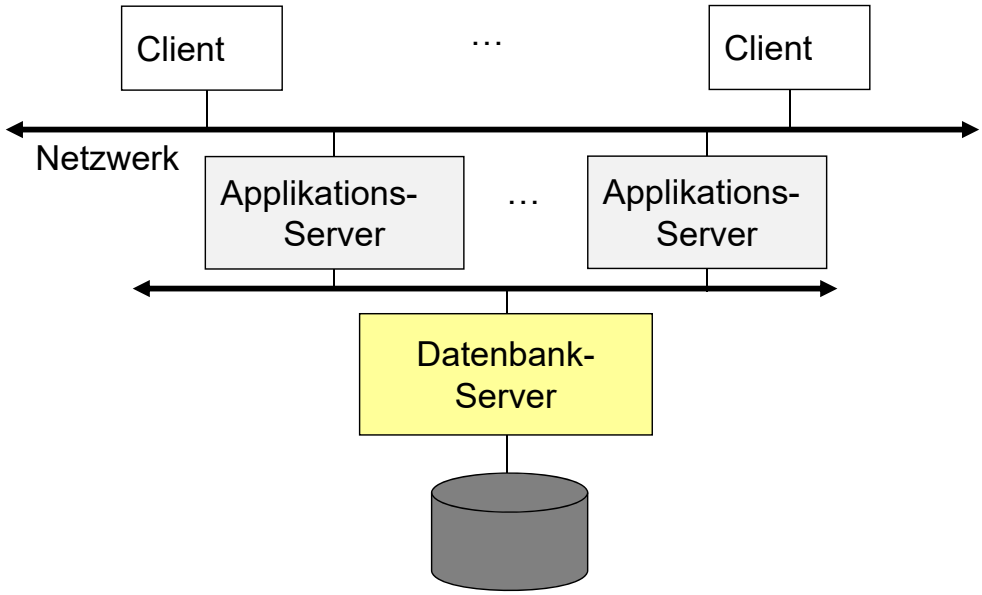
Kap. 1: Einführung

- Beschränkungen zentralisierter DBS
- Parallele vs. Verteilte DBS
- Anforderungen an Mehrrechner-DBS und für „Big Data“

- Arten der Parallelverarbeitung
- Speedup vs. Scaleup
- Gesetze von Amdahl und Gustafson

Zentralisierte DB-Verarbeitung



- meist relationale SQL-DBS mit Transaktionsverarbeitung (ACID)
- OLTP (Online Transaction Processing) und
  OLAP (Online Analytical Processing, Data Warehouse)

Beschränkungen zentralisierter DBS

- beschränkte Leistungsfähigkeit
  - Transaktionsraten
  - Zeitbedarf zur Auswertung großer Datenmengen
- unzureichende Skalierbarkeit
  - Big Data
  - wachsende Nutzerzahlen (Web-Buchungen …)
- unzureichende Verfügbarkeit
  - DBS-Absturz, Server-Ausfall
- keine Unterstützung dezentraler Organisationsstrukturen
  - Abhängigkeit von zentralem Datenmanagement

Notwendigkeit von *verteilten und parallelen Datenbanksystemen (Mehrrechner-DBS)* bzw. *Datenmanagementsystemen (DMS)*

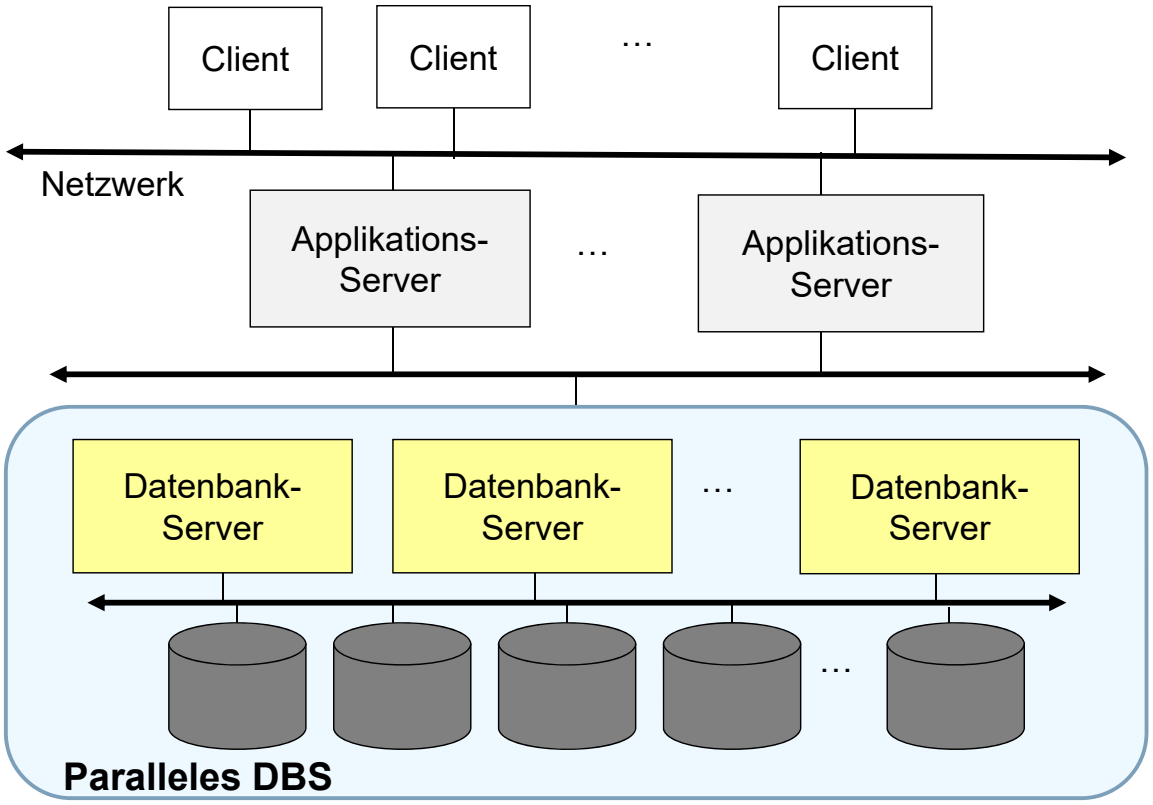
- Einsatz mehrerer Rechner zur koordinierten Verarbeitung von Daten

Verteilung und Parallelität

- **verteiltes System**
  - besteht aus autonomen Subsystemen, die oft (weit) entfernt voneinander angeordnet sind, aber koordiniert zusammenarbeiten, um eine gemeinsame Aufgabe zu erfüllen
  - charakteristisches Kernproblem: **Mangel an globalem (zentralisiertem) Wissen**

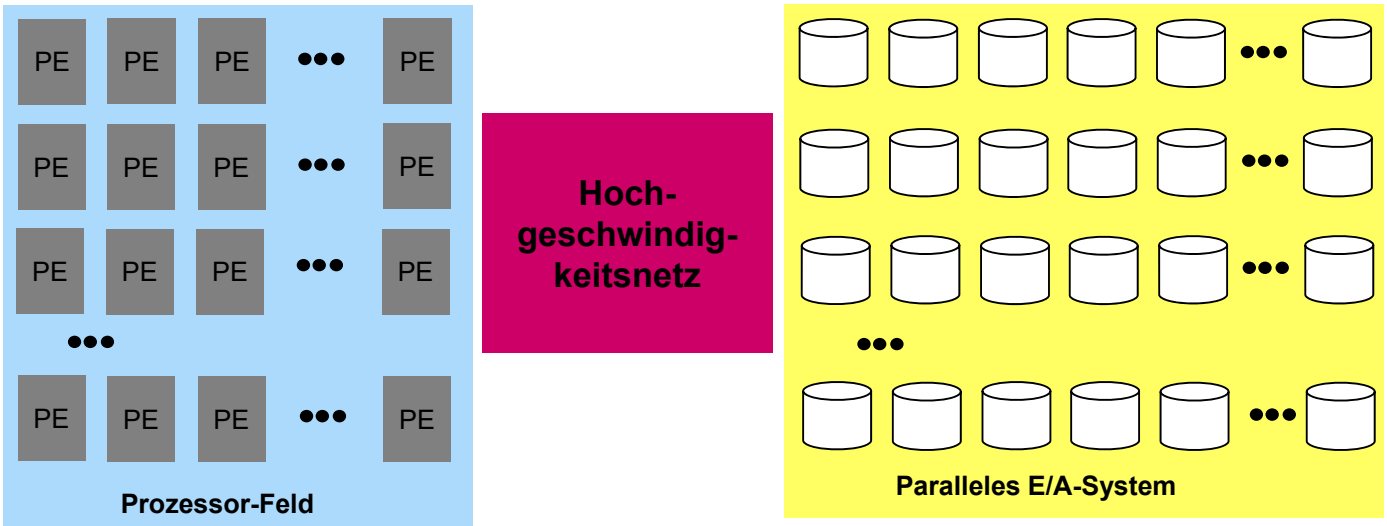
- **paralleles System**
  - besteht aus Vielzahl gleichartiger Subsysteme (Komponenten), die lokal zueinander angeordnet sind und nur einen geringen Grad an Autonomie aufweisen.
  - charakteristisch: **enge und hochgradig parallele Bearbeitung von Nutzeraufträgen**

Aufbau eines Parallelen DBS

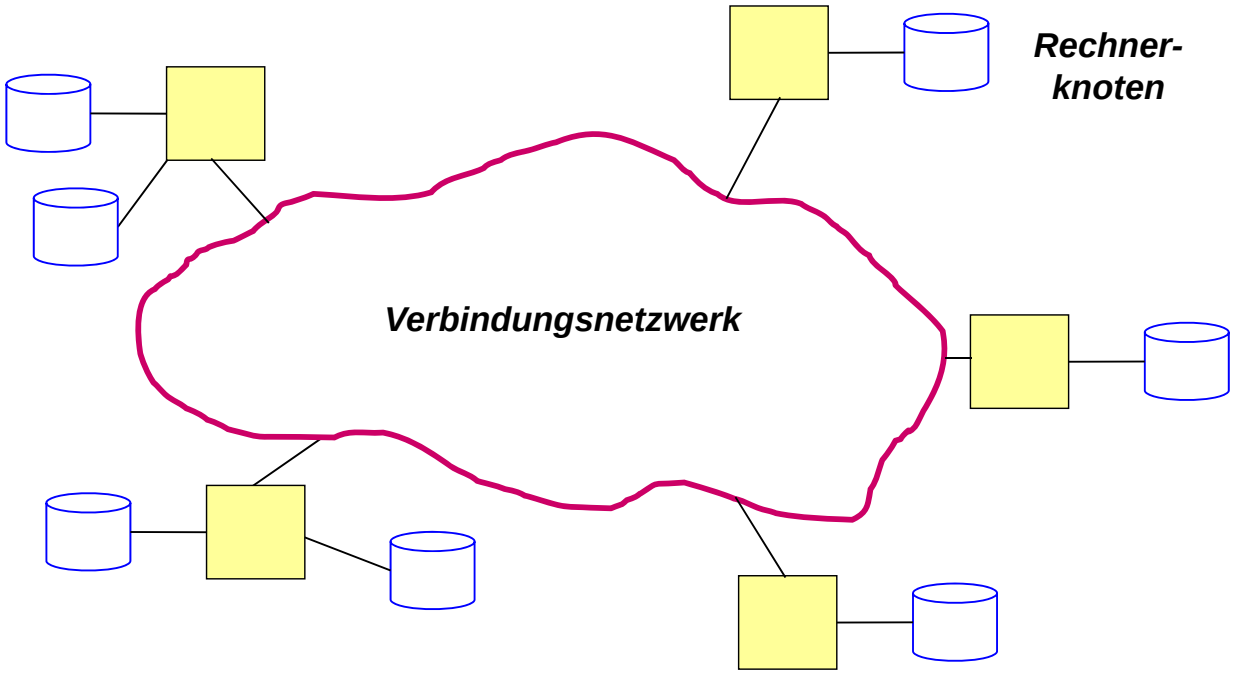


Parallele Datenbanksysteme (2)

- Architektur
  - lokale Verteilung (Cluster) der Rechner (Prozessorelemente, PE)
  - skalierbares Hochgeschwindigkeitsnetzwerk
  - E/A-Parallelität

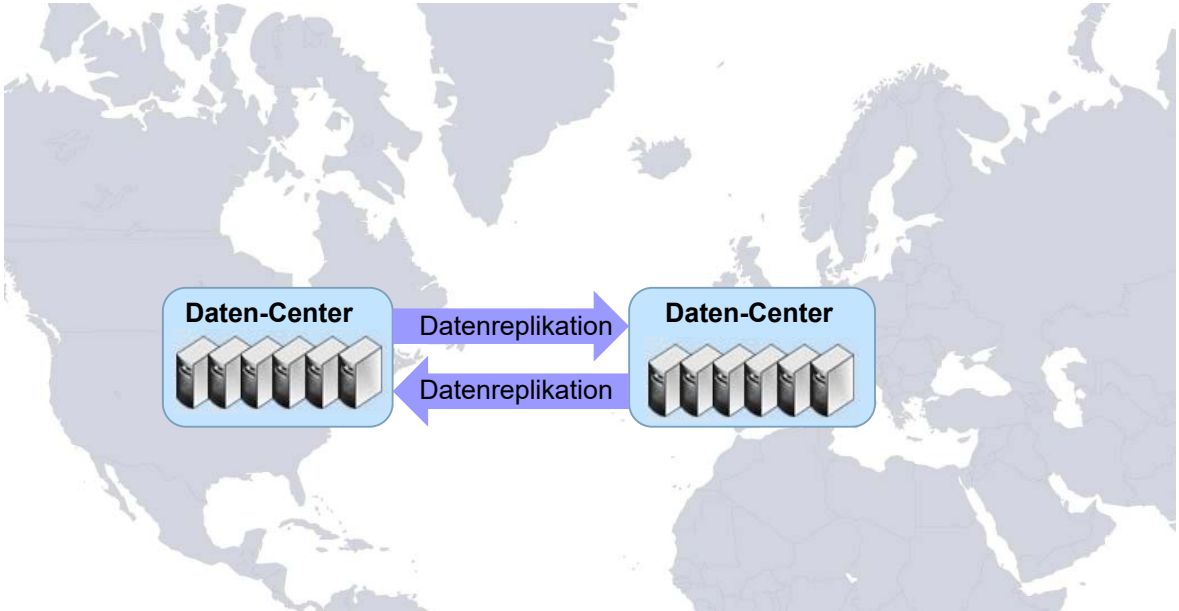


(Geographisch) Verteiltes DBS



Geo-Replikation

- ortsverteilte Kopplung mehrerer PDBS-Cluster (Data Center) mit Replikation der Daten
- hohe Verfügbarkeit auch bei Naturkatastrophen oder Terroranschlägen



Anforderungen an Mehrrechner-DBS

- hohe Leistung für OLTP und OLAP
  - hohe Transaktionsraten, kurze Antwortzeiten
- Skalierbarkeit
  - modulare Erweiterungsfähigkeit mit Rechnerzahl (Datenvolumen / Nutzerzahl)
- hohe Verfügbarkeit / Ausfallsicherheit
- Unterstützung dezentraler Organisationsstrukturen mit lokaler Datenhaltung (Wahrung einer hohen Knotenautonomie)
- koordinierter Zugriff auf unabhängige, heterogene Datenbanken (Datenintegration)
- hohe Verteilungstransparenz für DB-Benutzer (für Anwendungsprogramme bzw. Endbenutzer)
- hohe Kosteneffektivität
- einfache Nutzbarkeit / Administrierbarkeit

Anforderungen: Hohe Leistung

- hohe Transaktionsraten (Durchsatz)
  >> 1000 Transaktionen pro Sekunde (TPS), z.B. vom Typ'Kontenbuchung'
- kurze Antwortzeiten
  - Parallelisierung komplexer Anfragen
  - Akzeptanz für interaktive Nutzer
  - Auswertung von 10 TB bei max. Plattenbandbreite von 100 MB/s ?

- *Inter-Transaktionsparallelität:*
  - hohe Transaktionsraten für OLTP
  - lineares Durchsatzwachstum
- *Intra-Transaktionsparallelität*: kurze Antwortzeiten für daten- und/oder berechnungsintensive DB-Anfragen
  - Analysen auf großen Relationen: Scan, Join, Group-By ...
  - Data Mining-Auswertungen …

Anforderungen: Hohe Verfügbarkeit

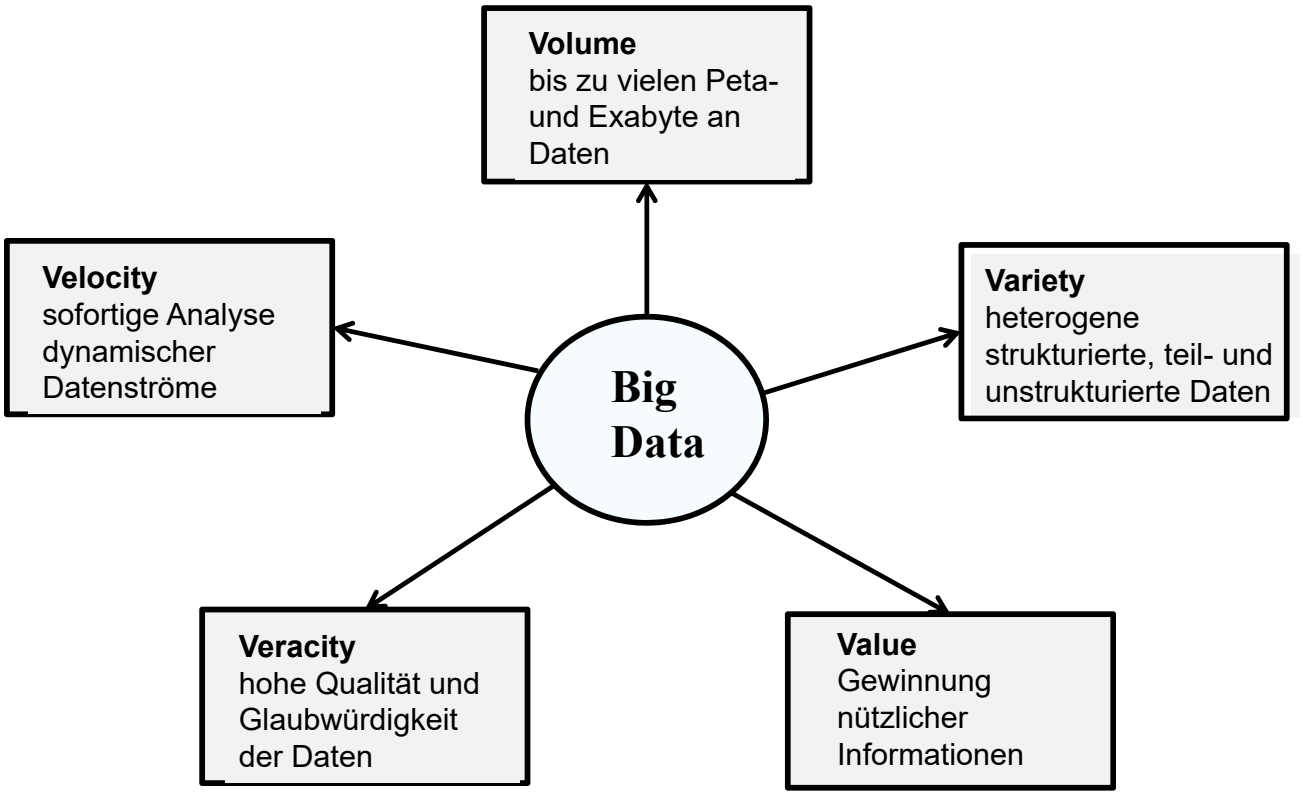
- Ziel: ständige Verfügbarkeit (24x7)
  - zumindest Tolerierung aller Einfachfehler
- Voraussetzungen:
  - redundante Systemkomponenten (Fehlertoleranz)
    - HW- und SW-Komponenten
    - Daten (Log, Spiegelplatten, replizierte DB)
  - automatische Fehlererkennung und –behandlung
  - Umkonfigurierbarkeit im laufenden Betrieb

Hohe Verfügbarkeit (2)

- MTTF (Meantime To Failure):
  - konventionelle Systeme: ca. 10-20 Tage
  - fehlertolerante Systeme: > 10 Jahre
- MTBF (Meantime Between Failures)
  - MTBF =MTTF(Meantime To Failure)+MTTR (Meantime to Repair)

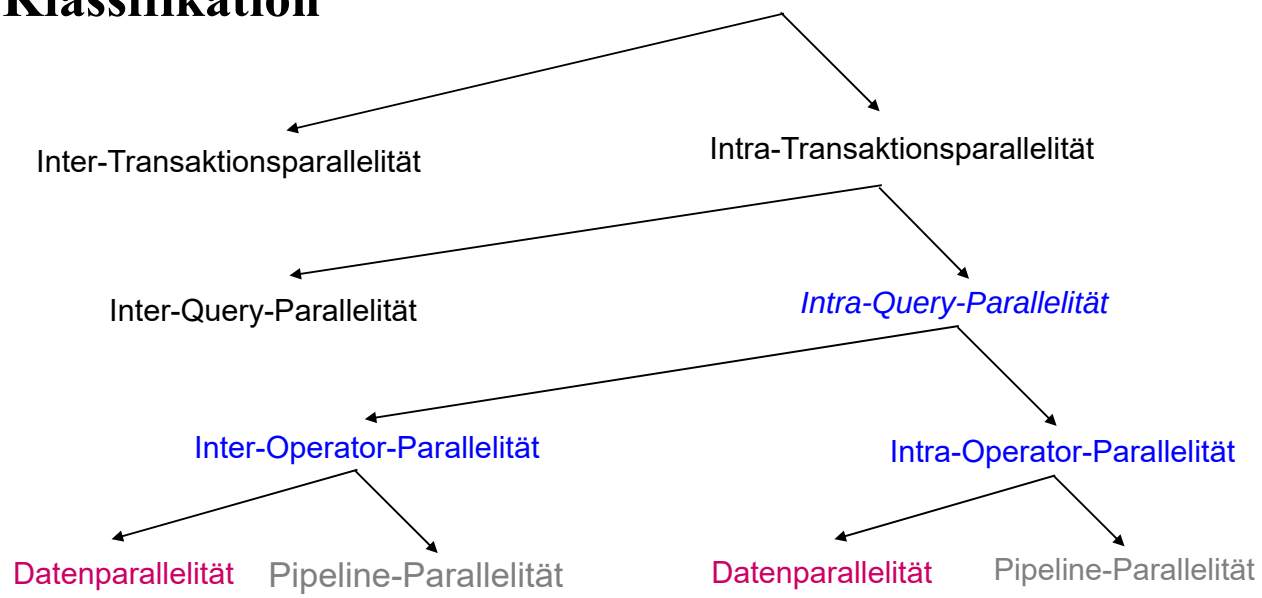
- Verfügbarkeit = MTTF / (MTTF + MTTR)
  - Beispiel Ausfall 1 h pro Jahr:
  - hohe Verfügbarkeit einzelner Komponenten reicht nicht (z.B. 25 Jahre MTTF pro Platte)

Anforderungen für „Big Data“



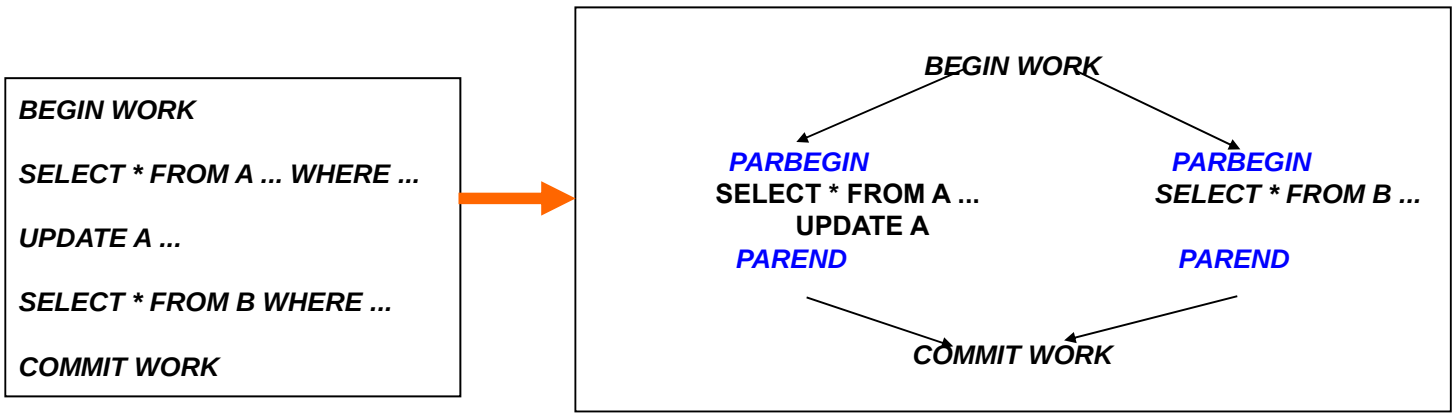
Arten der Parallelität

- Unterscheidungsmerkmale
  - Granularität der Parallelarbeit: Transaktion, Query, Operator
  - Datenparallelität vs. Funktionsparallelität (Pipeline-Parallelität)
  - Verarbeitungs- vs. E/A-Parallelität
- **Klassifikation**



Inter-Query-Parallelität

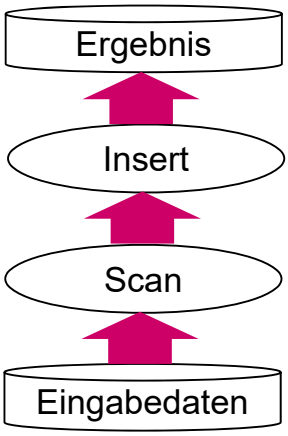
- parallele Bearbeitung unabhängiger DB-Operationen (Queries) eines Transaktionsprogrammes
- Programmierer muss Parallelisierung spezifizieren



- nur begrenzter Parallelitätsgrad möglich

Pipeline-Parallelität

- Datenfluss-Prinzip zum Datenaustausch zwischen Operatoren / Teiloperatoren
- frühzeitige Weitergabe von Tupeln bei Zwischenergebnissen
- Einsatz vor allem bei Inter-Operator-Parallelität
- Pipeline-Unterbrechung bei Operatoren, die vollständige Eingabe zur Ergebnisberechnung verlangen:
  - Sortierung
  - Duplikateliminierung
  - Gruppierung (GROUP BY)
  - Aggregatfunktionen, etc.
- Pipelines oft sehr kurz (≤ 10 Operatoren)

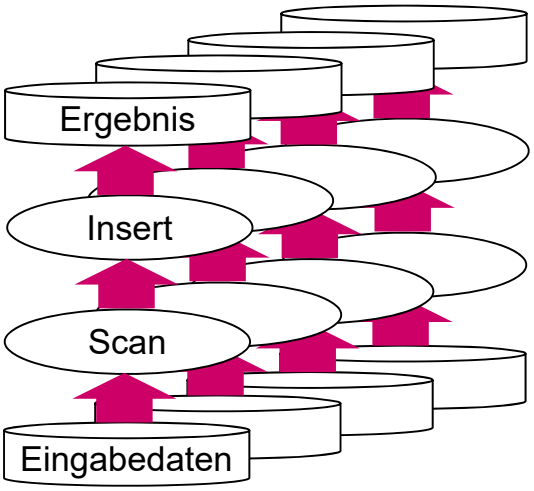


Pipeline-Parallelität

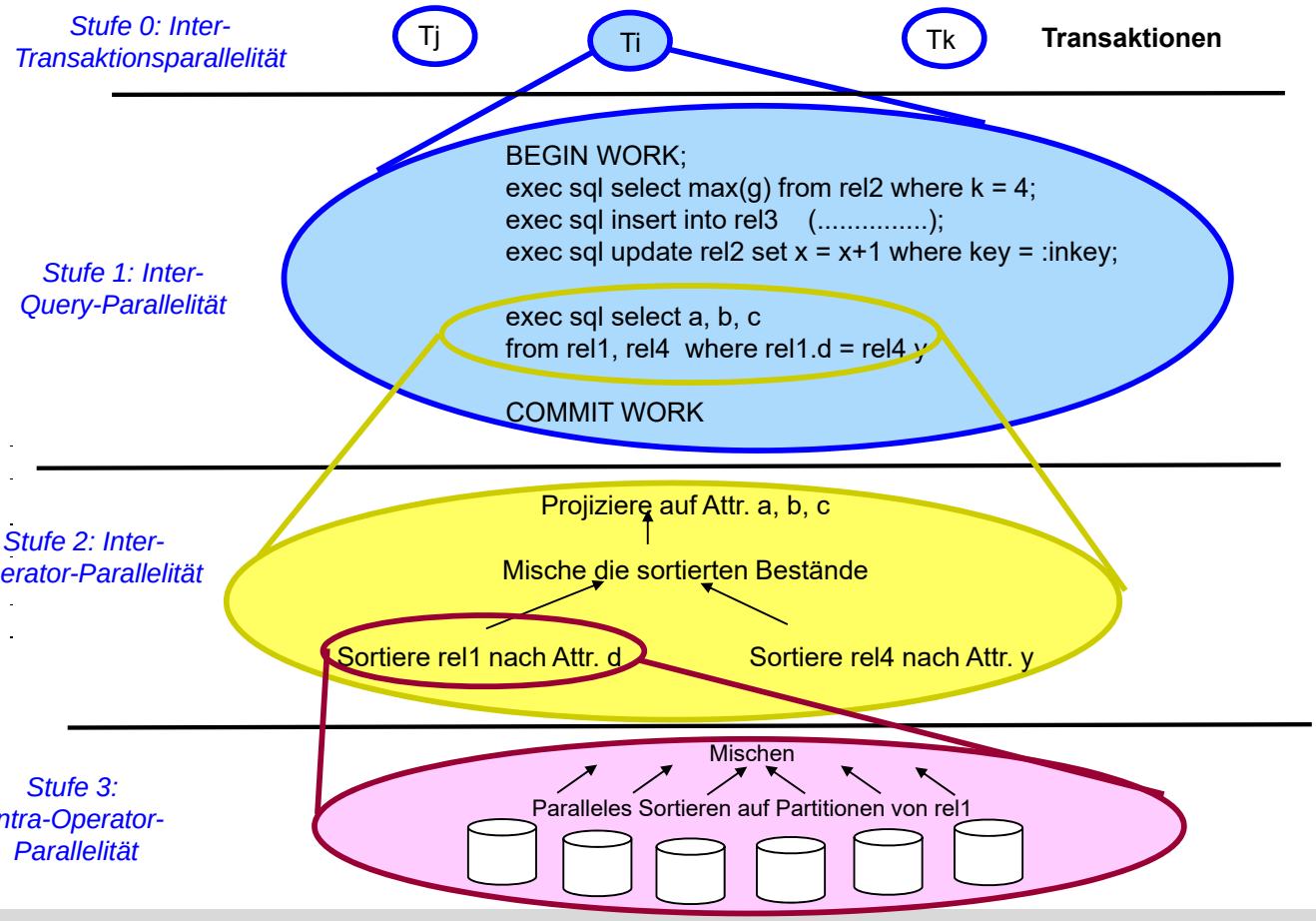
Daten-Parallelität

- basiert auf breiter (horizontaler) Datenverteilung (Declustering)
- parallele Bearbeitung von Teiloperationen auf disjunkten Datenmengen
- Parallelitätsgrad kann mit Datenumfang gesteigert werden
- kombinierbar mit Pipeline-Parallelität

Daten- und Pipeline-Parallelität



Gleichzeitiger Einsatz mehrerer Parallelisierungsarten



Leistungsmaße für Parallelverarbeitung: Speedup

- Vorgabe: konstante Datenbankgröße
- ***Antwortzeit-Speedup*** (batch speedup) misst Antwortzeitverbesserung für komplexe Operationen durch Parallelverarbeitung

Antwortzeit-Speedup (N) =

$$\frac{\text{Antwortzeit bei 1 Rechner}}{\text{Antwortzeit bei N Rechnern}}$$

- Ziel: lineare Antwortzeitverkürzung mit wachsender Rechneranzahl durch Einsatz von Intra-Transaktionsparallelität

Scaleup (1)

- Vorgabe: Datenbankgröße wächst linear mit der Rechneranzahl
- ***Durchsatz-Scaleup*** (OLTP scaleup) misst relatives Durchsatzwachstum (bei gegebener Antwortzeitrestriktion)

Durchsatz-Scaleup (N) =

$$\frac{\text{Transaktionsrate bei N Rechnern}}{\text{Transaktionsrate bei 1 Rechner}}$$

- Ziel: lineares Wachstum durch Nutzung von Inter-Transaktionsparallelität

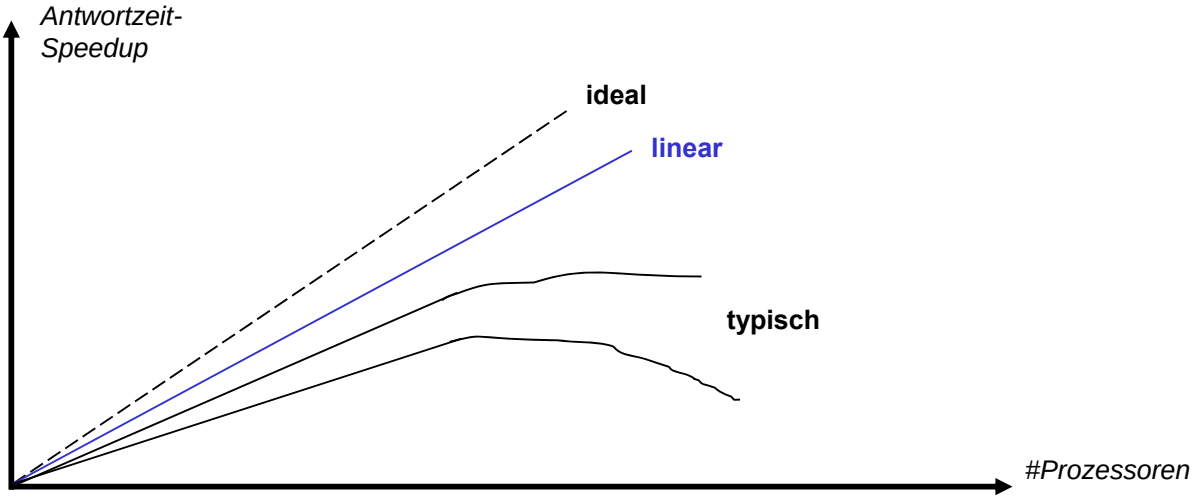
Scaleup (2)

- Antwortzeit-Scaleup (batch scaleup) misst Antwortzeitveränderung für komplexe Operationen auf unterschiedlichen Datenmengen

$$\text{Antwortzeit-Scaleup (N)} = \frac{\text{Antwortzeit bei N Rechnern}}{\text{Antwortzeit bei 1 Rechner}}$$

- Ziel: gleichbleibende Antwortzeit trotz wachsender DB durch Intra-Transaktionsparallelität

Grenzen der Skalierbarkeit



- maximale inhärente Parallelität ist begrenzt
- Startup- und Terminierungs-Overhead
- Interferenzen bei physischen und logischen Ressourcen
  - Überlastung einzelner Rechner, Sperrkonflikte, beschränkte Partitionierbarkeit von Daten und Anfragen
- Varianz (Skew) in den Ausführungszeiten der Teilaufgaben

Amdahl's Gesetz

- Speedup ist begrenzt durch nicht-optimierte (sequenzielle) Komponenten der Antwortzeit

$$\text{Antwortzeit-Speedup} = \frac{1}{(1 - F_{opt}) + \frac{F_{opt}}{S_{opt}}}$$

F_{opt} = Anteil der optimierten Antwortzeitkomponente $(0 \leq F_{opt} \leq 1)$

S_{opt} = Speedup für optimierten Antwortzeitanteil

- Beispiel: 5% sequenzieller Anteil -> $F_{opt} =$

$$\text{Antwortzeit-Speedup} = \frac{1}{(1 - \quad) + }$$

Gesetz von Gustafson

- optimistischere Abschätzung des erreichbaren Speedups
- Annahmen:
  - Problemgröße (Datenmenge) wächst linear mit Rechnerzahl
  - optimale Parallelisierung ist möglich
- Antwortzeit im 1-Prozessorfall sei $s+p$ (s/p sequenzieller/parallelisierbarer Anteil)
- n-fache Problemgröße
  - Bearbeitungszeit bei 1 Prozessor: $s + n*p$
  - Bearbeitungszeit bei n Prozessoren: $s + p$
- „skalierbarer" Speedup nach Gustafson:

$$\text{Antwortzeit-Speedup} \leq \frac{s + n * p}{s + p} = n - f * (n-1)$$

für $f = s / (s+ p)$

Zusammenfassung

- wesentliche Klassen von MRDBS:
  Verteilte DBS und Parallele DBS
- Parallele DBS: skalierbare Architektur mit lokaler Verteilung
  - hohe Leistungsfähigkeit (hohe Transaktionsraten, Parallelisierung komplexer Anfragen)
  - hohe Verfügbarkeit und Fehlertoleranz in allen Komponenten
  - modulare Erweiterungsfähigkeit
- Verteilte DBS: Unterstützung ortsverteilter DBS mit höherer Autonomie einzelner Knoten
- Big Data: extreme Skalierbarkeitsanforderungen, Analysefokus
- Unterstützung unterschiedlicher Arten von Intra-Transaktionsparallelität
- Speed-Up und Scale-Up-Metriken zur Parallelverarbeitung