MarkLogic®

DAS RELATIONALE MODELL WEITERGEDACHT

MARKLOGIC WHITEPAPER • SEPTEMBER 2015

Mit der Verarbeitung riesiger Mengen an vielfältigen und sich ständig wandelnden Daten in den unterschiedlichsten Formaten, mithilfe der relationalen Datenbanken auf die man seit 30 Jahren baut, sind Unternehmen heute zunehmend überfordert. Aus diesem Grund denken führende Unternehmen das relationale Modell weiter und gehen zu einer neuen Generation von Datenbanken über: MarkLogic.

EINSCHÄTZUNG

Braucht Ihr Unternehmen Veränderung, oder kommen Sie Ihrem Eindruck nach mit Ihren Daten gut zurecht? Die Antworten auf die nachstehenden Fragen ermöglichen eine grundlegende Einschätzung zu diesem Thema. Je häufiger die Antwort „Ja“ lautet, desto wahrscheinlicher ist es, dass Ihre derzeitigen Datenbanken den Anforderungen Ihres Unternehmens nicht mehr gerecht werden.

		JA	NEIN
FRAGEN ZUM UNTERNEHMEN	1. Gibt es unternehmenswichtige Daten, die nicht in einer Datenbank gespeichert sind?	✓	✗
	2. Bestehen mehrere verschiedene Datenbanken, die im Wesentlichen die gleichen Daten enthalten?	✓	✗
	3. Sind mehrere Datenquellen vorhanden, die nicht zentral durch die IT-Abteilung verwaltet werden?	✓	✗
	4. Gibt es große IT-Projekte, die das Budget gesprengt haben oder gescheitert sind?	✓	✗
	5. Sind manche Datenbankschemata so kompliziert, dass nur eine Handvoll Experten kompetent Auskunft darüber geben können?	✓	✗
FACHLICHE FRAGEN	6. Kommt es vor, dass die Datenmodellierung die Anwendungsentwicklung aufhält oder behindert?	✓	✗
	7. Gibt es relationale Tabellen, in denen Spaltennamen geändert oder umschrieben wurden, „damit es halt irgendwie funktioniert“?	✓	✗
	8. Gibt es monatlich mehrere Datenbankschema-Änderungen, und sind manche davon erfolglos?	✓	✗
	9. Verschlingt die Frage der Skalierung viel Zeit und Ressourcen?	✓	✗
	10. Kommt es zu Leistungsproblemen und Fehlern, die möglicherweise auf komplizierte Middleware zurückzuführen sind?	✓	✗

Inhalt

EINLEITUNG

Die vergangenen Jahrzehnte waren von einer rasanten technischen Entwicklung geprägt, und unsere Art zu wirtschaften hat sich dadurch in jeder Hinsicht verändert. Heute werden mehr Daten als je zuvor erfasst. Unternehmen haben immer neue Ideen für noch größere und intelligentere Anwendungen. Man sollte also meinen, dass sich auch die Datenbanken entsprechend gewandelt haben – schließlich speichern die Unternehmen in ihnen ihren wichtigsten Besitz, nämlich ihre *Daten*. Dies trifft aber weitgehend nicht zu.

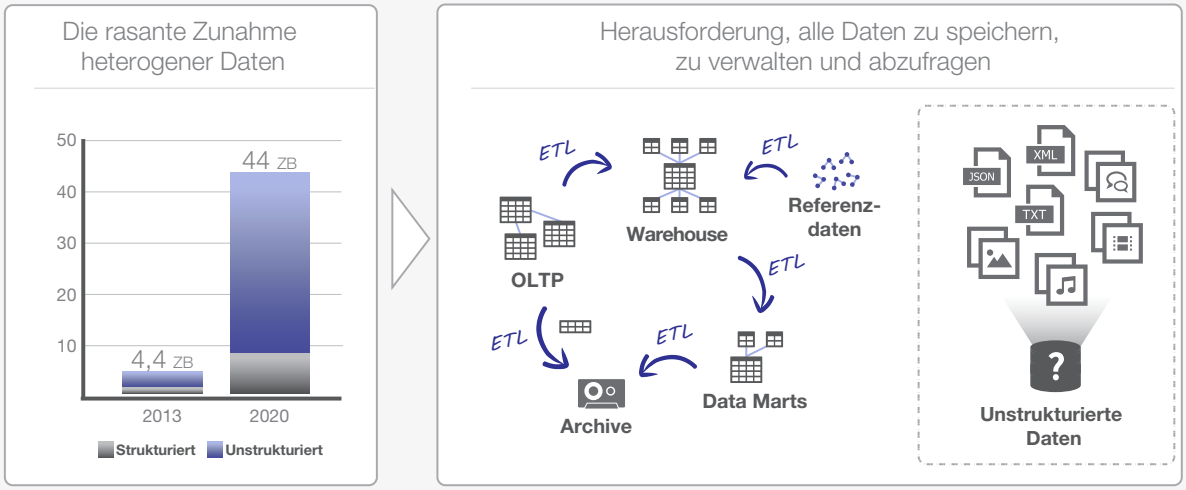
Die vorherrschende Technologie für die Speicherung und Verwaltung von Daten – die relationale Datenbank – sieht noch immer ungefähr so aus wie zum Zeitpunkt ihrer Einführung vor mehr als dreißig Jahren. Damals galten Daten als wenig umfangreich, sauber, gut strukturiert und statisch, denn nur in dieser Form konnte man sie speichern. In Wirklichkeit sind Daten aber ganz anders. Heutzutage sehen sich Unternehmen einer Welt mit umfangreichen, schnelllebigen, vielfältigen und volatilen Daten gegenüber. Sie arbeiten nicht mehr bloß mit einer Handvoll an Systemen, sondern mit Hunderten davon sowie mit Petabytes an Daten.

Die neue Welt des „Big Data“ bringt spannende Chancen mit sich, doch allzu oft wird sie als eine von vielen Herausforderungen abgetan, die nur die IT-Abteilung zu interessieren haben. IT-Abteilungen sind heute größtenteils damit beschäftigt, überhaupt den Kopf über Wasser zu halten, haben sie es doch mit einem komplexen Netz aus Datensilos und häufigen ETL-Prozessen (Extrahieren, Transformieren, Laden) zur Verschiebung von Daten zu tun. In diesem Klima basteln sich dann Mitarbeiter und einzelne Abteilungen ihre eigenen Lösungen. Inoffizielle „Schatten“-Prozesse machen sich breit, und es kommt zu Sicherheitslücken. In allen Branchen sind heute hohe Kosten und zeitintensive Projektabläufe zur Norm geworden. Die Unternehmen sind mit den ständigen Wartungszyklen zu sehr gefordert, um sich darauf konzentrieren zu können, all ihre Daten möglichst gut zu verwerten. Bis zu einem gewissen Grad sind all diese Herausforderungen darauf zurückzuführen, dass man mit relationalen Datenbanken Probleme zu lösen versucht, die dafür nie gedacht waren.

Heutzutage können Unternehmen nicht mehr blind auf das uniforme und unflexible relationale Modell bauen. Angesichts des herrschenden Veränderungsdrucks gehen sie zu neuen Arten von Datenbanken über. MarkLogic ist bei diesem Generationenwechsel an vorderster Front und bietet eine Datenbank, die für die Daten von heute besser gewappnet ist. Sie überzeugt mit einem flexibleren Modell für das Speichern, Verwalten und Durchsuchen riesiger und vielfältiger Datenmengen, ohne dabei Abstriche bei den Unternehmensfunktionen zu machen, die ein moderner Betrieb verlangt. Genau diese einzigartige Kombination hat es führenden Unternehmen ermöglicht, das relationale Modell weiterzudenken und so ihre Daten besser zu verwerten als jemals zuvor.

VERÄNDERUNG – WOZU?

Mit der Verarbeitung ihrer vielfältigen und sich ständig wandelnden Daten in den unterschiedlichsten Formaten sind Unternehmen heute zunehmend überfordert. Nur durch neue Ansätze können sie Herr dieses Problems werden und ihr Risiko minimieren, schneller intelligentere Anwendungen entwickeln und ihre Daten betriebswirtschaftlich besser verwerten.



> Die drei ‚Vs' von ‚Big Data' laufen auf eine zusammenfassende Erkenntnis hinaus: *Die Daten von heute sind umfangreich, schnelllebig, komplex und veränderlich*."

DAS ZEITALTER VON „BIG DATA"

Aufgrund der Grenzen der damaligen Technologie sahen Daten früher immer gleich aus. Sie gelangten langsam und geordnet in Rechenzentren und waren sauber in präkonfigurierten Zeilen und Spalten in miteinander verbundenen Tabellen organisiert. Veränderungen gingen langsam vor sich, sowohl in betriebswirtschaftlicher als auch in informatischer Hinsicht, und dies war auch kein Problem. Doch das waren die 80er-Jahre, und heute stellt sich die Situation ganz anders dar.

Die moderne Welt des „Big Data" kennt drei Prinzipien, auf Englisch gerne als die drei „Vs" bezeichnet: Volume, Velocity, Variety – Menge, Geschwindigkeit, Vielfalt. Zwei weitere „Vs" gewinnen ebenfalls zunehmend an Bedeutung: Veracity und Variability – Echtheit und Veränderlichkeit. All diese „Vs" laufen auf eine zusammenfassende Erkenntnis hinaus: *Die Daten von heute sind umfangreich, schnelllebig, komplex und veränderlich*.

VOLUME – MENGE

Die digitale Welt wächst jährlich um 40 %. Der weltweite Datenbestand dürfte von 2013 bis 2020 von 4,4 Zettabyte auf 44 Zettabyte anwachsen (ein Zettabyte entspricht einer Billion Gigabyte).[1] Papier hat als Speichermedium ausgedient, Datenbanken haben diese Rolle übernommen – und das heißt: Sie müssen *alles* speichern können. Unternehmen müssen heute mit mehr Daten in vielfältigeren Formen auf mehr verschiedenen Systemen umgehen als je zuvor und sollen dies auch noch effizient, sicher und kostengünstig bewerkstelligen. Die Datenspeicherkosten sinken weiterhin, und Verbraucher wie Behörden erwarten somit von den Unternehmen, dass sie alles speichern.

VELOCITY – GESCHWINDIGKEIT

Die moderne Welt ist in jeder Hinsicht schnelllebiger als früher. Daten entstehen schneller und verändern sich schneller. Auch was man aus den Daten erschließen will, ändert sich schneller angesichts ständig neuer Geschäftsanforderungen. Unternehmen müssen rasch wandelnde Marktdynamiken, Änderungen in der Unternehmungsführung, On-Demand-Dienste oder Übernahmen und Abspaltungen bewältigen. Entscheidungen werden heute innerhalb von Minuten und nicht Tagen getroffen, und die Daten zur Untermauerung dieser Entscheidungen müssen mit weniger Latenz und höherer Effizienz im richtigen Format bereitgestellt werden. Ob es um Daten für Sportveranstaltungen oder die Betrugsbekämpfung in einer Bank geht: *Echtzeitdaten* sind nicht mehr optional, sondern ein absolutes Muss. Auch der Zeitrahmen für die Entwicklung von Anwendungen ist knapper geworden. Man rechnet in Wochen, nicht mehr in Jahren. Die so entwickelten Anwendungen müssen dann Massen an Benutzern bedienen – Benutzer, die nicht lange warten wollen, weniger produktgebunden sind und einen höheren Grad an Personalisierung wünschen.

VARIETY – VIELFALT

Vielfalt ist unter den großen „Vs" eine der größten Herausforderungen. Die Daten von heute sind deutlich vielfältiger und heterogener als früher – nur etwa 20 % davon sind strukturiert (z. B. transaktionale oder tabellarische Daten), 80 % hingegen unstrukturiert (z. B. Dokumente, Textnachrichten, E-Mails, Bilder oder Videos).[2] Die neu verfügbaren unstrukturierten Datenquellen sind zweifellos problematisch. In einer Studie gaben 64 Prozent aller befragten Unternehmen an, dass der Hauptgrund für die Erwägung eines neuen „Big Data"-Ansatzes die Bewältigung der vielen unterschiedlichen neuen Echtzeitdatenquellen sei.[3] Doch noch problematischer ist womöglich die Vielfalt an strukturierten Daten. Unternehmen haben Mühe, mit den vielen verschiedenen Formen, Größen und Arten an schnell wachsenden und veränderlichen strukturierten Daten fertigzuwerden. Neue Anwendungen oder auch Fusionen und Übernahmen sowie die Wiederverwendung von Daten für einen anderen Zweck führen oft zu großen Unterschieden zwischen strukturierten Daten.

VERACITY – ECHTHEIT

Echtheit hat mir der Zuverlässigkeit bzw. Integrität der Daten zu tun. Daten sind Goldwert, und Unternehmen scheuen keine Mühen, um sicherzustellen, dass ihre Daten richtig und in keiner Weise beschädigt sind. Damit wird es immer wichtiger, die Herkunft bzw. den Lebenszyklus der Daten nachzuverfolgen: Wann wurden die Daten erfasst? Woher kommen sie? Wie und von wem wurden sie bearbeitet? Wie lange sollen sie gespeichert bleiben? Wie relevant sind sie für

1 IDC. *Digital Universe*. April 2014 <http://www.emc.com/leadership/digital-universe/2014iview/executive-summary.htm>

2 Khan et al. *Big Data: Survey, Technologies, Opportunities, and Challenges*. Scientific World Journal, 2014 <http://www.ncbi.nlm.nih.gov/pmc/articles/PMC4127205/#B53>

3 New Vantage Partners. *Big Data Executive Survey: Themes & Trends*. 2012 <http://newvantage.com/wp-content/uploads/2012/12/NVP-Big-Data-Survey-Themes-Trends.pdf>

> „Datensilos sind der Befragung nach das größte Erfolgshindernis im Big-Data-Bereich.“

einen bestimmten Zweck? Zusätzlich sind Unternehmen an strenge Richtlinien für den Umgang mit Daten gebunden, um den Zugriff der Benutzer auf die Daten auf granularer Ebene zu schützen. Diese Fragen werden schon allein aus rechtlichen Gründen immer wichtiger, ganz zu schweigen vom Potenzial für eine ausgefeiltere Datenanalytik.

VARIABILITY – VERÄNDERLICHKEIT

Unter Veränderlichkeit verstehen wir die Mehrdeutigkeit von Daten je nach Kontext. Brian Hopkins, Principal Analyst bei Forrester, hat sich mit diesem Thema befasst und Veränderlichkeit als „die Veränderlichkeit der Bedeutung (also Mehrdeutigkeit) in natürlichen Sprachen und die Rolle von Big Data bei der Lösung dieses Problems“ definiert.[4] Ein Beispiel wäre das Wort „Bank“ – reden wir von einem Finanzinstitut oder einer Sitzgelegenheit? Doch hierbei geht es nicht nur um Sprache. Es bestehen auch Unterschiede zwischen den Benennungen grundlegender Entitäten durch Benutzer und Datenmodellierer. Der amerikanische Bundesstaat „North Carolina“ wird zum Beispiel manchmal zu „N Carolina“ oder gar zu „NC“ abgekürzt. Wie kann eine Datenbank erkennen, dass damit das Gleiche gemeint ist, oder überhaupt die Idee eines „Bundesstaats“ verstehen? Menschen fällt es leicht, solche Dinge im Zusammenhang richtig zu interpretieren, doch den Datenbanken bereiten diese semantischen Herausforderungen Probleme. Je größer die Vielfalt an Daten, desto größer die Vielfalt an Benennungen für Menschen, Orte und Dinge – wodurch sich das Problem weiter verstärkt.

VERSUNKEN IM MORAST DER KOMPLEXITÄT

Die heutige Welt des „Big Data“ sollte eigentlich als Riesenchance begriffen werden, doch allzu oft wird sie nur als eine Herausforderung unter vielen angesehen. Heutzutage verbringen die IT-Abteilungen die meiste Zeit damit, angesichts dieser Komplexität einfach irgendwie den eigenen Kopf über Wasser zu halten. Wenn sie aber nicht stattdessen das Ruder ergreifen, drohen sie das ganze Schiff zum Sinken zu bringen.

DATENSILOS UND ETL

Datensilos sind der Befragung nach das größte Erfolgshindernis im Big-Data-Bereich.[5] Dies wird deutlich, wenn man sich die üblichen komplexen Unternehmensdiagramme ansieht: Da sind inkompatible Altsysteme mit anderen ebenso inkompatiblen Altsystemen verwoben, und das Ergebnis ist eine hochkomplexe und fragile Architektur, in der die Daten weder austauschbar noch verwendbar sind. In den meisten Unternehmen sind womöglich nur noch einige wenige Experten da, die die Zusammenhänge zwischen dieser Datenarchitektur und den komplizierten Geschäftsregeln begreifen. Umso weniger überrascht es, dass bei den meisten Business-Intelligence-Initiativen die meiste Zeit dafür aufgewandt wird, die einzelnen Datenquellen erst einmal auszumachen und zu analysieren.[6]

Datensilos sind nie beabsichtigt, sie entstehen im Zuge kurzsichtiger Lösungen. Die meisten Datenbanken sind nur für eine bestimmte Anwendung oder eine bestimmte Art von Daten gedacht. Um die Daten aus diesen Datenbanken herauszulösen und dann in einer anderen Datenbank für einen anderen Zweck einzusetzen, ist ein ETL-Prozess (Extrahieren, Transformieren, Laden) nötig, damit die Daten an das Schema der neuen Zieldatenbank angepasst werden können. ETL-Prozesse sind in den meisten Unternehmen laufend auf der Tagesordnung, und jedes Mal entsteht so ein neues Datensilo.

Mit zunehmender Anzahl an Datensilos wird es immer schwieriger, diese zu warten und miteinander zu verbinden. Letztendlich beginnen dann die Entwickler damit, Ad-hoc-Wartungscode zu „basteln“, um die verschiedenen Anwendungen miteinander zu verbinden. Doch damit weichen sie der wahren Ursache des Problems aus. Die Komplexität wächst nur noch weiter an, und am Ende funktioniert entweder etwas gar nicht mehr, oder aber neue Projekte werden so stark vernachlässigt, dass an Fortschritt nicht mehr zu denken ist.

4 Hopkins, Brian. „Blogging From the IBM Big Data Symposium – Big Is More Than Just Big“, 2011 <http://blogs.forrester.com/brian_hopkins/11-05-13-blogging_from_the_ibm_big_data_symposium_big_is_more_than_just_big>

5 Oracle. *IT Assessment Complexity Survey*. 2015 <http://www.oracle.com/us/corporate/features/it-complexity-assessment-survey-2281110.pdf>

6 Boris Evelson. *Boost Your Business Insights By Converging Big Data And BI*. Forrester, 25. März 2015 <https://www.forrester.com/Boost+Your+Business+Insights+By+Converging+Big+Data+And+BI/fulltext/-/E-RES115633>

> Die Anbieter von relationalen Datenbanken verkaufen den Benutzern nach wie vor ein Produkt aus den 90ern, basierend auf Codes aus den 80ern, konzipiert für die Probleme der 70er, beruhend auf einer Grundidee aus den 60ern.“

„SCHATTEN“-IT-PROZESSE UND SICHERHEITSLÜCKEN

CIOs verlieren zunehmend die Kontrolle über die Unternehmensdaten. Mitarbeiter und Abteilungen lösen ihre eigenen Probleme selbstständig und verwenden dazu Software, die nicht von einer zentralen IT-Abteilung kontrolliert oder verwaltet wird. Die meisten CIOs würden wahrscheinlich davon ausgehen, dass sie ein paar Dutzend „Schatten“-IT-Lösungen im Einsatz haben. In Wahrheit sind es wohl in den meisten Fällen Hunderte. In einer Studie stellte sich heraus, dass Unternehmen sage und schreibe 923 verschiedene Cloud-Dienste verwenden, von denen nur 9,3 % die jeweils unternehmenseigenen Sicherheitsanforderungen erfüllen.[7] Diese Entwicklung ist das unmittelbare Ergebnis einer gefühlten Untätigkeit angesichts der geschäftlichen Erfordernisse und setzt das Unternehmen als Ganzes unermesslichen Risiken und Effizienzverlusten aus.

Gleichzeitig werden die Kosten etwaiger Sicherheitslücken immer höher und Cyber-Kriminelle in ihren Angriffen immer durchtriebener. Der Ruf eines Unternehmens kann schon bei einem einzigen Datenleck schweren Schaden nehmen, und auch die Kosten sind nicht zu unterschätzen. Laut einer Studie kann eine einzige Cyber-Sicherheitsverletzung ein Unternehmen durchschnittlich 5,4 Mio. $ (rund 4,8 Mio. €) bzw. 188 $ (rund 167 €) pro Datensatz kosten.[8] Leider ist aber der Schutz der Daten schwieriger denn je, da die wild wuchernden Datensilos mehr Angriffspunkte, Sicherheitslücken und Datenlecks verursachen.

HOHE KOSTEN, GESCHEITERTE PROJEKTE UND INNOVATIONSUNFÄHIGKEIT

Traurig, aber wahr: Die meisten IT-Projekte werden nicht termingerecht und im veranschlagten Kostenrahmen fertiggestellt. Hohe Kosten und gescheiterte Projekte sind die Norm. Die Hälfte aller mit mehr als 15 Mio. $ (rund 13,5 Mio. €) veranschlagten IT-Projekte sprengen das Budget um 45 % und den Zeitrahmen um 7 %. Dabei liefern sie 56 % weniger Funktionen als vorgesehen. Als wäre das noch nicht genug, laufen 17 % aller IT-Projekte so schlecht, dass sie für das Unternehmen regelrecht existenzgefährdend sind.[9] Und das trotz langwieriger Planung, enormer Ressourcen und großer Teams an hervorragenden Mitarbeitern!

Während diese Teams nun an Projekten arbeiten, die mehr kosten und weniger bringen als geplant, fehlt ihnen die Zeit für innovative Projekte, die ausschlaggebend für den Unternehmenserfolg sind. Wie kann ein Unternehmen einen Mehrwert aus „Big Data“ schöpfen, wenn es keinerlei Ressourcen dafür verfügbar hat? Heutzutage fließen 95 % aller Datenbankausgaben in relationale Datenbanken. Doch nur 20 % der Unternehmensdaten sind darin gespeichert.[10] Damit bleiben nur 5 % für die Verwaltung der restlichen 80 % der Unternehmensdaten übrig. Ohne Veränderung werden CIOs und IT-Mitarbeiter mit zahllosen Altsystemen und -lasten beschäftigt bleiben und im Wettbewerb von jenen geschlagen werden, die mehr Ressourcen für Innovation und geschäftliche Erneuerung einsetzen.

DIE UNTAUGLICHKEIT RELATIONALER DATENBANKEN

Viele der Probleme, die wir heute mit „Big Data“ und der daraus resultierenden Komplexität haben, lassen sich auf relationale Datenbanken zurückführen. Relationale Datenbanken sind natürlich nicht grundlegend falsch konzipiert – sie waren nur nie dafür gedacht, mit den Daten von heute umzugehen. Deswegen sagte ja auch der frühere CIO der US-amerikanischen Regierung, Vivek Kundra, schon 2009: „Dieses Konzept von Daten in einer strukturierten, relationalen Datenbank ist Vergangenheit.“

Die relationale Datenbank wurde Ende der 70er-Jahre erfunden. Sie folgte auf die hierarchischen Mainframe-Systeme und wurde Anfang der 90er-Jahre zur ersten Wahl. Relationale Datenbanken wurden den Anforderungen der Computer-Frühzeit sehr gut gerecht. Sie ermöglichten die Entkopplung von Anwendungen und Daten, es war weniger eigener Code nötig, und die Benutzer hatten dank SQL als allgemein anerkannte Datenbanksprache mehr Kontrolle über die Datenabfragen.

7 Skyhigh. *Cloud Adoption and Risk Report – Q1 2015*. Juni 2015 <http://info.skyhighnetworks.com/rs/skyhighnetworks/images/WP%20CARR%20Q1%202015.pdf>

8 Ponemon Institute. *2013 Cost of Data Breach Study: Global Analysis*. Symantec. 2013 <https://www4.symantec.com/mktginfo/whitepaper/053013_GL_NA_WP_Ponemon-2013-Cost-of-a-Data-Breach-Report_daiNA_cta72382.pdf>

9 McKinsey & Company. „Delivering large-scale IT projects on time, on budget, and on value“, Oktober 2012 <http://www.mckinsey.com/insights/business_technology/delivering_large-scale_it_projects_on_time_on_budget_and_on_value>

10 Carl Olofson. *Worldwide Database Management Systems 2014–2018 Forecast and 2013 Vendor Shares*. IDC, Juni 2014 <http://www.idc.com/getdoc.jsp?containerId=248952>

> „Dieses Konzept von Daten in einer strukturierten, relationalen Datenbank ist Vergangenheit.“
>
> Vivek Kundra, Federal CIO, 21. Juli 2009, Open Government and Innovations Conference

In ihrer nun schon fast 40-jährigen Geschichte haben sich die relationalen Datenbanken laufend verbessert, und um die Produkte der jeweiligen Anbieter haben sich entsprechende Ökosysteme gebildet. Das grundlegende Modell für den Umgang mit Daten blieb jedoch unverändert. Fakt ist: Die Anbieter von relationalen Datenbanken verkaufen den Benutzern nach wie vor ein Produkt aus den 90ern, basierend auf Code aus den 80ern, konzipiert für die Probleme der 70er, beruhend auf einer Grundidee aus den 60ern.[11] Heute aber können relationale Datenbanken die Anforderungen der Unternehmen nicht mehr erfüllen, wie im Folgenden näher ausgeführt.

NICHT DAFÜR KONZIPIERT, MIT VERÄNDERUNG UMZUGEHEN

In relationalen Datenbanken sind die Daten in Zeilen und Spalten organisiert, ganz ähnlich wie in Arbeitsblättern in Microsoft Excel. Jede Zeile entspricht einem eindeutigen Eintrag, und jede Spalte beschreibt eindeutige Attribute. Eine der Spalten wird als Primärschlüssel gewählt, mit dem jede Reihe in der Tabelle eindeutig identifiziert werden kann.

Wenn Sie beispielsweise eine relationale Datenbank für Kunden und die von diesen bestellten Produkte modelliert haben, könnten Sie zuerst eine Tabelle mit der Bezeichnung „Kunden“ und einer Spalte „Kunden-ID“ erstellen. Dann würden Sie weitere Spalten für die jeweiligen Eigenschaften der Kunden erstellen, z. B. „Vorname“, „Nachname“ und „Adresse“, und den für die entsprechende Spalte geltenden Datentyp definieren.[12] Nun verknüpfen Sie die „Kunden-ID“ mit einer anderen Tabelle namens „Bestellungen“, in der Informationen über die Käufe der verschiedenen Kunden gespeichert werden sollen. Jede Zeile in der Tabelle „Bestellungen“ verfügt über ihre eigene eindeutige ID sowie zusätzlich eine Referenz zum Primärschlüssel der Tabelle „Kunden“ (sog. „Fremdschlüssel“).

Nach dem gleichen Prinzip erstellen Sie weitere Tabellen und stellen dabei sicher, dass Ihr Design allen Vorgaben hinsichtlich der Entitäts- und Referenz-Integrität gerecht wird und alles ordnungsgemäß „normalisiert“ ist. Zeilen dürfen sich also nicht wiederholen, alle Zeilen müssen von ihrem Primärschlüssel abhängig sein, und Informationen dürfen nicht doppelt in verschiedenen Tabellen vorkommen. Diese Vorgaben stellen die Einheitlichkeit der Daten sicher und ermöglichen rasche Abfragen – die Trümpfe des relationale Modells. Bei der hier beschriebenen Konzeption des Datenmodells oder -schemas entscheidet ein speziell dafür abgestelltes Team, welche Tabellen benötigt werden und wie die Spalten benannt werden sollen. Dies ist ein wichtiger Prozess, dessen Endergebnis oft stolz in einem großen Beziehungsdiagramm (Entity-Relationship-Diagramm, ERD) dargestellt wird, das dann ausgedruckt und an prominenter Stelle aufgehängt wird.

Dieser Ansatz birgt zweierlei Probleme. Erstens kann der Prozess je nach Größe der Datenbank Monate, wenn nicht gar Jahre, dauern. Relationale Schemata sind komplex, und der Großteil der Modellierung muss schon *im Voraus* erfolgen, bevor noch irgendwelche Daten geladen wurden oder die zugehörige Anwendung entwickelt ist. Zweitens sind etwaige Änderungen, die nach der Entwicklung von auf der Datenbank basierenden Anwendungen erforderlich werden, mit erheblichem Zeit- und Arbeitsaufwand verbunden. Es kann sich hier noch einmal um Monate oder Jahre handeln. Das relationale Modell gleicht dem empfindlichen und komplexen Ökosystem des Regenwaldes: Eine kleine Änderung kann eine schädliche Kettenreaktion mit Auswirkungen auf die gesamte Datenbank und den Anwendungsstack auslösen. Selbst einfache Änderungen wie das Hinzufügen oder Entfernen einer Tabellenspalte kann sechsstellige Summen verschlingen.[13]

Veränderung ist heute allgegenwärtig, daher wird die Datenmodellierung aufgrund des bei relationalen Datenbanken erforderlichen Zeit- und Arbeitsaufwands zu einer großen Herausforderung. Jedes Jahr werden Milliardenbeträge für Datenmodellierungs- und ETL-Prozesse ausgegeben, um noch „perfektere“ Modelle zu erstellen und umzusetzen, die im Idealfall nie geändert werden müssen. Doch es gibt immer wieder neue Veränderungen.

11 Bereits 1969 veröffentlichte Edgar „Ted“ F. Codd seine berühmte Abhandlung intern bei IBM. 1970 wurde es dann der Öffentlichkeit zugänglich gemacht. (E.F. Codd, „A Relational Model of Data for Large Shared Data Banks.“ Communications of the Association for Computing Machinery, Bd. 13, Nr. 6, S. 377–387)

12 In den 80er-Jahren waren die Spaltennamen in relationalen Datenbanken auf acht Zeichen begrenzt und mussten einheitlich groß- oder kleingeschrieben sein. Damit lauteten typische Spaltennamen z. B. „fname“ (für „firstname“) oder „lname“ (für „lastname“). Nach SQL-Standard sind die Spaltennamen heute auf 30 Zeichen begrenzt.

13 Laut einem Kunden bei einem führenden Fortune-100-Technologieunternehmen kann das Hinzufügen einer neuen Spalte in seinem Betrieb bis zu ein Jahr in Anspruch nehmen und mehr als eine Million Dollar kosten. Bei komplexeren Datenmodellierungsprojekten mit Masterdatenmanagement sind noch längere Zeitrahmen von mehr als fünf Jahren bekannt.

> Es ist überraschend, dass 95 Prozent der gesamten Ausgaben für Datenbanken in relationale Datenbanken fließen, diese aber nur für den Umgang mit jenen 20 Prozent der Daten konzipiert sind, die in strukturierter Form vorliegen.“

NICHT FÜR HETEROGENE DATEN KONZIPIERT

Es ist überraschend, dass 95 Prozent der gesamten Ausgaben für Datenbanken in relationale Datenbanken fließen, diese aber nur für den Umgang mit jenen 20 Prozent der Daten konzipiert sind, die in strukturierter Form vorliegen.[14] Unternehmen sehen sich immer weniger in der Lage, dieser strukturierten Daten Herr zu werden. Zugleich werden die übrigen 80 Prozent völlig vernachlässigt, obwohl sie einen immensen potenziellen Wert darstellen.

Früher speicherten Unternehmen nur einige wichtige Transaktionsdaten sowie ein paar grundlegende Informationen über ihre Kunden. Heutige Unternehmen können nicht mehr so wählerisch sein. So ziemlich alles muss nun gespeichert werden. Da die Kosten für die dafür nötige Infrastruktur inzwischen zumutbar sind, können Unternehmen diese Chance ergreifen, um die Risiken zu minimieren und die Kosten zu senken. Es besteht auch eine entsprechende Erwartungshaltung vonseiten der Kunden, Partner und Aufsichtsbehörden an die Unternehmen: Sie sollen alles in einem brauchbaren Format speichern, das auch für sie von Nutzen ist.

Die wachsende Menge an *strukturierten* Daten ist ein Problem für relationale Datenbanken, weil jede Datenquelle eine andere Struktur hat. Die zum Umgang mit einer neuen Datenquelle erforderlichen Veränderungen sind, wie schon weiter oben besprochen, aufwendig und führen zu komplexeren Schemata. Das ist selbst dann der Fall, wenn die neuen Daten den gleichen Bereich oder die gleichen Konzepte abbilden.

Die wachsende Menge an *unstrukturierten* Daten stellt relationale Datenbanken ebenfalls vor Probleme. Die Zeilen und Spalten einer relationalen Datenbank eignen sich ideal zum Speichern von Feldwerten, doch die meisten Informationen bestehen aus mehr. Nehmen wir etwa die Krankenakte eines Patienten. Sie enthält **Werte** (Name, Geburtsdatum), **Beziehungen** (zu Angehörigen, Leistungserbringern, Symptomen, Medikamenten), **Geodaten** (Adressen), **Metadaten** (Datenherkunft, Sicherheitsmerkmale), **Bilder** (CT-Scans) und **Freitext** (ärztliche Notizen, Transkripte).

Nun stellen Sie sich vor, Sie müssten all diese Daten in einer Excel-Arbeitsmappe darstellen. Dazu wäre viel Einfallsreichtum nötig, und es müssten schwierige Entscheidungen getroffen werden: *Sollen umfangreiche Textblöcke aufgeteilt oder in eine einzelne Tabellenzelle gepresst werden? Wie sollen Daten aus neuen Datenquellen gespeichert werden, die später dazukommen? Wie viele Spalten sollten für die Metadaten vorgesehen werden? Wie steht es um die Beziehungen zwischen verschiedenen Entitäten? Wie steht es um die Struktur innerhalb des Dokuments? Welche Indizes sollten erstellt werden? Was ist, wenn ich die Daten nach einem Element filtern möchte, das nicht in einer Zeile oder Spalte definiert ist?*

So viel Arbeit investiert und so viele Abstriche auch gemacht wurden, um das relationale Modell für jeden Zweck einsetzen zu können, bleibt es doch dabei, dass es eben nicht für heterogene Daten konzipiert wurde.

NICHT AUF SKALIERBARKEIT UND ELASTIZITÄT AUSGERICHTET

Heutige Unternehmen haben Millionen an Benutzern und Petabytes an Daten. Sie führen ihre Anwendungen in der Cloud aus, um dynamische Inhalte an Millionen von PCs und Tablets sowie Mobilgeräte an den verschiedensten geografischen Standorten zu liefern. Um mit dieser neuen Situation umzugehen, brauchen die Unternehmen Skalierbarkeit (die Möglichkeit, neue Kapazitäten für neue Daten und Benutzer hinzuzufügen) und Elastizität (die *Flexibilität*, mit der sich das System skalieren lässt – meist geht es dabei um eine Rückskalierung bei nachlassender Nachfrage).

Die Skalierung relationaler Datenbanken ist leider eine große Herausforderung. Relationale Datenbanken sind dafür konzipiert, auf einem bestimmten Server ausgeführt zu werden, um die Integrität der Tabellenverknüpfungen zu erhalten und die Probleme verteilter Systeme zu vermeiden. Bei diesem System müssen die Unternehmenskunden größere, komplexere und teurere Hardware mit mehr Rechenleistung, Arbeitsspeicher und Speicherplatz erwerben, wenn ein System skaliert werden soll. Auch Upgrades sind nicht unproblematisch, da das Unternehmen dabei einen langwierigen Übernahmeprozess durchläuft, bei dem das System häufig auch noch offline sein muss. Parallel dazu steigt die Anzahl der Benutzer stetig an, so dass die Last – und damit das Risiko unzureichender Rechenressourcen – immer mehr zunimmt.

14 IDC, Juni 2014

> Das Problem dabei ist, dass den IT-Abteilungen diese Komplexität praktisch vom relationalen Modell aufgezwungen wird, weil dieses nicht dafür konzipiert wurde, Informationen auf die richtige Art und zum richtigen Zeitpunkt an verschiedene Benutzergruppen zu liefern.“

Um diesen Problemen entgegenzutreten, haben die Anbieter von relationalen Datenbanken eine ganze Reihe an Verbesserungen eingeführt. Mit den heutigen weiterentwickelten relationalen Datenbanken können sie komplexere Architekturen einsetzen und auf ein „Master-Slave“-Modell zurückgreifen. Die „Slaves“ sind dabei zusätzliche Server, die replizierte oder horizontal fragmentierte (auf mehrere Server bzw. Hosts aufgeteilte) Daten verarbeiten können, um den „Master“-Server zu entlasten. Andere Verbesserungen – wie gemeinsam genutzter Speicher, speichergestützte Verarbeitung (In-Memory-Verarbeitung), ein effizienterer Einsatz von Replikaten, verteilte Cache-Server sowie weitere neue relationale Architekturen – haben die Skalierbarkeit relationaler Datenbanken zweifellos verbessert. Hinter den Kulissen erkennt man aber leicht, dass es sich um ein einziges System mit einem einzelnen Ausfallpunkt („single point of failure“) handelt.[15]

Die Optimierung der relationalen Datenbanken bringt auch hohe Kosten und große Nachteile mit sich. Wenn zum Beispiel Daten über eine relationale Datenbank verteilt sind, kommen zur Beibehaltung der Leistung im Normalfall vordefinierte Abfragen zum Einsatz. Mit anderen Worten steht dem Mehr an Leistung ein Weniger an Flexibilität gegenüber. Außerdem sind relationale Datenbanken nicht dafür konzipiert, wieder hinunterskaliert zu werden – sie bieten also *keinerlei* Elastizität. Nach der Verteilung der Daten und der Zuweisung des zusätzlichen Speicherplatzes ist es fast unmöglich, diese Verteilung rückgängig zu machen.

NICHT FÜR HETEROGENE WORKLOADS KONZIPIERT

Die Fähigkeit, „heterogene Workloads“ zu verarbeiten, besteht darin, sowohl mit betrieblichen als auch mit analytischen Workloads zurechtzukommen. Betriebliche Workloads betreffen die tagtäglichen Geschäftstransaktionen, die in Echtzeit erfolgen, z. B. Käufe von einer großen Anzahl an Kunden. Unter analytischen Workloads hingegen versteht man Vorgänge im Bereich der Business Intelligence und des Data Mining, wenn also zum Beispiel ein Analyst die Gesamtheit aller in einem bestimmten Zeitraum getätigten Käufe untersuchen möchte.

Mitte der 90er-Jahre kam es zu einer Aufteilung in Datenbanken, die für betriebliche Workloads optimiert sind, bekannt als OLTP-Systeme (Online Transaction Processing), und Datenbanken, die für analytische Workloads optimiert sind, die sog. OLAP-Systeme (Online Analytical Processing). Bei OLTP-Systemen ist die Datenmodellierung für die auf der Datenbank beruhende Anwendung optimiert, also für einheitliche, schnelle Transaktionen. Bei OLAP-Systemen hingegen ist die Datenmodellierung für detaillierte Analysen optimiert, einschließlich Aggregaten und Trends. Bald schon hatte man elegante Modelle entwickelt, und die Experten diskutierten über die beste Art der Datenmodellierung für verschiedene Szenarien. Damit fanden Begriffe wie „Sternschema“, „Schneeflockenschema“ und „OLAP-Würfel“ Eingang in den Fachjargon der Datenmodellierer.

Die Aufteilung in betriebliche und analytische Systeme trug leider zur Entstehung verschiedenartiger Data Marts, Data Warehouses, Referenzdatenspeicher und Archive bei, deren Zahl sich zwingend ständig erhöhte. Daten aus betrieblichen Systemen wurden mit ETL-Prozessen in ein zentrales Data Warehouse verschoben, das die Basis für alle geschäftlichen Entscheidungen sein sollte. Doch dies funktionierte nicht mehr, sobald neue und andersartige Fragen nicht mehr beantwortet werden konnten. Daher wurde ein weiterer ETL-Prozess eingeführt, um eine bestimmte Teilmenge der Daten in einen Data Mart zu verschieben. Andere Systeme wiederum wurden eingerichtet, um Referenzdaten zu erfassen. Schließlich erfasste ein Archivsystem alle historischen Daten aus all diesen Systemen. Jedes Mal, wenn eine neue Frage zu beantworten war oder eine neue Anwendung entwickelt werden sollte, wurde ein neues, besseres Modell erstellt – und kein Modell glich dem anderen. Was mit einem einfachen Schema und einer Handvoll an Datenbanken begonnen hatte, vervielfachte sich bald zu Hunderten davon.

Dies ist einer der Gründe, aus denen IT-Abteilungen heute den Großteil ihrer Zeit und ihres Geldes auf die reine Wartung der Unzahl an verschiedenen Systemen im Unternehmen verwenden. Das Problem dabei ist, dass den IT-Abteilungen diese Komplexität praktisch vom relationalen Modell aufgezwungen wird, weil dieses nicht dafür konzipiert wurde, Informationen auf die richtige Art und zum richtigen Zeitpunkt an verschiedene Benutzergruppen zu liefern.

15 Oracle RAC zum Beispiel ist eine relationale „Cluster“-Datenbank mit einem Cluster-Dateisystem. Dahinter verbirgt sich aber nach wie vor ein gemeinsames Festplatten-Subsystem.

> Das Ergebnis der herkömmlichen relationalen Architektur sind somit Leistungseinbußen und potenzielle Fehler im Code.“

PASSEN NICHT ZUR MODERNEN ANWENDUNGSENTWICKLUNG

Moderne Anwendungen werden in objektorientierten Programmiersprachen wie Java, JavaScript, Python oder C# entwickelt, um nur einige zu nennen. Diese Sprachen behandeln Datenstrukturen als „Objekte“, die sowohl Daten als auch Code enthalten (d. h. Attribute und Methoden). Das Problem dabei ist, dass sich diese Art des Umgangs mit Daten grundlegend von der Verarbeitung von Daten durch relationale Datenbanken unterscheidet. Dadurch kommt es zum sogenannten „Impedance Mismatch“, also einer Unverträglichkeit zwischen Datenbank- und Anwendungsprogrammierung.

Um dieses Problem zu umschiffen, verwenden die Entwickler eine Technik, die man als objektrelationale Abbildung (Object-Relational Mapping, ORM) bezeichnet. Darunter versteht man eine bidirektionale Active-Active-Verknüpfung zwischen den Objekten der Anwendungsebene und den Daten, wie sie im Schema der relationalen Datenbank abgebildet sind. Dank ORM können die Entwickler mit Geschäftsregeln und -logiken arbeiten und die aus Anwendungsentwicklungssicht sinnvollsten Ansichten generieren. Bei diesem Ansatz werden Datenbanken einfach als Orte betrachtet, an denen Daten persistent gespeichert und gespeicherte Prozeduren definiert sind. Es steht eine breite Palette an ORM-Tools zur Verfügung, die die Anwendungsentwicklung mit relationalen Datenbanken vereinfachen. Als Beispiele für solche ORM-Tools wären etwa Hibernate für Java, ActiveRecord für Ruby on Rails, Doctrine für PHP und SQLAlchemy für Python zu nennen.

Leider gilt ORM aber auch als halbgare Notlösung für ein systemisches Problem bei relationalen Datenbanken. ORM wurde sogar schon als „Vietnam der Informatik“ bezeichnet, weil es „ein Sumpf ist, der zunächst überwindbar erscheint, mit der Zeit aber immer tiefer wird und den Benutzer schließlich ganz herabzieht, indem es ihn in eine Situation ohne klare Grenzlinien, ohne klare Erfolgsbedingungen und ohne klaren Ausweg versetzt“.[16] Auch in vielen anderen Publikationen wird dargelegt, dass ORM mehr Schaden als Nutzen bringt.[17]

Statt die interessanten Aspekte der Daten eines Objekts zu bewahren, extrahiert ORM die Daten und reißt sie auseinander, wodurch der Aufwand wächst. Dies geschieht zudem, nachdem die Daten im Zuge der Normalisierung schon einmal auf verschiedene Tabellen aufgeteilt wurden. Erinnern wir uns an das Beispiel mit der Krankenakte: Denken Sie daran, wie viele verschiedene Daten, die in einer relationalen Datenbank auf verschiedene Tabellen aufgeteilt werden müssen, Teil dieser Akte sind. Nach dem „Auseinanderreißen“ der Daten und deren Verteilung auf die Tabellen müssen sie dann auf der Anwendungsebene für den Benutzer wieder zusammengefügt oder aggregiert werden. Das bedeutet viel unnötigen Aufwand, viele Verknüpfungen und entweder ein eigenes Framework oder viele Abfragen über mehrere Tabellen (JOIN-Abfragen). Nur so erhält der Benutzer eine tatsächliche Ansicht des eigentlichen Formulars oder Dokuments.

Das Ergebnis der herkömmlichen relationalen Architektur sind somit Leistungseinbußen und potenzielle Fehler im Code. Angesichts der schnelllebigen Anwendungsentwicklungszyklen von heute und des Wunsches der Benutzer nach mehr Interaktivität und schnelleren Reaktionen zeigt das relationale Modell seine Schwächen. Statt nach Notlösungen für das unverträgliche relationale Modell zu suchen, gehen die Entwickler zu neuen Modellen mit einem geringeren Abstraktionsgrad und besserer Leistung über.

EINE DATENBANK DER NEUEN GENERATION FÜR DIE DATEN VON HEUTE

MarkLogic ist eine NoSQL-Datenbank (NoSQL steht für „Not only SQL“ – „nicht nur SQL“), die bei der Ablösung der unflexiblen relationalen Datenbanken der vergangenen drei Jahrzehnte eine Vorreiterrolle einnimmt. MarkLogic bietet eine Vielzahl an Funktionen, die gut zur heutigen Datenwelt passen. Besonders vier Eigenschaften machen es aber wahrhaft einzigartig:

1. Ein **flexibles Datenmodell** zur Speicherung der vielfältigen, unterschiedlich gearteten und veränderlichen Daten von heute
2. **Integrierte Such- und Abfragefunktionen**, mit denen sich Daten zu jedem beliebigen Zeitpunkt besser verwerten lassen
3. **Skalierbarkeit und Elastizität**, um mit massiven, veränderlichen Datenmengen zurechtzukommen
4. **Unternehmensfunktionen** zur Unterstützung missionskritischer Unternehmensanwendungen

16 Ted Neward. Blogbeitrag in „The Blog Ride“, 26. Juni 2006 <http://blogs.tedneward.com/2006/06/26/The+Vietnam+Of+Computer+Science.aspx>

17 Siehe *OrmHate* von Martin Fowler <http://martinfowler.com/bliki/OrmHate.html>, *Object-Relational Mapping Is the Vietnam of Computer Science* von Jeff Atwood, *ORM Is an Anti-Pattern* von Laurie Voss <http://seldo.com/weblog/2011/08/11/orm_is_an_antipattern>, *ORM is An Offensive Anti-Pattern* von Yegor Bugayenko <http://www.yegor256.com/2014/12/01/orm-offensive-anti-pattern.html> und viele andere Beiträge.

> Das erste MarkLogic-Projekt hat 60 Tage gedauert. Der Zeitaufwand mit der bestehenden Technologie wurde auf 3.000 Tage geschätzt.“

Paolo Pelizzoli, Global Head of Architecture, Global Technology Operations bei Broadridge Financial Solutions

ÜBERBLICK ÜBER DIE ALLEINSTELLUNGSMERKMALE VON MARKLOGIC

FLEXIBLES DATENMODELL

MarkLogic ist eine Multi-Modell-Datenbank zur nativen Speicherung und raschen Abfrage von JSON- und XML-Dateien, RDF-Tripeln, Geodaten und großen Binärdateien (z. B. Bildern, Videos). Dadurch eignet sich MarkLogic viel besser zum Umgang mit einer breiten Palette an verschiedenen Datentypen als relationale Datenbanken. Auch Veränderungen des Datenmodells angesichts sich ändernder Daten sind leichter zu bewerkstelligen.

Sowohl JSON als auch XML sind Dokumentenformate. Sie sind die Hauptformate zur Speicherung von Daten mit MarkLogic. Im Vergleich zu den Tabellen relationaler Datenbanken sind Dokumenten für Menschen viel angenehmer zu lesen und bieten einen natürlicheren Zugang zur Modellierung der vielfältigen, veränderlichen und komplexen Daten, mit denen es moderne Unternehmen zu tun haben. Durch die Verwendung eines umfassenderen Datenmodells können die Unternehmen einen höheren Mehrwert aus ihren Daten ziehen.

Dokumente passen auch besser zur modernen Anwendungsentwicklung, weil bei ihnen das Problem des *Impedance Mismatch* wegfällt. Das Dokumentenmodell ermöglicht es den Entwicklern, die Integrität der Daten auf allen Ebenen des Anwendungsstacks zu wahren. So können die Entwickler beispielsweise sowohl in der Datenbank als auch auf der Anwendungsebene und auf der Benutzeroberfläche durchgängig JSON verwenden. Dies ist ein agilerer Ansatz, der auch gut zur zunehmenden Verwendung von JavaScript bei der Entwicklung moderner Webanwendungen passt.

Ein weiterer Vorteil des Datenmodells von MarkLogic besteht darin, dass vor dem Laden der Daten kein Schema und keine Struktur definiert werden müssen. Diese Herangehensweise nennt man „Schema-agnostisch“. MarkLogic ermöglicht seinen Benutzern die Speicherung von Dokumenten mit unterschiedlichen Schemata sowie die Veränderung einzelner Schemata ohne Beeinträchtigung der anderen. Dadurch wird auch die schnelle Kombination relationaler Datentabellen mit verschiedenen Modellen möglich – all dies innerhalb von MarkLogic.

MarkLogic verfügt zudem über Graphdatenbankfunktionen, da es nativ RDF-Tripel speichern kann, die Sprache der Semantik. Im Prinzip ist die Semantik ein Datenmodell, bei dem zwei Entitäten (Personen, Orte oder Dinge) anhand der zwischen ihnen bestehenden Beziehung miteinander verknüpft werden und so ein Tripel bilden. Wenn Tripel miteinander verknüpft werden, bilden sie ein Diagramm, das maschinenlesbar ist und aus dem neue Tatsachen abgeleitet werden können. MarkLogic kann Hunderte von Milliarden an Tripeln direkt neben oder sogar innerhalb von JSON- oder XML-Dokumenten speichern.

MarkLogic ist die einzige Enterprise-Datenbank, die Dokumenten- und Tripel-Speicherung kombiniert. Dank dieser einzigartigen Funktion ist es weniger zeitaufwendig und kompliziert, die Daten im sinnvollsten Format zu modellieren, und Unternehmen können aus ihren Daten einen höheren Mehrwert schöpfen. Insgesamt ist das Datenmodell von

VERGLEICH VON NOSQL-DATENBANKEN

Aufgrund des dringenden Bedarfs an Veränderung sind in den vergangenen Jahren zahlreiche neue Datenverwaltungstechnologien aus dem Boden geschossen. Sie alle wollen bessere Möglichkeiten für den Umgang mit den Daten von heute bieten. Es gibt viele Open-Source-NoSQL-Datenbanken unterschiedlichen Typs, etwa mit Dokumenten-, Graph-, Spalten- oder Key-Value-Speichern. Sie weisen zwar einige Gemeinsamkeiten auf, z. B. die Skalierungsmöglichkeiten mit Commodity-Hardware, letztendlich unterscheiden sie sich aber doch deutlich. Wenn Sie die NoSQL-Landschaft besser kennenlernen möchten, können Sie sich das eBook, *Enterprise NoSQL for Dummies* herunterladen; es ist kostenlos erhältlich unter info.marklogic.com/nosql-for-dummies.html.

> MarkLogic ist speziell dafür konzipiert, mit der Menge, Vielfalt und Geschwindigkeit von ‚Big Data' zuerechtzukommen."

MarkLogic viel flexibler als das relationale Modell. Es bietet eine Plattform zur schnelleren Entwicklung von Anwendungen sowie ein größeres Maß an Agilität, um ad hoc mit Veränderungen zurechtzukommen.

Die dadurch erzielten Verbesserungen gegenüber relationalen Datenbanken können immens sein: „Das erste MarkLogic-Projekt hat 60 Tage gedauert. Der Zeitaufwand mit der bestehenden Technologie wurde auf 3.000 Tage geschätzt", berichtet Paolo Pelizzoli, Global Head of Architecture, Global Technology Operations bei Broadridge Financial Solutions.

INTEGRIERTE SUCHE UND ABFRAGE

Damit Daten schnell und genau durchsucht werden können, braucht eine Datenbank Indizes. Ein Datenbankindex funktioniert ähnlich wie der Index am Ende eines Buches – eine Auflistung der Informationen im Buch, die sich dadurch leicht auffinden lassen, ohne das gesamte Werk durchlesen zu müssen. Bei den meisten Datenbanken wird die Indizierung als eine der Datenspeicherung nachgeordnete Aufgabe betrachtet. Die Indizierung ist ein schwieriger Prozess, bei dem die Benutzer sich klar werden müssen, welche Indizes zur Beantwortung welcher Fragen gebraucht werden, wie sich jeder Index auf die Leistung auswirkt und wie die Indizes gewartet werden sollen. Für die Volltextsuche sind dann zusätzlich auch noch Volltextindizes nötig. Bei relationalen Datenbanken muss dafür neben der Datenbank selbst zusätzliche Software eingerichtet und gewartet werden.

MarkLogic funktioniert anders: Es bietet eine unerreicht starke Indizierungsfunktion sowie Volltextsuche als integralen Bestandteil des Produkts. MarkLogic indiziert Inhalt und Struktur der Daten schon beim Laden und verfügt über zahlreiche Indizes (z. B. Bereichsindizes, Tripelindex, Geodatenindex), die aktiviert oder deaktiviert werden können. Die Indizes von MarkLogic machen es zum Kinderspiel, mit den verschiedensten Abfragesprachen – JavaScript, XQuery, SPARQL (der Abfragesprache für Semantik) und natürlich SQL – sowohl einfache als auch komplexe Abfragen durchzuführen. Ein Beispiel für eine komplexe Abfrage wäre etwa: „Finde alle Einkünfte und Ranglistenplätze von Profisportlern, mit denen Michael Jordan im Laufe seiner Karriere gespielt hat. Beschränke die Ergebnisse auf Sportler, die in New York wohnen und nach Januar 2015 in seriösen Nachrichtenquellen Erwähnung fanden. Reihe die Ergebniskandidaten nach Relevanz auf."

Die Beantwortung solcher mehrdimensionalen Fragen ist alles andere als trivial und wäre mit einer relationalen Datenbank entweder sehr schwierig oder sogar unmöglich. Eine relationale Datenbank hätte Mühe mit der Modellierung der Beziehung zwischen Michael Jordan und den Sportlern, mit denen er zusammen gespielt hat, sowie mit dem Auffinden der Erwähnungen in Nachrichtenquellen, bei denen es sich ja um Textdokumente handelt. Eine relationale Datenbank kann auch keine Aufreihung nach Relevanz vornehmen, wie es etwa eine Suchmaschine wie Google macht – sie würde einfach nur eine simpel nach bestimmten Werten geordnete Ergebnisliste liefern.

Diese Art von komplexen Abfragen kann MarkLogic hingegen mit relativ wenig Code lösen. Die Probleme der Relevanz und der Volltextsuche werden durch das umfassende Datenmodell und die leistungsstarken Indizes von MarkLogic gelöst, die dafür konzipiert sind, nicht nur die gleichen Fragen zu beantworten, die man mit SQL an eine relationale Datenbank stellen würde, sondern auch viele andere. Selbst wenn sich die Fragen ändern, bereitet das dem System keine Probleme. MarkLogic kann auch mit neuen und unerwarteten Abfragen umgehen, ohne dass die Benutzer dafür die Daten und Indizes neu konfigurieren müssten, wie bei einer relationalen Datenbank. Als wäre es damit noch nicht genug, liefert MarkLogic auch bei Hunderten von Terabyte an Daten die Ergebnisse in wenigen Millisekunden, und dies im Rahmen eines Systems mit konsequenter und zuverlässiger Datenpflege.

SKALIERBARKEIT UND ELASTIZITÄT

Die Beschränkungen von Einzelserver-Architekturen kennt MarkLogic nicht. Es ist für riesige Dimensionen auf verteilten Systemen konzipiert. MarkLogic skaliert „horizontal", d. h., es wird auf mehreren Servern ausgeführt, die zusammenarbeiten, indem jeder einen Teil der Last übernimmt. Mit diesem Ansatz kann MarkLogic mit Hunderten von Servern, Petabytes an Daten und Milliarden von Dokumenten arbeiten – und ermöglicht gleichzeitig die Verarbeitung von Tausenden von Transaktionen pro Sekunde. All dies schafft es auch auf kostengünstiger Commodity-Hardware in jeder Umgebung, ob auf hauseigener Hardware oder in einer Cloud-Umgebung wie Amazon Web Services.

> MarkLogic hat sich in missionskritischen Systemen des US-amerikanischen Verteidigungsministeriums, großer Investmentbanken, Gesundheitsdienstleister, internationaler Medienhäuser und in vielen anderen harten Branchen *in der Praxis bewährt*."

Die massive Skalierbarkeit ist beeindruckend, aber noch wichtiger ist vielleicht die *Elastizität* von MarkLogic. Die einzigartige Architektur von MarkLogic macht es möglich, in einem Cluster rasch und unkompliziert Knoten hinzuzufügen oder zu entfernen. Damit kann die Datenbank auch ohne kostenintensives Over-Provisioning den Leistungsanforderungen gerecht werden. Es gibt keine komplexe horizontale Fragmentierung der Daten und keine architektonischen Notlösungen. Die Daten werden automatisch gleichmäßig neu im Cluster verteilt, wenn Knoten hinzugefügt oder entfernt werden. Dies ist einer der Gründe für die hohe Benutzerfreundlichkeit von MarkLogic auf dem Gebiet der Verwaltung.

UNTERNEHMENSFUNKTIONEN

Es ist eine weitverbreitete Fehlannahme, dass NoSQL-Datenbanken nicht für „seriöse" Anwendungsbereiche geeignet sind, sondern nur für Jungunternehmen oder als Aufbewahrungsort für nicht betriebskritische Daten. Bei MarkLogic ist das schlicht und einfach falsch.

MarkLogic bietet all die kritischen Unternehmensfunktionen, die die modernsten relationalen Datenbanken so zuverlässig machen und bei der Speicherung und Verwaltung von Unternehmensdaten absolut unentbehrlich sind. Hier einige der wichtigsten Unternehmensfunktionen von MarkLogic:

- **ACID-Transaktionen** zur Gewährleistung der Datenkonsistenz und zur Vermeidung von Datenverlust oder Beschädigung der Daten
- **Höchste Sicherheitszertifizierungen**, sodass MarkLogic auch in Unternehmensrechenzentren verwendet werden kann
- **Hochverfügbarkeit und Disaster Recovery** zur Sicherstellung der ständigen Datenverfügbarkeit
- **Leistungsüberwachung** zur genauen Nachverfolgung der Bereitstellung und Verwendung von Ressourcen
- **Unternehmensverwaltungstools** mit automatisierten Abläufen für häufige Aufgaben

Bei MarkLogic sind alle diese Funktionen mehr als bloße Behauptungen. Sie haben sich alle in missionskritischen Systemen des US-amerikanischen Verteidigungsministeriums, großer Investmentbanken, Gesundheitsdienstleister, internationaler Medienhäuser und in vielen anderen harten Branchen *in der Praxis bewährt*.

FÜHRENDE UNTERNEHMEN AUF ERFOLGSKURS MIT MARKLOGIC

Hunderte von Unternehmen haben sich für Wandel und Innovation entschieden – mit MarkLogic als Motor für ihre geschäftliche Zukunft. Hier einige Bespiele für diese zahlreichen Erfolgsgeschichten.

OPERATIONAL TRADE SYSTEM BEI EINER GROSSEN BANK

Eine der fünf größten Investmentbanken hat 20 relationale Datenbanken durch nur eine MarkLogic-Datenbank ersetzt. Der Derivatehandel der Bank läuft nun über MarkLogic. Es werden mehr als 100.000 Geschäfte pro Tag verwaltet; zu jedem beliebigen Zeitpunkt befinden sich über 32 Mio. Live-Geschäfte im System. Dieses hohe Volumen bringt ein Cashflow-Risiko von mehr als 100 Mio. $ mit sich. Zusätzlich zu den dank MarkLogic erzielten massiven Kosteneinsparungen hat die Bank nun auch einen globalen, einheitlichen und genauen Echtzeitüberblick über ihr Derivatgeschäft.

REGISTRIERUNG VON MILLIONEN KRANKENVERSICHERTER US-BÜRGER ÜBER HEALTHCARE.GOV

Die Centers for Medicare & Medicaid Services (CMS) stellen über HealthCare.gov Zugang zu medizinischer Versorgung für Millionen von Amerikanern bereit. Dazu müssen die Versicherungsansprüche anhand verschiedener bundesweiter Datenquellen überprüft und Hunderttausende von Benutzern gleichzeitig bedient werden – all das ohne Datenverluste. In den ersten zwei Jahren haben sich dank dem System 12 Mio. Amerikaner für eine Krankenversicherung registriert. Dieser Erfolg steht in starkem Kontrast zu vielen der gescheiterten Krankenversicherungssysteme auf bundesstaatlicher Ebene. Ein Bundesstaat hat angesichts dieses Misserfolgs sogar einen Anbieter einer relationalen Datenbank verklagt.[18]

18 Shelby Stebens. „Oracle sues Oregon officials in healthcare website dispute". Reuters, 27. Februar 2015 <http://www.reuters.com/article/2015/02/27/us-usa-healthcare-oregon-idUSKBN0LV2LK20150227>

> MarkLogic hat uns bei der Beschleunigung der Produktentwicklung und der Verbesserung der Zusammenarbeit zwischen Benutzern und IT-Abteilung unterstützt. Die IT-Abteilung wird nun als geschäftskritischer Teil der Lösung und nicht mehr als Hindernis bei der Lösungsfindung angesehen."
>
> Andrea Powell, CIO bei CABI (The Centre for Biosciences and Agriculture International)

BESSERE LEISTUNG FÜR DEN iPLAYER-STREAMINGDIENST DER BBC

Der iPlayer ist der Videostreamingdienst der BBC in Großbritannien und Nordirland. Eine einzige Sendung wird heute im iPlayer bisweilen mehr als 3 Mrd. Mal angesehen. Um den massiven Skalierbarkeits- und Leistungsanforderungen gerecht zu werden, ist das Team der BBC von einer relationalen Technologie auf MarkLogic umgestiegen, das nun als Hauptkomponente für die Speicherung und Bereitstellung von Metadaten über ihre Sendungen dient. Die BBC hatte schon für die Olympischen Spiele 2012 erfolgreich eine dynamische Content-Delivery-Plattform auf der Grundlage von MarkLogic entwickelt. Diese Skalierbarkeit und Flexibilität wollte man nun auch für den iPlayer nutzen. Nach der Implementierung dauerten Abfragen, die mit SQL noch 20 Sekunden in Anspruch genommen hatten, mit MarkLogic nur noch 20 Millisekunden – eine Verbesserung um Größenordnungen.

NEUE MÖGLICHKEITEN FÜR IHRE DATEN JENSEITS DES RELATIONALEN MODELLS

Den Schritt von der alten in die neue Welt zu wagen, mag zunächst mit Ängsten verbunden sein. Daher ist es oft besser, mit einem kleinen Projekt anzufangen und dann aufzustocken. Nachstehend finden Sie einige Empfehlungen, die Ihnen bei der Planung des Umstiegs auf NoSQL helfen sollen.

EMPFOHLENE WEITERE SCHRITTE

BETONUNG DER DRINGLICHKEIT

Es ist wichtig, dass der Bedarf an Veränderungen erkannt wird, damit der Umstieg auf NoSQL nicht an falscher Gleichgültigkeit scheitert. Dringlichkeit erleichtert die Umsetzung Ihrer Vision und sorgt dafür, dass Sie Rückendeckung erhalten.

DAS RICHTIGE TEAM

Erfolg braucht nicht nur großartige Technologien, sondern auch großartige Mitarbeiter und Prozesse. Es ist wichtig, sowohl die Entscheidungsträger als auch die handelnden Personen im Unternehmen zu kennen und zu verstehen, welche Prozesse der Umsetzung der Initiative abträglich und welche nützlich sein können.

DAS RICHTIGE PROJEKT

Es ist von höchster Bedeutung, mit dem richtigen Anwendungsbereich zu beginnen. Oft empfiehlt es sich, mit einem kleinen, aber wirkungsvollen Projekt anzufangen, bei dem jede unnötige Beeinträchtigung vermieden wird. Die Experten von MarkLogic können Sie beraten, um sicherzustellen, dass Ihr Projekt sich für die Arbeit mit MarkLogic eignet.

FRÜHER KONTAKT MIT MARKLOGIC

Es stehen eigene MarkLogic-Experten zur Verfügung, die zusammen mehr NoSQL-Erfahrung mitbringen als irgendein Team im Unternehmen. Es ist wichtig, sie möglichst früh in der Entwicklungsphase einzubeziehen. Zu diesem Zeitpunkt spielt der Support eine besonders große Rolle.

WEITERE INFORMATIONEN

- **eBook „NoSQL for Dummies"** – Kostenloses eBook mit einem Überblick über NoSQL-Datenbanken info.marklogic.com/nosql-for-dummies
- **Was ist MarkLogic?** – Mehr über die einzigartigen Funktionen von MarkLogic erfahren Sie unter marklogic.com/what-is-marklogic
- **Whitepaper „Inside MarkLogic"** – Erklärung der Infrastruktur, die MarkLogic so leistungsstark macht marklogic.com/resources/inside-marklogic-server
- **Terminvereinbarung** – Sie können Ihren individuellen Anwendungsbereich mit einem Vertreter von MarkLogic besprechen; wenden Sie sich dazu unter sales@marklogic.com an uns