Programmierung von Systemen – 18 – NoSQL

Matthias Tichy & Stefan Götz | SoSe 2020

NoSQL

- NoSQL das erste Mal von Carlo Strozzi verwendet für eine DB ohne SQL-Zugriff ➔ no SQL
- Seit Anfang 2009 Wiederentdeckung des Begriffs für DB-Systeme, welche Abstand vom relationalen Konzept nehmen ➔ not only SQL
- "ur"alte Ansätze Hierarchie-DB, Netz-DB sind bereits Vertreter dieser "neuen" Idee

NoSQL – Warum?

- Aufkommende intensive Nutzung des Web (2.0) führt(e) zu Leistungsproblemen bei relationen Datenbanken
- SQL-DB effizient bei häufigen, kleinen Transaktionen
- heute aber:
  - Graphstrukturen
  - Indexierung von Dokumenten
  - Web-Seiten mit hoher Last (nur lesender Zugriff)
  - Location Based Services (Umkreissuche)
  - Time-Series Datenbanken

NoSQL – Warum?

- Performanz:
  - Lösung der Performanzprobleme früher über Aufrüstung des Servers (Scale-Up)
  - Heute reicht das nicht mehr, daher Lösung meist über Replikation der DB auf mehrere Knoten (Scale-Out)

- Konsistenz?!
  - NoSQL-Systeme häufig kein ACID-Modell, sondern lockerere Sicht auf Konsistenz (BASE = basically available, soft state, eventually consistent)

NoSQL – Warum?

- Schema:
  - Schemas oft zu starr und unflexibel für Anpassungen
- Suchanfragen gerade bei LBS komplexer:
  - "Welche E-Tankstelle kann ich in maximal 50km erreichen?"
  - "Welcher Briefkasten, der heute noch geleert wird, kann ich in 10 Minuten zu Fuß oder mit dem Fahrrad erreichen?"

Was, wenn nicht SQL?

- Key/Value Stores
  z.B. Dynamo, Voldemort, Riak
- Wide-Column-Stores
  z.B. HBase, Cassandra
- Document-Stores
  z.B. MongoDB, CouchDB
- Graphdatenbanken
  z.B. Neo4j, SonesDB

Key-Value-Store

- einfaches Schema aus Schlüssel und Wert
- sehr gute Performanz auf sehr einfachen Abfragen
- Replikation einfach
- Verteilung über hierarchische Indizes möglich

Wide-Column-Store

- auch Column-Family-System genannt
- sehr viele Zeilen mit sehr vielen Spalten
- jede Zeile kann unterschiedliche Spalten haben
- jede Zeile kann mehrere Instanzen von einer Spalte haben (Timestamp)

Document-Store

- Datenbank besteht aus einzelnen (hierarchischen) Dokumenten
- oft JSON- oder XML-Format
- Zugriff über eindeutigen Identifikator (evtl. als Pfad)
- semantische Zusammenhänge können besser erhalten bleiben als bei Key-Value-Stores

Graph-DB

- keine Hierarchie, sondern Graph als Verknüpfungsstruktur
- Semantic-Web-Forschung starker Treiber
- grundlegende Struktur oft Property-Graph:
  - Knoten und Kanten können mit Eigenschaften und Gewichtungen versehen werden