregel beachten. Diese besagt:

$$\alpha \twoheadrightarrow \beta \Rightarrow \alpha \twoheadrightarrow \mathcal{R} - \alpha - \beta.$$

Für das Stammbaumschema gelten folgende FDs und MVDs:

$\{\text{Kind}\} \rightarrow \{\text{Mutter, Vater}\}$		(1)
$\{\text{Kind, Oma}\} \rightarrow \{\text{Opa}\}$		(2)
$\{\text{Kind, Opa}\} \rightarrow \{\text{Oma}\}$		(3)
$\{\text{Mutter, Vater}\} \twoheadrightarrow \{\text{Kind}\}$		(4)
$\{\text{Mutter, Vater}\} \twoheadrightarrow \{\text{Oma, Opa}\}$	Komplementregel zu MVD (4)	(5)
$\{\text{Kind}\} \twoheadrightarrow \{\text{Oma, Opa}\}$	Komplementregel zu FD (1)	(6)
$\{\text{Kind, Oma}\} \twoheadrightarrow \{\text{Mutter, Vater}\}$	Komplementregel zu FD (2)	(7)
$\{\text{Kind, Opa}\} \twoheadrightarrow \{\text{Mutter, Vater}\}$	Komplementregel zu FD (3)	(8)

Man erhält z.B. MVD (6) aus FD (1), indem man die Komplementregel anwendet: Es gilt nach FD (1)  $\{\text{Kind}\} \rightarrow \{\text{Mutter, Vater}\}$ . Also insbesondere  $\{\text{Kind}\} \twoheadrightarrow \{\text{Mutter, Vater}\}$ . Daraus folgt  $\{\text{Kind}\} \twoheadrightarrow \{\text{Oma, Opa}\}$ .

Die Zerlegung der Relation erfolgt nun nach dem 4. NF Zerlegungsalgorithmus. Hierzu muss eine geltende, nicht-triviale MVD gefunden werden, deren linke Seite nicht Schlüssel der Relation ist. Ein Beispiel hierfür ist die MVD  $\{\text{Mutter, Vater}\} \twoheadrightarrow \{\text{Kind}\}$ . Zerlegt man die ursprüngliche Relation anhand dieser MVD, so ergeben sich die zwei Relationen

$$R_1 : \{\text{Mutter, Vater, Kind}\} \text{ sowie}$$

$$R_2 : \{\text{Mutter, Vater, Oma, Opa}\}, \text{ oder } \{\text{Mutter, Vater, Oma, Opa}\}.$$

Diese sind in 4. NF, womit der Algorithmus terminiert.

Eine Zerlegung anhand der MVD  $\{\text{Kind}\} \twoheadrightarrow \{\text{Mutter, Vater}\}$  ist ebenfalls möglich. Daraus ergeben sich die beiden Relationen

$$R_1 : \{\text{Kind, Mutter, Vater}\}$$

$$R_2 : \{\text{Kind, Oma, Opa}\}, \text{ oder } \{\text{Kind, Oma, Opa}\}.$$

Der Algorithmus terminiert auch hier bereits nach der ersten Iteration.

Eine weitere Lösung ergibt sich aus der FD(1)  $\{\text{Kind}\} \rightarrow \{\text{Mutter, Vater}\}$ , denn aus ihr lässt sich auch  $\{\text{Kind}\} \rightarrow \{\text{Mutter}\}$  und  $\{\text{Kind}\} \rightarrow \{\text{Vater}\}$  ableiten (Dekompositionsregel). Desweiteren gilt, dass jede FD auch eine MVD ist, also  $\{\text{Kind}\} \twoheadrightarrow \{\text{Mutter}\}$  und  $\{\text{Kind}\} \twoheadrightarrow \{\text{Vater}\}$ .

Zerlegt man die Relation anhand der MVD  $\{\text{Kind}\} \twoheadrightarrow \{\text{Mutter}\}$ , so erhält man

$$R_1 : \{\text{Kind, Mutter}\}$$

$$R_2 : \{\text{Kind, Vater, Oma, Opa}\}, \text{ oder } \{\text{Kind, Vater, Oma, Opa}\}.$$

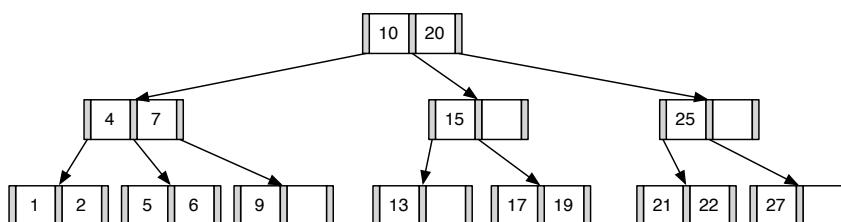
$R_2$  ist noch nicht in 4. NF. Eine weitere Zerlegung anhand  $\{\text{Kind}\} \twoheadrightarrow \{\text{Vater}\}$  oder  $\{\text{Kind}\} \twoheadrightarrow \{\text{Oma, Opa}\}$  gibt dann

$$R_1 : \{\text{Kind, Mutter}\}$$

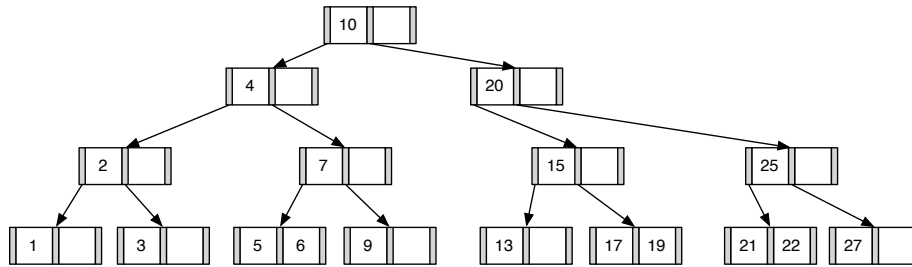
$$R_{2.1} : \{\text{Kind, Vater}\}$$

$$R_{2.2} : \{\text{Kind, Oma, Opa}\}, \text{ oder } \{\text{Kind, Oma, Opa}\}$$

## Hausaufgabe 2 (Klausuraufgabe aus dem WS2010!)

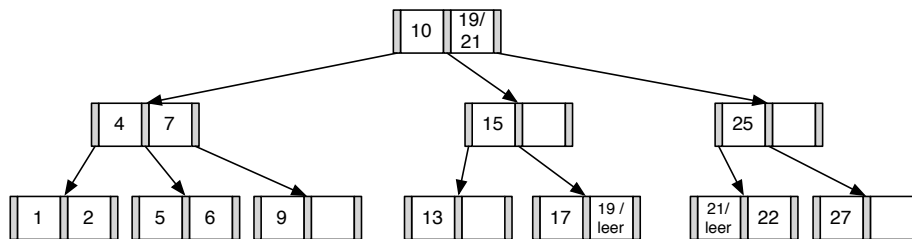


1. Fügen Sie die 3 in den gezeigten B-Baum ein. Zeichnen Sie das Endergebnis. Zeichnen Sie jeweils den kompletten Baum oder machen Sie **deutlich**, falls Teile des Baumes unverändert bleiben. Verwenden Sie den aus der Vorlesung bekannten Algorithmus. Das Ergebnis sieht wie folgt aus:



2. Entfernen Sie aus dem **ursprünglichen Baum** den Eintrag 20. Zeichnen Sie das Ergebnis der Operation. Sollte es mehrere richtige Lösungen geben, genügt es, wenn Sie hier eine angeben. Zeichnen Sie jeweils den kompletten Baum oder machen Sie **deutlich**, falls Teile des Baumes unverändert bleiben. Verwenden Sie den aus der Vorlesung bekannten Algorithmus.

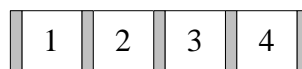
Das Ergebnis sollte wie folgt aussehen:



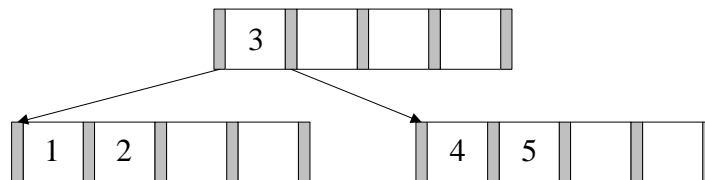
### Hausaufgabe 3

Fügen Sie in einen anfänglich leeren B-Baum mit  $k = 2$  die Zahlen eins bis zwanzig in aufsteigender Reihenfolge ein. Was fällt Ihnen dabei auf?

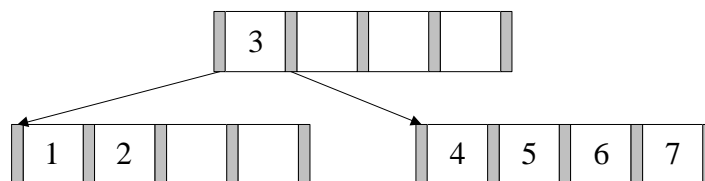
Nachdem man die Zahlen 1 bis 4 eingefügt hat, liegt folgender B-Baum vor:



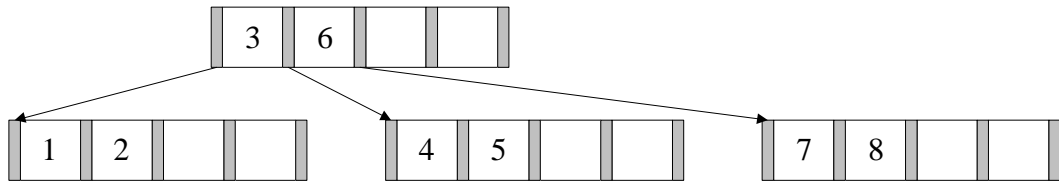
Beim Einfügen von 5 wird der Knoten gespalten und man erhält eine neue Wurzel.



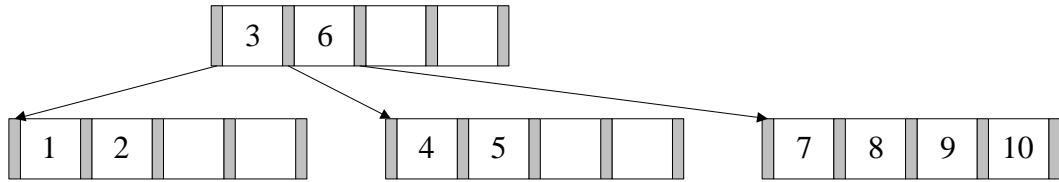
Die nächsten beiden Zahlen lassen sich wieder ohne Probleme einfügen.



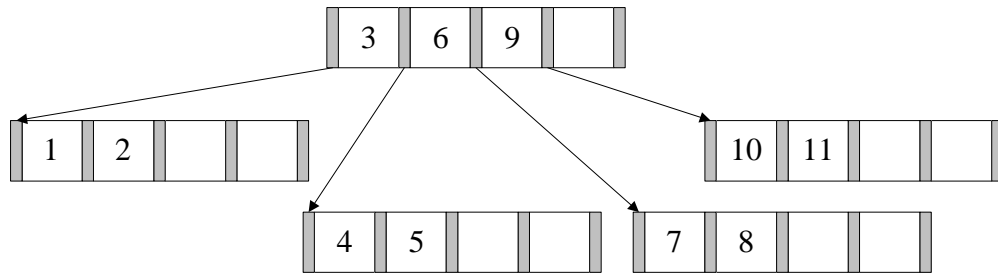
Beim Einfügen der 8 kommt es erneut zum Überlauf. Die 6 wandert in die Wurzel.



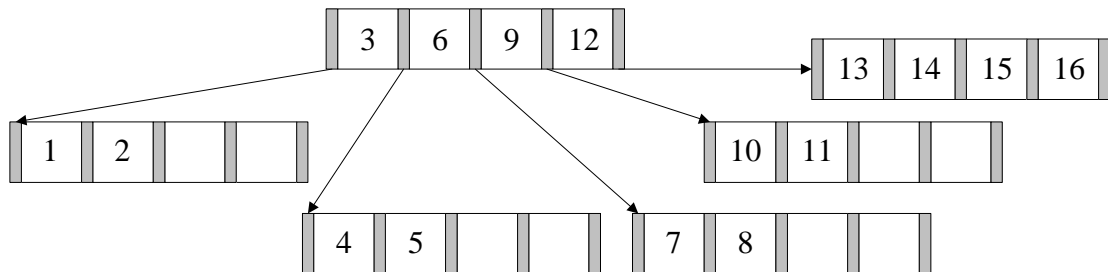
9 und 10 lassen sich wieder ohne Probleme einfügen. Bei 11 kommt es zum Überlauf.



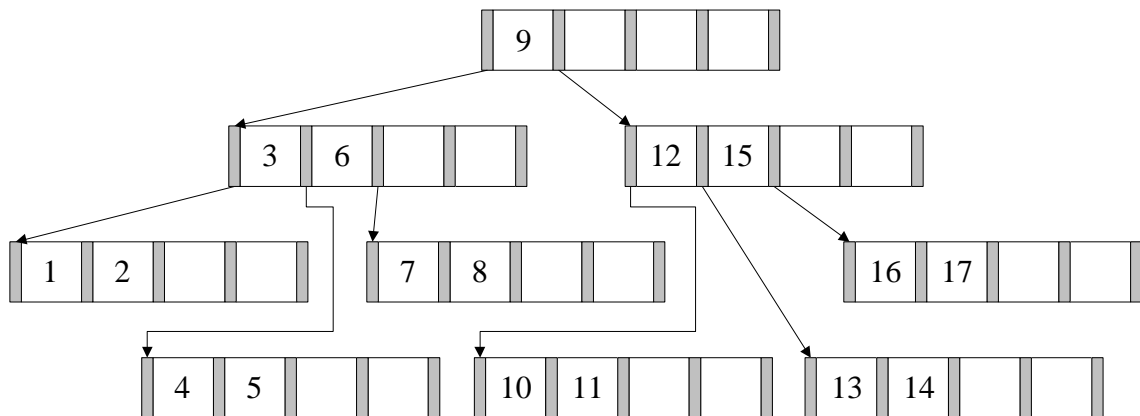
Nach dem Aufspalten erhält man dann:



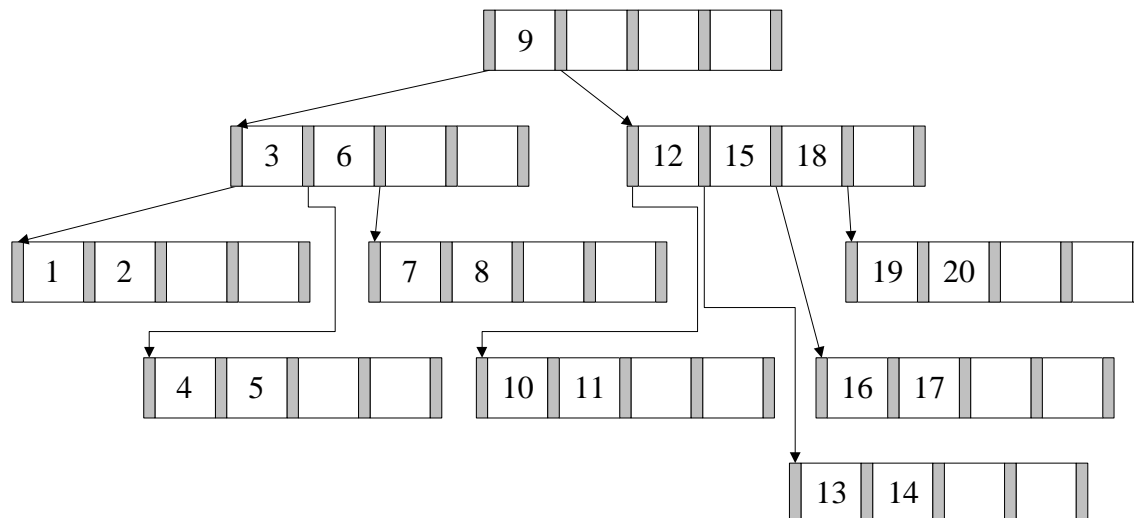
Es werden nun die nächsten Zahlen bis 16 analog eingefügt.



Bei 17 kommt es dann wieder zum Überlauf.



Fügt man nun noch die restlichen Zahlen ein, erhält man folgenden B-Baum:



Es fällt auf, dass der B-Baum nahezu minimale Auslastung aufweist. Dies liegt daran, dass eine aufsteigende Zahlenfolge sequentiell in den Baum eingefügt wird. Nach dem Aufspalten einer Seite in zwei Seiten werden dann in die Seite, die die kleineren Datensätze enthält, keine weiteren Werte mehr eingefügt. Allgemein ist das sortierte Einfügen der Schlüssel in einen B-Baum eine sehr schlechte Idee, da dies zu einer sehr geringen Auslastung führt.

#### Hausaufgabe 4

Bestimmen Sie  $k$  für einen B-Baum, der die folgenden Informationen aller Menschen auf der Erde (ca. 10 Milliarden) enthalten soll: Namen, Land, Stadt, PLZ, Straße und Hausnummer (insgesamt ca. 100 Byte). Dabei ist die Steuernummer eindeutig und 64 Bit lang und wird im B-Baum als Suchschlüssel verwendet. Gehen Sie bei der Berechnung davon aus, dass eine Speicherseite 16KiB groß ist und ein Knoten des B-Baums möglichst genau auf diese Seite passen sollte.

Zunächst gibt uns die Information über die zu erwartende Anzahl von Einträgen im B-Baum einen Hinweis auf die Größe eines Verweises in den Speicher. Wir betrachten gängige Architekturen und sehen, dass eine Speicherung auf einer Maschine, deren Adressierbarer Speicher lediglich  $2^{32}$  byte groß ist, nicht ohne weiteres möglich ist. Wir gehen im Verlauf der Aufgabe also von einer 64 bit Architektur aus, bei der ein Zeiger eine Größe von 8 byte hat.

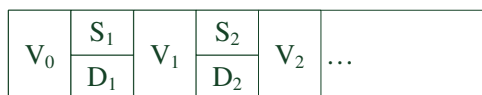


Abbildung 1: Struktur eines B-Baum Knotens

Um nun  $k$  zu bestimmen, muss die von  $k$  abhängige Größe eines Knotens maximiert werden, so dass dieser gerade noch auf eine Seite der Größe 16KiB passt. Die Struktur eines solchen Knotens ist in Abbildung 1 dargestellt. Zu beachten ist, dass ein komplett gefüllter Knoten genau  $2k$  Schlüssel/Daten Paare, jedoch  $2k + 1$  Verweise speichern muss.

Die Berechnung ergibt sich wie folgt:

$$\begin{aligned}2k * (\text{Schlüsselgröße} + \text{Datengröße}) + (2k + 1) * \text{Zeigergröße} &\leq 16KiB \\2k * (8\text{bytes} + 100\text{bytes}) + (2k + 1) * 8\text{bytes} &\leq 16384\text{bytes} \\2k * 116\text{bytes} + 8\text{bytes} &\leq 16384\text{bytes} \\2k &\leq 16376\text{bytes}/116\text{bytes} \\k &\leq (16376\text{bytes}/116\text{bytes})/2 \approx 70,59\end{aligned}$$

Da der Knoten „innerhalb“ einer Speicherseite liegen soll und daher nicht überlappen darf, sollte  $k$  idealerweise auf 70 gesetzt werden, keinesfalls auf 71.

### Zusatzaufgabe 1 (wird nicht in der Übung besprochen)

Betrachten Sie ein abstraktes Relationenschema  $\mathcal{R} = \{M, N, V, T, P, PN\}$  mit den FDs

- $M \rightarrow M$
- $M \rightarrow N$
- $V \rightarrow T, P, PN$
- $P \rightarrow PN$

1. Bestimmen Sie alle Kandidatenschlüssel.
2. In welcher Normalform befindet sich die Relation?
3. Bestimmen Sie zu den gegebenen FDs die kanonische Überdeckung.
4. Falls nötig, überführen Sie die Relation verlustfrei und abhängigkeitsbewahrend in die dritte Normalform.

### Kandidaten Schlüssel

$$\{V, M\}$$

### Normalform

1. NF

### Kanonische Überdeckung

Linksreduktion:

$$\begin{aligned}M &\rightarrow M \\M &\rightarrow N \\V &\rightarrow T, P, PN \\P &\rightarrow PN\end{aligned}$$

Rechtsreduktion:

$$\begin{aligned}M &\rightarrow \emptyset \\M &\rightarrow N \\V &\rightarrow T, P \\P &\rightarrow PN\end{aligned}$$

Zusammenfassen der übrigen FDs:

$$\begin{aligned}M &\rightarrow N \\M &\rightarrow \emptyset \\V &\rightarrow T, P \\P &\rightarrow PN\end{aligned}$$

Entfernen von FDs mit leere Menge auf der rechten Seite:

$$\begin{aligned}M &\rightarrow N \\V &\rightarrow T, P \\P &\rightarrow PN\end{aligned}$$

### Synthesealgorithmus

Ergebnis:

$$\{\{\underline{M}, N\}\}, \{\{\underline{V}, T, P\}\}, \{\{\underline{P}, PN\}\}, \{\{\underline{V}, M\}\}$$