Relationale Datenbanken im Spiegel neuer Speichertechnologien

Differenzierung statt Ablöse

Vor nicht allzu langer Zeit wurden relationale Datenbanken als „Commodity" gesehen. Gemeint ist damit, dass Datenbanken einen simplen Dienst zur Verfügung stellen, der klar definiert ist und bei dem es nicht relevant ist, welches Produkt diesen Dienst erbringt. Dieser Dienst ist – einfach ausgedrückt – der „Persistency Layer" einer Anwendung und umfasst die Operationen Lesen, Schreiben und Verändern von Daten. Zur Speicherung wird das relationale Datenmodell verwendet. Getrieben durch neue Anwendungen und die damit verbundenen Herausforderungen wird von Datenbanken heute jedoch wesentlich mehr verlangt. Daher sehen viele das Ende der bestehenden Systeme kommen. Neue Arten von Datenbanken laufen den bewährten Produkten den Rang ab. Zu diesen neuen Arten gehören beispielsweise NoSQL-Datenbanken, Hadoop-Systeme und In-Memory-Datenbanken.

Unternehmen wie Google, Yahoo, Facebook und Twitter, die durch die Verarbeitung großer Datenmengen angetrieben werden, haben eine enorme Flut neuer Ideen und Technologien losgetreten, wie mit Daten umgegangen werden kann. Außerdem haben Web-2.0-Programmierer inzwischen gelernt, ohne SQL auf Daten zuzugreifen. Es vergeht kaum eine Woche ohne die Ankündigung eines neuen Datenbank-Produkts, das die Lösung aller Probleme traditioneller Datenbanken verspricht. Und in der Tat gibt es Technologien, die besser skalieren, flexibler oder schneller sind. Die neuen Technologien haben mittlerweile einen so hohen Reifegrad, dass auch Unternehmen, deren Hauptgeschäft nicht die Verarbeitung von Daten ist, von diesen neuen Trends profitieren und ihre traditionellen relationalen Datenbanken ablösen können.

Ein Umstieg muss jedoch gut überlegt sein. Man muss verstehen, dass all diese neuen Technologien Fähigkeiten auf Kosten anderer Eigenschaften hinzugewinnen. Manche reduzieren die Flexibilität, um Performance zu erreichen, andere reduzieren die Performance, um Flexibilität zu gewinnen, und einige verzichten auf die Möglichkeit, Daten zu verändern, um extrem schnell einfügen zu können. Es können nicht alle Eigenschaften in einem Produkt vereint werden, und daher finden sich auch kaum neue Technologien unter den wohl bekannten TPC-C- und TPC-H-Benchmarks. Werfen wir also einen Blick auf die wichtigsten Strömungen und diskutieren die möglichen Einsatzfelder und Einschränkungen.

In Memory: Renaissance der relationalen Datenbanken

Traditionelle Datenbanken entstanden zu einer Zeit, in der Hauptspeicher extrem knapp war. Daher werden die Daten auf Festplatte gespeichert, und nur ein kleiner Ausschnitt daraus befindet sich im Hauptspeicher. Der Zugriff auf die Daten erfolgt nur im Hauptspeicher und ein intelligenter Algorithmus sorgt dafür, dass nur die benötigten Daten im Hauptspeicher liegen. Daten müssen dazu von der Platte in den

Hauptspeicher und bei Änderungen auch vom Hauptspeicher zurück auf die Platte transportiert werden. Durch die Fortschritte in der Speichertechnologie stehen heute Rechner zur Verfügung, die sehr viel Hauptspeicher haben. Außerdem haben aktuelle Rechner hohe Rechenleistung, was die Möglichkeit eröffnet, auch eine aufwändige Datenkomprimierung zu verwenden.

In-Memory-Datenbanken nutzen dies. Sie speichern ihre Daten nicht mehr auf Platte, sondern halten sie komplett im Hauptspeicher des Rechners vor. Bei analytischen In-Memory-Datenbanken wird durch die Organisation der Daten in Spalten und die eingesetzten Komprimierungsalgorithmen erreicht, dass größere Datenmengen im Hauptspeicher abgebildet werden können. Traditionelle Datenbanken, die so viel Hauptspeicher verwenden, dass alle Daten darin Platz finden, können mit In-Memory-Datenbanken meist nicht konkurrieren, weil die Zugriffsmechanismen darauf ausgelegt sind, dass nur ein Teil der Daten im Hauptspeicher ist. In-Memory-Datenbanken sind speziell darauf optimiert, dass alle Daten immer im Hauptspeicher liegen. Dadurch erreichen sie deutlich höhere Verarbeitungsgeschwindigkeiten.

Die meisten In-Memory-Datenbanken basieren wie die traditionellen Datenbanken auf dem relationalen Datenmodell und nutzen SQL für den Zugriff darauf. Funktional existiert daher kaum ein Unterschied zwischen diesen beiden. Passt die Datenbank in den Hauptspeicher, ist die Verarbeitungsgeschwindigkeit von In-Memory-Datenbanken um einen Faktor bis zu etwa 100 schneller. Da mehr Hauptspeicher als bisher benötigt wird und dieser immer noch vergleichsweise teuer ist, ist sorgfältig zwischen Kosten und Nutzen abzuwägen. Wichtig ist auch, dass bei analytischen In-Memory-Datenbanken durch die Komprimierung eine Verlangsamung beim Speichern der Daten typisch ist. Kommt es also auf die extrem schnelle Speicherung von Daten an, ist ein In-Memory-System nicht immer die beste Wahl.

NoSQL-Datenbanken: Hoch effiziente Speichersysteme für spezielle Anwendungen

NoSQL-Datenbanken haben ihren Ursprung in der Web-Entwicklung. Schnelle Caches waren für die effiziente Verarbeitung von Web-Objekten notwendig, und traditionelle Datenbanken boten nicht die entsprechende Leistung bei vertretbaren Kosten. Aus diesen Caches entwickelten sich Datenspeicher, die sehr schnell sogenannte Schlüssel-Wert(Key-Value)-Paare speichern und lesen konnten. Die Form der Speicherung unterscheidet sich also grundlegend von relationalen Datenbanken: Das zugrunde liegende Datenmodell ist nicht mehr eine Menge von Tabellen mit Zeilen und Spalten, sondern lediglich eine Menge von Schlüssel-Wert-Paaren. Typische Operationen liefern für einen Schlüssel den Wert oder speichern zu einem Schlüssel den Wert. Um diese Operationen auszuführen, wird eine Programmierschnittstelle angeboten, die von der Anwendung genutzt werden muss. Daher kommt auch der Begriff „no SQL". Heute haben sich diese Datenbanken weiter entwickelt und bieten meist zusätzlich eine einfache SQL-Schnittstelle an. Daher heißt NoSQL heute auch „not only SQL". Bei den NoSQL-Datenbanken lassen sich vier Untertypen unterscheiden:

- Bei den reinen Key-Value-Datenbanken ist der Value-Teil nur ein Block von Daten, der nicht weiter vom System strukturiert wird. Die Anwendung ist selbst für den Inhalt verantwortlich.

- Bei den spaltenorientierten noSQL-Datenbanken ist der Value-Teil in Spaltengruppen organisiert, die wiederum jeweils eine beliebige Menge von Spalten enthalten können. Dadurch können Operationen auch direkt auf Teilinformationen zu einem Schlüssel zugreifen.

- Bei Dokumenten-Datenbanken ist der Value-Teil ein Dokument, das einer der Datenbank bekannten Strukturierung unterliegt. Typisch sind dort JSON- oder XML-Strukturen. Auch hier besteht die Möglichkeit, direkt auf Teil-Informationen zu einem Schlüssel zuzugreifen.

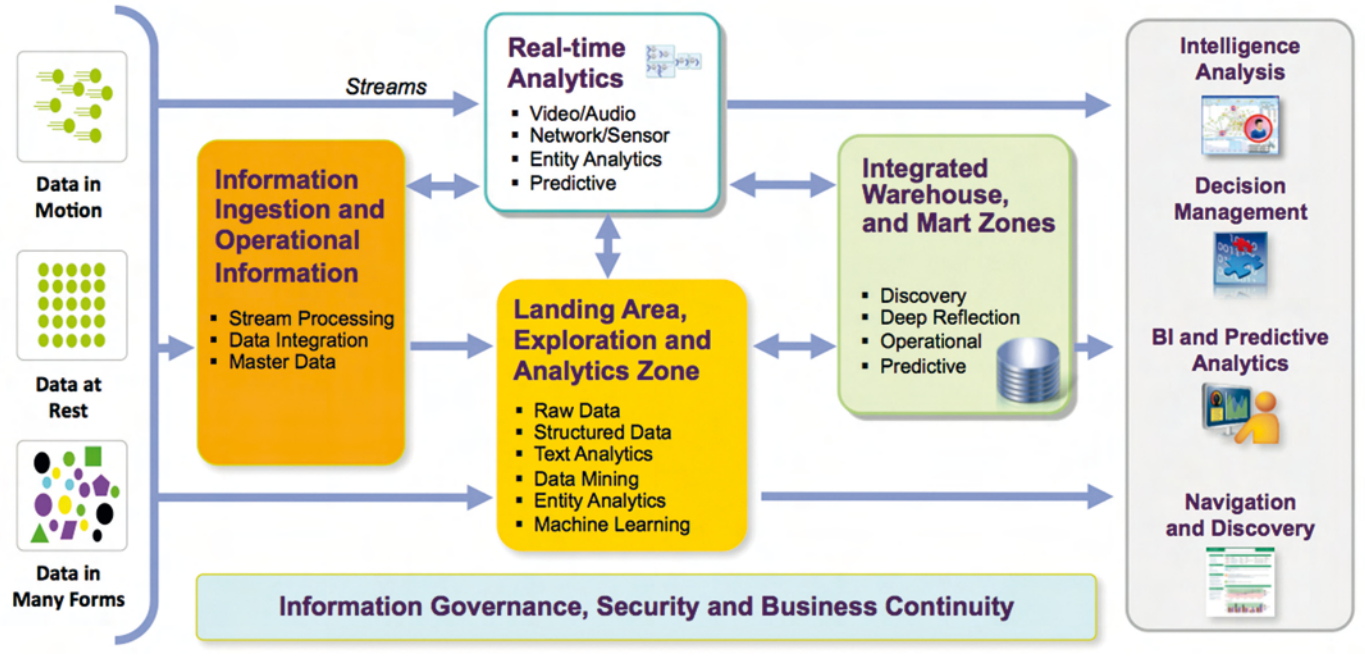
- Die vierte Gruppe von noSQL-Datenbanken bilden die Graph-Datenbanken. Diese sind auf die Verarbeitung von Graphen optimiert, die beispielsweise bei den Beziehungsnetzwerken in sozialen Medien entstehen.

Durch die Reduktion auf ein einfaches Basis-Datenmodell (Key-Value) erreichen NoSQL-Datenbanken oft eine deutlich höhere Verarbeitungsgeschwindigkeit als relationale Systeme. Es wird direkt über einen Schlüssel auf die Daten zugegriffen, und dort finden sich alle benötigten Informationen. Es sind keinerlei Joins oder Interpretationen notwendig. Allerdings sind diese auch ohne deutlichen Overhead

nicht möglich. Die Flexibilität der relationalen Datenbanken kommt indes genau aus der sorgfältigen Modellierung – unabhängig von einem speziellen Anwendungsfall. Bei NoSQL-Datenbanken müssen die zusammenhängenden Daten bereits zusammen abgelegt sein, oder die Anwendung muss sie selbst zusammentragen. Während traditionelle Datenbanken Mechanismen mitbringen, um durch Transaktionen Manipulationen über mehrere Tabellen sicher auszuführen, können noSQL-Systeme meist nur Transaktionen über ein einzelnes Objekt garantieren. Die Beschränkung auf die Key-Value-Mimik und die damit verbundene Vermeidung von Joins erlaubt es NoSQL-Datenbanken, den Datenbestand sehr einfach aufzuteilen. Das sogenannte Sharding splittet die Datenbank in mehrere Unterdatenbanken, die verteilt gespeichert werden können. Dadurch können NoSQL-Datenbanken leicht skalieren und auch für große Datenmengen und Benutzerzahlen sehr gute Leistung garantieren. Hinzu kommt außerdem, dass das Sharding automatisch erfolgt und für die Anwendungen transparent ist. Neben der Skalierung ist auch die Verfügbarkeit eine Stärke von NoSQL-Datenbanken. Die Replikation auf eine oder mehrere Kopien – in Kombination mit dem Sharding – ermöglicht eine hoch effiziente und verfügbare Infrastruktur. Die Kopien werden jedoch meist auf Kosten der Konsistenz asynchron erzeugt. Diese kann jedoch wieder hergestellt werden, indem das Lesen mehrerer Kopien erzwungen wird. Aber es ist klar, dass das Ziel hier die Balance zwischen Konsistenz, Verfügbarkeit und Skalierung ist. Traditionelle Datenbanken sind hier fest gefügt und erlauben kaum Kompromisse bei der Konsistenz. Ist man jedoch bereit, darauf einzugehen, bieten NoSQL-Datenbanken interessante Optionen.

Hadoop: Grundbaustein für Big-Data-Lösungen

Getrieben durch das Thema Big Data wird die Plattform Hadoop stark diskutiert. Hadoop ist keine Datenbank im engeren Sinne, sondern eine Kombination aus einem verteilten Dateisystem und einem Framework für die Ausführung von Algorithmen auf den verteilt gespeicherten Dateien. In Hadoop 1.0, dem aktuellen Standard, können nur Algorithmen ausgeführt werden, die in das Schema Map/Reduce passen. Dabei ist der gewünschte Algorithmus auf zwei Funktionen aufzuteilen (eben eine Map- und eine Reduce-Funktion), die dann durch Hadoop ausgeführt werden. Der Vorteil von Hadoop besteht in der Flexibilität. Da Daten nur in einem Dateisystem abgelegt werden, ist kein Schema zur Speicherung nötig. Das ist bei relationalen Datenbanken anders, weil dort die zu speichernden Daten zuerst in das (relationale) Schema der Datenbank überführt werden müssen. Bei einer relationale Datenbank spricht man daher auch von schema-on-write. Bei einem Hadoop-System wird zwar zum Speichern kein Schema benötigt, aber wenn die Daten gelesen werden sollen, muss eine Struktur über die Daten gelegt werden. Man spricht daher von schema-on-read. In dieser Schema-Verlagerung liegt die große Flexibilität von Hadoop. Neue Datenquellen können direkt erschlossen werden, ohne zuerst Datenschemata und Transformationsprozesse entwerfen zu müssen. Hadoop erlaubt, optimalerweise durch zusätzliche Werkzeuge gestützt, einen explorativen Umgang mit den Daten. Statt zuerst Definitionen vorzunehmen, wird in den Daten selbst nach Strukturen gesucht. Auf Basis dieser Strukturen können dann leicht Analysen durchgeführt werden.



Quelle: IBM

Durch den Aufbau ist Hadoop ein Batch-orientiertes System. Kurze Antwortzeiten sind nicht seine Stärke. Es ist zwar heute oft nicht mehr notwendig, direkt Map/Reduce-Funktionen zu entwerfen, weil es Werkzeuge gibt, die dies übernehmen. Aber dennoch ist der Reifegrad der Hadoop-Landschaft bei weitem nicht mit dem traditioneller Datenbanken vergleichbar.

Ein wichtiger Trend ist aktuell SQL-on-Hadoop. Dabei wird der Zugang auf die Daten in Hadoop durch die Abfragesprache SQL ermöglicht. Fast alle Hersteller arbeiten an dieser Technologie, aber es gibt zwei große Herausforderungen: zum einen die Reduktion der Ausführungsgeschwindigkeit und zum anderen die Unterstützung für einen möglichst großen Teil des SQL-Standards. Es sind hier durchaus Fortschritte sichtbar, aber die Qualität und Geschwindigkeit eines optimierten relationalen Datenbanksystems ist noch lange nicht erreichbar. Als Basis für Analysen und Reporting-Umgebungen ist daher eine relationale Datenbank immer noch die beste Wahl.

Mit Hadoop 2.0 werden Änderungen diskutiert, welche die Fähigkeiten von Hadoop deutlich erweitern werden. Neben Map/Reduce werden so auch andere Zugriffe auf die Daten möglich werden, wodurch sich sicherlich ein breiteres Anwendungsspektrum von Hadoop erschließen wird. Diese Entwicklungen werden jedoch noch mindestens bis 2014 benötigen, um die nötige Reife für einen produktiven Einsatz zu erreichen.

Weiterentwicklungen bei relationalen Datenbanken

Wie schon bei den Entwicklungen im Bereich In-Memory-Datenbanken zu beobachten, greifen die Hersteller der traditionellen relationalen Datenbank die Trends auf und erweitern die bestehenden Systeme. Plattenbasierte Datenbanken bieten nun die Möglichkeit, Tabellen direkt im Hauptspeicher abzulegen. Damit profitieren sie von den damit verbundenen Geschwindigkeitsvorteilen, kombinieren dies aber mit den bereits bestehenden Fähigkeiten etwa hinsichtlich Sicherheit, Datenschutz und Betriebskonzepten. Dadurch können für Unternehmen Vorteile gegenüber Neuimplementierungen entstehen, weil Bestehendes nur erweitert werden muss.

Ähnliche Weiterentwicklungen sind auch aus Richtung der NoSQL-Datenbanken zu beobachten. So unterstützen einige relationale Datenbanken bereits die Schnittstellen weit verbreiteter NoSQL-Graph- oder Dokumenten-Datenbanken. Dies ermöglicht den Einsatz von Werkzeugen, die diese Schnittstellen nutzen, ohne die NoSQL-Datenbanken selbst implementieren zu müssen. Es ist also immer abzuwägen, welche Merkmale für eine Datenbank oder den Betrieb derselben wesentlich sind. Nicht immer ist der Wechsel weg von den bestehenden relationalen Datenbanken nötig oder vorteilhaft.

Ausblick auf eine zukünftige Informationsarchitektur

In der Vergangenheit wurde versucht, die Informationslandschaft in einem Unternehmen auf möglichst wenige Bausteine zu konzentrieren. Die gewünschte Konsolidierung auf ein unternehmensweites Data Warehouse ist ein typischer Ausdruck dieses Bestrebens. Es ist heute jedoch offensichtlich, dass für eine effiziente Implementierung oft eine Kombination verschiedener Technologien notwendig ist. Eine moderne Informationsarchitektur trägt dem mit der Einführung besonderer Zonen Rechnung: Diese Zonen erlauben die Bearbeitung spezieller Aufgaben mit den optimal dazu passenden Technologien. Zu den Zonen, die bereits in der heutigen Informationsarchitektur eines Unternehmens häufig zu finden sind, gehören die Data Warehousing- und Data Mart Zone sowie die Ingestion Zone, in der das ETL (Extract, Transform, Load – nach Wikipedia ein Prozess, bei dem Daten aus mehreren gegebenenfalls unterschiedlich strukturierten Datenquellen in einer Zieldatenbank vereinigt werden) seinen Platz findet. Neu hinzu kommen nun Zonen für die Datensammlung (Landing Zone), explorative Analysen (Sandboxing Zone) und Echtzeit-Analysen (Real-time Analytics Zone).

Es ist schwer vorauszusehen, wie sich die sehr heterogene und komplexe Datenbankwelt weiter entwickeln wird. Es zeichnet sich aber ab, dass es keinen Ersatz für die relationale Datenbank geben wird, sondern diese durch entsprechende Erweiterungen weiterhin ihren Platz in den Unternehmen behält. Neue Technologien werden dann bei Bedarf hinzugefügt und ergänzen das, was bereits heute existiert. ■

WILFRIED HOGE,
Senior IT Architect bei der IBM Software Group