Apache Cassandra als Beispiel eines Wide Column Stores

Tom Delle

Zusammenfassung: Dieses Dokument dient der Zusammenfassung des gleichnamigen Vortrages im Oberseminar *Datenbanksysteme aktuelle Trends* im Sommersemester 2015 - Informatik Master HTWK-Leipzig.

Keywords: Apache Cassandra, Wide Column Store, Big Data, NoSQL

1. EINLEITUNG

Im Rahmen dieses Oberseminars steht das Thema *Big Data* im Fokus der Vortragsreihe. In diesem Vortrag wird ein wichtiger Vetreter der *No-SQL-Datenbanken* vorgestellt. Die Wide-Column-Store Datenbank Apache Cassandra wurde von Facebook entwickelt um den riesigen Datenmengen dieses sozialen Netzwerkes gerecht zu werden. Der Auslöser der Entwicklungen war dabei das *Inbox-Search-Problem* [1] bei einem Stand von 100 Millionen Nutzern und 7TB Inbox-Daten.

2. WIDE COLUMN STORES

Das Alleinstellungsmerkmal steckt hier bereits im Namen. Wide-Column-Stores können Datensätze mit beliebiger Spaltenanzahl aufnehmen. Ein Datensatz kann bis zu 2 Milliarden Spalten besitzen. Speziell die Datenspeicherung unterscheidet sich von anderen *No-SQL*-Vertretern.

2.1 Datenmodell

Das Datenmodell basiert auf Tabellen(table) wobei Datensätze in Zeilen modelliert werden. Jede Zeil ist durch eine Zeilenid(kann auch zusammengesetzt sein) eindeutig identifiziert. Jede Zeile kann bis zu 2 Milliarden Spalten aufnehmen. Spalten werden logisch in Spaltenfamilien (Column-Families) gruppiert.

2.2 Vorteile

Wide-Column-Stores eignen sich für sehr großen Datenmengen von unstrukturierten Daten. Speicherbedingt sind spaltenweise Aggregationen und Zugriffe sehr performant. Das dynamische Hinzufügen von Spalten erlaubt eine größere Schemaflexibilität als in relationalen Datenbanksystemen.

2.3 Nachteile

Operationen auf viele Spalten bzw. über mehrere Spaltenfamilien ist durch die physisch spaltenorientierte Speicherung sehr aufwendig. Auch das einfügen von neuen Zeilen fordert hohe I/O-Kosten.

[1] https://www.facebook.com/notes/facebook/inbox-search/20387467130

3. APACHE CASSANDRA

Apache Cassandra ist laut *DB-Ranking* [2] die beliebteste *Wide-Column-Store*-Datenbank und nach *MongoDB* die beliebteste *No-SQL*-Datenbank. Cassandra ist ein verteiltes Speichersystem mit dem Datenmodell von Google's *BigTable* und der verteilten Architektur von Amazon's *Dynamo*.

Abbildung 1. Entstehung Apache Cassandra

3.1 Datenmodell

Das Datenmodell von Apache Cassandra weicht etwas von dem der *Wide-Column-Stores* ab. Abbildung 2 veranschaulicht das Modell.

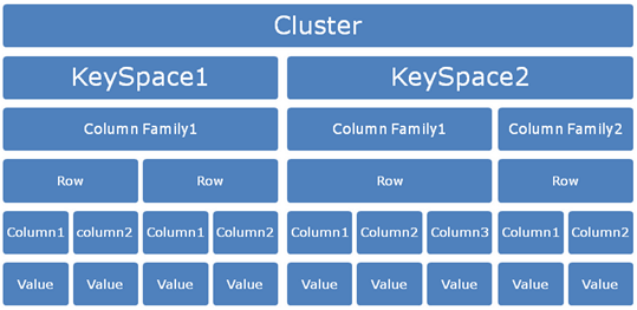

Abbildung 2. Datanmodell

[2] http://db-engines.com/de/ranking

Um die Vorstellung zu vereinfachen kann man von einem Containermodell ausgehen. Ein Cluster entspricht dabei dem Datenbankserver und enthält eine oder mehrere Datenbanken(Keyspaces). Eine Spaltenfamilie(Column-Family) entspricht einer Tabelle und enthält Zeilen(Rows), welche mit einer eindeutigen Zeilenid identifiziert werden. Jede Zeile enthält beliebig viele(<2 Mrd.) Spalten, wobei jede Spalte ein *Key-Value*-Paar ist.

3.2 Systemarchitektur

Cassandra ist ein verteiltes Speichersystem ohne *Single Point of Failure(SPOF)* und ausgelegt um auf mehreren huntert Knoten zu laufen. Dies stellt einige Anforderungen an die Systemarchitektur welche in dieser Sektion kurz vorgestellt werden.

Token Range
Jeder Knoten(Node) im Cassandra Cluster wird anteilig in einen Bereich von 128 Bit eingeteilt.

Partitionierung
Die Partitionierung erfolgt über den zugeordneten Bereich des *Token Rank*. Um einen Datensatz(Partition) zu verteilen wird ein 128Bit MD5-Hash des Primärschlüssels(1. Spalte) erzeugt. So kann jeder Datensatz einem eindeutigen Bereich bzw. Knoten im *Token Range* zugeordnet werden(siehe Abbildung 3).

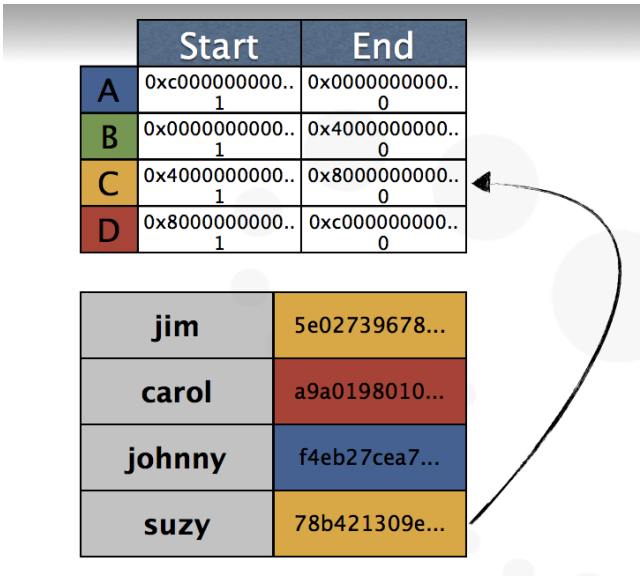

Abbildung 3. Partitionierung

Replikation
Die Replikation wird für die gesamte Datenbank(Keyspace) gesetzt und umfasst einen Replikationsfaktor und eine Replikationsstrategie. Der Replikationsfaktor gibt die Anzahl der Replikationen an. Die Replikationsstrategie gibt an wie die Kopien auf weitere Knoten aufgeteilt werden sollen. Beispielsweise verteilt die *SimpleStrategy* die Kopien einfach im Uhrzeigersinn entlang der *Token Range*.

3.3 Anfrageschnittstellen

Cassandra liefert bereits eine breite Auswahl an sogenannten Client-Drivers. Dies sind Bibliotheken für die meisten Programmiersprachen wie Java, Python, C# und weitere. Cassandra bietet auch ein Entwicklungstool [3] an. Alle Anfragen basieren dabei auf der *Cassandra Query Language(CQL)*

CQL
Die *Cassandra Query Language* [4] ist das Primärinterface zum Cassandra DBMS. Die Anfragesprache ist SQL ähnlich und kann über die Client-Drivers sowie über die *CQL-Shell* bedient werden.

4. DEMONSTRATION

Im Rahmen des Vortrages wird es eine Live-Demonstration geben welche mit dem *Cassandra-Cluster-Manager* durchgeführt wird. Diese Tool erlaubt die simple Konfiguration eines Test-Clusters für demonstrative Zwecke. In Produktivumgebungen sollte dieses Tool nicht genutzt werden.

5. ZUSAMMENFASSUNG

Natürlich können nicht alle relevanten Punkte des Vortrages bzw. von Apache Cassandra in diesem Abstract behandelt werden. Es wurde ein Überblick des Vortrages zu schaffen und es wurde versucht wichtige Punkte in aller Kürze darzustellen. Apache Cassandra hat enormes Potential speziell seit der Eingliederung als Apache Top-Level Projekt(2010). Als Abschlussmotivation soll noch ein Zitat von Aaron Turner von Synfinatic aufgeführt werden:

took me 10hrs to notice a cassandra node hat a hw failure because everything just kept working.

LITERATUR

Thomas Kudraß. Taschenbuch Datenbanken *ISBN* 978-3446435087
Datastax Docs. http://docs.datastax.com
DB-Ranking. http://db-engines.com/
Apache Cassandra. http://cassandra.apache.org/

Sowie alle Quellen des Vortrages.

[3] http://www.datastax.com/what-we-offer/products-services/devcenter
[4] https://cassandra.apache.org/doc/cql/CQL.html