Zugriffsbeschleunigung

- **Indexe**
- **Clustering (Datenballung)**

Indexe

Oft: **wenige Tupel als Anfrage-Ergebnis**

Aber: **alle Daten müssen durchsucht werden ohne Zusatzinfo**

Idee: **Ausnützen der Direktzugriffsmöglichkeit des Sekundärspeichers durch Index**

Ziel: **Schneller Zugriff auf bestimmte Attribute**

für viele Anfragetypen

mit wenig zusätzlichem Speicherbedarf

und niedrigem Pflegeaufwand

Indextypen

- **Primärindexe**
  legen physische Anordnung der Daten fest
  meist: Primärindex auf Primärschlüssel
- **Sekundärindexe**

➔ **1 Primärindex, mehrere Sekundärindexe**

Achtung: Suchkriterium Index nennt man Schlüssel
≠ Schlüssel Relation

```
create index SemesterInd on Studenten(Semester);
drop index SemesterInd;
```

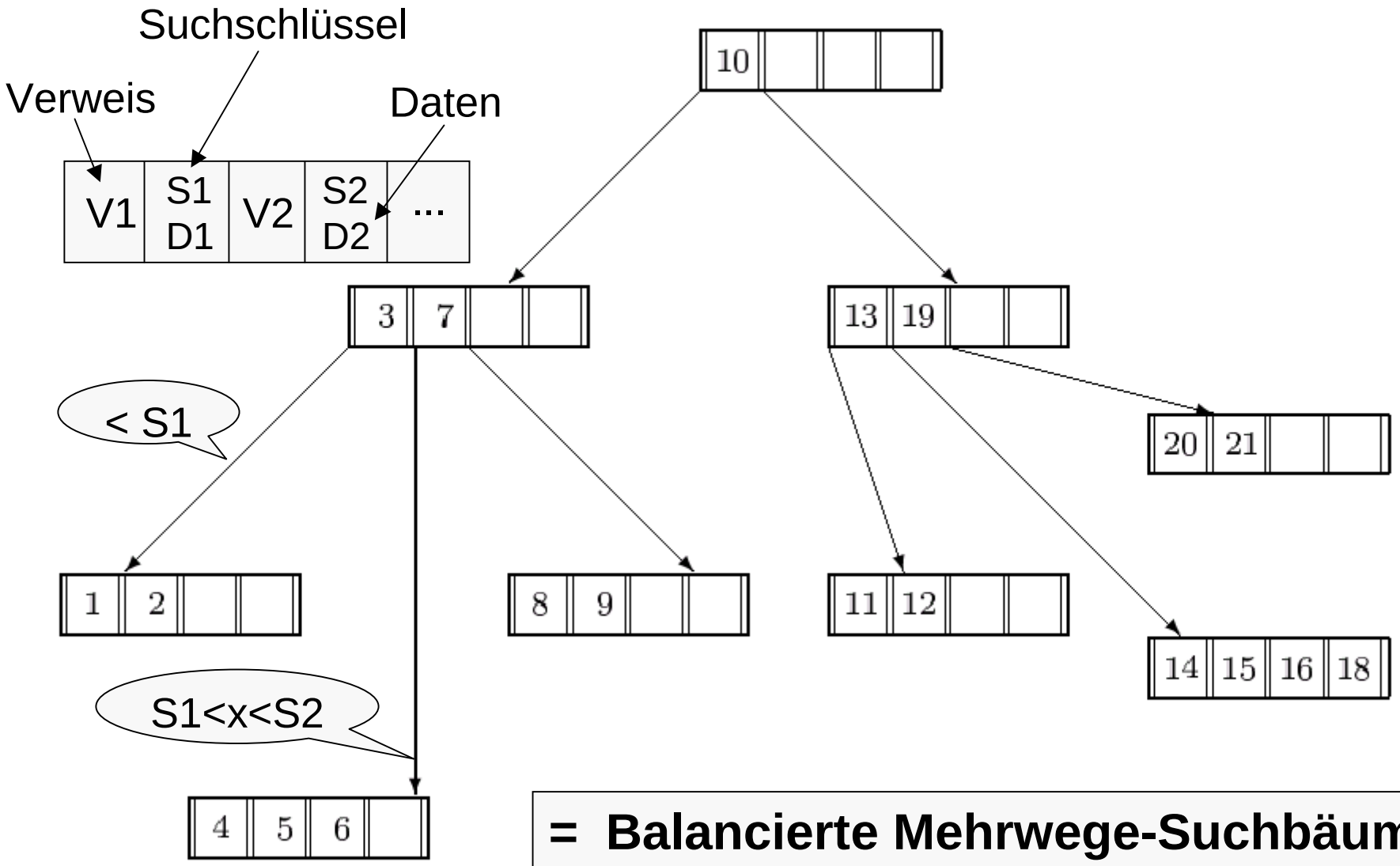
Arten von Indexen

- **Sequentielle Speicherstrukturen (ISAM)**
- **Baumstrukturen (Schlüsselvergleich)**
  Beispiel: B-Baum
- **Hashing (Schlüsseltransformation)**
  max. 2 Seitenzugriffe für ein Datum

Anfragearten:

- **exact-match query**
- **partial-match query**
- **range query**

B-Bäume



= Balancierte Mehrwege-Suchbäume für den Hintergrundspeicher

B-Baum von Grad k:

1. Jeder Weg von der Wurzel zu einem Blatt hat die gleiche Länge.
2. Jeder Knoten außer der Wurzel hat mindestens k und höchstens $2k$ Einträge. Die Wurzel hat höchstens $2k$ Einträge. Die Einträge werden in allen Knoten sortiert gehalten.
3. Alle Knoten mit n Einträgen, außer den Blättern, haben $n + 1$ Kinder.
4. Seien $S_1, \ldots, S_n$ die Schlüssel eines Knotens mit $n + 1$ Kindern. $V_0, V_1, \ldots, V_n$ seien die Verweise auf diese Kinder. Dann gilt:
   (a) V_0 weist auf den Teilbaum mit Schlüsseln kleiner als S_1.
   (b) V_i $(i = 1, \ldots, n - 1)$ weist auf den Teilbaum, dessen Schlüssel zwischen S_i und S_{i+1} liegen.
   (c) V_n weist auf den Teilbaum mit Schlüsseln größer als S_n.
   (d) In den Blattknoten sind die Zeiger nicht definiert.

Einfügen neues Objekt in B-Baum

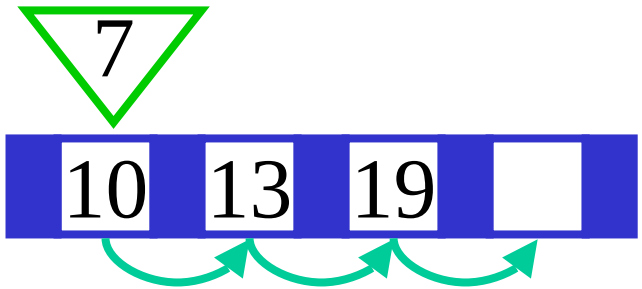
1. Führe eine Suche nach dem Schlüssel durch; diese endet (scheitert) an der Einfügestelle.

2. Füge den Schlüssel dort ein.

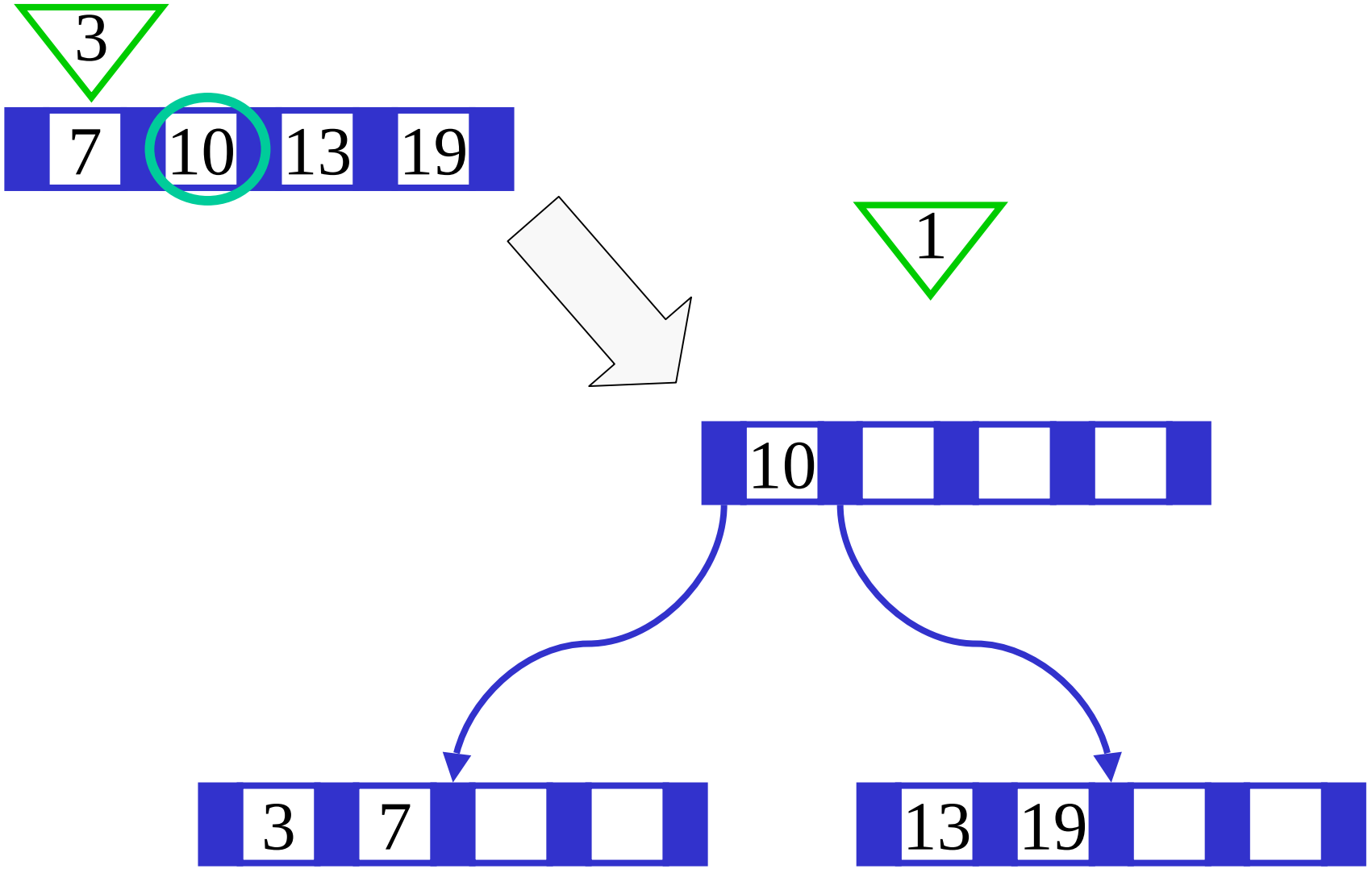
3. Ist der Knoten überfüllt, teile ihn
   - Lege einen neuen Knoten an und belege ihn mit den Schlüsseln, die rechts vom mittleren Eintrag des überfüllten Knotens liegen.
   - Füge den mittleren Eintrag im Vaterknoten des überfüllten Knotens ein.
   - Verbinde den Verweis rechts des neuen Eintrags im Vaterknoten mit dem neuen Knoten

4. Ist der Vaterknoten jetzt überfüllt?
   - Handelt es sich um die Wurzel, so lege eine neue Wurzel an.
   - Wiederhole Schritt 3 mit dem Vaterknoten.

Sukzessiver Aufbau eines B-Baums



Sukzessiver Aufbau eines B-Baums

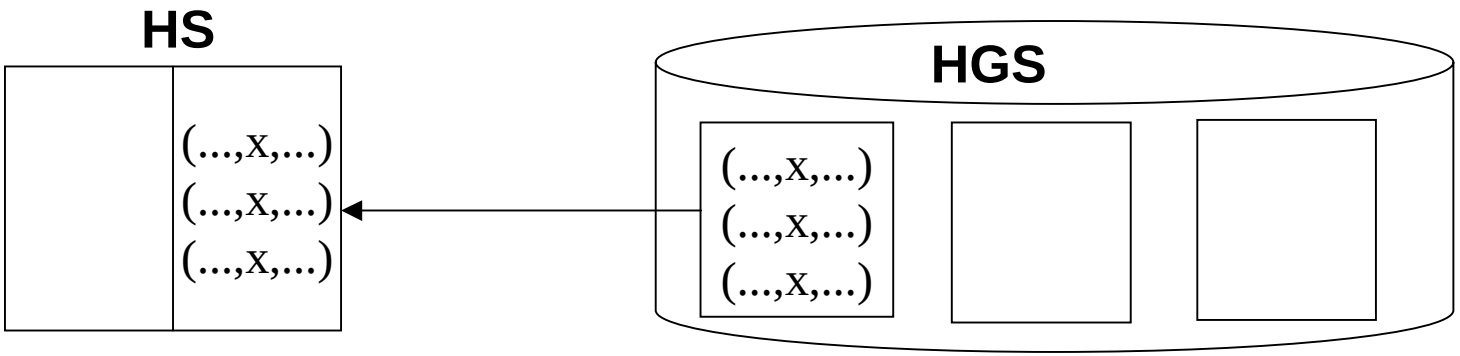


Clustering

Clustering =

Ballung logisch verwandter Datensätze, d.h. logisch zusammengehörige Daten werden räumlich zusammenhängend gespeichert (1 Seite)

➔Optimierung Zugriffs- und Ladezeiten



Primärindex legt physische Anordnung fest

➔Clustering kann genutzt werden