Skalierbare Datenbanksysteme (NoSQL)-Document Stores (CouchDB) für das Cloud Computing-Seminar an der HS-Mannheim im SS2012

Marcel Sinn

Hochschule Mannheim
Fakultät für Informatik
Paul-Wittsack-Straße 10
68163 Mannheim
marcel.sinn@stud.hs-mannheim.de

Zusammenfassung Diese Ausarbeitung zeigt die Vorteile von NoSQL Datenbanken - speziell Document Stores - auf, anhand des Beispieles CouchDB. Dem Leser werden Grundlagen vermittelt, aber auch tiefgreifendere Einblicke in die API, Funktionsweise sowie Funktionalitäten gegeben. Zum Schluss werden die Vor- wie Nachteile gegeneinander abgewogen, und die Einsatzgebiete von Document Stores aufgezeigt.

In der heutigen Zeit des Internets werden verteilte Anwendungen immer selbstverständlicher. Um mit der Zeit zu gehen, wurden einige Alternativen zu den traditionellen SQL-Datenbanken entwickelt, die den Anforderungen von verteilten Anwendungen gerecht werden.

1 Die Geschichte von CouchDB

Die Abkürzung CouchDB bedeutet 'Cluster of unreliable commodity hardware Data Base'

(Zu Deutsch: 'Datenbank auf einem Cluster aus unzuverlässiger Standardhardware'.)

Sie ist ein Akronym und entstand erst später.

Entwickelt wird CouchDB seit 2005 von Damien Katz, der davor Senior Developer bei Lotus Notes war, eine der (wenn nicht der) bekannteste Document Store. Sein Ziel war es eine Dokumentenorientierte Datenbank mit MapReduce-Ansatz zu entwickeln.

Seit 2007 wurde CouchDB der eigene Port 5984 von der IANA zugesichert, und ist somit nun offizieller Standard Port von CouchDB.

Ab 2008 wurde CouchDB als 'BrustkastenProjekt' bei der Apache Foundation aufgenommen, und unterliegt nun dessen Bestimmungen.

Noch im gleichen Jahr im November wurde CouchDB aufgewertet zu einem vollwertigen Apache Projekt und ist jetzt gleichbedeutend mit z.B.: Apache und Tomcat Webserver

Die aktuellste Version ist 1.2.0 die am 10. April 2012 veröffentlicht wurde.

2 Was ist ein Document Store ?

Als ein Document Store wird eine dokumentenorientierte Datenbank bezeichnet. Dokument Datenbanken zählen zur NoSQL (das so viel wie 'not only SQL' beudetet) Bewegung. Diese Idee ist aber wesentlich jünger, sie entstand also erst nach den Dokumentdatenbanken. Bei den oben genannten Document Stores bilden Dokumente die Grundeinheit der Datenbank, und nicht wie bei SQL die Relationen.

In SQL mussten die Dokumente noch mit vordefinierten festem Schemata in den Datenbanken hinterlegt werden, hier ist dies nicht mehr der Fall: Dokumente können beliebig viele, und neue Attribute, ohne dazu Änderungen vornehmen zu müssen, enthalten.

Als Dokumenten können verstanden werden:

- Strukturierte Daten: Zum Beispiel Textdateien
- Binary Large Object: Zum Beispiel Videofilme

3 Warum CouchDB ?

In diesem Kapitel sollen einige Vorteile von CouchDB dargestellt werden, die die Vorzüge von Dokumentdatenbanken beschreiben:

- Das Motto 'Entspann dich' wird bei CouchDB ganz groß geschrieben. Darunter fallen verschiedene Ideen, wie das leicht zu verstehende Grundkonzept CouchDBs, welches das Erlernen erleichtern soll. Ein anderer großer Vorteil besteht im Produktivbetrieb, hier punktet CouchDB mit seiner fehlertoleranten Architektur, die so geschafften ist, das sich Fehler nur auf einzelne Requests beschränken und nicht fortpflanzen.

- 'Nicht im Weg zu stehen, wenn kreative Menschen versuchen ein Problem zu lösen' ist der nächste Punkt. CouchDB möchte den Entwickler nicht bremsen oder in seiner Kreativität einschränken indem es mit proprietären oder schwer Erlernbaren Technologien aufwartet. CouchDB nutzt eine sehr einfach zu bedienende REST basierende API. Diese Technologie setzt auf dem Altbewährten HTTP-Protokoll auf, und ist somit bewährt.

– Durch den Einfluss der Apache Foundation besitzt CouchDB eine sehr gute, und vor allem einfach gehaltene Dokumentation, die es jedem ermöglichen sich in CouchDB einzuarbeiten.

– Des Weiteren ist CouchDB auf wechselnde Last ausgelegt, welches im heutigen Internetzeitalter eine sehr wichtige Eigenschaft ist, und die Einsatzgebiete von CouchDB auf viele Anwendungsszenarien erweitert.

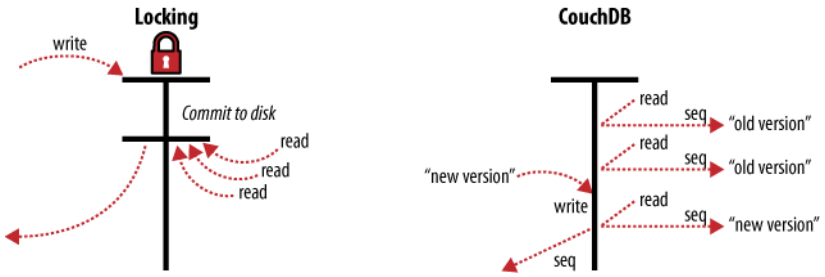
– CouchDB besteht aus 'in sich abgeschlossenen' Daten. Warum dies ein Vorteil ist, machen wir uns an einem kleinen Beispiel klar: Eine Rechnung enthält alle nötigen Informationen wie, Preis, Produkt/Dienstleistung, Diensterbringer, Dienstnehmer, Datum usw. Jeder schätzt es all jene Informationen auf einen Blick zu haben. Bei Document Stores werden solche Daten, wie man es aus dem Leben gewöhnt ist, genauso gespeichert, als ein in sich abgeschlossenes Dokument, das alle nötigen Informationen vor Ort und Stelle bereithält.

  Im Vergleich zu SQL würde es so aussehen dass: 'Jede Rechnung wird als Zeile in einer Tabelle gespeichert. Diese Zeile verweist auf andere Zeilen in anderen Tabellen ? eine für den Käufer, eine für den Verkäufer, eine für jedes Teil was verkauft wurde und noch mehr Zeilen, die wiederum die verkauften Teile genauer beschreiben.'

  Dies zeigt sehr gut das der Einsatz von relationalen Datenbanken nicht immer die beste Alternative da stellt. Zugleich beschreibt die Idee, Daten in sich abgeschlossen zu halten, eine der wichtigsten Eigenschaften von Dokumentdatenbanken.

– Syntax und Semantik spielt ebenfalls eine wichtige Rolle. Nimmt man Visitenkarten als Beispiel, wird sehr gut deutlich worin der Vorteil besteht. Visitenkarten haben viele Gemeinsamkeiten wie Vor-, und Nachname, aber auch manchmal Unterschiede wie das vorhanden sein einer Fax-Nummer. Trotz dieser kleinen Unterschiede erkennen wir Menschen an den Gemeinsamkeiten dass es sich um eine Visitenkarte handelt. In SQL müsste die Information 'Kein Fax' extra mitgespeichert werden, bei Document Stores kann diese Information einfach weggelassen werden ? durch das einfache weglassen wird ebenfalls zum Ausdruck gebracht das 'kein Fax vorhanden' ist, so wie es bei uns Menschen auch funktioniert.

– Ein Weiterer Grund für CouchDB ist die integrierte Funktionalität MVCC (Multiversion Concurrency Control). Diese trägt dazu bei Lese- und Schreibblockaden zu vermeiden. Des Weiteren werden Dokumente in der Datenbank versioniert, d.h. das beim Ändern eines bereits vorhandenen Dokuments automatsch eine neue Version angelegt wird, auf die jeder Zeit wieder zurückgesprungen werden kann.



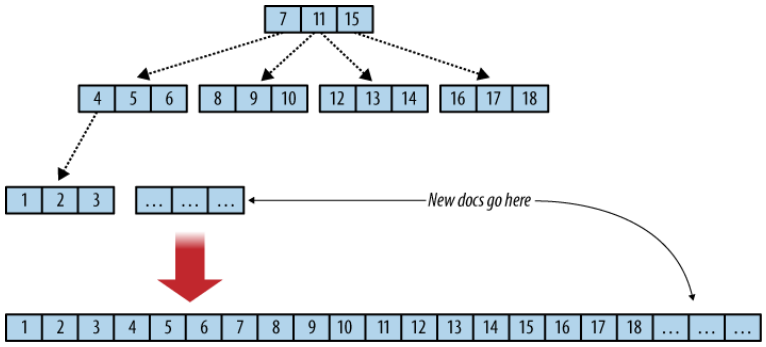
– Eine Besonderheit ist die verwendete Programmiersprache Erlang. Diese ist von Haus aus darauf ausgelegt folgende Eigenschaften in sich zu vereinen: Parallelität, hohe Verfügbarkeit, Fehlertoleranz, auswechseln von Modulen zur Laufzeit.

  Diese Eigenschaften mussten gegeben sein da damit z.B.: Vermittlungsstellen von Telefonnetzen entwickelt wurden, bei denen die obig genannten Eigenschaften unabdingbar sind.

– CouchDB benutzt 'B+Trees' als innere Datenstruktur.

> From a practical point of view, B-trees, therefore, guarantee an access time of less than 10 ms even for extremely large datasets.
>
> —*Dr. Rudolf Bayer, inventor of the B-tree*

  Daraus lässt sich entnehmen das CouchDB vor allem für sehr große Datenmengen gut geeignet ist. Weitere Vorteile sind leichte iterierbarkeit, Blätter die immer sortiert sind und die Geschwindigkeit die mit der 'B+Tree'-Datenstruktur einhergehen.



Zum Schluss nochmal die Wichtigsten Eigenschaften auf einen Blick:

CouchDB

Vorteile:

+ Flexibel, da die Dokumente keinem Schema entsprechen müssen;

+ Dokumente entsprechen sehr gut natürlichen Objekten: einfaches Design, kein objektrelationales Mapping und keine Normalisierung;

+ einfache Bedienung: einfache Web-Schnittstellen und bekannte Techniken wie HTTP-REST, JSON und Javascript;

+ einfacher bidirektionaler Replikationsmechanismus;

+ skaliert sowohl nach oben (Cluster) als auch nach unten (Smartphone).

Nachteile:

- Genügt keinen hohen Konsistenzanforderungen.

4 Die API

Die API von CouchDB kann grob in vier Bereiche aufgeteilt werden:

- Server
- Datenbanken
- Dokumente
- Replikation

Aus diesen werden nun exemplarisch einige Funktionen vorgestellt.

4.1 Server

Um den Server aufzurufen reicht es folgende URL im Browser einzutippen:

http://127.0.0.1:5984/

Als Antwort des Servers erhält man

‘couchdb’:‘Welcome‘,‘version’:‘1.2.0’

In diesem JSON-String sind ein Willkommensgruß und die installierte Version von CouchDB enthalten.

4.2 Datenbanken

Um eine neue Datenbank auf dem Server zu erstellen muss ein PUT-Command an den Server geschickt werden das wie folgt aufgebaut sein muss

```
PUT http://127.0.0.1:5984/albums
```

‘albums’ ist in diesem Falle der Name der neu anzulegenden Datenbank Konnte die Aktion erfolgreich ausgeführt werden Antwortet der Server mit

```
‘ok’:true
```

4.3 Dokumente

Möchte man ein neues Dokument in einer Datenbank speichern, sieht der Befehl dafür wie folgt aus

```
PUT ‘http://127.0.0.1:5984/albums/6e1295ed6c29495e54cc05947f18c8af
-d ‘‘title’:‘There is Nothing Left to Lose’,‘artist’:‘Foo Fighters’’
```

Wie zu erkennen wird wieder der PUT-Command benötigt. ‘albums’ ist der Datenbankname in dem das Dokument gespeichert werden soll ‘6e1295ed6c29495e54cc05947f18c8af’ ist die eindeutige ID des zu speichernden Dokumentes.

Der Parameter ‘-d’ gibt zu erkennen das das Dokument direkt mit Daten initialisiert werden soll.

Diese sind in JSON-Codiert, und enthalten in dem obigen Beispiel die Informationen ‘title’ mit dem Wert ‘There is Nothing Left to Lose’, außerdem ‘artist’ mit dem Wert ‘Foo Fighters’.

5 Funtkionen

Im Folgenden werden einige wichtige Funktionen von CouchDB vorgestellt.

5.1 Design Dokumente

Diese Art von Dokumenten haben eine spezielle Notation der ID, die mit ‘_design/’ beginnen muss, damit sie als solche identifiziert werden kann.

Sie enthalten JavaScript Code für Anwendungen, die zur Laufzeit vom internen Query Server, der die Funktionen der Design Dokumente ausführt, unter anderem an den externen JavaScript Interpreter weitergeleitet wird.

```
{
  "_id" : "_design/example",
  "views" : {
    "foo" : {
      "map" : "function(doc){ emit(doc._id, doc._rev)}"
    }
  }
}
```

In diesem Beispiel wird ein Design Dokument mit dem Namen ëxampleängelegt. Es enthält eine view namens 'foo' die eine 'map' Funktion implementiert, einfach die ID und die dazugehörige Revision eines jeden Dokumentes ausgibt. Aufgerufen werden könnte dieses Dokument mit:

http://127.0.0.1:5984/basic/_design/example/_view/foo

5.2 Views

Sie werden für den gleichen Sinn und Zweck wie der SELECT Befehl bei SQL verwendet. Es ist möglich mit Views Dokumente zu filtern, sie in einer bestimmten Reihenfolge anzuzeigen, Indizes zu erstellen um Dokumente anhand von Werten oder Struktur zu finden, um Indizes für die Herstellung von Beziehungen zu nutzen, und zu guter Letzt auch zur Ausführung von verschiedenen Berechnungen.

```
function(doc) {
  if(doc.date && doc.title) {
    emit(doc.date, doc.title);
  }
}
```

Diese View gibt einfach, falls vorhanden, das Datum eines jeden Dokumentes und dessen Titel aus.

5.3 Validierung

Es ist möglich Validierungsfunktionen zu implementieren, mit denen Benutzer nur bestimmte Dokumente speichern können. Diese Validierung kann auf drei Arten geschehen: 1. Inhalt 2. Struktur 3. Anfragender User

```
function(newDoc, oldDoc, userCtx) {
  throw({forbidden : 'no way'});
}
```

Diese Validierungsfunktion würde jede Art von speichern einfach Verbieten.

5.4 Show

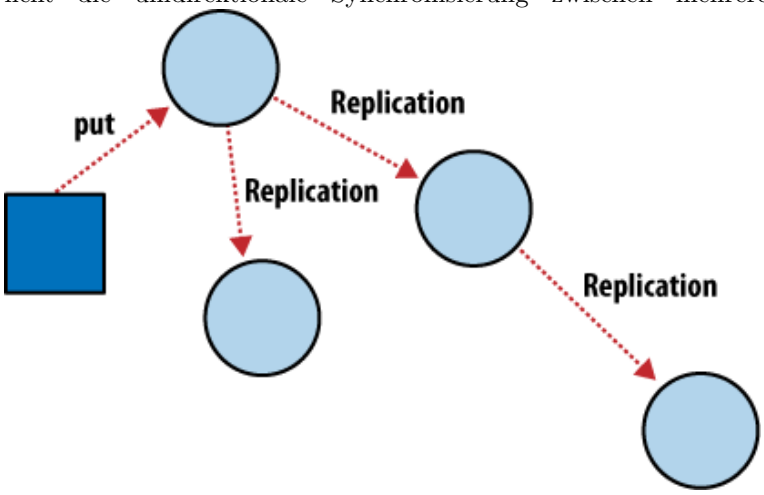
Eine sehr mächtige Möglichkeit JSON Objekte als HTML,XML,PNG usw. rendern zu lassen. Damals war dies Aufgabe eines Application Servers, CouchDB jedoch enthält diese Funktionalität schon von Haus aus.

```
function(doc, req) {
  return '<h1>' + doc.title + '</h1>';
}
```

Diese Show-Funktion würde in HTML gerendert eine Überschrift mit dem Dokumenttitel ergeben.

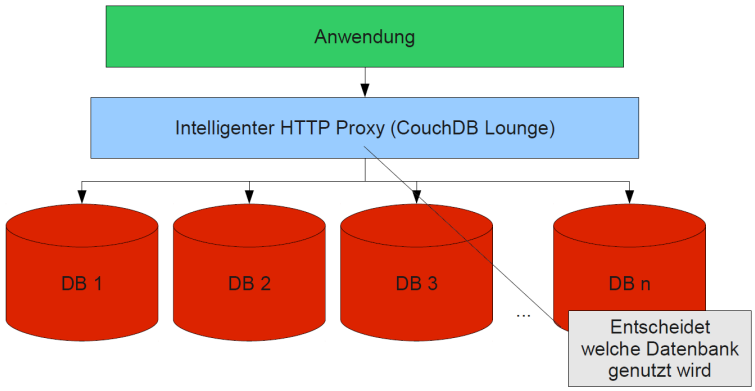
6 Replikation

Die Replikation ist ebenfalls eine Funktionalität die bereits von Haus aus dabei ist. Mit ihr ist es möglich mehrere CouchDB Instanzen uni- oder bidirektional zu synchronisieren. Die folgende Abbildung verdeutlicht die unidirektionale Synchronisierung zwischen mehreren Instanzen.



7 Clustering

Es ist auch möglich Cluster mit CouchDB zu betreiben. Hierfür wird spezieller, intelligenter HTTP Proxy-Server namens ‘CouchDB Louge’ benötigt, welcher eine Schicht zwischen den eigentlichen Datenbankinstanzen und der Anwendung dastellt. Dieser Entscheidet je nach Auslastung der Datenbankserver, welche Instanz die Anfrage bearbeiten soll.



8 Schlusswort

Wie zu sehen war sind Document Stores bereits eine ältere Technologie, die erst jetzt durch das heutige Internet gefragt sind. CouchDB bietet als Einstieg eine sehr mächtige Plattform, die viele Dienste wie Replikation schon von Haus aus mitbringen. Die Möglichkeit des Clusterings macht CouchDB auch für große verteilte Anwendungen brauchbar. Durch die gegebene Funktionalitäten ist es sogar machbar komplette Stand-Alone Anwendungen in CouchDB zu programmieren, die auch das Offline arbeiten ermöglichen.

Literatur

1. Wassilios Kazakos, Andreas Schmidt, Peter Tomczyk, D. Gündisch, T. Marz, Guido Moerkotte, A. Valikov. Springer, ISBN 978-3-540-41956-3.
2. J. Chris Anderson, Jan Lehnardt and Noah Slater. O Reilly