DHBW Stuttgart

Datenbanken I

Kapitel 9 – Interne Speicherung der Daten

Modul: T2INF2004

Hinweis

Inhalte Vorlesung

1. Grundlagen und Begriffsdefinitionen
2. Der konzeptionelle Datenbankentwurf (ER-Modell)
3. Der relationale Entwurf (logischer Entwurf)
4. Die relationale Entwurfstheorie (Normalformen)
5. Einführung zum Datenbankentwurf

6.1 Die Sprache SQL (Teile DDL und DML)

7. Relationale Algebra (eine formale Sprache)

6.2 SQL-Teil2 (Teile DQL und Views)

8. Transaktion und Mehrbenutzersysteme
9. **Interne Speicherorganisation (Indexstrukturen)**

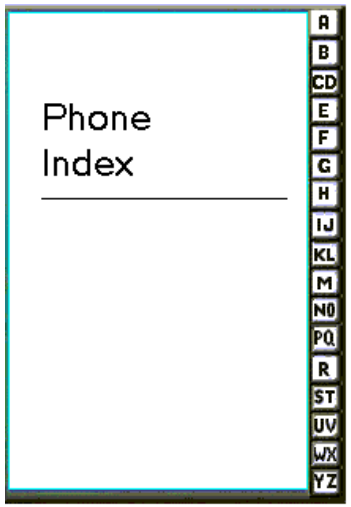
Speicherung von Daten

Speicherung und Zugriff auf Daten

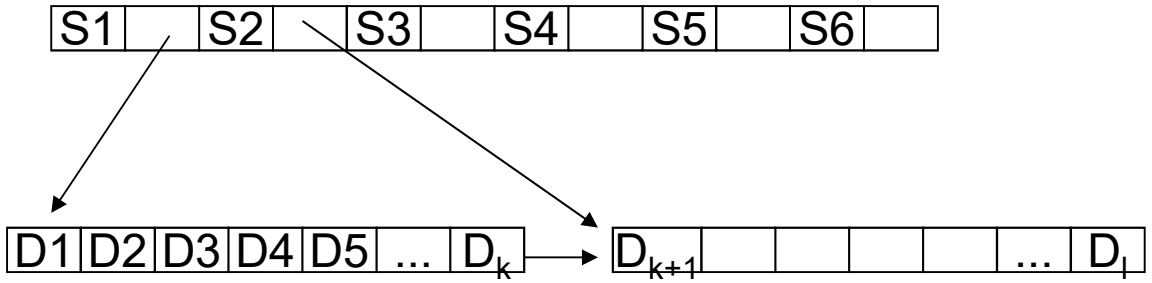
- Sequentielle Ablage (Sortierte Ablage auf einem Speichermedium z.B. Bandlaufwerk)
- Direkter Zugriff (Mit einer Adresse können Datensätze direkt angesprochen werden)
- Index-sequentieller Zugriff (Zugriff auf sequentielle gespeicherte Datensätze erfolgt über eine Indextabelle)
- Baumstruktur (Hierarchische Ablage von Datensätzen in Baumform)
- Schlüsselbäume (Zusammenfassung von Schlüssel und Ablage in B- oder Mehrfachbäumen)

Index-Sequentieller Zugriff (ISAM)

ISAM - Index-Sequential Access Method



⇨ Geordnetes Speichern der Datensätze und Schlüssel - Indexseiten sequentiell im Speicher abgelegt



S2 >= D1, ..., Dk > S1

⇨ *Suchen:* Binärsuche im Index - lineares Durchsuchen der Seite und ihrer Nachfolgeseiten bis zu einem „zu großen“ Schlüssel

⇨ *Einfügen:* Aufwändig! Bei voller Datenseite Ausgleich mit Nachbarseite versuchen, ansonsten neue Seite anlegen und Index verschieben

Infos und Regeln für B-Bäume

- Ein Baum besteht aus Knoten und Blättern.
- Jeder Knoten außer der Wurzel hat einen Elternknoten und mehrere (oder keinen) Kindknoten.
- Ein Knoten ohne Kinder heißt Blatt.
- Die Schlüsseleinträge in den Knoten sind sortiert.
- Jeder Knoten hat Zeiger auf seine Kinder.
- Zusätzlich zu den Zeigern enthält jeder Knoten noch weiter Informationen (Daten oder Zeiger auf Datensatz).
- B-Bäume („B“ auch manchmal für balanced) sind immer ausgeglichen.
- Alle Blattknoten sind auf dem selben Niveau.
- Bei einem Blattknoten sind die Zeiger auf die Kindknoten alle NULL.
- Jede Seite enthält zwischen k und 2k Elemente.
- Mit B-Bäumen werden Index-Strukturen in Datenbanken verwaltet.
- B-Bäume verringern die Anzahl der Zugriff gegenüber Binär-Bäumen.

B-Bäume

B-Bäume

⇨ Im Hauptspeicher können Binärbäume als Speicherstruktur verwendet werden - diese sind jedoch nicht effizient auf die Seitenstruktur des Hintergrundspeichers abbildbar.

⇨ B-Bäume (Bayer-Bäume) begrenzen die Zahl der Seitenzugriffe

Definition: B-Baum vom Grad k

Jeder Knoten hat mindestens $\lceil k/2 \rceil$ und höchstens k Schlüsseleinträge.

Der Wurzelknoten hat zwischen 1 und k Schlüsseleinträge.

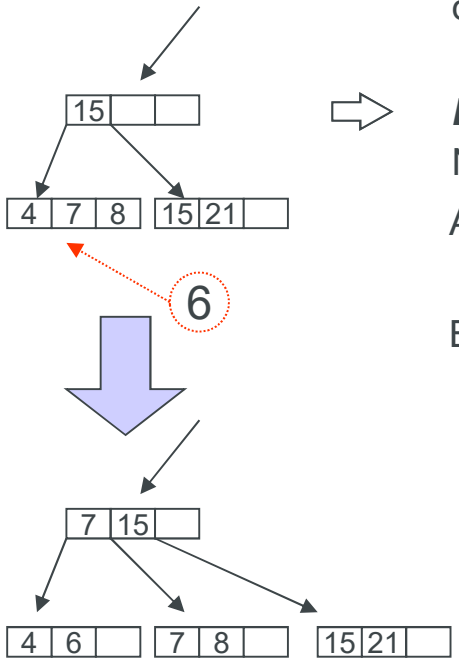
Die Schlüsseleinträge in den Knoten sind sortiert.

Jeder Knoten außer den Blattknoten, der s Schlüsseleinträge besitzt s+1 Kindknoten

Seien S1 ... Sn die Schlüsseleinträge eines Knotens und T0, ..., Tn die Verweise auf Kindknoten, dann gilt:

T0 verweist auf Kindknoten, dessen Schlüsseleinträge kleiner als S1 sind
Ti verweist auf Kindknoten, dessen Schlüsseleinträge >= Si und kleiner als Si+1 sind
Blattknoten besitzen keine weiteren Verweise

B-Bäume



⇨ ***Auffinden*** eines Datensatzes

Navigation durch den B-Baum. Anzahl der Seitenaufrufe durch Höhe des Baumes begrenzt. (logarithmische Komplexität!)

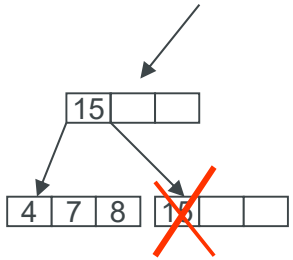
⇨ ***Eintragen*** eines Datensatzes

Navigation durch den B-Baum bis zur entsprechenden Seite s.

A) Seite s ist noch nicht voll : eintragen des Datensatzes und anpassen der Schlüsseleinträge in den Knoten bis zur Wurzel.

B) Seite s ist voll: Es wird eine neue Seite r angefordert und die Elemente von s, sowie das neue Element gleichmäßig auf s und r verteilt. Anschließend wird die Seite r als neues Element in den Vaterknoten von s eingefügt.

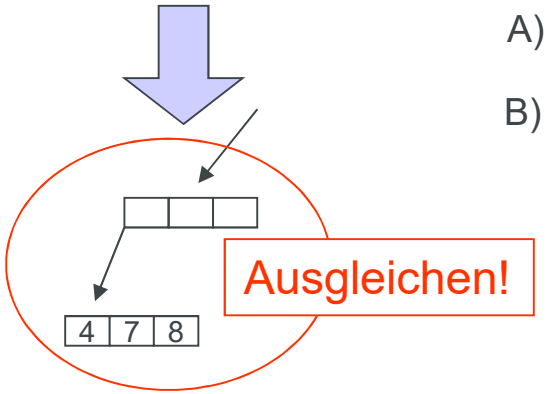
B-Bäume



⇨ ***Löschen*** eines Datensatzes

Navigation durch den B-Baum bis zum Auffinden des Datensatzes in einem Blattknoten s.

Wird der Datensatz im Blattknoten s gefunden, so wird er gelöscht. War es der erste Datensatz dieser Seite, so müssen die Schlüsseleinträge des B-Baums entlang des Pfades zur Wurzel überprüft werden. War es der letzte Datensatz in dieser Seite, so wird die Seite freigegeben und als Kindknoten aus dem Vaterknoten p des B-Baums gelöscht.



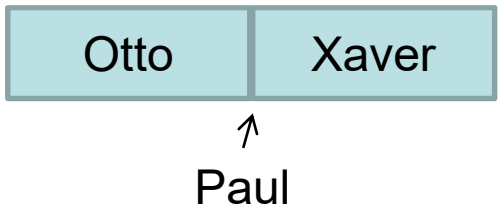
A) Knoten p hat noch mehr als $\lceil k/2 \rceil$ Kinder: löschen des Verweises.

B) Knoten p hat nun weniger als $\lceil k/2 \rceil$ Kinder: Ausgleichen mit Nachbarknoten der mehr als $\lceil k/2 \rceil$ Kinder hat. Gibt es keinen solchen, so wird p gelöscht und die Kindknoten dem Nachbarknoten zugeschlagen. (Rekursion!)

Aufbau eines B-Baumes

Wurzel mit 2 Schlüssel

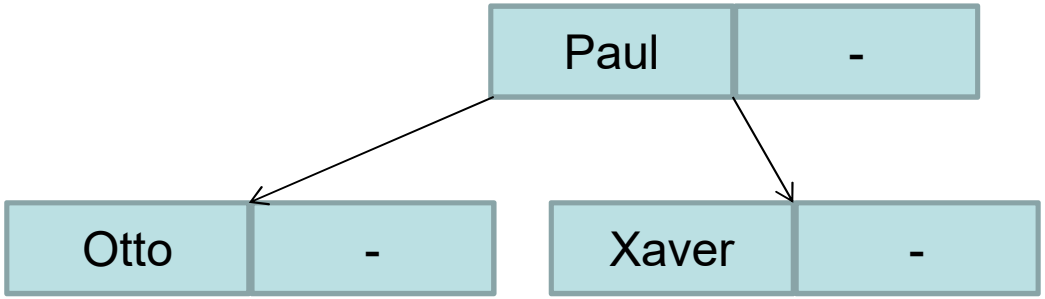
Einfügen eines neuen Schlüssels: Paul



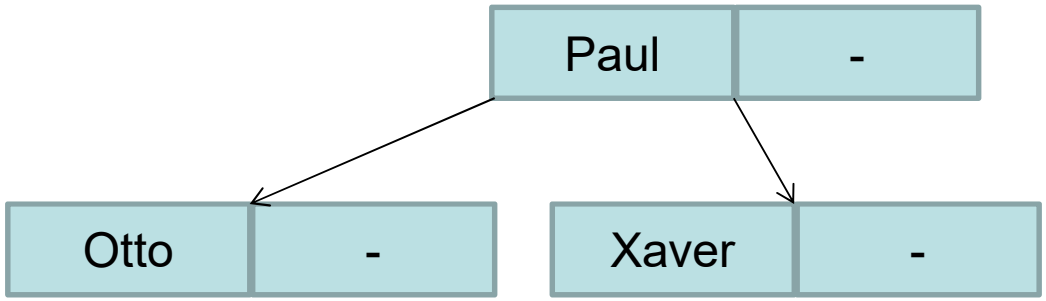
Aufbau eines B-Baumes

Wurzel mit 2 Schlüssel

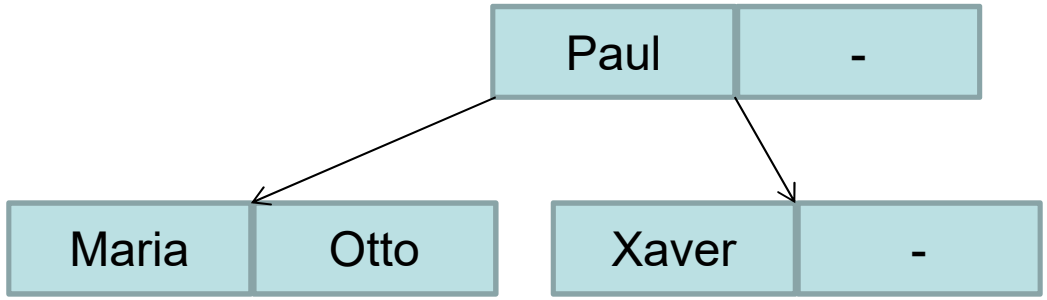
Einfügen eines neuen Schlüssels: Paul



Aufbau eines B-Baumes

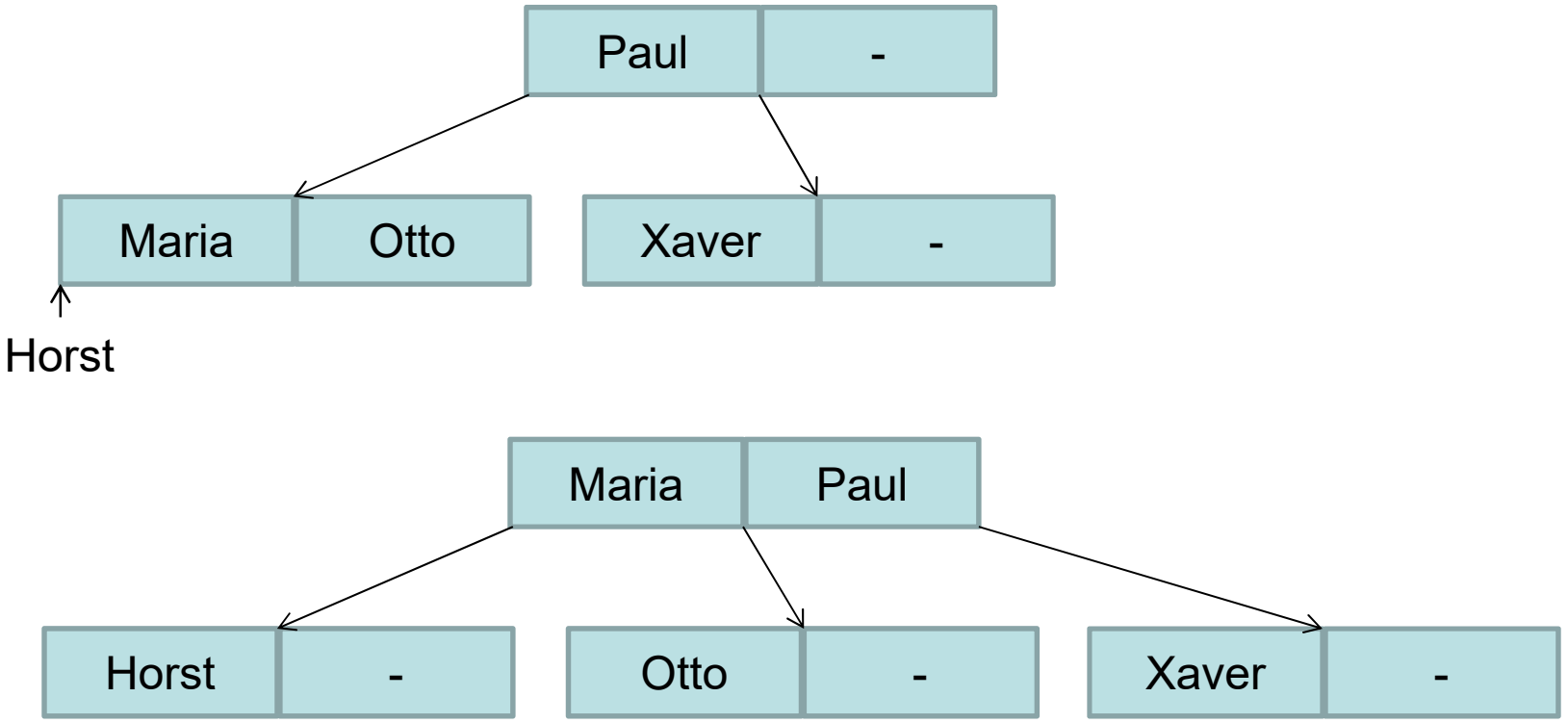


Einfügen weiterer Schlüssel: **Maria**, Horst



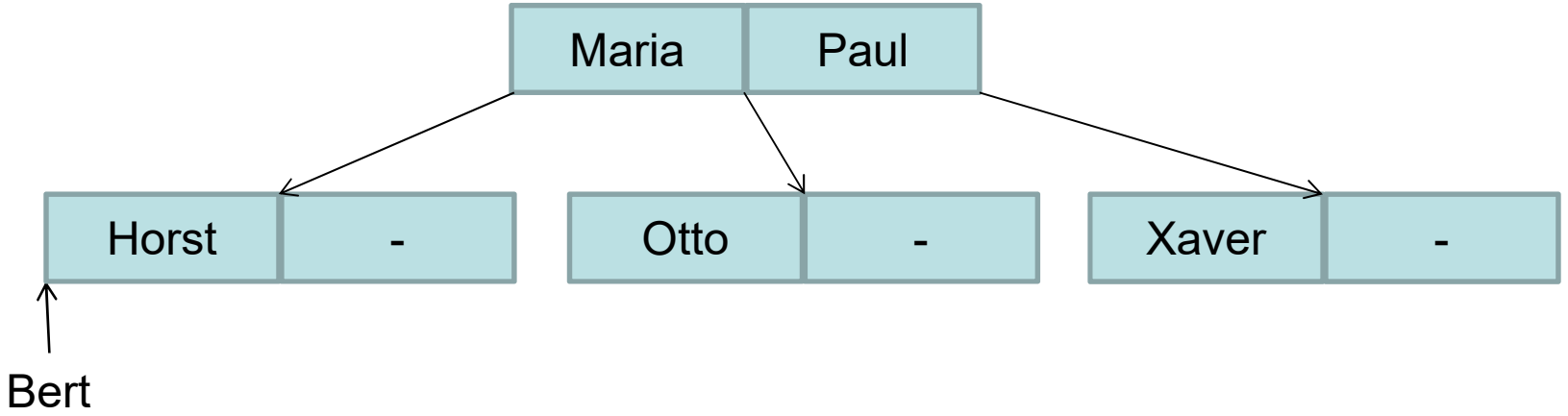
Aufbau eines B-Baumes

Einfügen weiterer Schlüssel: Maria, **Horst**



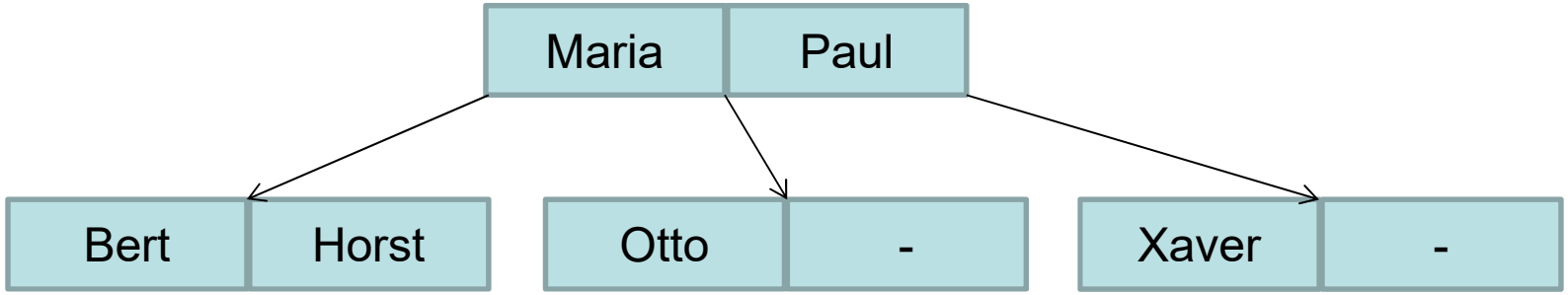
Aufbau eines B-Baumes

Einfügen weiterer Schlüssel: **Bert**, Oskar, Jonas



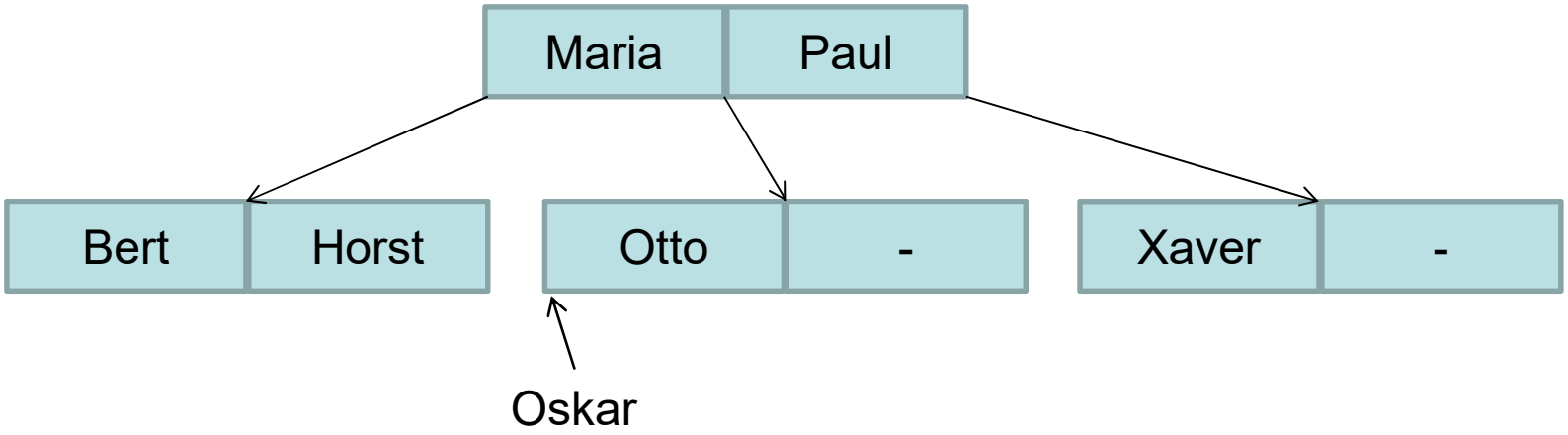
Aufbau eines B-Baumes

Einfügen weiterer Schlüssel: **Bert**, Oskar, Jonas



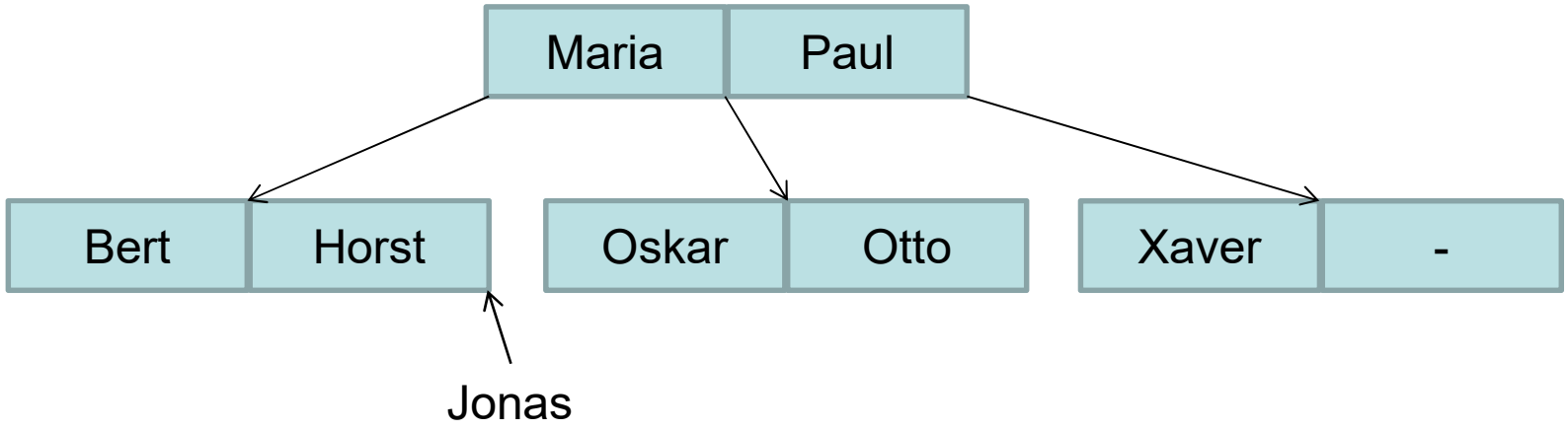
Aufbau eines B-Baumes

Einfügen weiterer Schlüssel: Bert, **Oskar**, Jonas



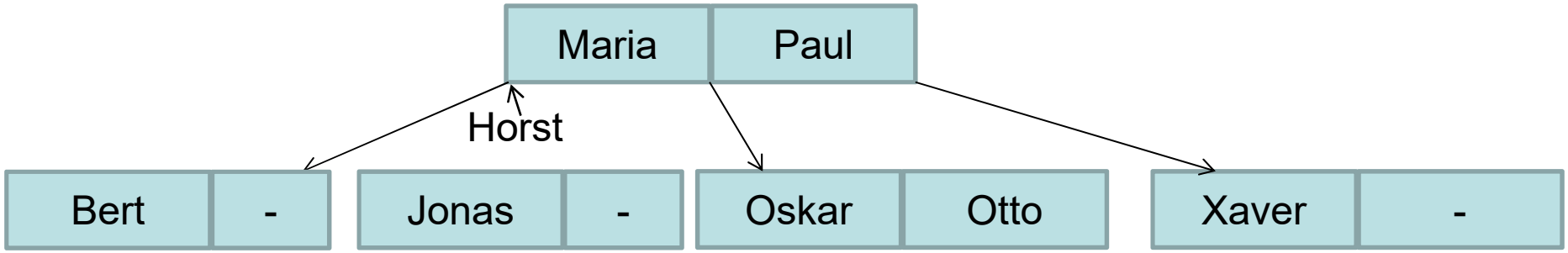
Aufbau eines B-Baumes

Einfügen weiterer Schlüssel: Bert, Oskar, **Jonas**

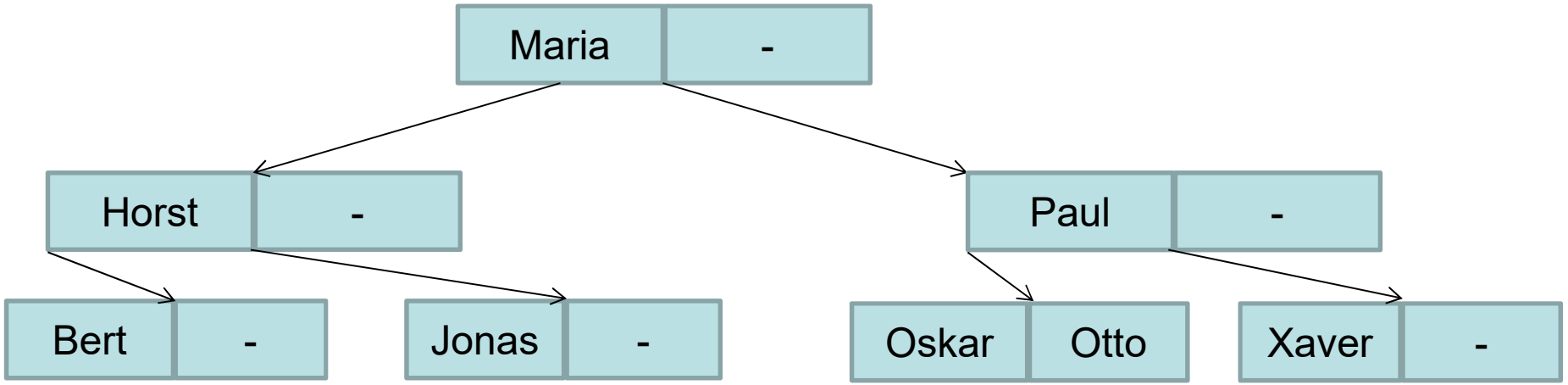


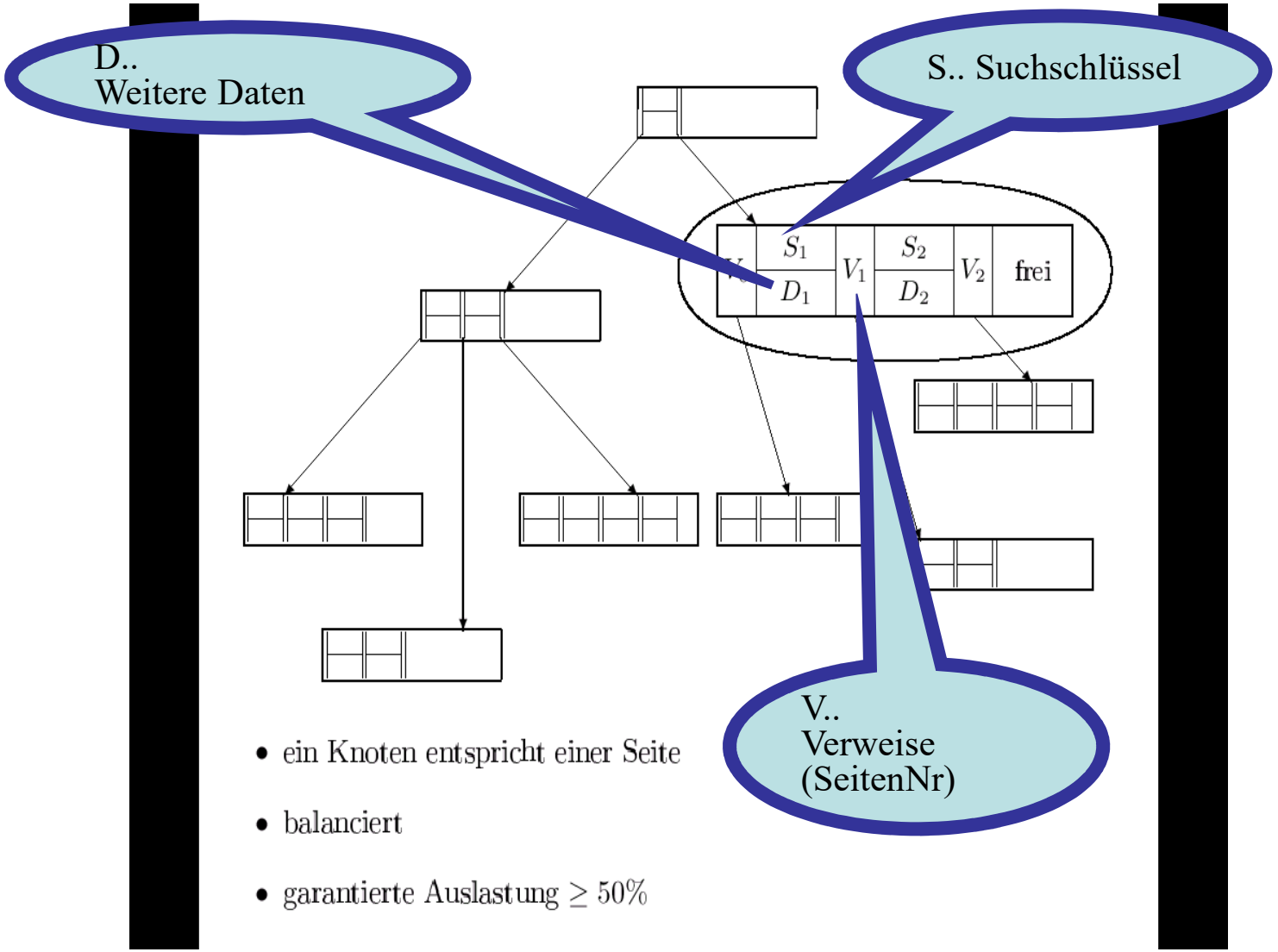
Aufbau eines B-Baumes

Einfügen weiterer Schlüssel: Bert, Oskar, **Jonas**



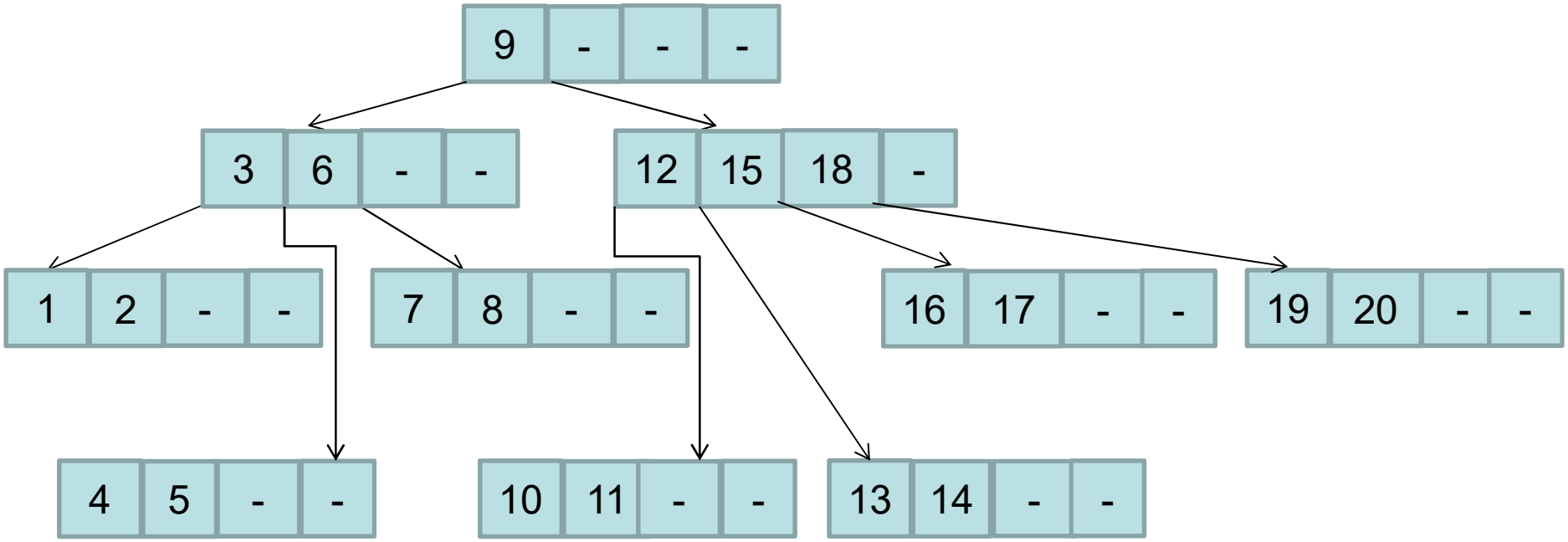
Aufbau eines B-Baumes




D..
Weitere Daten
S.. Suchschlüssel
S_1
D_1
V_1
S_2
D_2
V_2
frei
V..
Verweise
(SeitenNr)
• ein Knoten entspricht einer Seite
• balanciert
• garantierte Auslastung ≥ 50%

Aufbau eines B-Baumes

Übung: Fügen Sie in einen anfänglich leeren Baum mit K=4 die Zahlen eins bis zwanzig in aufsteigende Reihenfolge ein. Was fällt Ihnen dabei auf?



Indizes

- Beim Anlegen des Primärschlüssels wird automatisch ein Index angelegt.
- Beim Zugriff auf einen Datensatz über den Primärschlüssel kann dieser Index verwendet werden.
- Vorteil eines Index: Gewünschter Datensatz kann über wenige Zugriffe (im B-Baum) gefunden werden.
- Der Zugriff auf alle anderen Attribute der Tabelle erfolgt über einen Tabellenscan.
- Zugriffe über einen Tabellenscan können sehr performanceintensiv sein.
- Indizes können nachträglich auf alle Attribute der Tabelle angelegt werden.

```
CREATE INDEX [Index_Name]
ON TabellenName
      (Attribut);
```

- Der `Index_Name` ist nicht zwingend erforderlich.
- `Attribut` ist der Spaltennamen für den ein Index angelegt wird.
  Auch auf mehrere Spalten möglich.

Indizes

Szenario: Von der Buchhaltung des Softwarehauses wird immer wieder aus abrechnungstechnischen Gründen abgefragt, auf welche Auftrags- und Leistungs-Nummer zu einem bestimmten Projekten gebucht wurden .
Wie lautet dann hierzu die Abfrage?

Aufgabe:

1. Führen Sie das oben angegebene Select aus und lassen Sie sich vom Optimizer erklären wie er diese Abfrage ausführt.
2. Interpretieren Sie diesen Query Plan.

Indizes

- Bei häufigen Zugriffen und vielen Projekten wäre es sinnvoll auf das Attribut `zu_projekt` (Projektnummer) einen Index zu legen.

Aufgabe: Legen Sie selbstständig diesen Index an.

- Anzeigen/überprüfen des Index erfolgt durch anzeigen der Tabelle *PG-Indexes*.

```
Select * from PG_Indexes
where tablename ='leistung'
;
```

Indizes

Data Output | Explain | Notifications

Primärindex

	schemaname name	tablename name	indexname name	tablespace name	indexdef text
1	public	leistung	leistung_pkey	[null]	CREATE UNIQUE INDEX leistu...
2	public	leistung	pr_nr_leistung	[null]	CREATE INDEX pr_nr_leistung ...

Neuer Index

Aufgabe: Select nochmals ausführen und wieder den Plan anschauen.
Was fällt Ihnen auf?

- Der Optimizer sucht sich den für ihn günstigsten (Cost) Weg aus.
- Dies ist hier nicht die Verwendung des neuen Index.

Ob und wenn ja, in welcher Form ein Index zum Datenbankzugriff verwendet wird, ist die Entscheidung des DBMS und damit des Optimizers.

Indizes

Wie lässt sich nun die Verwendung eines Index demonstrieren?

- Wir verwenden die Datenbank „World_DB“ in der mehrere tausend Städte und die dazugehörigen Länder eingetragen sind.
- An der Tabelle „*City*“ der Datenbank zeigen wir die Verwendung eines Index.

Data Output | Explain | Notifications

	id [PK] integer	name text	countrycode character (3)	district text	population integer
1	1	Kabul	AFG	Kabol	1780000
2	2	Qanda...	AFG	Qandahar	237500
3	3	Herat	AFG	Herat	186800
4	4	Mazar-...	AFG	Balkh	127800
5	5	Amster...	NLD	Noord-Holl...	731200

Aufgabe: Führen Sie nachfolgenden Select aus und interpretieren Sie wieder.

Indizes

Aufgabe:

1. Legen Sie einen Index auf „city“ und führen Sie wieder den Select durch.
2. Führen Sie ein weiteres Select mit nachfolgender Bedingung aus.

```
where name  like 'Kaf%'
;
```

3. Interpretieren Sie die Ergebnisse wieder.

Indizes

Wird ein angelegter Index nicht verwendet so können die Statistiken, auf die der Optimizer zugreift, noch aktualisiert werden.
Dies erfolgt mit nachfolgendem Befehl.

```
analyze city
;
```

Ende Kapitel 9