NoSQL-Databases

Präsentation für
Advanced Seminar "Computer Engineering",
Matthias Hauck,
matthias.hauck@stud.uni-heidelberg.de

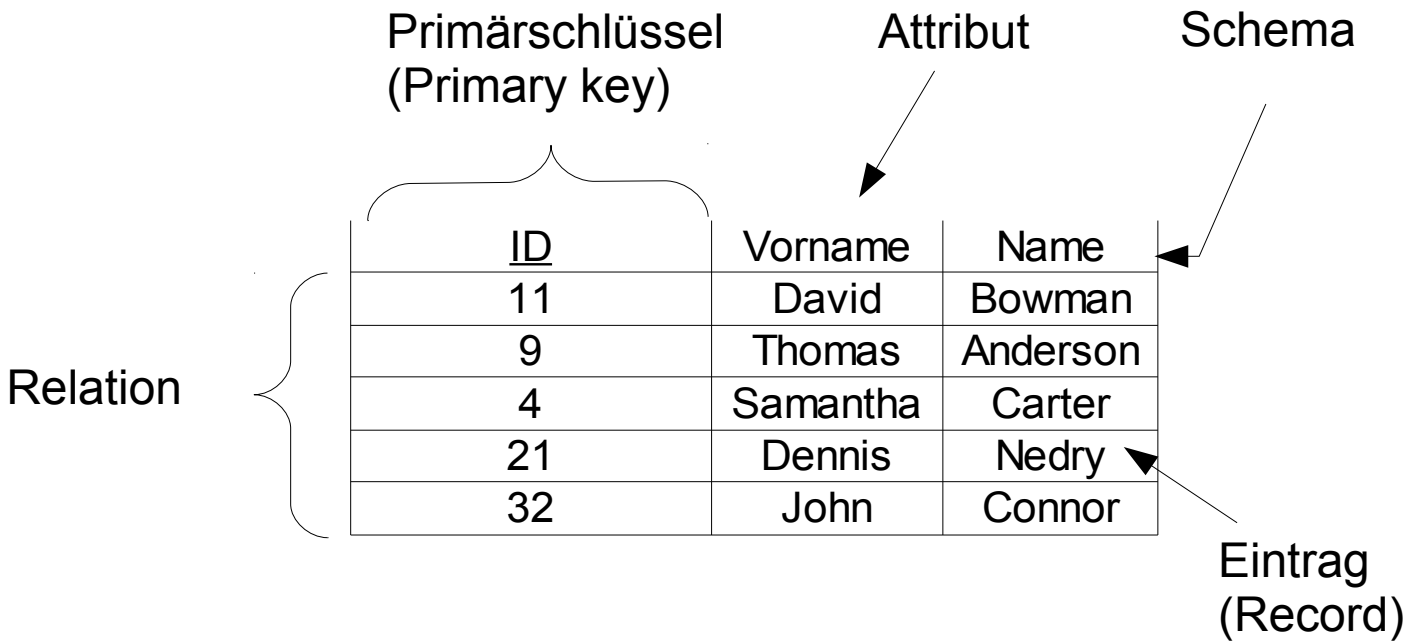
Klassische SQL-Datenbanken

- Anwendungsgebiet:
  - Geschäftsanwendungen
  - Behördenanwendungen
- Anwendungsklassen
  - Online Transaction Processing(OLTP)
  - Online Analytic Processing(OLAP)

Klassische SQL-Datenbanken

- Focus auf Robustheit
  - ACID
- Relationales Datenmodel
  - Star(Performanz)
  - Zeilen Orientiert(Daten anhängen)
- Ausgereift
  - Ursprung in den 70er
- Teuer
  - Oracle Database Enterprise Edition(pur) 47.500 $/Prozessor

Relationales Datenmodel



Beispiel Abfrage:
select LV.Name from Stud, Belegungen, LV where Name=Nedry,
Stud.ID = Belegungen.ID, Belegungen.ID=LV.ID

Neue Anforderungsprofile

- Webanwendungen
  - interaktive Datenzugriff
  - Verfügbarkeit wichtiger als Konsistenz
  - Anforderungen der Daten variiert
- Big Data
- Mobile Anwendungen
  - Synchronisation
  - Wenig Leistungsfähige Hardware

Anforderungen

- Extreme Skalierbarkeit
  - Viele Anfragen
  - Viele (stark wachsende) Daten
- Hohe Verfügbarkeit
- Schnell ändernde Software
- Niedrige Kosten
  - Geringer administrativer Aufwand
  - Standard Hardware

Lösungen
(Evolution)

- Cache Layer für SQL Datenbank
  - Memcached
- Middelware
  - Gizzard
- NoSQL

Lösung: NoSQL

- Ausrichtung auf Cluster
  - Teilweiser Verzicht auf Abstraktion
- Flexible Schemas
- Verzicht auf Features
- Ein Interface das nicht SQL verwendet

Fokus der Entwicklung

- Anpassung an die Aufgabenfelder
- Unterstützung verteilter Systeme
  - Replikation
  - Verteilung von Daten
- Optimierung der Performanz/Durchsatz

Arten von NoSQL-Datenbanken

- Key-Value Store
  - MemcacheDB, Amazon Dynamo
- Document Store
  - Lotus Notes, couchDB
- Spalten-orientierte Datenbanken
  - Bigtable, Cassandra
- Graph Datenbanken
  - Neo4j

Weitere Klassifikationen möglich!

Key/Value Store

- Einfacher Aufbau
- Einfache Abfragen
- Value wird nicht ausgewertet

Key	Value
Bunsen	
Buchteln	
Winter	

Document Store

- Strukturierte Daten
  - Gruppierung von gemeinsam genutzten Daten
- Flexibles Schema

```
"oname":"The Silence of the Lambs"
"name“: { "en":"The Silence of the Lambs"
          "de":"Das Schweigen der Lämmer"
          "fi": "Uhrilampaat" }
"regie":"Jonathan Demme"
"schauspieler":{"Jodie Foster",
                "Anthony Hopkins"}
"laufzeit":"113 min"
```

Spalten-orientierte Datenbanken

oder: Wide Column Store / Column Families

- Tabelle mit Zeilen und Spalten
- Zeilen besitzen unbegrenzte Anzahl an Spalten
- Spalten gehören zu einer Spalten Familie
- Die Anzahl der Spalten Familien ist begrenzt

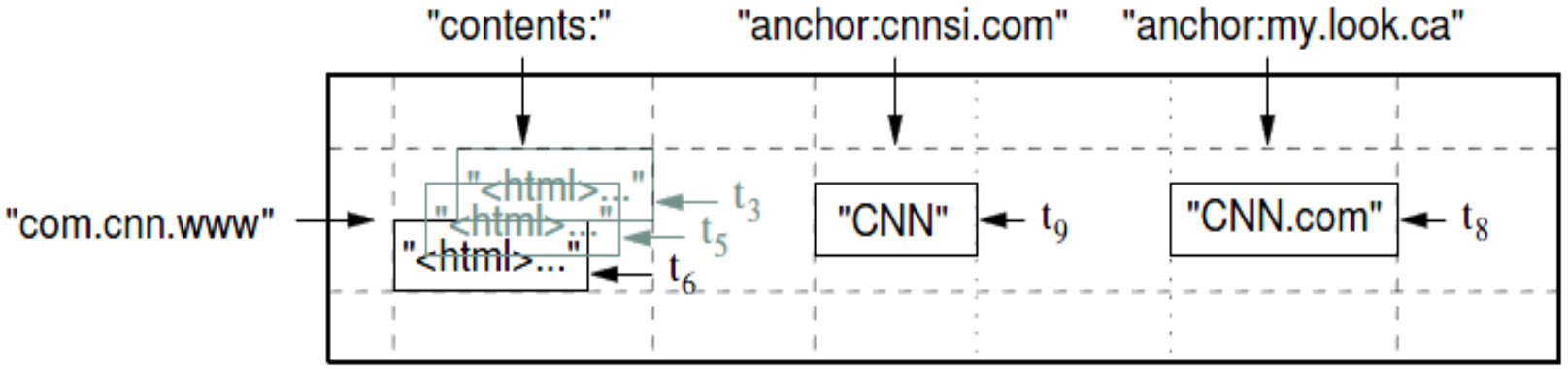
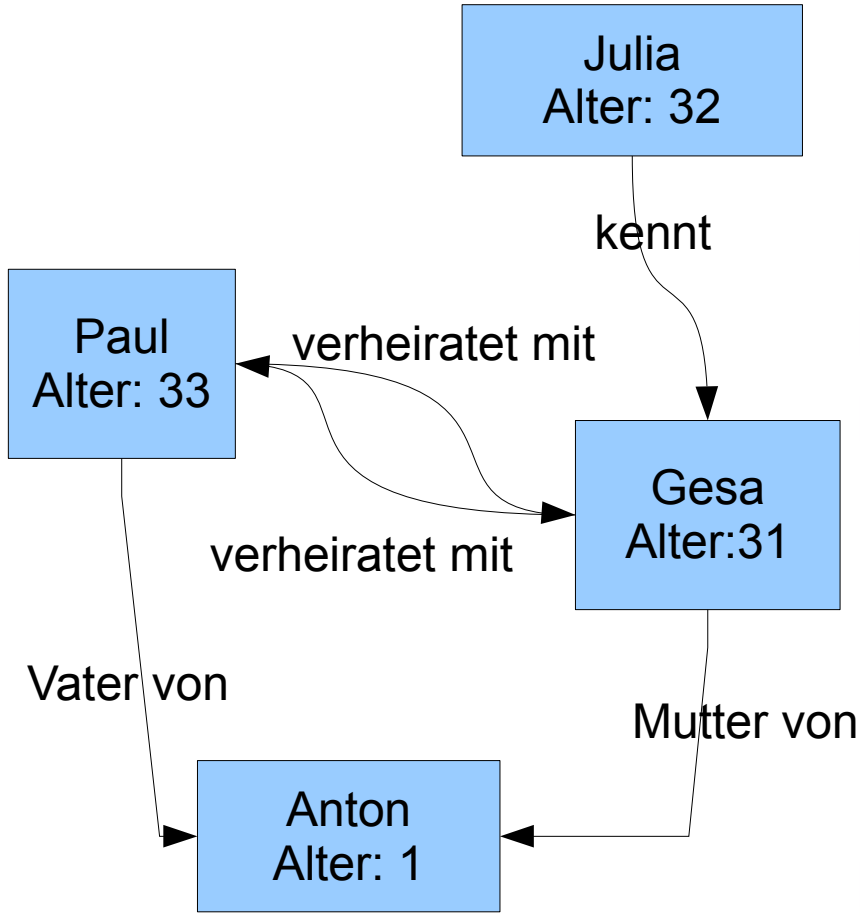


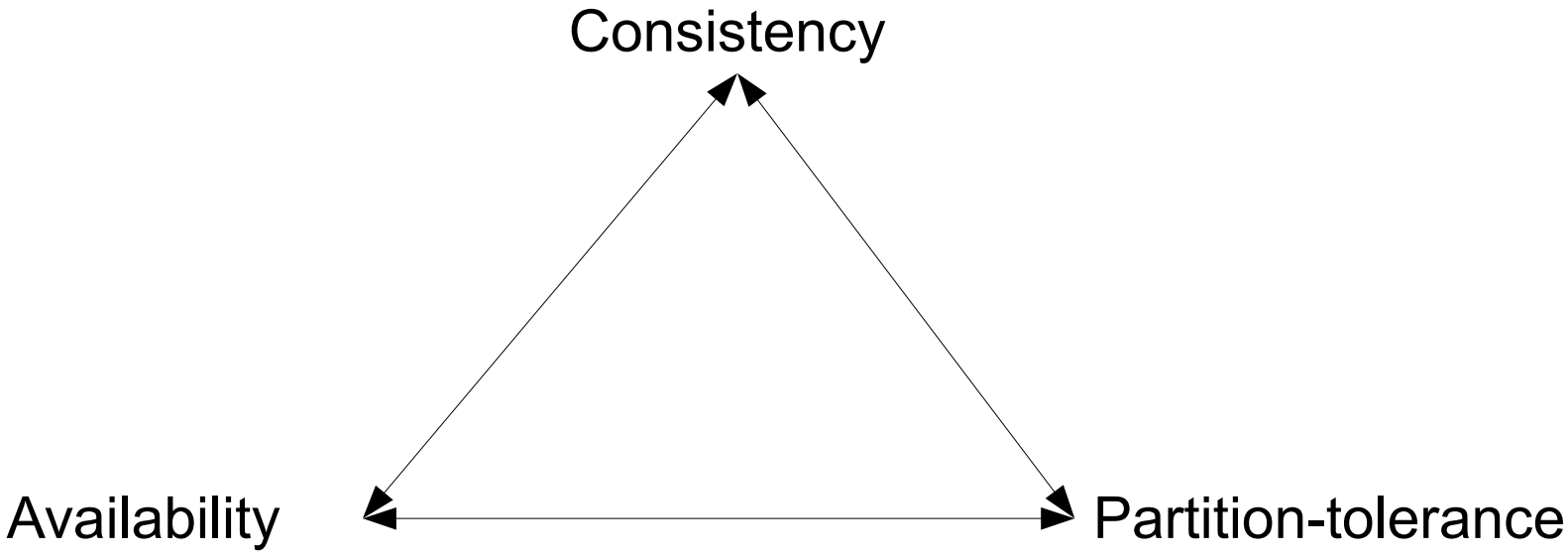
Bild: Google: Bigtable: A Distributed Storage System for Structured Data

Graph Datenbanken

- Modellierung der Daten als Graph
- Unterstützung von Graphen Verarbeitung



Verteilte Systeme: CAP-Theorem



Verteilte Systeme: BASE

Basically **a**vailable, **S**oft-state, **E**ventual consistency

- Updates nicht sofort im ganzen System sichtbar
  - Varianten möglich
  - Nicht alle Knoten müssen gleichzeitig Commiten
- Applikation muss Inkonsistenzen tolerieren

Verteilte Systeme:
BASE vs. ACID

ACID	BASE
Strong consistency	Weak consistency – stale data OK
Isolation	Availability first
Focus on “commit”	Best effort
Nested transactions	Approximate answers OK
Availability?	Aggressive (optimistic)
Conservative (pessimistic)	Simpler!
Difficult evolution (e. g. schema)	Faster
	Easier evolution

Quelle: Brewer, Eric A.: *Towards Robust Distributed Systems*. Portland, Oregon, July 2000. – Keynote at the ACM Symposium on Principles of Distributed Computing (PODC) on 2000-07-19., Folie 13

Verteilte Systeme: Sharding

- Automatische Verwaltung
  - Load Balancing
  - Daten Verteilung/Redistribution
- Lookup Mechanismus
  - Statisch/Dynamisch
  - Hash, Lookup Server
- Membership Managment

Performanz: Elimination von Overhead

- Kommunikation Frontend/Backend
- Logging
- Locking
- Latching
- Buffermangment

Performanz: Elimination von Overhead

- Kommunikation Frontend/Backend
  - Verlagerung von Logik in Client Lib
  - Komplexe Operationen auf dem Server(Stored procedures)
  - MapReduce Integration
- Logging
- Locking
- Latching
- Buffermangment

Performanz: Elimination von Overhead

- Kommunikation Frontend/Backend
- Logging
  - Verzicht auf Logs
  - Replikation/GEO distribution
- Locking
- Latching
- Buffermangment

Performanz: Elimination von Overhead

- Kommunikation Frontend/Backend
- Logging
- Locking
  - Verzicht auf Transaktionen(über mehr als einen Eintrag)
- Latching
- Buffermangment

Weitere Optimierungen

- In Memory DB
  - Performanz
  - Einfacheres Zugriffsmanagement
- Überarbeitung des Storage Layouts
  - Kompression

Kritik an NoSQL

- NoSQL hat nichts mit SQL zu tun
- Nichts neues
- Nicht standardisiert
- Kein kommerzieller Support / Lizenzprobleme
- Evatual Consistency ist kompliziert

Fazit & Ausblick

- SQL Datenbanken sind relevant
- NoSQL Datenbanken sind relevant
- NoSQL wird reifen
- Neue Technologien kommen
  - NewSQL