NoSQL Datenbanken – Beispielsysteme

Oberseminar: Datenbanksysteme – Aktuelle Trends

Sven Walter

22. Mai 2014

1 Key/Value-System – Redis

1.1 Allgemein

Redis [1] ist eine *Key/Value-Datenbank* oder genauer genommen ein *Datenstrukturserver*. Die gesamten Daten werden im Hauptspeicher gespeichert. Somit ist Redis außerdem eine *In-Memory-Datenbank*.

Redis is in *C* geschrieben und steht und *BSD* Lizenz.

Es gibt viele Clients in vielen Programmiersprachen [2]. Außerdem wird ein CLI-Programm mitgeliefert und es gibt eine direkte TCP-Schnittstelle auf Port 6379, auf die direkt zugegriffen werden kann (z.b *netcat*).

1.2 Datenstrukturen

Neben Zeichenketten als übliche Datenstrukturen, gibt es noch Listen, Mengen und Maps. Diese verschiedenen Strukturen unterstützen unterschiedliche Operationen. Beispiele dafür sind `PUSH` und `POP` für Listen oder das Bilden von Schnittmengen und Vereinigung von Mengen.

1.3 Weitere Funktionen

1.3.1 Persistenz

Obwohl Redis eine *In-Memory-Datenbank* ist, können die Daten persistent [3] gespeichert werden. Dafür gibt es die zwei Verfahren *RDB* und *AOF*. Ersteres speichert die Daten in regelmäßigen Abständen auf die Festplatte, während *AOF* jede Schreiboperation in eine Logdatei schreibt und diese somit wiederherstellbar macht.

1.3.2 Ablauf der Einträge

Jedem Schlüssel kann eine Gültigkeitsdauer (`EXPIRE`) zugewiesen werden [3]. Diese Schlüssel werden nach Ablauf dieser Dauer automatisch gelöscht.

1.3.3 Transaktionen

Redis unterstützt ebenfalls Transaktionen [3]. Dadurch kann eine Folge von Befehlen atomar ausgeführt werden. Hierbei ist zu beachten, dass diese Transaktionen keine Wiederherstellung erlaubt.

1.3.4 LRU Cache

Redis kann als *Least-Recently-Used-Cache* [3] verwendet werden. Das bedeutet, dass eine Obergrenze für die Speichernutzung festgelegt werden kann und alte automatisch entfernt werden. Für die Auswahl des zu entfernenden Schlüssel gibt es verschiedene Regeln.

1.3.5 Replikationen

Es wird *Master-Slave-Replikation* [3] unterstützt. Die *Slaves* können hierachisch angeordnet werden und bespielsweise genutzt werden, um Leseoperationen auf verschiedene Server zu verteilen oder die Persistenz auf einen anderem Server durchzuführen.

1.3.6 Partitioning

Partitionierung [3] ist das Verfahren, unterschiedliche Schlüssel auf unterschiedliche Server zu verteilen.

1.3.7 Pipelining

Als *Pipelining* [3] wird das Verfahren bezeichnet, mehrere Anfragen nacheinander an den Server senden zu können, ohne auf die jeweilige Antwort warten zu müssen.

1.3.8 Nachrichtendienst

Redis implemeniert auch einen einfachen *Publish/Subscribe*-Nachrichtendienst.

1.4 Verwendung

Redis wird bei einigen großen Unternehmen verwendet. Beispiels weise nutzen *Twitter* [4], *GitHub* [5] und *Stackoverflow* [6] *Redis* zum cachen ihrer Daten.

2 Document Store – MongoDB

2.1 Allgemein

MongoDB [7] ist eine schemafreie dokumentenorientierte Datenbank. Es steht unter *AGPL 3* Lizenz und ist momentan die verbreiteste *NoSQL*-Datenbank und an fünfter Stelle unter allen Datenbanken [8].
Es gibt offizielle Schnittstellen für viele Programmiersprachen [2], die bei *MongoDB* als Treiber bezeichnet werden. Zusätzlich gibt es viele inoffizielle Werkzeuge, wie beispielsweise *ObjectMapper*. Diese ermöglichen es Objekte von objektorientierten Programmiersprachen direkt als Dokumente abzulegen.

2.2 Datenstrukturen

Die Dokumente werden in *JSON*-ähnlichen Dokumenten gespeichert. Dieses Format heißt *BSON* und ist eine binäre Version von *JSON*. Ein einzelnes *JSON*-Objekt ist ein *Document* und ist das äquivalent eines Tupel in relationalen Datenbanken. Eine Menge von Dokumenten wird als *Colleciton* bezeichnet und ist das äquivalent einer Tabelle.

2.3 Weitere Funktionen

2.3.1 Komplette Indizes

MongoDB unterstützt das anlegen eigener Indizes [7]. Dabei ist es auch möglich einen Index für Schlüssel eines Unterobjektes oder einer Liste zu setzen.

2.3.2 Replikation

Es wird *Master-Slave-Replikation* [7] unterstützt.

2.3.3 Sharding

Als *Sharding* [7] wird die Verteilung der Daten auf unterschiedliche Server bezeichnet. Dadurch wird horizontale Skalierung ermöglicht.

2.3.4 Aggregation

MongoDB unterstüzt viele Operationen zur *Aggregation*. Neben den typischen Operationen wie `min`, `max`, `avg` und `sum` gibt es noch weitere, wie beispielsweise `MapReduce` oder die Möglichkeit eigene *JavaScript*-Funktionen zur Gruppierung zu übergeben.

2.3.5 GridFS

GridFS ist eine Spezifikation, um Dateien speichern zu können. Können Daten gespeichert werden, die größer als maximale Dokumentengröße von `16 MiB` sind.

2.4 Verwendung

MongoDB wird von vielen großen Unternehmen verwendet [10]. Beispielsweise wird es von *Expedia*, *LinkedIn* und *SourceForge* als Teil der Datenbankinfrastruktur verwendet.

3 Graphendatenbank – Neo4j

3.1 Allgemein

Neo4j [11] ist eine Graphendatenbank unter *GPLv3* Lizenz oder einem alternativen komerziellen Lizenzmodell. Es gibt ein *CLI*-Programm, eine Webschnittstelle zur Administration und eine *REST*-Schnittstelle. Durch letzteres ist es besonders einfach die Datenbank in die verschiedenen Programmiersprachen einzubinden.

3.2 Datenstruktur

Wie bei Graphen üblich besteht die Datenbank aus Knoten und Beziehungen. Die Beziehungen sind gerichtent, aber diese Richtungen müssen bei der Abfrage nicht unbedingt beachtet werden. Zusätzlich können die Knoten und Beziehungen mit Eigenschaften versehen werden. Außerdem können den Knoten *Labels* zugewiesen werden, die den Art des Knoten beschreibt (z.B. „Profil“ oder „Nachricht“).

3.3 Weitere Funktionen

3.3.1 ACID

Neo4j ist ACID-konform [12], was bei NoSQL-Datenbanken selten ist.

3.3.2 Optionale Schemata

Es werden optionale Schemata unterstützt, die bei *Neo4j* *Constraints* genannt werden. Damit kann die Sturktur der Graphen festgelegt werden. Beispielsweise kann ein bestimmter Wert für einen bestimmtes *Label* als *Unique* spezifiziert werden.

3.3.3 Indizes

Neo4j unterstützt ebenfalls Indizes für die Eigenschaften eines Knotens oder einer Kante.

3.4 Verwendung

Neo4j wird von vielen großen Unternehmen verwendet [13]. Darunter sind Unternehmen die *ebay*, *Chip Online*, *Deutsche Telekom* und *TomTom*.

Quellen

1. **„Redis“** – offizielle Webseite – http://redis.io/ – Stand: 16. Mai 2014
2. **„Redis Clients“** – http://redis.io/clients – Stand: 16. Mai 2014
3. **„Redis Documentation“** – http://redis.io/documentation – Stand: 16. Mai 2014
4. **„Timelines at Scale“** – Vortrag eines Twitter Ingeneurs – http://www.infoq.com/presentations/Twitter-Timeline-Scalability – Stand: 20. Mai 2014
5. **„How We Made GitHub Fast“** – Vortrag eines GitHub Ingeneurs –https://github.com/blog/530-how-we-made-github-fast – Stand: 20. Mai 2014
6. **„Does Stack Overflow use caching and if so, how?“** – http://meta.stackexchange.com/a/69172 – Stand: 20. Mai 2014
7. **„MongoDB“** – offizielle Webseite – http://www.mongodb.org/ – Stand: 17. Mai 2014
8. **„DB-Engines Ranking“** – http://db-engines.com/de/ranking – Stand: 20. Mai 2014
9. **„MongoDB Drivers“** – http://docs.mongodb.org/ecosystem/drivers/ – Stand: 17. Mai 2014
10. **„Production Deployments“** – http://www.mongodb.org/about/production-deployments/ – Stand: 21. Mai 2014
11. **„Neo4j“** – ofizielle Webseite – http://neo4j.com/ – Stand: 18. Mai 2014
12. **„The Neo4j Manual v2.03“** – http://docs.neo4j.org/chunked/stable/ – Stand: 18. Mai 2014
13. **„Neo4j Customers“** – http://neo4j.com/customers/ – Stand: 21. Mai 2014