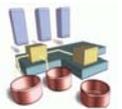


Kap. 1: Einführung

- Anforderungen
 - Leistung
 - Verfügbarkeit
 - Erweiterbarkeit
- Verteilte vs. Parallele DBS
- Arten der Parallelverarbeitung
- Speedup vs. Scaleup
- Amdahl's Gesetz



Verteilung und Parallelität

- Ein **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

- Ein **paralleles System** besteht aus einer 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



Mehrrechner-Datenbanksysteme

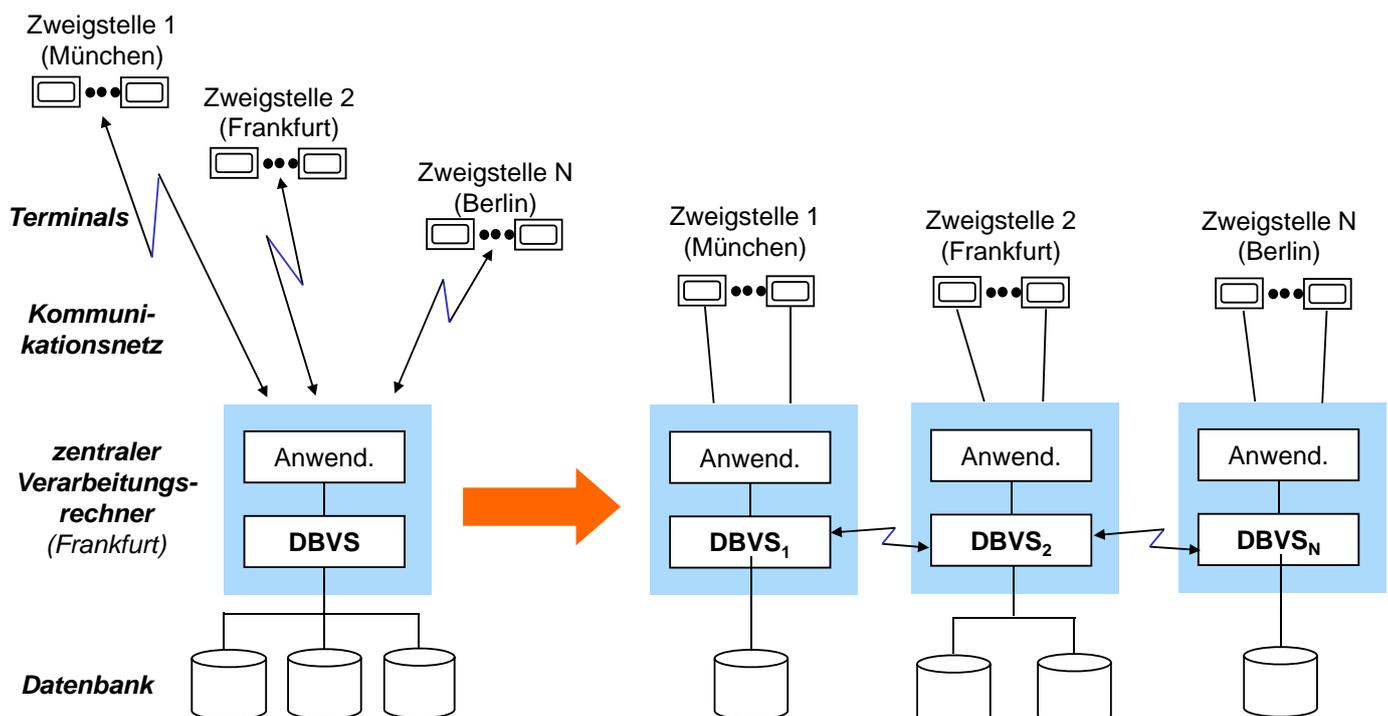
Einsatz mehrerer Rechner/DBVS zur koordinierten Verarbeitung von Datenbankoperationen

Anforderungen:

- Hohe Leistung: hohe Transaktionsraten, kurze Antwortzeiten
- Modulare Erweiterungsfähigkeit / Skalierbarkeit
 - Durchsatz / Antwortzeiten verbessern sich linear mit Rechnerzahl
- Hohe Verfügbarkeit / Fehlertransparenz
- Koordinierter Zugriff auf unabhängige, heterogene Datenbanken
- Unterstützung geographisch verteilter Datenbanken (Wahrung einer hohen Knotenautonomie)
- Verteiltransparenz für DB-Benutzer (für Anwendungsprogramme bzw. Endbenutzer)
- Hohe Kosteneffektivität
- Einfache Handhabbarkeit / Administrierbarkeit



Von zentralisierten zu verteilten DBS



Parallele Datenbanksysteme

■ Typische Leistungsmerkmale bei sequentieller Verarbeitung:

- Platte 10 MB/s
- Suchen (Relationen-Scan) 2 MB/s
- Sortieren 0.2 MB/s
- Verbund (Join) ?

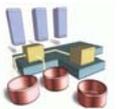
Bearbeitungszeit für eine 10 TB Datenbank ?

■ *Intra-Transaktionsparallelität*: kurze Antwortzeiten für daten- und/oder berechnungsintensive DB-Anfragen

- Operationen auf großen Relationen: Scan, Join, ...
- Data Mining-Auswertungen
- Multimedia-Anwendungen •••

■ *Inter-Transaktionsparallelität*:

- hohe Transaktionsraten für OLTP
- lineares Durchsatzwachstum

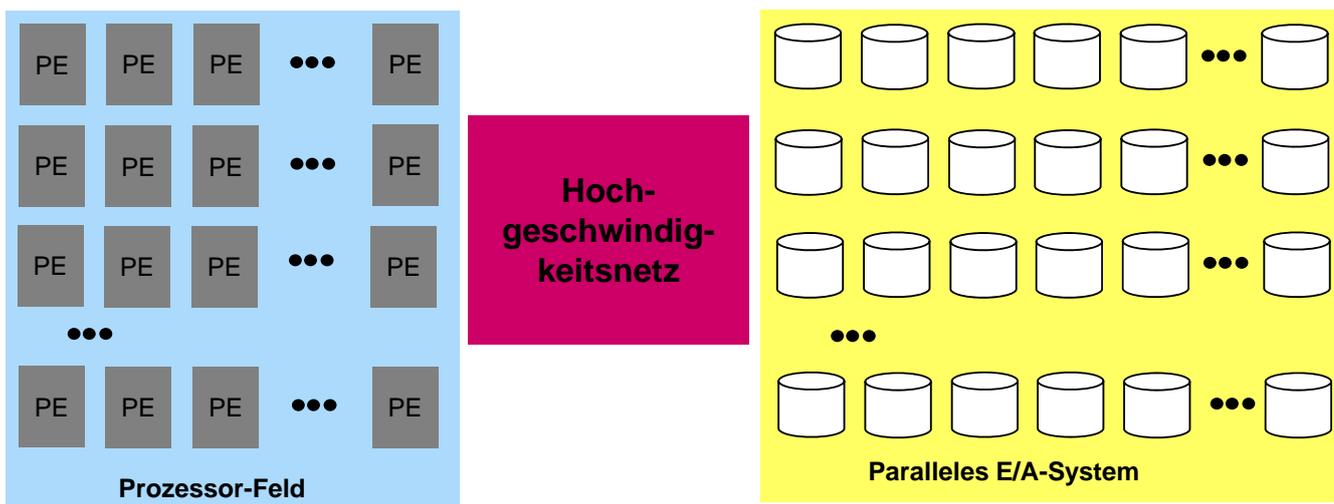


Parallele Datenbanksysteme (2)

■ Spezieller Typ von Mehrrechner-DBS mit den Hauptzielen: hohe Leistung, Verfügbarkeit, Skalierbarkeit und Kosteneffektivität

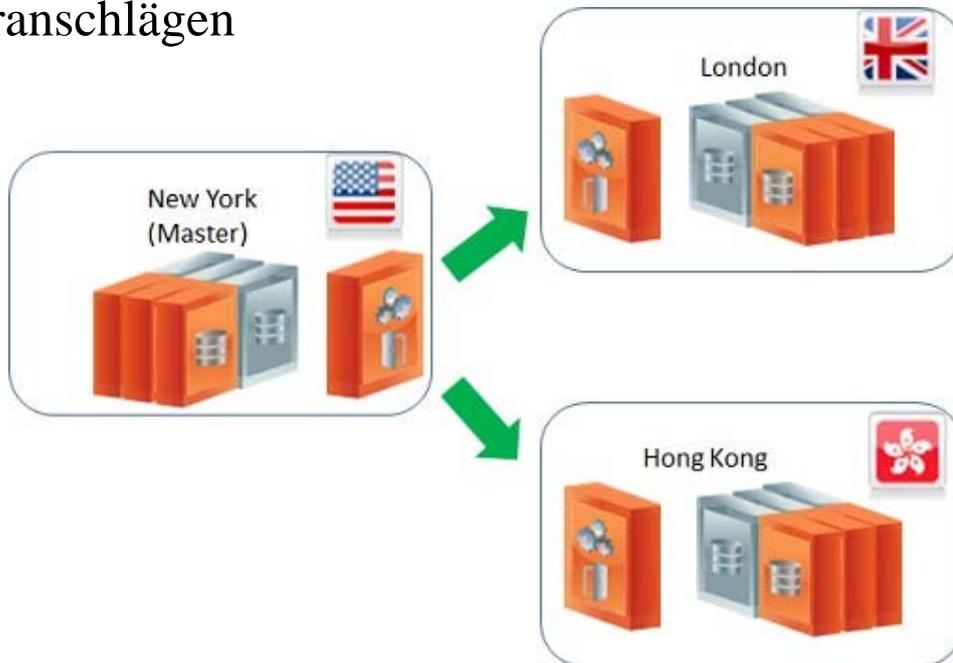
■ Architektur

- Parallelrechner mit hoher Anzahl von Standardprozessoren
- lokale Verteilung (Cluster)
- skalierbares Hochgeschwindigkeitsnetzwerk
- E/A-Parallelität

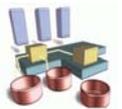


Ortsverteilte Data Center

- Ortsverteilte Kopplung mehrerer PDBS-Cluster mit Replikation der Daten
- Hohe Verfügbarkeit auch bei Naturkatastrophen oder Terroranschlägen



Quelle: gigaspaces.com



Anforderungen: Hohe Leistung

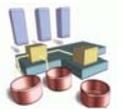
- Hohe Transaktionsraten (Durchsatz)
 - >> 1000 Transaktionen pro Sekunde (TPS)
(z.B. vom Typ 'Kontenbuchung')
- Kurze Antwortzeiten
 - Parallelisierung komplexer Anfragen
 - Akzeptanz für interaktive Nutzer
- Ständig steigende Leistungsanforderungen:
 - wachsende Zahl von Benutzern
 - ständiges Wachstum der Datenmengen („Big Data“)
 - komplexere Aufträge

=> Einsatz von Mehrrechner-DBS erforderlich



Beispiele für hohe Leistungsanforderungen

- OLTP / E-Commerce
 - mehrere 1000 TPS bei Antwortzeit < 2 s
- Telefonvermittlung
 - pro Telefonat ist Benutzerprofil aus DB zu lesen und Abrechnungssatz zu schreiben
 - oft > 15.000 Transaktionen pro Sekunde; Antwortzeit < 0.2 s
- Entscheidungsunterstützung / Data Warehousing
 - Auswertung komplexer Ad-Hoc-Anfragen auf TB von Daten
 - Durchsatz von 10 Queries pro Sekunde mit Antwortzeit < 30 s
- Bio-Datenbanken (z.B. Clusteranalyse auf Genexpressionen)
- ...



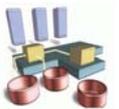
Anforderungen: Hohe Verfügbarkeit

- Ziel: Continuous Operation (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
- MTBF (meantime between failures):
 - konventionelle Systeme: ca. 10-20 Tage
 - fehlertolerante Systeme: > 10 Jahre
- Verfügbarkeit = $MTBF / (MTBF + MTTR)$



Analyse von Ausfallursachen

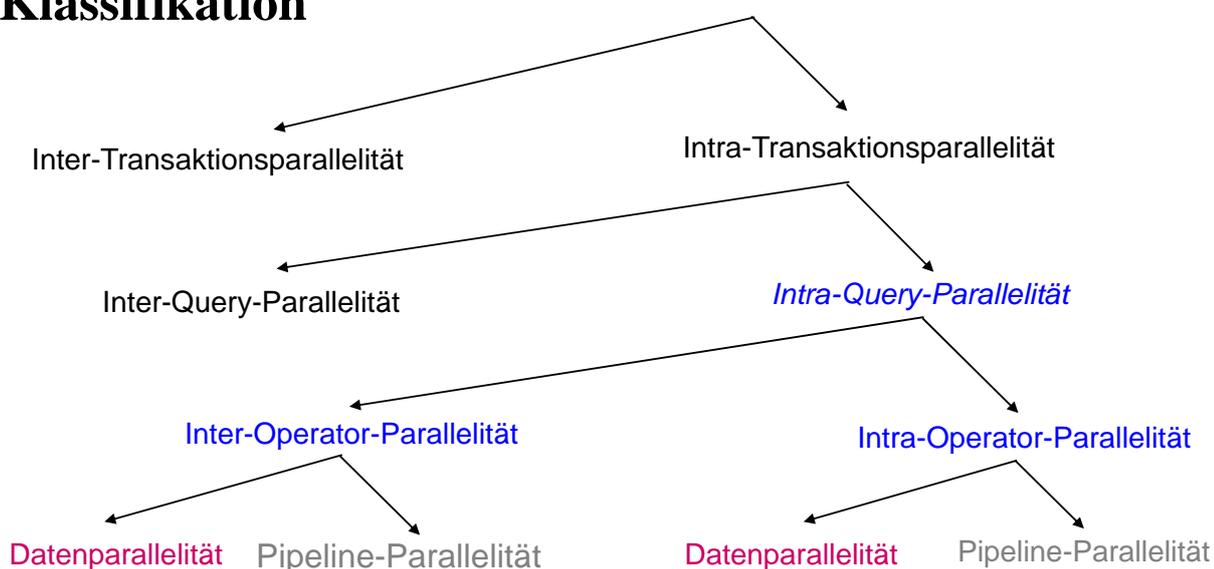
- empirische Auswertungen, u.a. bei Tandem-Anlagen
- starke Verbesserungen bei Hardware
- zunehmende Automatisierung bei Operating
- Software-Fehler zunehmend dominierende Ausfallursache
 - stark wachsender Umfang an Anwendungs- und System-SW
 - gestiegene SW-Komplexität
 - Zunehmende Zahl von Abhängigkeiten zu anderen Programmen und Hardware-Varianten
 - Testen skaliert nicht; unzureichend für hohe Software-Qualität



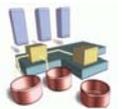
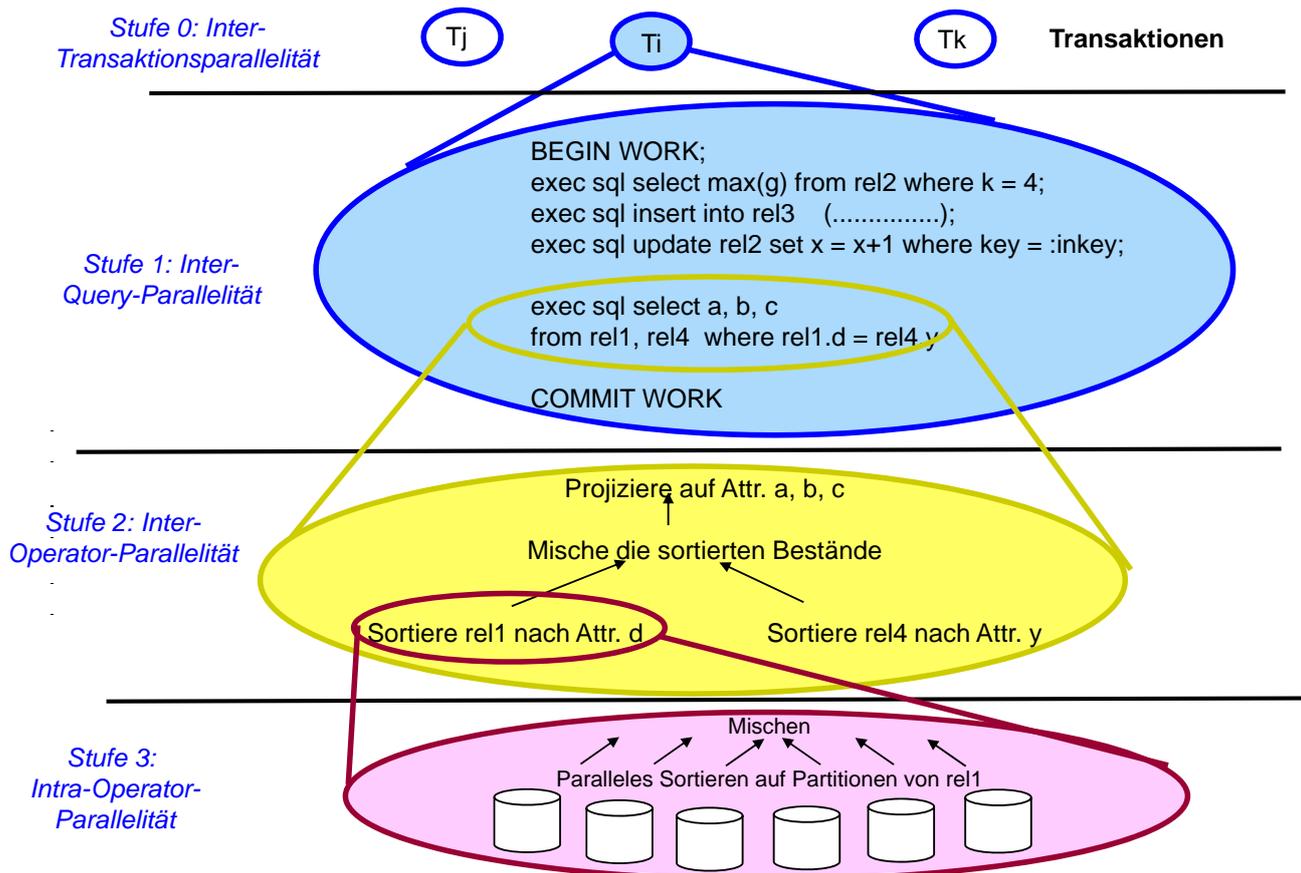
Arten der Parallelität

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

■ Klassifikation

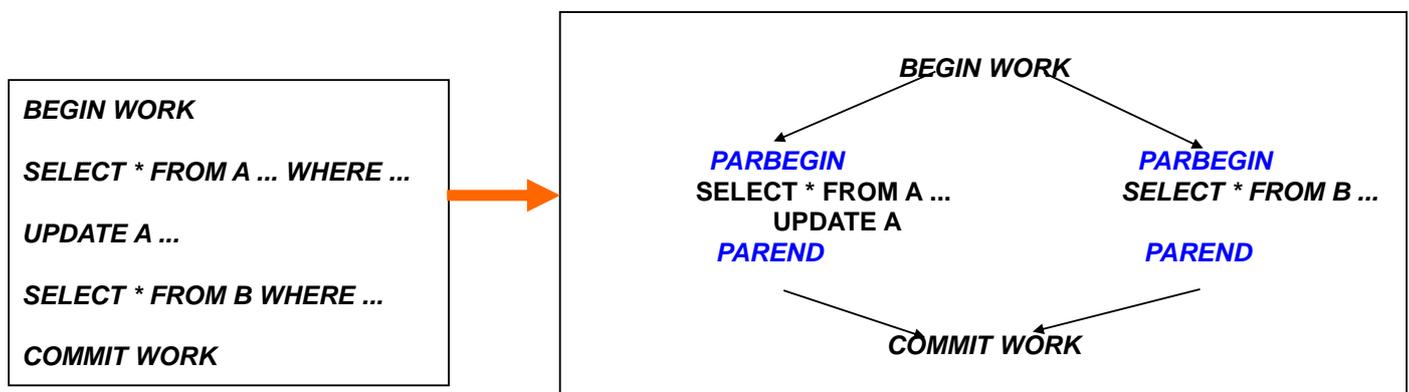


Gleichzeitiger Einsatz mehrerer Parallelisierungsarten



Inter-Query-Parallelität

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

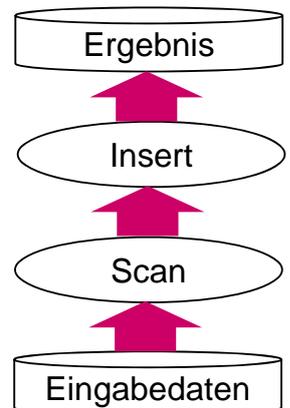


- nur begrenzter Parallelitätsgrad möglich

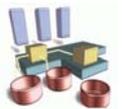


Pipeline-Parallelität

- 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)



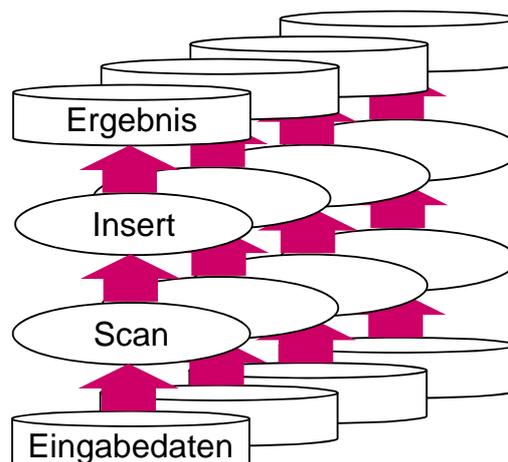
Pipeline-Parallelität



Daten-Parallelität

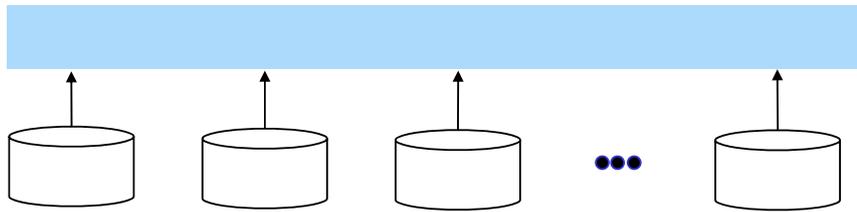
- basiert auf breiter (horizontaler) Datenverteilung (Declustering)
- parallele Bearbeitung von Teiloperationen auf disjunkten Datenmengen
- Parallelitätsgrad kann mit Datenumfang gesteigert werden
- Kombinierbar mit Pipeline-Parallelität

Daten- und Pipeline-Parallelität



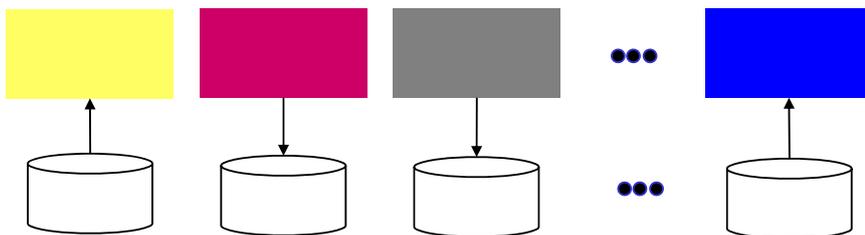
E/A-Parallelität

- Voraussetzung: Declustering von Dateien über mehrere Platten
- *Intra-E/A-Parallelität* (Zugriffsparallelität)

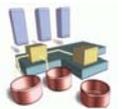


Parallele Ausführung eines E/A-Auftrags (wichtig für Datenparallelität)

- *Inter-E/A-Parallelität* (Auftragsparallelität)



Parallele Ausführung unabhängiger E/A-Aufträge verschiedener Platten (wichtig für Inter-Transaktions-Parallelität, Inter-Query-Parallelität)



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



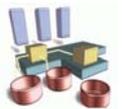
Scaleup (1)

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

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



- **Ziel:** lineares Wachstum durch Nutzung von Inter-Transaktionsparallelität



Scaleup (2)

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

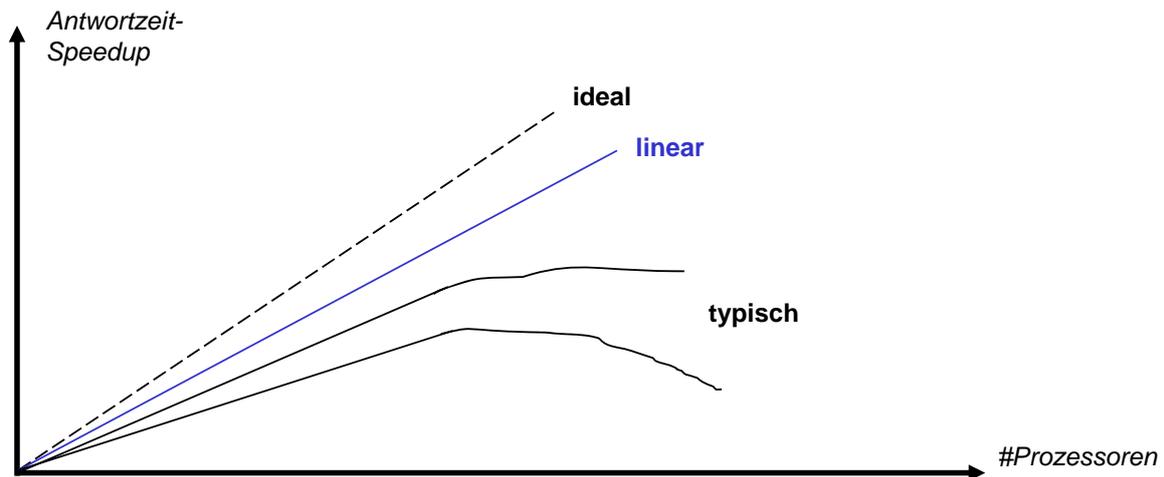
$$\text{Antwortzeit-Scaleup (N)} = \frac{\text{Antwortzeit bei N Rechnern}}{\text{Antwortzeit bei 1 Rechner}}$$



- **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 Transaktionen / Anfragen
- Varianz (Skew) in den Ausführungszeiten der Teilaufgaben



Amdahls Gesetz

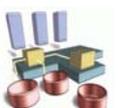
- Speedup ist begrenzt durch nicht-optimierte (sequentielle) Komponenten der Antwortzeit

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

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

S_{opt} = Speedup für optimierten Antwortzeitanteil

- Beispiel: 5% sequentieller Anteil ->



Zusammenfassung

- wesentliche Klassen von MRDBS: Verteilte DBS und Parallele DBS
- Parallele DBS: skalierbare Architektur mit lokaler Verteilung
 - Hohe Leistungsfähigkeit (hohe Transaktionsraten, Parallelisierung komplexer Anfragen)
 - Unterstützung sehr großer DB
 - Hohe Verfügbarkeit und Fehlertoleranz in allen Komponenten
 - Modulare Erweiterungsfähigkeit, u. a.
- Unterstützung unterschiedlicher Arten von Intra-Transaktionsparallelität
- Speed-Up und Scale-Up-Metriken zur Parallelverarbeitung
- Verteilte DBS: Unterstützung ortsverteilter DBS mit höherer Autonomie einzelner Knoten

