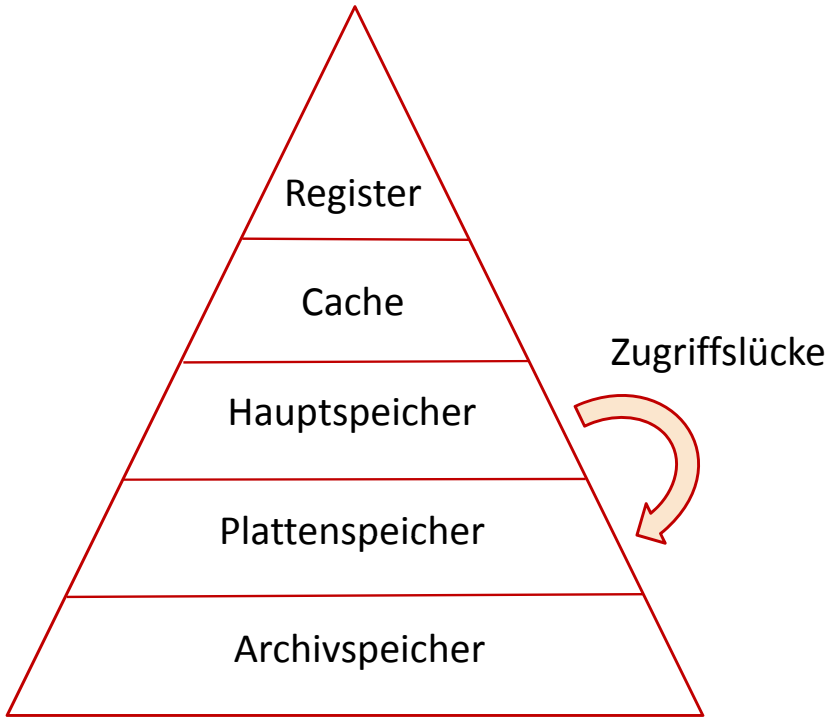
Indexe

DAS DYNAMISCHE INHALTSVERZEICHNIS EINER DATENBANK

Speicherhierarchie



Problem: Zugriffslücke (Zeit, mechanisch)

Ziel: schneller Zugriff und Transfer nur der Daten, die wirklich gebraucht werden (kleines Datenvolumen)

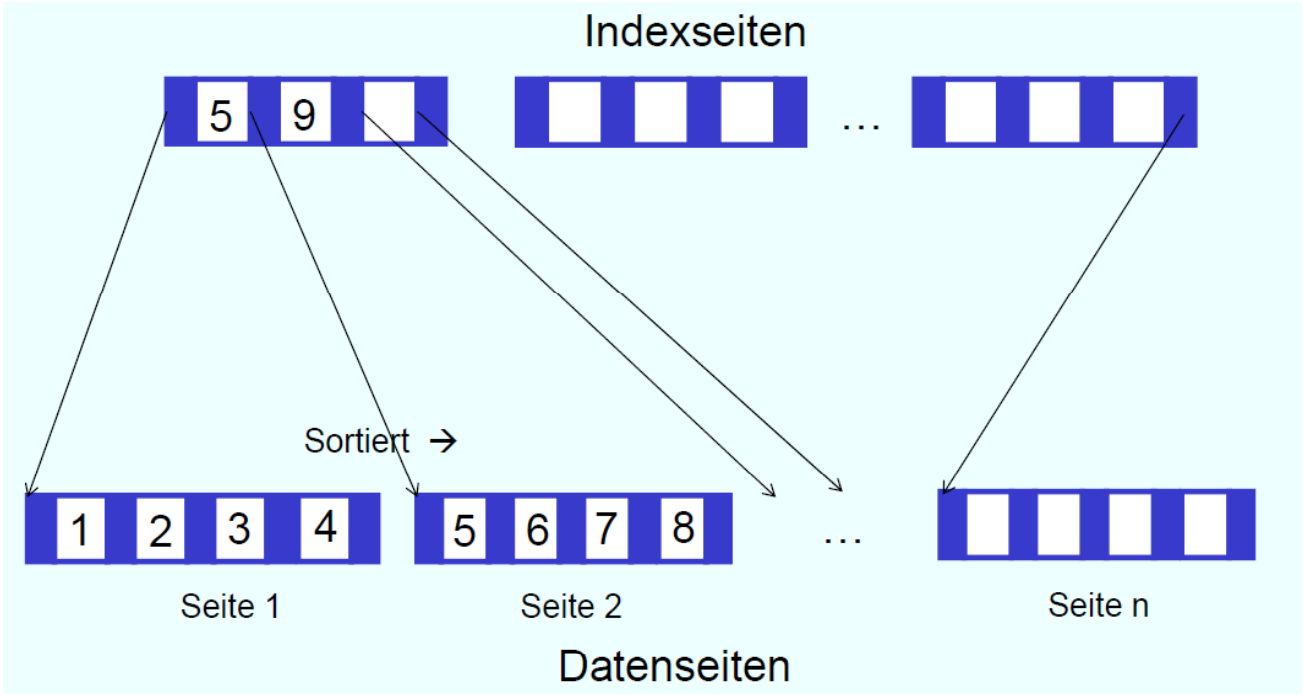
Zwei Ansätze:

Hierarchisch → Bäume

Partitionierung → Hashing

Hierarchische Indexstrukturen - ISAM

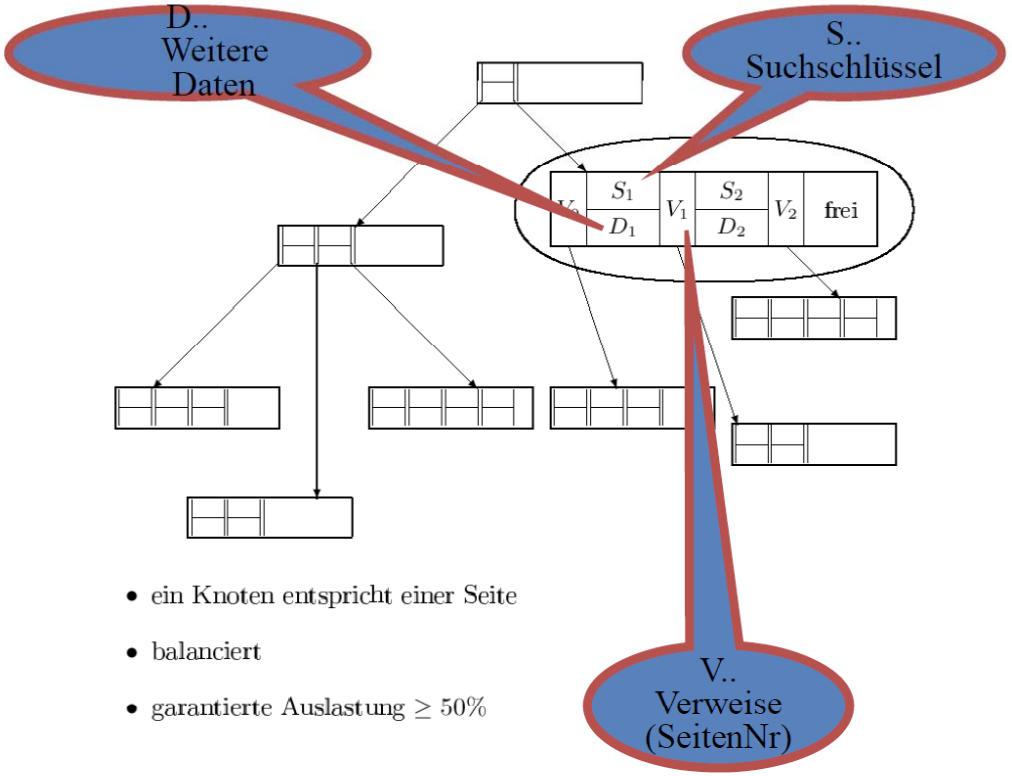
ISAM: Indexseiten und Datenseiten



Hierarchische Indexstrukturen – B-Bäume

ISAM

B-Bäume: Erstellung von Indexseiten für Indexseiten -> stärkere Untergliederung durch Knoten -> Zugriff schneller

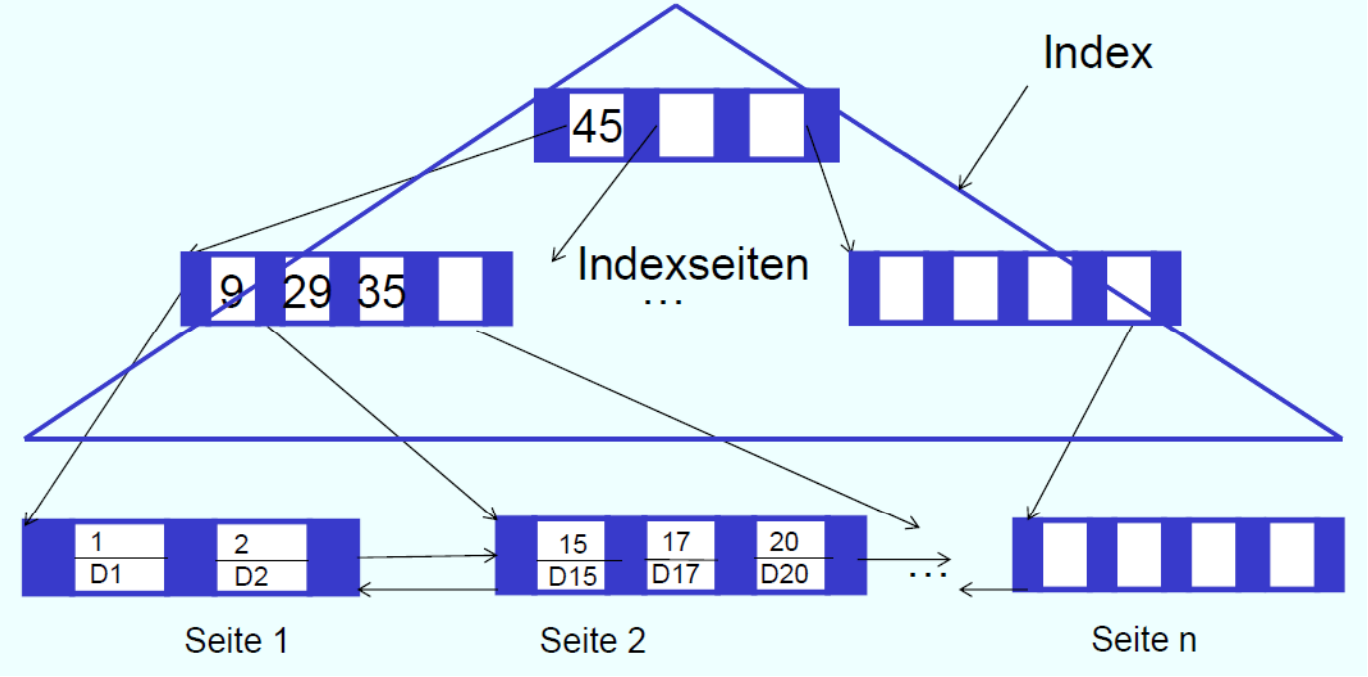


Hierarchische Indexstrukturen – B+ Bäume

B-Bäume

B+ -Bäume:

Daten nur in den Blattknoten



Hierarchische Indexstrukturen

CREATE [UNIQUE] INDEX index_name

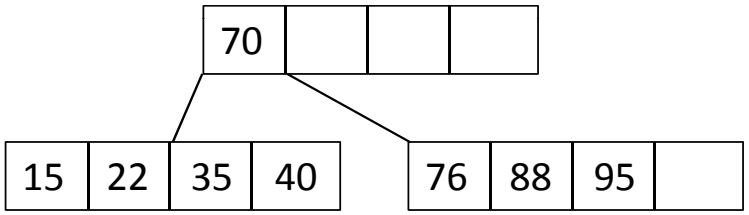
ON table_name (column_name1 [, column_name2, …])

Beispiel:

CREATE INDEX full_name

ON Person (Name, Vorname)

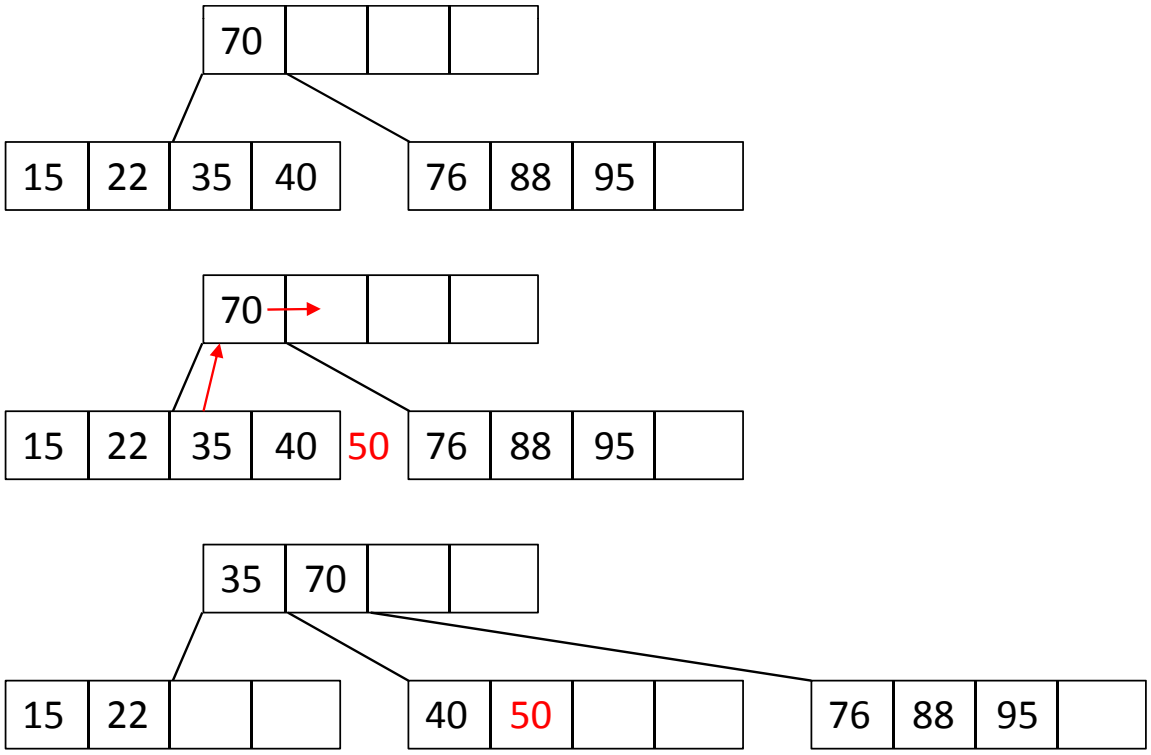
Beispiel Index einfügen



Grad 2

50 ?

Beispiel Index einfügen



Partitionierung - Hashing

Speicherung von Tupeln in festgelegten Speicherbereich

Optimale Hashfunktion gibt es nicht (Kollision)

+ einfache Implementierung

- Kollisionsbehandlung notwendig
- Vorreservierung des Speicherplatzes
- nicht dynamisch
- nur Punktanfragen (keine Bereiche)