Datenmodellierung in der Anwendungsentwicklung mit NoSQL-Datenbanken

Stefanie Scherzinger
OTH Regensburg
stefanie.scherzinger@oth-regensburg.de

Abstract: NoSQL-Datenbanken sind gerade in der Webentwicklung zunehmend beliebt. Oft sind es die großen Datenmengen, die es zu verwalten gilt, mitunter sind diese Systeme aber auch wegen ihrer Schema-Flexibilität für agile Entwicklungsteams interessant. Indem viele NoSQL-Datenbanken keine Unterstützung für die Definition, Einhaltung und Wartung eines globalen Schemas bieten, verlagern sich klassische Aufgaben des Datenbankmanagementsystems in die Anwendungssoftware. Dieser Vortrag gibt einen Überblick über konkrete Herausforderungen, die sich in der Praxis beim Entwurf eines Datenmodells für *Key-Value*- und Dokumenten-Datenbanken ergeben. Dazu zählen eine Modellierung, die atomare Updates ermöglicht, das Vermeiden von *Hot-Spot*-Datenobjekten, wie sie durch hochfrequente, parallele Schreibzugriffe gegen dasselbe Objekt verursacht werden, sowie Strategien zum Umgang mit kontinuierlicher Schema-Evolution. Der Vortrag zeigt auf, dass gerade die Datenbank-Community mit ihrem Erfahrungsschatz im Schema-Management und ihrem breiten Fundus an formalen Methoden hier einen wertvollen Beitrag leisten kann.

Während der letzten Jahrzehnte wurde die Architektur von Desktop- und Webanwendungen nahezu ausnahmslos von relationaler Datenbanktechnologie dominiert. Es gibt jedoch Sparten, in denen NoSQL-Datenbanksysteme eine attraktive Alternative darstellen. Unter dem Begriff "NoSQL" tummelt sich mittlerweile ein bunter Zoo von Systemen. So sehr sich die einzelnen Produkte voneinander unterscheiden, so zeichnen sie sich typischerweise in den "3 *V*s" aus: *Volume*, die Fähigkeit, auf große Datenmengen zu skalieren, *Velocity*, die Fähigkeit, laufend neu hinzukommende Datensätze zu verarbeiten, und *Variety*, die Fähigkeit, mit stark heterogenen Datenstrukturen zu arbeiten.

Unabhängig davon, welches "*V*" letztlich bei der Entscheidung für eine NoSQL-Datenbank maßgeblich ist, stellen sich in der Anwendungsentwicklung gegen diese Systeme Herausforderungen, auf die das grundständige Informatikstudium oft nur ungenügend vorbereitet [ST14b]. Dieser Vortrag gibt einen Überblick über die gängigen *Pain Points* in der Datenmodellierung bei der Anwendungsentwicklung gegen NoSQL-Datenbanksysteme, wie etwa *Key-Value*- und Dokumenten-Datenbanken:

- Transaktionen mit ACID-Verhalten sind in Systemen, die darauf ausgelegt sind, auf zehntausenden physischen Knoten verteilt zu laufen, nicht uneingeschränkt möglich. Naiv entwickelte Anwendungen zeigen daher oft *Eventual Consistency*-Effekte, welche zu erheblichen Irritationen bei den Anwendern führen können. So sind eben vollzogene Änderungen an Daten, z.B. nach Ausfüllen eines Formulars, womöglich nicht unmittelbar sichtbar.

Um Änderungen an mehreren Objekten atomar durchführen zu können, sind Entwickler mitunter gezwungen, an sich logisch eigenständige Objekte gemeinsam auf ein persistentes Objekt abzubilden [SF12]. Dies läuft entgegen der in der Datenbankvorlesung vermittelten Strategie, die Daten möglichst zu normalisieren. Manche Systeme erlauben es auch, logisch eigenständige Objekte 1:1 auf persistente Objekte abzubilden und diese dann zu gruppieren. Die Objekte in solchen *Entity Gruppen* werden physisch im verteilten System kollokiert, was atomare Änderungen und strikt konsistenten Zugriff innerhalb der Gruppe ermöglicht (vgl. [San12, BBC$^+$11]). Somit ist der Gestaltungsspielraum bei der Datenmodellierung systemspezifisch.

- Das häufige Ändern einzelner sog. *Hot-Spot*-Objekte stellt für hochverteilte Systeme typischerweise ein Problem dar. Naiv entwickelte Anwendungen leiden daher ausgerechnet während Lastspitzen unter Laufzeitfehlern und fehlgeschlagenen Schreibzugriffen. Hier spielt die Datenmodellierung eine tragende Rolle, wenn solche Effekte vermieden werden sollen [SDAIDF13, ST14a].
- Indem viele NoSQL-Systeme kein festes Schema fordern, erlauben sie eine flexiblere Handhabung heterogener Daten bei Schema-Evolution. Langfristig erschwert eine hohe Heterogenität in den persistenten Objekten jedoch die Wartbarkeit der Anwendung. In der Praxis behilft man sich mit Migrations-Skripten (d.h. "Custom Code"), um die Datenbasis *eager* zu migrieren. Alternativ werden Object-Mapper eingesetzt, die einzelne persistente Objekte beim Laden in die Anwendung transformieren. Dadurch wird die Datenbasis inkrementell bzw. *lazy* migriert. Ob *eager* oder *lazy*, diese Ansätze sind bisher bloße Notbehelfe, fehlt es doch an einem theoretisch fundierten Rahmenwerk, das ein systematisches Testen und eine saubere Migrationssemantik überhaupt erst ermöglicht [KSS14, SKS13].

Diese *NoSQL Pain Points* zeigen insbesondere die enge Verflechtung zwischen logischem und physischem Datenbankentwurf auf, die in den meisten NoSQL-Datenbanken gegeben ist. So führt ein ungünstig gewähltes Datenmodell dazu, dass ACID-Eigenschaften bei Änderungen nicht gewährleistet werden können, oder Anwendungen unter hoher Schreiblast nicht skalieren. Es gilt zudem, die Entwickler von der bestehenden Notwendigkeit zu befreien, sich mit den Eigenheiten der jeweiligen NoSQL-Datenbank eingehend beschäftigen zu müssen. Konkret ist ein neues Ökosystem an Werkzeugen zu entwickeln, die um die proprietären Funktionalitäten dieser Systeme wissen und die Entwickler aktiv darin unterstützen, ein geeignetes Datenmodell zu entwerfen.

Dieser Vortrag zeigt auf, welchen wertvollen Beitrag die Datenbank-Community bei der Entwicklung solcher Werkzeuge für die Software-Technik leisten kann.

Biographie: Stefanie Scherzinger ist seit 2012 Professorin an der OTH Regensburg. Sie promovierte 2008 an der *Universität des Saarlandes*, mit Zwischenstationen an der *TU Wien* und der *Cornell University*. Danach erwarb sie Industrieerfahrung als Softwareentwicklerin bei IBM und Google. Aus dieser Zeit stammen mehrere erfolgreiche Patentanmeldungen, sowie einschlägige Erfahrungen in der professionellen Anwendungsentwicklung mit NoSQL-Datenbanken. Ihre aktuelle Forschung widmet sich Problemen im Datenmanagement, die aus ihrer Berufspraxis motiviert sind.

Literatur

[BBC+11] Jason Baker, Chris Bond, James C. Corbett, JJ Furman et al. Megastore: Providing Scalable, Highly Available Storage for Interactive Services. In *Proc. CIDR*, 2011.

[KSS14] Meike Klettke, Stefanie Scherzinger und Uta Störl. Datenbanken ohne Schema? *Datenbank-Spektrum*, 14(2), 2014.

[San12] Dan Sanderson. *Programming Google App Engine*. O'Reilly Media, Inc., 2. Auflage, 2012.

[SDAIDF13] Stefanie Scherzinger, Eduardo Cunha De Almeida, Felipe Ickert und Marcos Didonet Del Fabro. On the Necessity of Model Checking NoSQL Database Schemas when Building SaaS Applications. In *Proc. TTC 2013*, 2013.

[SF12] Pramod J. Sadalage und Martin Fowler. *NoSQL Distilled: A Brief Guide to the Emerging World of Polyglot Persistence*. Addison-Wesley Professional, 1. Auflage, 2012.

[SKS13] Stefanie Scherzinger, Meike Klettke und Uta Störl. Managing Schema Evolution in NoSQL Data Stores. In *Proc. DBPL*, 2013.

[ST14a] Stefanie Scherzinger und Andreas Thor. AutoShard - Declaratively Managing Hot Spot Data Objects in NoSQL Data Stores. In *Proc. WebDB*, 2014.

[ST14b] Stefanie Scherzinger und Andreas Thor. Cloud Technologien in der Hochschullehre. *Datenbank-Spektrum*, 14(2), 2014.