

2. Klassifikation von Mehrrechner-DBS

- Merkmale für PDBS/VDBS
 - **Räumliche Verteilung**: ortsverteilt oder lokal
 - **Rechnerkopplung**: enge, lose oder nahe Kopplung
 - **Externspeicheranbindung**: partitioniert oder gemeinsam ('shared')
- Weitere Klassifikationsmerkmale
 - Funktionale Spezialisierung vs. funktionale Gleichstellung
 - Integrierte vs. heterogene/föderierte MRDBS
- Systemansätze zur Datenintegration
- Grobbewertung von MRDBS-Alternativen
- PDBS-Architekturen
 - **Scale-Up** vs. **Scale-Out**
 - Shared-Nothing vs. Shared-Disk



Räumliche Verteilung

- **ortsverteilt**:
 - unterstützt dezentrale Organisationsstrukturen
 - unterstützt Katastrophen-Recovery (replizierte DB an entfernten Knoten)
 - relativ langsame Kommunikation
 - Signallaufzeiten > 100 ms
 - aufwändige Protokolle (> 10.000 Instruktionen pro Send/Receive)
- **lokal**:
 - schnelle Rechnerkopplung (gemeinsame Speicher bzw. Hochgeschwindigkeitsnetz)
 - effektive dynamische Lastverteilung möglich
 - bessere Voraussetzungen für Intra-Transaktionsparallelität
 - einfachere Administration

	Durchmesser	Latenzzeit	Bandbreite (Mbit/s)		Übertragung 10 KB	
			1990	2010	1990	2000
Cluster	20 m	1 us	1000	10000	0,01 ms	0,001 ms
LAN	1 km	10 us	10	1000	10 ms	0,1 ms
WAN	10.000 km	100 ms	0,05	100	1.700 ms	101 ms



Enge Rechnerkopplung (tightly coupled systems)

■ Eigenschaften

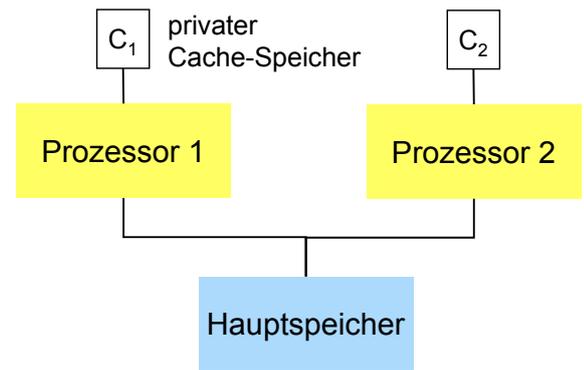
- gemeinsamer Hauptspeicher
- 1 Kopie von Software-Komponenten (BS, DBMS)
- HW-Cache pro Prozessor

■ Vorteile:

- weit verbreitet
- wenig neue DB-Probleme (DBS-Ausführung in mehreren Prozessen)
- effiziente Kommunikation über Hauptspeicher
- Lastbalancierung durch Betriebssystem
- Single System Image

■ Nachteile:

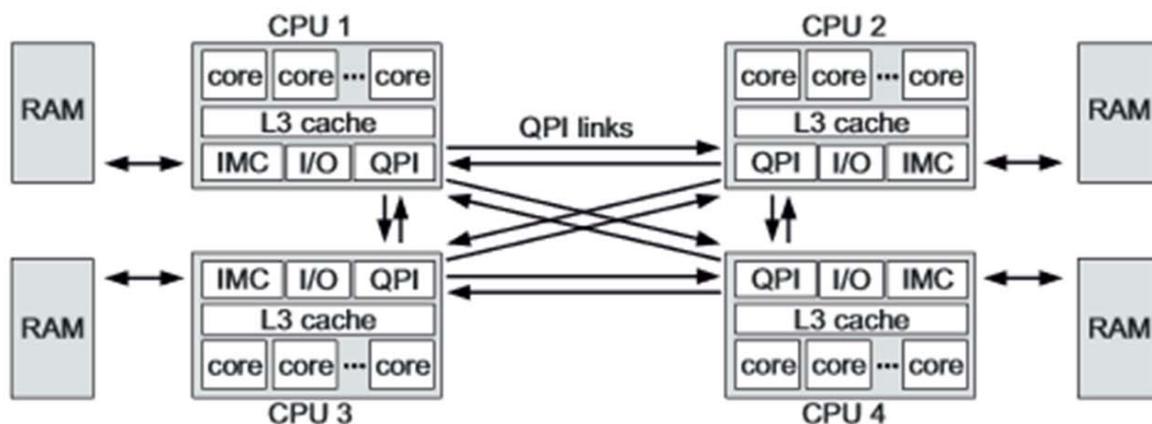
- mangelnde Fehlerisolation
- oft begrenzte Erweiterbarkeit und Skalierbarkeit
- Cache-Kohärenz



NUMA-Architekturen

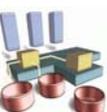
■ Non-Uniform Memory Access (NUMA) bei hoher Prozessorzahl

■ Beispiel Intel



■ Memory-Zugriff über Links wesentlich langsamer

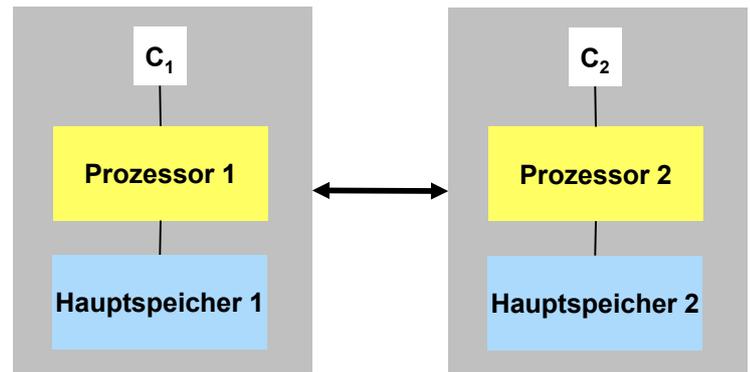
- Datenpartitionierung und -lokalität innerhalb eines NUMA-Servers verbessert Leistung für DB-Anwendungen



Lose Rechnerkopplung (loosely coupled systems)

■ Eigenschaften

- N selbständige Rechner (pro Knoten eigener Hauptspeicher, eigene Software-Kopien)
- Kommunikation über Nachrichtenaustausch



■ Vorteile:

- höhere Fehlerisolation/Verfügbarkeit
- bessere Erweiterbarkeit

■ Nachteile:

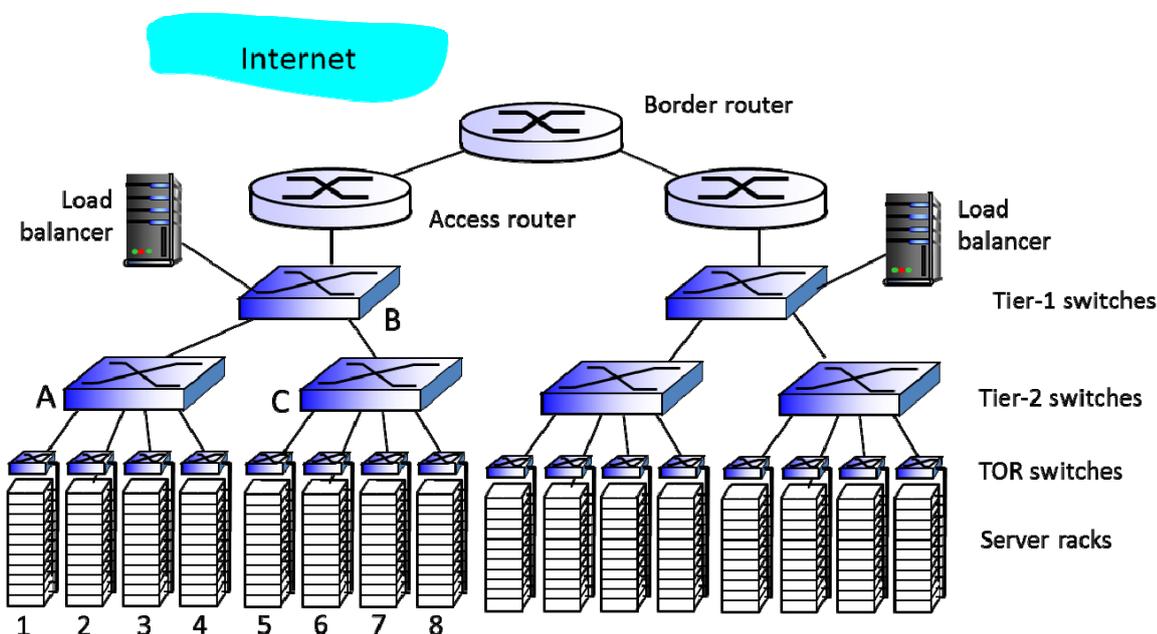
- Nachrichtenaustausch aufwendig (Kommunikations-Overhead)
- kein 'single system image'



Lose Kopplung im Data Center

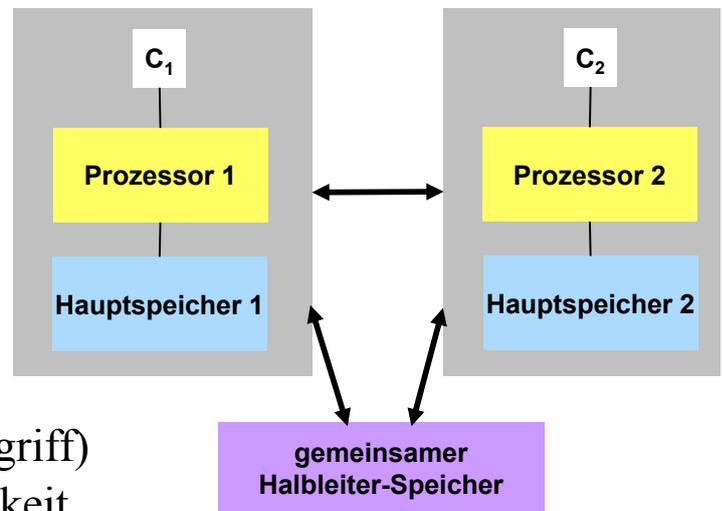
■ mehrstufige Kopplung mit Server-Racks

- Tausende von Rechnern
- schnellere Kommunikation innerhalb Rack



Nahe Rechnerkopplung (closely coupled systems)

- Kompromiß zwischen enger und loser Kopplung
 - effizientere Kommunikation als mit loser Kopplung unter Beibehaltung einer ausreichenden Fehlerisolation und Erweiterbarkeit
- Merkmale
 - N selbständige Rechnerknoten
 - gemeinsame Halbleiter-Speicherbereiche
 - lokale Rechneranordnung
- Speichereigenschaften
 - schneller, synchroner Zugriff (kein Prozeßwechsel während Zugriff)
 - i.a. keine Instruktionsadressierbarkeit
 - ggf. nichtflüchtig
- Unterstützung z.B in IBM z/OS-Mainframes



Externspeicheranbindung

gemeinsam:

jeder Prozessor kann alle Externspeicher / Daten direkt erreichen

- lokale Rechneranordnung
- lose oder nahe Kopplung bzw. enge Kopplung
- hohes Potential zur Lastbalancierung

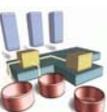
partitioniert:

Externspeicher sind primär nur je einem Knoten zugeordnet

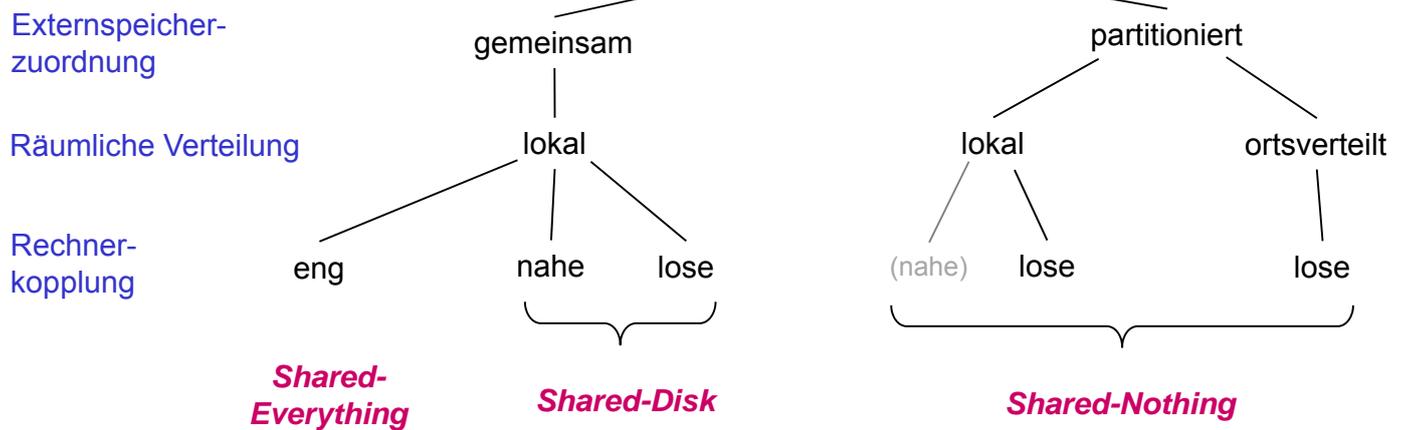
- lokale oder ortsverteilte Rechneranordnung
- i. a. lose Rechnerkopplung
- verteilte Transaktionsausführung, um auf entfernte Daten zuzugreifen

verteilte/parallele **In-Memory-Datenbanksysteme** ?

- enge Kopplung: gemeinsam genutzte In-Memory-DB
- lose Kopplung: partitionierte In-Memory-DB



Shared -*-Systeme

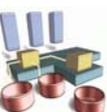


- **Parallele DBS**: lokal verteilte Shared -*-Systeme
- **Verteilte DBS**: ortsverteilte DBS
- **Hybride Ansätze**
 - eng gekoppelte Knoten in lose gekoppelten Systemen
 - ortsverteilte Data Center mit PDBS pro Center

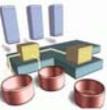
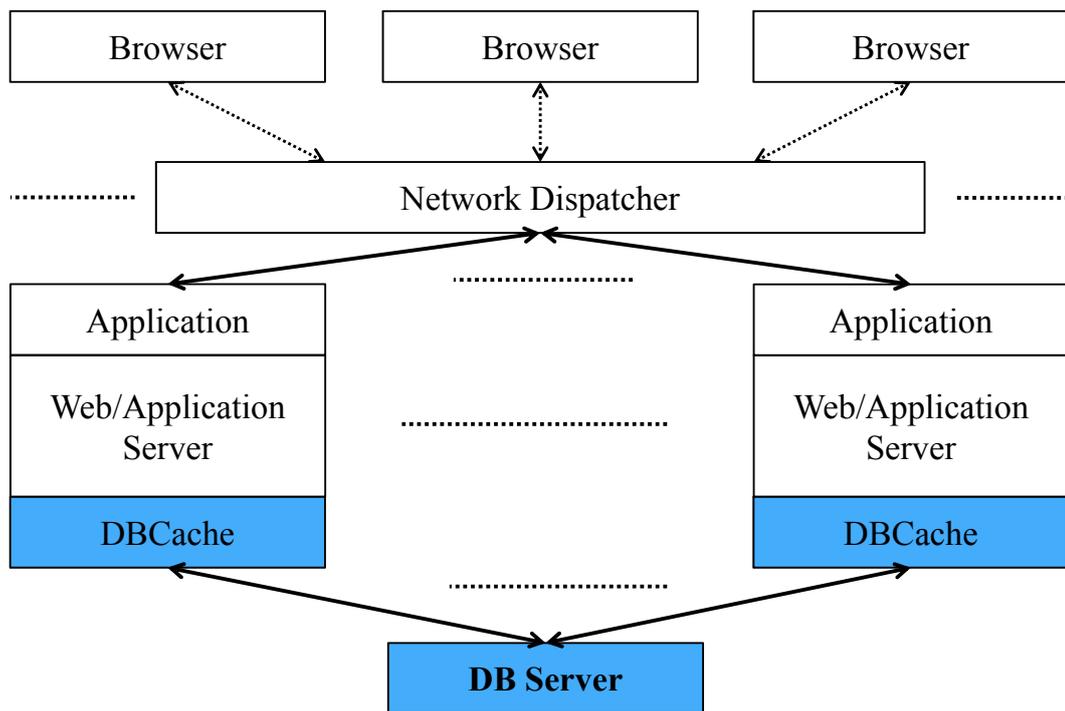


Verteilung der Funktionalität

- **funktionale Gleichstellung** („horizontale Verteilung“)
 - jeder Knoten besitzt gleiche Funktionalität bzgl. DB-Verarbeitung
 - i.a. vollständige DBMS pro Knoten
 - Replikation der Funktionen
- **funktionale Spezialisierung** („vertikale Verteilung“)
 - Partitionierung von Funktionen
 - Beispiele:
 - **DB-Maschinen** mit Spezialprozessoren für bestimmte DB-Funktionen (Join-Prozessor, Sortier-Prozessor, etc.)
 - Intelligente Platten-Kontroller z.B. für Selektion/Projektion (**Oracle Exadata**)
 - Optimierung von DB-Funktionen durch programmierbare/konfigurierbare GPUs und FPGA (Bsp. IBM Netezza)
 - Web-Informationssysteme (Multi-Tier-Architekturen) mit DB-Server und DB-Verarbeitung auf Applikations-Server
 - Datenintegrationsansätze, z.B. Föderierte DBS mit Query-Mediator
 - Spezialisierung erschwert Lastbalancierung, Erweiterbarkeit und Fehlertoleranz
- **hybride Partitionierung + Replikation von DBS-Funktionen**



Web-Informationssysteme mit Mid-Tier Caching



Integrierte vs. heterogene/föderierte MRDBS

■ integrierte Mehrrechner-DBS

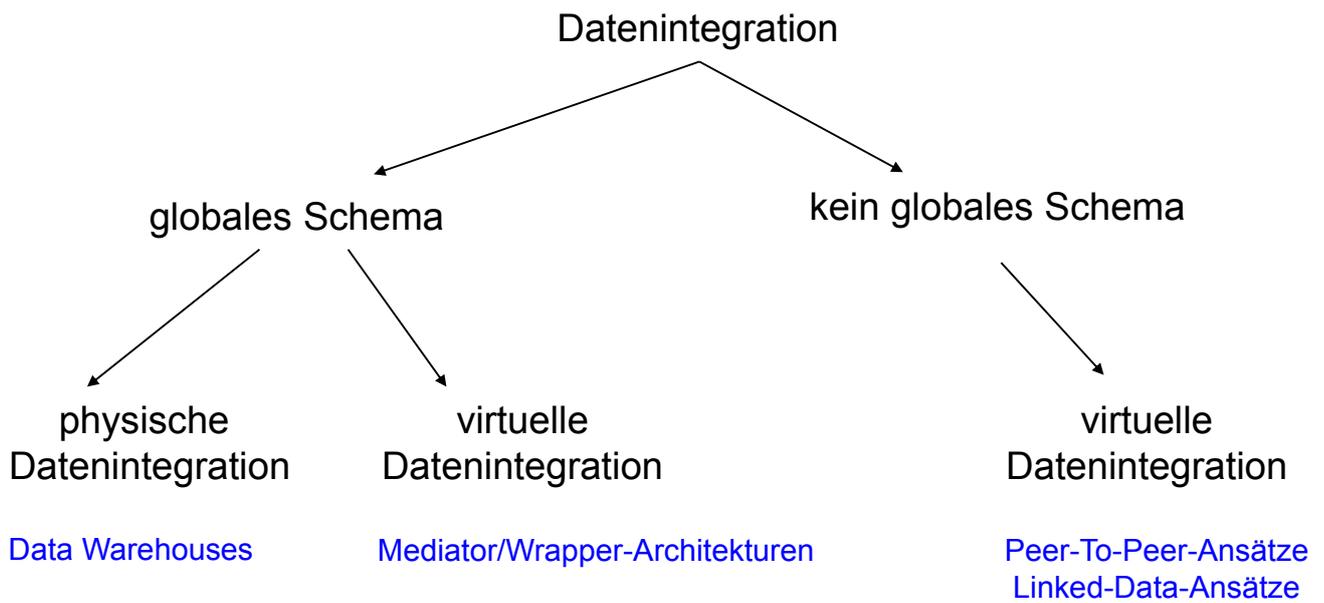
- 1 logische Datenbank: DB-Zugriff wie im zentralen Fall (Verteiltransparenz für AP)
- **Top-Down-Ansatz** zur Verteilung einer DB
- homogenes MRDBS (z. B. identische DBMS-Instanzen)
- geringe Autonomie für beteiligte DBMS
- Beispiele: Verteilte DBS, Parallele DBS

■ heterogene/ föderierte Mehrrechner-DBS

- **Bottom-Up-artige Kopplung** existierender Datenbanken
- weitgehend unabhängige DBMS mit privaten konzeptionellen DB-Schemata
- partielle Zulassung externer Zugriffe (Kooperation)
- Heterogenität bei Datenmodellen und Transaktionsverwaltung möglich
- Probleme mit semantischer Heterogenität
- Verteilungstransparenz i.a. nur bedingt erreichbar
- verschiedene Alternativen zur Datenintegration

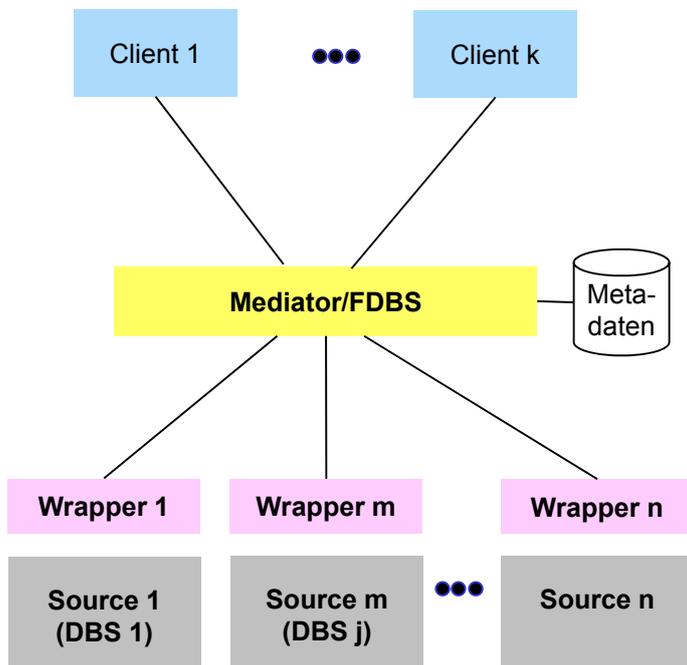


Heterogene DBS

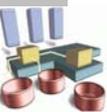
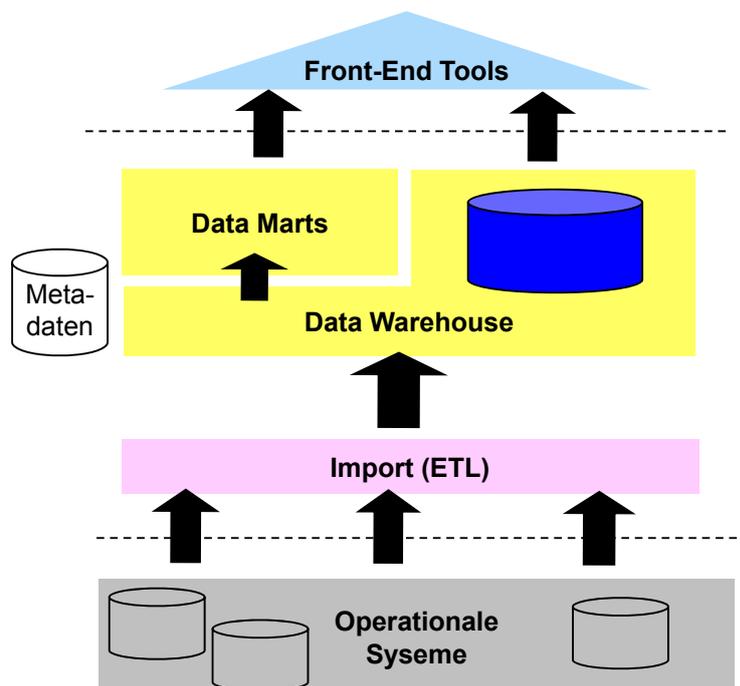


Grundlegende Alternativen zur Datenintegration

Virtuelle Integration
(Mediator/Wrapper-Architekturen,
föderierte DBS)

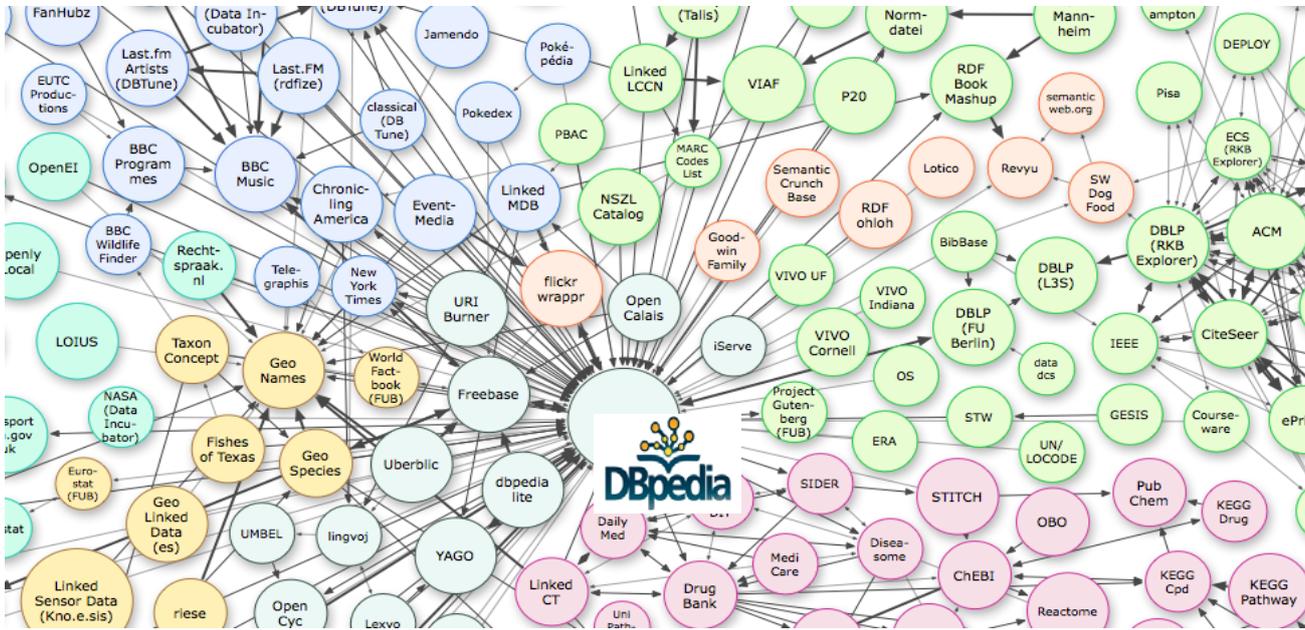


Physische (Vor-) Integration
(Data Warehousing)

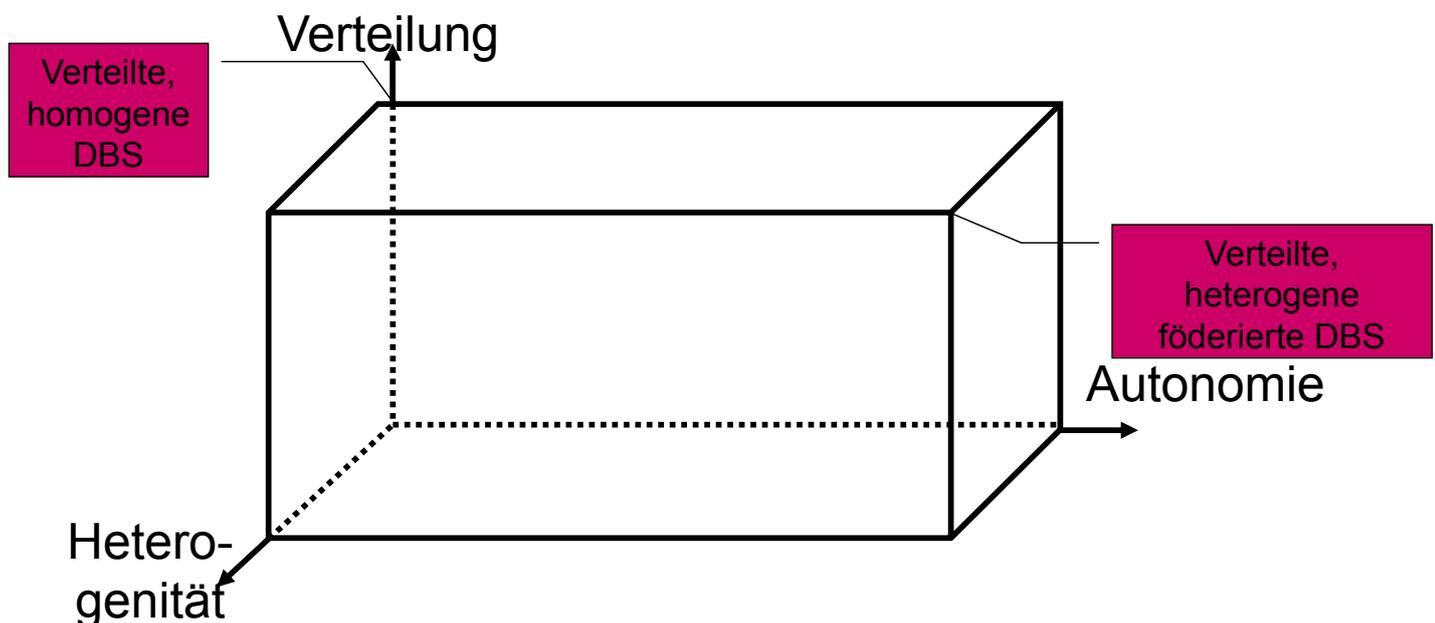


Linked Open Data

- vernetzte RDF-basierte Datenquellen
- Links (Mappings) zwischen Instanzen und Ontologie-Konzepten ermöglichen Datenintegration



Klassifikation nach Özsu/Valduriez



Grobbewertung von Mehrrechner-DBS

	Parallele DBS (SD, SN)	Verteilte DBS	Föderierte DBS
Hohe Transaktionsraten	++	o/+	o
Intra-TA-Parallelität	++	o/+	-/o
Skalierbarkeit	+ / ++	o/+	o
Verfügbarkeit	+	+	-
Verteilungstransparenz	++	+	o
Geographische Verteilung	-	+	+
Knotenautonomie	-	o	+
DBS-Heterogenität	-	-	+
Administration	o	-	- / --

■ Big Data Architekturen

- Nutzung von Hadoop-Cluster mit Analyse-Frameworks / SQL-on-Hadoop
- NoSQL auf Shared-Nothing-Cluster
- parallele DBS / NewSQL z.B. mit In-Memory-Datenbanken



3 Stufen der Verteilung

1. Scale-Up: mehrere Prozessoren innerhalb von 1 Knoten (Shared Everything)

- Sehr effiziente Kommunikation; Datenaustausch über Haupt- oder Externspeicher
- Direkter Zugriff auf gesamte Datenbank für alle DBMS-Instanzen; zentrale Datenstrukturen (Sperrtabelle, DB-Puffer, etc.)
- Vglw. einfache DB-Administration
- Wird von allen DBS-Herstellern unterstützt (Microsoft, Oracle, IBM ...)
- Begrenzte Erweiterbarkeit und Verfügbarkeit
- SMP-Leistungsfähigkeit reicht für Mehrzahl von Datenbanken
- Relativ hohe Kosten im High-End-Bereich

2. Scale-Out: SN/SD/Hybrid-Cluster

- Hohe Skalierbarkeit durch unabhängige Rechnerknoten (kein gemeinsamer Hauptspeicher, lokale Software)

3. Verteiltes DB-Mirroring (für SE, SD oder SN)



Skalierbarkeit: Scale-Up vs. Scale-Out

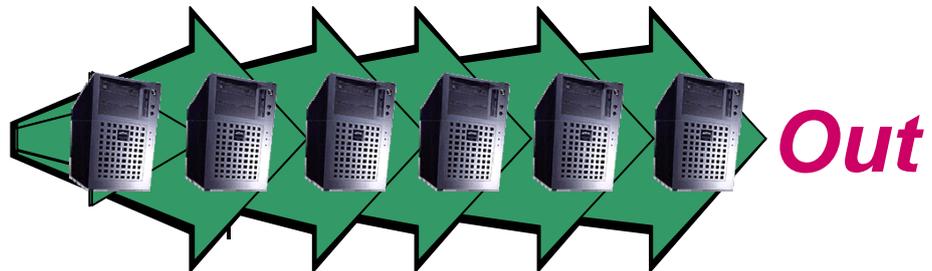


“Scale Up”

- Schnellere SMP-Knoten
- Shared Everything

“Scale Out”

- N unabhängige Rechner (z.B. Commodity-Server)
- Hinzufügen neuer Server nach Bedarf
- Shared Nothing oder Shared Disk (Cluster)

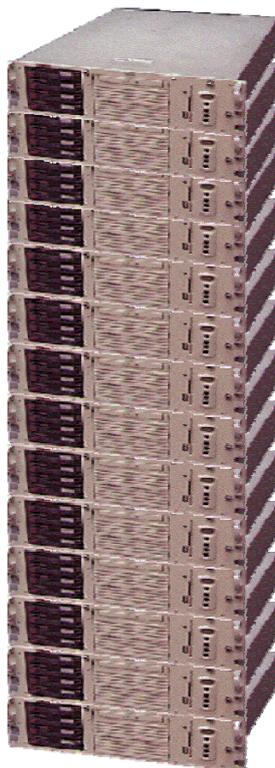


Scale-Out

- sehr viele preiswerte Standard-Knoten (Blades)
 - geringer Administrationsaufwand
 - leichte Erweiterbarkeit
 - ausreichend für gut partitionierbare Lasten aus einfachen Operationen
 - Bsp.: Hadoop-Cluster, Data Center von Google / Amazon etc.

VS.

- moderate Zahl von High-End-Servern



DB-Mirroring für Hochverfügbarkeit



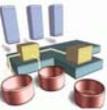
DB-Änderungen



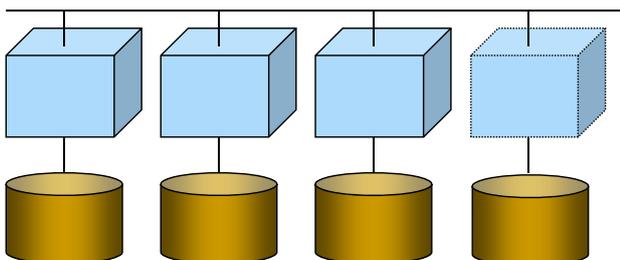
Primärsystem

Sekundärsystem

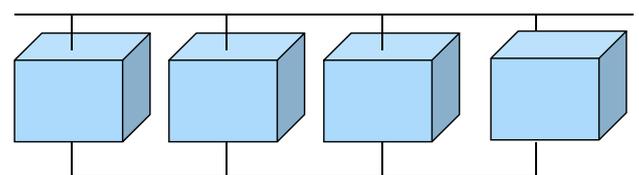
- komplette DB-Kopie an entferntem System (Geo-Replikation)
- fortlaufende Übertragung aller Änderungen aus Primärsystem (z.B. Log-Transfer) und Anwendung auf Kopie
- Schutz auch gegenüber Katastrophen
- anwendbar für alle PDBS-Architekturen im Primärsystem: SE (z.B. MS SQL-Server), SN, SD und Kombinationen



Shared Nothing (SN) vs. Shared Disk (SD)



SN



SD

- ❖ Teradata
- ❖ IBM DB2 (Windows, Unix)
- ❖ MS Parallel Data Warehouse
- ❖ viele NoSQL-Systeme

- ❖ Oracle (RAC, Exadata)
- ❖ IBM DB2 z/OS, DB2 PureScale
- ❖ SAP Sybase

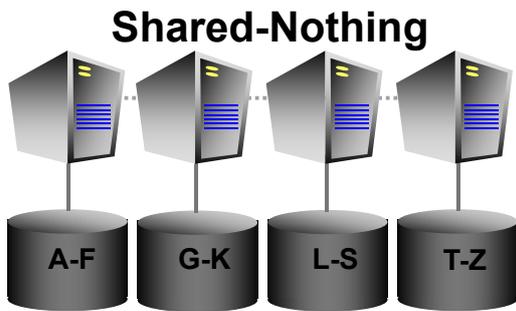
- SN auf SD-Hardware realisierbar und umgekehrt
- wesentlich ist Sicht der DBMS-Instanzen

SN: Partitionierung der Datenbank unter den DBMS-Instanzen mit daraus abgeleiteter Anfrage- und Transaktionsverarbeitung

SD: direkter Zugriff auf gesamte DB für jedes DBMS; lokale Puffer und Sperrtabellen, Datenaustausch über Externspeicher bzw. Nachrichten

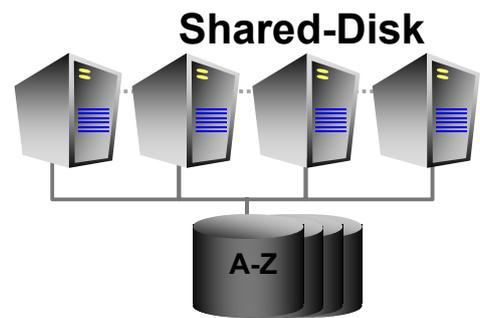


SN vs. SD: Leistungsfähigkeit



statische Datenpartitionierung bestimmt Ausführungsort von DB-Operationen und damit Kommunikationsaufwand

geringe Möglichkeiten zur dynamischen Lastbalancierung



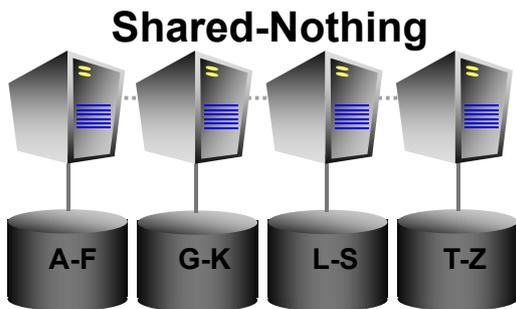
hohe Flexibilität zur Parallelisierung und Lastbalancierung aufgrund Erreichbarkeit aller Daten von jedem Knoten

hoher Aufwand für Synchronisation und Kohärenzkontrolle

- Lokalität ermöglicht Einsparungen
- nahe Kopplung reduziert Overhead

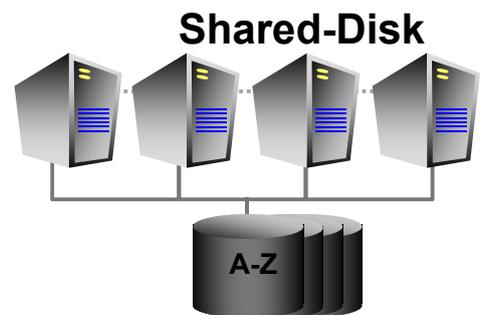


SN vs. SD: Erweiterbarkeit



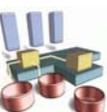
Hinzufügen neuer Knoten hardwareseitig einfach (viele Knoten sind möglich)

neuer Rechner erfordert physische Neuauflteilung der Datenbank ($N \rightarrow N+1$)

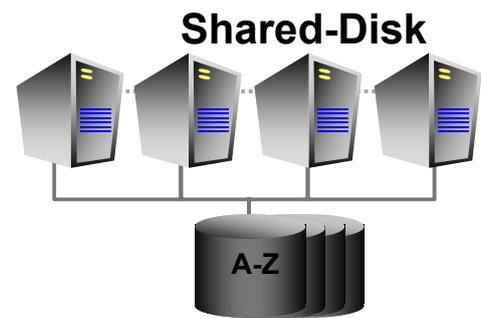
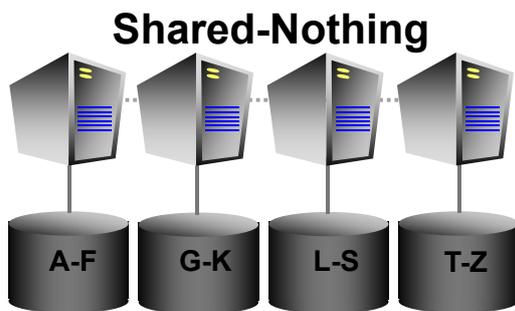


keine physische (Neu-) Aufteilung der DB bei neuem Rechner

direkte Plattenanbindung kann Rechneranzahl begrenzen



SN vs. SD: Recovery

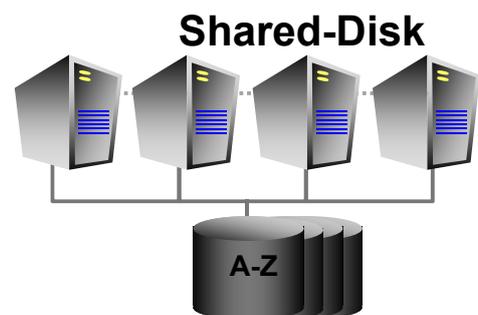
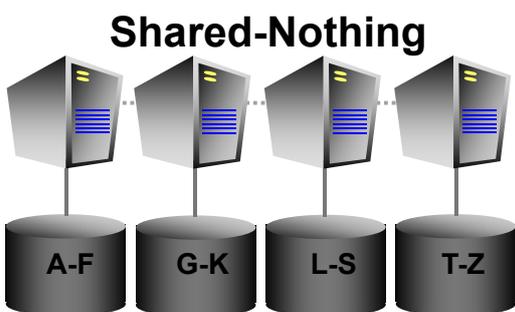


Übernahme/Recovery der betroffenen Partition durch anderen Rechner vorzusehen (ggf. Überlastungsgefahr)

gesamte DB bleibt nach Rechnerausfall erreichbar
komplexe Crash-Recovery
Erstellung einer globalen Log-Datei



SN vs. SD: Technische Probleme



DB-Partitionierung bzgl. Rechner
verteilte und parallele
Anfrageverarbeitung
verteiltes Commit-Protokoll
globale Deadlock-Behandlung
Replikationskontrolle
Katastrophen-Recovery ...

DB-Partitionierung bzgl. Platten
parallele Anfrageverarbeitung
globale Synchronisation
Kohärenzkontrolle
globaler Log, Crash-Recovery
Lastbalancierung
Katastrophen-Recovery ...



Zusammenfassung

- vielfältige Anforderungen an Mehrrechner-Datenbanksysteme führen zu verschiedenen Architekturtypen:
 - Parallele DBS, Verteilte DBS, Föderierte DBS, Data Warehouses ...
- Klassifikationsmerkmale
 - räumliche Verteilung, Rechnerkopplung, Externspeicheranbindung, integrierte/homogene vs. föderierte / heterogene DBS, funktionale Spezialisierung vs. Gleichstellung
- Parallele DBS:
 - Ziele: hohe Leistung/Parallelisierung, hohe Verfügbarkeit, Skalierbarkeit
 - lokale Rechneranordnung
 - Hauptansätze: Shared-Everything, Shared-Disk, Shared-Nothing
- Verteilte DBS: ortsverteilte, integrierte MRDBS (globales Schema)
- PDBS-Vergleich
 - Skalierbarkeit: Scale-Up (SE) einfacher umsetzbar als Scale-Out
 - DB-Mirroring für Hochverfügbarkeit
 - Scale-Out: Shared-Disk vs. Shared-Nothing

