Data-Warehousing 3.0 – Die Rolle von Data-Warehouse-Systemen auf Basis von In-Memory-Technologie

Maik Thiele, Wolfgang Lehner, Dirk Habich

Lehrstuhl Datenbanken, Institut für Systemarchitektur
Technische Universität Dresden, Fakultät Informatik
01069 Dresden
{maik.thiele, wolfgang.lehner, dirk.habich}@tu-dresden.de

Abstract: In diesem Beitrag widmen wir uns der Frage, welche Rolle aktuelle Trends der Hard- und Software für Datenbanksysteme spielen, um als Enabler für neuartige Konzepte im Umfeld des Data-Warehousing zu dienen. Als zentraler Schritt der Evolution im Kontext des Data-Warehousing wird dabei die enge Kopplung zu operativen Systemen gesehen, um eine direkte Rückkopplung bzw. Einbettung in operationale Geschäftsprozesse zu realisieren. In diesem Papier diskutieren wir die Fragen, wie In-Memory-Technologie das Konzept von Echtzeit-DWH-Systemen unterstützt bzw. ermöglicht. Dazu stellen wir zum einen eine Referenzarchitektur für DWH-Systeme vor, die insbesondere push- und pull-basierte Datenversorgung berücksichtigt. Zum anderen diskutieren wir die konkrete Rolle von In-Memory-Systemen mit Blick auf konkrete Aspekte wie der Frage optionaler Persistenzschichten, Reduktion der Batchgröße, Positionierung von In-Memory-Techniken für den Aufbau eines Corporate Memorys und die schnelle Bereitstellung externer Datenbestände zur Unterstützung situativer BI-Szenarien.

1 Einleitung

Data-Warehouses (DWH) haben seit ihrer ersten Erwähnung 1988, damals noch unter dem Begriff „Information Warehouse“, einen enormen Entwicklungsprozess durchlaufen, der durch aktuelle Trends wie beispielsweise hin zur Integration semi-strukturierter Datenbestände oder die Versorgung Echtzeit zunehmend verstärkt wurde. Hatten in der Vergangenheit Data-Warehouse-Systeme lediglich den Auftrag der Informationsversorgung von Management und Wissensarbeitern, entwickeln sie sich schrittweise zur zentralen Plattform für die integrierte Informationsversorgung eines Unternehmens. Oftmals wird in diesem Zusammenhang daher bereits von „Corporate Memory“ oder „Information Hub“ gesprochen. Diese Erweiterung schließt sowohl strategische und taktische Analysen basierend auf vorberechneten Berichten oder multidimensionalen Würfeln als auch die Unterstützung operativer Geschäftsprozesse mit ein, so dass die Feedback-Schleife von den DWH-Infrastrukturen zu operationalen Systemen realisiert wird. Letzteres macht insbesondere die Integration von Daten in Echtzeit notwendig, was jedoch im Konflikt zur Philosophie klassischer Data-Warehouse-Systeme steht, d.h. einen konsistenten Snapshot zu liefern, der als Gegenstand einer ausführlichen Analyse dient.

Der Begriff Echtzeit im Kontext von DWH-Systemen beschreibt dahingehend auch die Forderung, Änderungen in der Diskurswelt zeitnah im Data-Warehouse abzubilden. Aufgrund der Mehrdeutigkeit des Echtzeit-Begriffs wird in der Literatur auch von Near-Real-Time-, Right-Time-, On-Time- oder Living-DHWs gesprochen. Dabei muss man sich die Frage stellen, inwieweit die klassischen aus der Literatur bekannten DWH-Referenzarchitekturen [Le02,BG09,KR02] noch Gültigkeit besitzen und welchen Einfluss ein Design auf Ebene der Infrastruktur auch Auswirkung auf die technische Grundlage hat. All diese Aspekte wurden noch nicht, durch das von Inmon et al. beschriebene DWH 2.0 [ISN08] adressiert. Inmon et al. haben in ihrem Buch lediglich das Konzept des DWH um die Integration von semistrukturierten Datenquellen erweitert und den Begriff des „DWH als lebendes Archiv“ postuliert. Im Sinne einer Evolution wollen wir mit der zusätzlichen Betrachtung von In-Memory-Technologien sowie der Integration von Daten in Echtzeit das sogenannte DWH 3.0 positionieren.

Mit der Frage nach dem Data-Warehouse „der nächsten Generation“ müssen auch mögliche Ausprägungen einer Architektur diskutiert werden. Galt bisher üblicherweise eine homogene technische Plattform (mit unterschiedlichen Datenbanken etc.) als Standardlösung, so werden aktuell vermehrt zum Beispiel Map-Reduce-Systeme zur Vorverarbeitung insbesondere zur Entity-Extraktion im Rahmen einer Textanalyse positioniert; relationale Ansätze kommen erst für die späteren Schichten des DWH-Stacks zum Einsatz. Spielen Map-Reduce-Systeme eine Rolle im Kontext der funktionalen Erweiterung von DWH-Ansätzen durch Integration semi-strukturierter Datenbestände, fokussiert sich dieser Beitrag auf die Erweiterung mit Blick auf Echtzeit-DWHs auf Basis von main-memory-zentrischer Datenbanktechnologie.

Den Trend, spezielle (Teil-)Systeme als Lösung spezifischer Probleme aufgreifend, kritisieren Stonebraker et al. [SBC+07, SMA+07] die Produktstrategie der kommerziellen Datenbankanbieter, die dem Paradigma „one size fits all“ folgend, ihre Altsysteme stetig erweitern, um neuen Anforderungen gerecht zu werden. Stattdessen wird ein „rewrite from scratch“ gefordert, indem für spezifische Probleme wie Datenstromverarbeitung, Textsuche, datenintensive Analysen oder Informationsintegration angepasste Lösungen geschaffen werden. Diese Entwicklung hin zu neuen Lösungen für die Datenverarbeitung und -analyse wird zusätzlich noch durch aktuelle Entwicklungen im Hardwarebereich verstärkt. Die zunehmende Anzahl von Rechenkernen in Many-Core-Architekturen, der Einsatz spezialisierter Prozessoren wie GPU’s (engl. Graphic Processing Unit) oder FPGA (engl. Field Programmable Gate Array) und vor allem die stetig wachsenden Hauptspeichergrößen haben eine Vielzahl neuer Datenbanksysteme hervorgebracht. Ein zusätzlicher Treiber auf der Hardware-Seite ist der Trend hin zu energieverbrauchsoptimierten Systemen, im Zuge dessen neue Prozessortypen und damit einhergehend neue Datenbanklösungen entstanden sind. Im Kontext dieser sehr Hardware-zentrischen Softwareentwicklung spricht man auch von einem “Hardware-Software Co-Design.”

Die enorme Bedeutung der Datenanalyse in vielen Bereichen des Lebens hat zusätzlich dazu geführt, dass ehemals kleinen Nischen groß genug geworden sind, so dass sich Anbieter speziell angepasster Datenbanklösungen behaupten können. Allerdings ist zu

bemerken, dass in den letzten beiden Jahre getrieben durch mehrere Übernahmen (z. B. Sybase durch SAP, Netezza durch IBM, Vertica durch HP) bereits ein deutlich sichtbare Konsolidierung eingesetzt hat.

Der Schritt weg von General-Purpose- hin zu Speziallösungen ist im Bereich der DWH Systeme keineswegs neu. Die Trennung operationaler und analytischer Datenverarbeitung wurde durch das Konzept des Data-Warehousing seit jeher als Konzept postuliert. Inwiefern diese Unterscheidung jedoch angesichts der Forderungen nach Datenanalysen in Echtzeit und den zur Verfügung stehenden technischen Möglichkeiten, insbesondere der main-memory zentrischen Datenverarbeitung noch sinnvoll erscheint, ist im Folgenden zu diskutieren. Dazu sollen in Abschnitt 3 die Kernaspekte einer DWH-Architektur der „nächsten Generation“, welche die Informationsversorgung in ihrer ganzen Breite – von historischen bis hin zu aktuellen Daten – abdecken kann, betrachtet werden. Gleichermaßen ist die Instanziierung potentieller DWH-Architekturen durch konkrete Data-Management-Technologien zu diskutieren. Insbesondere wird untersucht inwiefern In-Memory-Datenbanken den Anforderungen moderner DWH-Anwendungen gerecht werden und welche Implikationen sich daraus für eine DWH-Infrastruktur ergeben. Dieser Diskussion vorangestellt werden zunächst in Abschnitt 2 aktuelle Architekturansätzen zur Datenanalyse vorgestellt. Der Artikel endet mit einer Zusammenfassung und einem Ausblick in Abschnitt 4.

2 Trends in Hardware- und DBMS-Architekturen

Aus den zunehmenden Anforderungen an Systeme zur Datenanalyse sind in den letzten Jahren eine Reihe von Architekturansätzen entstanden. Die wesentlichen Techniken – Parallelisierung, Einsatz spezieller Hardware und die massive Nutzung vom Hauptspeicher –werden in diesem Kapitel in aller Kürze vorgestellt und diskutiert.

Parallelisierung

Ein wichtiger Ansatz, um trotz steigender Anzahl von Anfragen und größer werdenden Datenvolumen Antwortzeiten für einen interaktiven Betrieb zu erreichen, ist die durchgängige Parallelisierung innerhalb von DBMS-Architekturen. Hier werden die folgenden drei Ansätze unterschieden: Shared-Memory, Shared-Disk und Shared-Nothing. Der einfachste aber zugleich auch am wenigsten leistungsfähige Ansatz ist die Shared-Memory-Architektur (z. B. Microsoft SQL Server). Alle Prozessoren teilen sich gemeinsamen Haupt- und Plattenspeicher, wodurch die Implementierung dieser Systeme vereinfacht wird, da auf verteilte Sperrprotokolle verzichtet werden kann. Die Skalierung wird dadurch beeinträchtigt, da sich alle Prozessoren den gleichen Bus für E/A-Operationen und Speicherzugriffe teilen müssen. Ähnlichen Beschränkungen unterliegt die Shared-Disk-Architektur, wobei voneinander unabhängige Verarbeitungsknoten mit jeweils eigenem Hauptspeicher auf einen gemeinsamen Plattenspeicher zugreifen. Oracle RAC und darauf aufbauend die Exa*-Serie bilden ein Beispiel für diese Architektur.

Ein höheren Grad an Skalierung wird üblicherweise mit einer Shared-Nothing-Architektur, auch als Massive-Parallel-Processing- oder kurz MPP-Architektur bezeichnet, erzielt. In diesem Ansatz besitzen alle Verarbeitungsknoten ihren eigenen Haupt- und üblicherweise auch Festplattenspeicher.

Die Datentabellen werden horizontal partitioniert und auf die Verarbeitungsknoten verteilt. Pufferspeicher und Sperrtabellen werden lokal für jeden Verarbeitungsknoten gehalten. Bekannte Vertreter dieser Architektur sind zum Beispiel Teradata, Netezza (mittlerweile IBM) und Greenplum (mittlerweile EMC^2). Zusätzlich interessant werden MPP-Systeme beim Einsatz einfacher Commodity-Hardware, wodurch sich kostengünstig sehr leistungsfähige Systeme konfigurieren lassen. In extremer Form wird dies durch das Map-Reduce-Framework von Google umgesetzt womit nebenläufigen Berechnungen, wie zum Beispiel die Vorverarbeitung unstrukturierter Dokumente, auf viele tausend Knoten skaliert werden können.

Hardwarebeschleunigung

Einen anderen Weg beschreiten Systeme, die durch den Einsatz spezieller und meist proprietärer Hardware Datenbankoperationen beschleunigen. Xtreme Data Appliance ergänzt beispielsweise einen Standardprozessor um einen FPGA-Chip. Der FPGA-Chip dient dabei der Beschleunigung von Kernroutinen bei der Anfrageverarbeitung. Netezza verwendet ebenfalls FPGAs und ergänzt mit diesen den Festplatten-Controller zur Datenvorverarbeitung. Weiterhin existieren Forschungsprojekte zur Nutzung spezieller Mehrkernarchitekturen, wie etwa Grafikkartenprozessoren oder kombinierte Systeme wie die Intel Sandybridge oder AMD Bulldozer-Architektur.

Aus Sicht datenintensiver Data-Warehouse-Anwendungen in Kombination mit Echtzeit-Anforderungen, werden jedoch vor allem hauptspeicherorientierte Ansätze relevant, die neben einer leseoptimierten Datenablage auch schreiboptimiert mit einer Historisierung der Datenbestände ermöglichen. Entsprechende Systeme werden im folgenden Abschnitt diskutiert.

Hauptspeicherdatenbanken und spaltenbasierte Speicherung

Die Optimierung von Data-Warehouse-Anfragen sowohl für Lese- aus auch für Schreiboperationen wird vor allem durch die Reduzierung des E/A-Aufwandes erreicht. Auf Seiten der Algorithmen und Software wird dies vor allem durch Materialisierung von Aggregationen, Reduzierung von Zwischenergebnissen durch den Optimierer, spaltenorientierte Speicherung, Kompression usw. erreicht. Auf Hardwareseite ist der E/A-Aufwand durch die extensive Nutzung vom Hauptspeicher reduzierbar, indem die Daten vollständig im Arbeitsspeicher abgelegt und somit Festplattenzugriffe überflüssig werden. Die effiziente Persistierung von Schreiboperationen wird dabei durch effiziente Datenstrukturen und Gruppen-Commits unterstützt. Bei bis zu zwei Terabyte Hauptspeicher, die heute schon für einen einzelnen Server verfügbar sind, können bereits viele kleinere Data-Warehouse-Szenarien komplett im Hauptspeicher abgebildet werden.

In Anlehnung an einen Vortrag von Jim Gray aus dem Jahr 2006 mit dem Titel „Tape is Dead, Disk is Tape, Flash is Disk, RAM Locality is King“ wurde aufgrund der wachsenden Hauptspeichergrößen die Phrase um „Memory is the new disk“ erweitert. Im Vergleich zu dem rasanten Wachstum bei den Hauptspeichergrößen hat sich allerdings die Speicherbandbreite nur relativ gering verbessert. Um trotzdem die Prozessoren optimal auszulasten, ist es notwendig die vorhandenen mehrstufigen Cache-Architekturen nutzbar zu machen und spezielle Datenbankoperatoren und Datenstrukturen zu entwickeln.

Durch den Einsatz von Kompressionsalgorithmen, die eine signifikante Reduzierung des Datenvolumens erlauben, ist es möglich auch sehr große Datenbanken komplett im Hauptspeicher abzulegen. Damit einhergehend werden in OLAP-Szenarien die Daten häufig nicht mehr zeilen-, sondern spaltenweise abgelegt. Erfolgt der Zugriff auf Tabelleneinträge in OLTP-Anwendungen meist bezogen auf einen ganzen Datensatz, was insbesondere auch für Einfüge-, Lösch- und Aktualisierungsoperationen gilt, so ist die zeilenorientierte Speicherung im Vorteil. Wird jedoch, wie für OLAP-Anfragen typisch, meist nur auf wenige Spalten einer Tabelle zugegriffen, so reduziert die spaltenorientierte Speicherung den E/A-Aufwand und damit die Kosten der Anfrageverarbeitung erheblich. Ohne effiziente Datenstrukturen sind Änderungsoperationen in spaltenorientierten Systemen in der Regel teurer, da die Daten eines Datensatzes nicht hintereinander gespeichert, sondern auf mehrere Spalten verteilt werden müssen.

Die aufeinanderfolgende Speicherung von Daten gleichen Typs bei spaltenorientierten Datenbanksystemen erlaubt den Einsatz vielfältiger Kompressionstechniken, wie zum Beispiel Lauflängenkodierung, Delta-Kodierung oder Wörterbuchkompression. Dadurch wird der E/A-Aufwand im Vergleich zu zeilenorientierten Systemen nochmals um ein Vielfaches reduziert. Die Anfrageverarbeitung wird durch den Einsatz von Kompressionstechniken jedoch nicht notwendigerweise verzögert. Oftmals kann die Datenverarbeitung direkt auf den komprimierten Daten durchgeführt werdenImplementiert wurde die spaltenorientierte Speicherung und Anfrageverarbeitung sowohl durch Forschungsprototypen wie C-Store (wurde in Vertica kommerzialisiert und 2011 von HP übernommen) und Monet-DB sowie in kommerziellen Systemen wie Sybase IQ (diskbasiert, mittlerweile zu SAP gehörend). Für sehr große Data-Warehouse-Systeme ist durch die zusätzliche Anwendung einer Shared-Nothing-Architektur (z.B. SAP BW Accelerator bzw. SAP HANA) eine nahezu beliebige Skalierung möglich.

3 Die Data-Warehouse-Architektur 3.0

Das klassische DWH-Referenzmodell beschreibt die Rolle der einzelnen Datenschichten einer DWH-Architektur (Arbeitsbereich, Konsolidierte Basisdatenbank, DWH-Datenbank, Data Marts) sowie den Datenfluss zwischen diesen Komponenten. Aktuelle Fragestellungen hinsichtlich des Aspekts der Echtzeit und damit einhergehend zu Batchgrößen, optionalen Persistenzschichten, alternativen Zugriffssemantiken usw. bleiben jedoch unberücksichtigt.

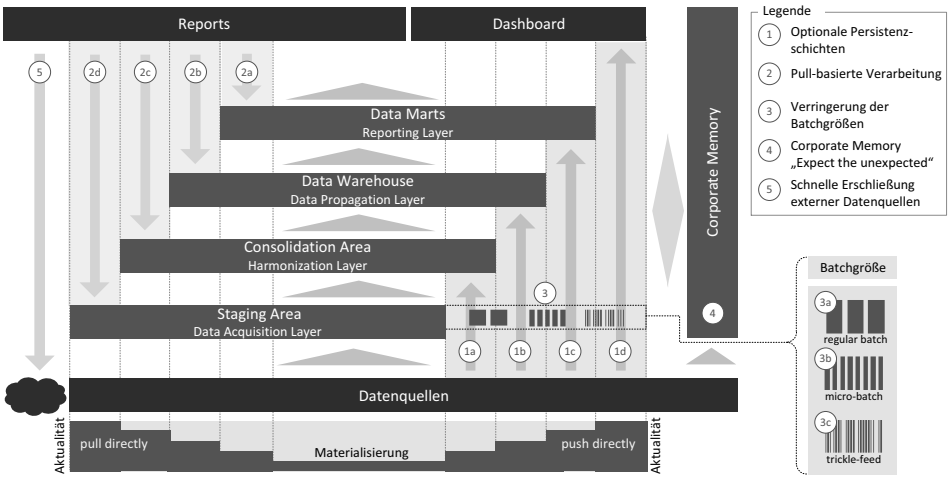

Abbildung 1: Die fünf Aspekte einer Echtzeit-Data-Warehouse-Architektur.

Unter Berücksichtigung der Trends im Bereich der DBMS-Architekturen (siehe Abschnitt 2) sowie den zunehmenden Anforderung nach Datenanalysen in Echtzeit wollen wir im Folgenden das klassische DWH-Referenzmodell auf seine Schwächen untersuchen und schrittweise erweitern.

Optionale Persistenzschichten

Ein DWH-System speist sich aus einer großen Menge üblicherweise heterogener Quellsysteme, deren Daten in einem aufwändigen Prozess bereinigt, transformiert, konsolidiert und in eine für die Analyse optimierte Darstellung gebracht werden müssen. An dieser schrittweisen Verfeinerung der Daten sind eine Vielzahl von Transformationen und Repräsentationen in den jeweiligen Persistenzschichten beteiligt, die aufeinander aufbauend jeweils einen weiteren Verfeinerungszustand der Ursprungsdaten materialisieren. Diese von der klassischen Referenzarchitektur vorgeschrieben Materialisierung der Zwischenstände führt, zusammen mit den notwendigen Transformationsprozessen und Qualitätssicherungsmaßnahmen, zu Verzögerung beim Laden und läuft damit der Echtzeitanforderung zuwider. Daher ist fallbasiert zu entscheiden, für welche Daten die Implementierung des gesamten DWH-Stacks und die damit verbundene redundante Datenhaltung wirklich adäquat ist oder ob nicht einzelne Schichte lediglich virtuell vorhanden sind (Abbildung 1, 1a bis 1d).

In welchem Umfang und welche Zwischenstände persistiert werden müssen, gibt die Echtzeit-DWH-Architektur nicht vor. Diese Entscheidung wird vielmehr anhand gegebener technischer oder juristischer Anforderungen im Unternehmen, wie etwa Governance-Bestimmungen, Möglichkeit der Datenrückverfolgung (engl. data lineage) oder bestimmte Gesetzgebungen, gefällt. Eine gute Übersicht der Kriterien für Datenpersistenz in DWH-Systemen findet sich in [WK11].

Wird in den Datenquellen dasselbe Datenmodell wie im DWH verwendet, so kann die technische Harmonisierung der Daten entfallen und damit die Schicht der konsolidierten Basisdatenbank übersprungen werden (Fall 1b in Abbildung 1). Ebenso verhält es sich mit dem Bereich der Staging Area zur temporären Zwischenablage in das DWH eingebrachter Quelldatenbestände. Diese Schicht kann ebenfalls entfallen, falls keine Entkoppelung der Quellsystemen und kein Vorhalten der Daten zur Rekonstruktion des Datenverfeinerungsprozesses im Fehlerfall des DWHs notwendig ist (Fall 1a) und die Datenbestände direkt aus den Quellsystemen übernommen werden können. Des Weiteren sind Szenarien realisierbar (zum Beispiel „Dashboards"), in denen die Daten der Quellsysteme aufgrund ihrer Beschaffenheit keiner weiteren Transformationen bedürfen, so dass sie direkt an die Data Marts bzw. die Reporting-Schicht propagiert werden können (Fall 1c und 1d).

Rolle von main-memory zentrischen Systemen: *DWH-Architekturen sind in der Praxis nicht nur auf die vier, in Abbildung 1 dargestellten Schichten beschränkt. Stattdessen ist in vielen Szenarien, zur Beschleunigung analytischer Data-Warehouse-Anfragen, eine vielstufige Data-Mart-Schicht, bestehend aus einer Reihe von einander abgeleiteter Materialized Views, zu finden. Diese unterliegen hohen Wartungskosten und verringern je nach Wartungsstrategie die Aktualität der Daten. Solche klassischen Datenbank-Tuning-Mechanismen sind einem In-Memory-DB-System obsolet. Sobald die Daten vollständig im Hauptspeicher vorgehalten werden können, sind auch datenintensive Operationen direkt auf den Basisdaten berechenbar. Somit kann durch den Einsatz von In-Memory-Techniken der DWH-Stack vereinfacht und die Kosten für das mehrfache Kopieren und Materialisieren von Daten reduziert werden. Zusätzlich dazu wird auch die Administration von Data-Warehouse-Systemen vereinfacht und damit ihre Gesamtbetriebskosten gesenkt.*

Im Moment sind In-Memory-Datenbanken noch auf eine zusätzliche festplattenbasierte Backupplattform angewiesen, um Persistenz zu gewährleisten. Durch aktuelle Weiterentwicklungen bei den nichtflüchtigen Speichern (NVRAM, Non-Volatile Random-Access Memory) ist jedoch absehbar, dass in Zukunft diese ebenfalls redundante Datenhaltung entfallen und sich der DWH-Stack noch stärker konsolidieren wird.

Pull- vs. Push-basierte Verarbeitungssemantik

Unter Echtzeitfähigkeit eines DWH-Systems wird die Fähigkeit verstanden, Daten schnell in das DWH zu integrieren und hinsichtlich der Datenanalysen aufzubereiten. Die zeitliche Verzögerung vom Eintreten eines Ereignisses in der Diskurswelt bis hin zur Datenbereitstellung im DWH oder im Data Mart wird als Datenverzögerung bezeichnet. Jede Schicht und jede Verarbeitungsstufe des klassischen DWH-Stacks wirkt sich negativ auf die Datenverzögerung aus. Diese kann, wie bereits skizziert, durch das Überspringen von Persistenzschichten minimiert werden.

Neben dieser push-basierten Verarbeitungssemantik von Daten ist für ein echtzeitfähiges DWH-System auch ein pull-basierter Datenzugriff notwendig und strukturell zu unterstützen (Abbildung 1, 2a bis 2d). Dies setzt voraus, dass die Nutzer des DWH die verschiedenen Verarbeitungszustände der Daten kennen und sich somit bewusst für noch nicht vollständig verfeinerte, dafür aber aktuelle Daten entscheiden können. Abhängig von der adressierten Datenschicht, vom Arbeitsbereich (Fall 2d in Abbildung 1) bis hin zu den Data Marts (Fall 2a), kann somit mit abnehmender Abhängigkeit vom Datenproduktionsprozess frühzeitig auf aktuelle Daten zugegriffen und natürlich im Rahmen des DWH mit anderen Datenbeständen aus dem Produktionsprozess verknüpft werden.

Rolle von main-memory zentrischen Systemen: *Betrachtet man die pull-basierte Verarbeitung im Zusammenhang mit den im vorherigen Abschnitt vorgeschlagenen optionalen Persistenzschichten so resultiert dies in einer schrittweisen Aufhebung der Trennung zwischen OLTP- und OLAP-Systemen (z. B. die Kombination der Fälle 1b und 2b in Abbildung 1). Statt der bisher durch den Batchbetrieb getrennten Verarbeitung von Anfragen und Aktualisierungen ist im Kontext eines Echtzeit-DHWs eine gleichzeitige Verarbeitung von lesenden und schreibenden Transaktionen notwendig. Die Konsolidierung beider Welten in einem System und die daraus resultierenden gemischten Workloads erfordern eine leistungsfähige Anfrageverarbeitung wie sie nur durch In-Memory-Datenbanken bereitgestellt werden kann. Allerdings müssen weiterhin die klassischen ACID-Eigenschaften gewährleistet und innerhalb einer OLAP-Session ein konsistenter Datenzustand garantiert werden. Dabei sollen jedoch weder die schreibenden OLTP-Zugriffe, noch die komplexeren OLAP-Anfragen verzögert werden. Die In-Memory-Datenbank HyPer [KN11] gewährleistet dies beispielsweise durch einen hardware-unterstützten Replikationsmechanismus der jeweils einen konsistenten Snapshot der transaktionalen Daten vorhält. Die wesentliche Idee dieses Ansatzes besteht darin, dass die Snapshots durch Anwendung des Fork-Betriebssystemaufrufs auf den OLTP-Prozess erzeugt werden können. Durch die entsprechend gegeben Prozessorunterstützung bei der Verwaltung des virtuellen Speichers, sind hohe Durchsatzraten seitens des OLTP-Workloads aber auch geringe Antwortzeiten bei den OLAP-Anfragen erreichbar. Ein anderer Ansatz wird durch das Hyrise-System [GKP+09] implementiert. Mit Hilfe einer automatisierten vertikalen Partitionierung der Daten können gemischte OLTP/OLAP-Workloads jeweils optimal unterstützt werden. Dazu werden die Partitionsgrößen basierend auf einem Modell über die Cache-Misses und den entsprechenden Zugriffscharakteristika bestimmt. Schaut man sich die beiden Ansätze an so ist festzustellen, dass zur Unterstützung gemischter OLTP/OLAP-Workloads jeweils hybride Ansätze erforderlich sind um beiden Anfragentypen optimal zu unterstützen.*

Diese Entwicklung hin zu hybriden Datenbankarchitekturen ist auch anhand der Produktstrategie der großen Datenbankhersteller absehbar. So existieren mit FlexStore von Vertica, der NewDB von SAP bzw. mit Oracle 11g Release 2 bereits Systeme die hybriden Datenstrukturen unterstützen.

Reduktion der Batchgröße

Eine Reduzierung der Datenverzögerung kann neben der optionalen Persistierung und dem pull-basierten Datenzugriff vor allem durch die Optimierung der beteiligten Transformationsprozesse erreicht werden. Dazu sind die beteiligten ETL-Prozesse (Extraktion, Transformation und Laden) zu optimieren [TVS07] und mit ausreichenden Hardware-Ressourcen zu provisionieren. Diese Maßnahmen vorausgesetzt, ist eine entscheidende Verringerung der Datenverzögerung vor allem durch kürzere Ladezyklen zu erzielen (Abbildung 1, 3a bis 3c). Die klassischen Ladeprozesse basierend auf täglichen Batches („regular batches“, Fall 3a) werden dabei in Abhängigkeit der Aktualitätsanforderung der Anwendung durch kleinere Batchgrößen („micro batches“, Fall 3b) bzw. durch einen kontinuierlichen Strom von Aktualisierungen („trickle feed“, 3c) ersetzt. Das Data Warehouse der Zukunft ist damit in der Lage die komplette Informationsversorgung eines Unternehmens, vom klassischen ETL bis hin Datenstromsystemen, abzudecken. Dabei geht es nicht um die Ablösung existierender Techniken sondern vielmehr um die Schaffung einer generalisierten DWH-Architektur anhand derer entsprechend konforme Installationen instanziiert werden können. Mit der in Abbildung 1 dargestellten Verringerung der Aktualisierungsperiodizität müssen sich allerdings auch die auf die Verarbeitung von Massendaten spezialisierten ETL-Prozesse ändern. Dies schließt etwa die Entwicklung nicht-blockierender ETL-Operatoren wie zum Beispiel für die Erzeugung von Surrogatschlüsseln ein [PSV+08]. Der Strom kontinuierlicher Aktualisierungen kann darüber hinaus einer Ablaufplanung unterworfen werden. Dazu sind die Anforderungen der DWH-Anwender hinsichtlich der Aktualität der Daten zu ermitteln und die Aktualisierungen entsprechend zu priorisieren [TL09].

Rolle von main-memory zentrischen Systemen: *Die Verringerung der Datengranularität für Aktualisierungen bis hin auf Tupelebene schließt die Verwendung effizienter Bulk-Load-Mechanismen aus wodurch Ladeprozess extrem verlangsamt werden. Um diese Nachteil zu umgehen wurde mit RiTE [TPL08] einen Middleware entwickelt die den Ladeprozess anfragetrieben optimiert. RiTE wurde unter Verwendung einer hauptspeicherbasierten und schreiboptimierten Datenstruktur umgesetzt und ist dadurch der Lage eine große Menge von Änderungen zwischen zu speichern und diese bedarfsgesteuert per Bulk-Load in das DWH laden. Mit Hyper [KN11] und Hyrise [GKP+09] existieren darüber hinaus schon hybride, d.h. sowohl schreib- als auch leseoptimierte Datenstrukturen. Wie sich solche Ansätze allerdings unter extrem hohen Aktualisierungsraten, etwa bei der Integration von Sensordatenströmen oder Click-streams in ein DWH verhalten bleibt zu untersuchen.*

Corporate Memory

Im Kontext von Echtzeit-DWH-Systemen wird häufig der Begriff der situativen Datenanalyse (engl. situational BI) verwandt. Die zentrale Eigenschaft der situativen Datenanalyse im Vergleich zu Echtzeit-DWH-Systemen besteht darin, dass der Umfang der Datenbereitstellung dem DWH-Betreiber a priori nicht bekannt ist. Dennoch muss es möglich sein, die Bedürfnisse der DWH-Anwender zeitnah zu befriedigen.

Derartige Szenarien kommen klassischerweise aus dem Bereich der wissenschaftlichen Datenanalyse, wobei erst schrittweise die unterschiedlichen Muster in den Daten aufgedeckt werden. Zur Unterstützung solcher Anforderungen ist eine zusätzliche Datenschicht, der sogenannte Corporate Memory (Abbildung 1, Fall 4) notwendig.

Ein Corporate Memory (CM) ist innerhalb der Referenzarchitektur so positioniert, dass die Daten in dem gleichen Umfang und in der gleichen Granularität wie sie von den Datenquellen zur Verfügung gestellt, gespeichert und mittelfristig archiviert werden. Getreu dem Motto „expect the unexpected" werden zunächst unnötige Attribute nicht ausgeblendet, sondern proaktiv für zukünftige Anforderungsänderungen vorgehalten. Das CM erlaubt damit höchstmögliche Flexibilität bei sich ändernden und nicht vorhersehbaren Nutzeranforderungen und erfüllt gleichzeitig die Rolle einer „Recovery-Schicht", aus welcher Transformationsprozesse im Fehlerfall wiederholt werden können. Neben den Daten aus den Quellsystemen werden im CM auch aufbereitete Datenbestände aus den anderen Schichten archiviert, so dass oftmals auch zwischen „Aquisition Memory" und „Reporting Memory" unterschieden wird.

Rolle von main-memory zentrischen Systemen: *In Hinblick auf den Umfang der Daten die in einem CM gespeichert werden, ist der Einsatz von In-Memory-Datenbanken für diesen Zweck innerhalb den nächsten Jahre, ausgeschlossen. Zusätzlich sind die Antwortzeiten bei einem CM weniger relevant so dass Hauptspeicherlösungen nicht erforderlich sind. Mittelfristig wird der CM daher als Backbone einer DWH-Landschaft, bestehend aus einer oder mehrere In-Memory-Datenbanken, fungieren. Schaut man jedoch ein wenig in die Zukunft und extrapoliert die Entwicklung der Hauptspeichergrößen insbesondere bei nichtflüchtigen Speichertechnologien ist eine technische Migration des CM in eine Hauptspeicherdatenbank durchaus denkbar.*

Schnelle Erschließung externer Datenquellen

Neben der hinsichtlich Umfang und Detailgrad proaktiven Speicherung von Daten besteht im Kontext von Echtzeit-DWH des Weiteren die Anforderung bisher unbekannte, zum großen Teil externe Datenquellen zeitnah zu erschließen (Abbildung 1, Fall 5). Ein typischer Anwendungsfall ist etwa die Einbindung offener Daten (engl. open data) bei der Erstellung von Geschäftsanalysen. Klassische DWH-Architekturen sind aufgrund der Forderungen nach hoher Datenqualität und Konsistenz an wohldefinierte Entwicklungszyklen gebunden und daher zu wenig agil um situativen Bedürfnisse nach zusätzlichen Datenquellen zu befriedigen. Zur Unterstützung situativer Datenanalysen existieren diverse Methoden und Lösungsansätze sowohl auf organisatorischer als auch auf technischer Ebene. So müssen externe Datenquellen mit einer Dienstschnittstelle beschrieben sein, so dass eine einheitliche Abstraktionsschicht existiert, auf die dann aufgesetzt werden kann. Für weiterführendere Betrachtungen sei an dieser Stelle auf [TL11] verwiesen.

Rolle von main-memory zentrischen Systemen: *Durch den Einsatz von hauptspeicherbasierten Systemen, können externe Datenquellen ohne weiteren Einsatz von Datenbank-Tuning-Mechanismen zur Nutzung bereitgestellt werden.*

Durch den Wegfall technischer Hilfsstrukturen wie beispielsweise Indexstrukturen, die für einen effizienten Zugriff auf Plattensysteme notwendig sind, reduziert sich die „time to consumption" beträchtlich. Anders formuliert ist eine situative Integration und Nutzung externer Datenquellen mit klassischen Systemansätzen nur sehr aufwändig realisierbar.

4. Zusammenfassung

Eine klare Trennung zwischen operativen und analytischen Systemen, wie sie bisher aus unterschiedlichen, insbesondere auch technischen Gründen propagiert wurde, hat sich überlebt. Nur durch Kombination beider Welten ist eine echte Rückkopplung der aus DWH-Infrastrukturen gewonnenen Erkenntnisse zurück in die operativen Geschäftsprozesse möglich. Die wesentlichen Unterscheidungsmerkmale von Echtzeit-Data-Warehouse-Systemen im Vergleich zur klassischen DWH-Architektur bestehen zum einen in kürzeren Aktualisierungszyklen und zum anderen durch die Möglichkeit, dass Zugriffe auf die Datenbestände nicht nur auf die Data-Marts beschränkt sind, sondern entlang des Datenverfeinerungsprozess in jeder Prozessstufe ausgeführt werden können. Die gleichzeitige Verarbeitung von lesenden und schreibenden Transaktionen stellt hohe Anforderungen an die Leistungsfähigkeit eines Datenbanksystems und ist nur durch hauptspeicherbasierte Techniken zu gewährleisten. In-Memory-Datenbanken spielen somit eine Schlüsselrolle bei der Umsetzung der Vision eines DWH 3.0. Weiterhin gilt, dass dem Aspekt der situativen Datenanalyse („Self-service-BI") bereits beim Entwurf des Systems Rechnung sowohl aus technischer als auch aus organisatorischer Perspektive getragen werden muss. Dem „Corporate Memory" kommt im „DWH der Zukunft" dabei eine wesentlich prominentere Stellung zu als in den klassischen Ansätzen.

Literaturverzeichnis

[BG09] Bauer, A.H., Günzel, H.H.: Data-Warehouse-Systeme. dpunkt, Heidelberg (2009).

[GKP+09] Grund, M., Krüger, J., Plattner, H., Zeier, A., Cudre-Mauroux, P., Madden, S.: Hyrise: a main memory hybrid storage engine. In: Proc. VLDB Endow., vol. 4, pp. 105–116, November 2010.

[ISN08] Inmon, W.H., Strauss, D., Neushloss, G.: DW 2.0: The Architecture for the Next Generation of Data Warehousing. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (2008).

[KN11] Kemper A., Neumann T.: HyPer – A Hybrid OLTP&OLAP Main Memory Database System Based on Virtual Memory Snapshots. In: ICDE 2011.

[KR02] Kimball, R., Ross, M.: The Data Warehouse Toolkit: The Complete Guide to Dimensional Modeling, 2nd edn. John Wiley & Sons, Inc., New York, NY, USA (2002).

[Le02] Lehner, W.: Datenbanktechnologie für Data-Warehouse-Systeme: Konzepte und Methoden, 1. Auflage edn. dpunkt-Verlag (2003).

[PSV+08] Polyzotis, N., Skiadopoulos, S., Vassiliadis, P., Simitsis, A., Frantzell, N.E.: Meshing streaming updates with persistent data in an active data warehouse. IEEE Trans.Knowl. Data Eng. 20(7), 976-991 (2008).

[SBC+07] M. Stonebraker, C. Bear, U. Cetintemel, M. Cherniack, T. Ge, N. Hachem, S. Harizopoulos, J. Lifter, J. Rogers, and S. B. Zdonik. One size fits all? Part 2: Benchmarking studies. In Third Biennal Conference on Innovative Data Systems Research (CIDR 2007), pages 173-184, 2007.

[SMA+07] M. Stonebraker, S. Madden, D. J. Abadi, S. Harizopoulos, N. Hachem, and P. Helland. The end of an architectural era: (it's time for a complete rewrite). In VLDB '07: Proceedings of the 33rd international conference on Very large data bases, pages 1150-1160. VLDB Endowment, 2007.

[TBL09] Thiele, M., Bader, A., Lehner, W.: Multi-objective scheduling for real-time data warehouses. In: In Proceedings der 12. GI-Fachtagung für Datenbanksysteme in Business, Technology und Web, pp. 307-326. GI (2009).

[TL11] Thiele, M., Lehner, W.: Real-Time BI and Situational Analysis (Chapter). In BUSINESS INTELLIGENCE APPLICATIONS AND THE WEB: MODELS, SYSTEMS AND TECHNOLOGIES, IGI-Global, Editors: Marta Zorrilla, Jose-Norberto Mazón, Óscar Ferrández, Irene Garrigós, Florian Daniel and Juan Trujillo, ISBN-13: 9781613500385.

[TPL08] Thomsen, C., Pedersen, T.B., Lehner, W.: Rite: Providing on-demand data for right-time data warehousing. In: ICDE '08: Proceedings of the 2008 IEEE 24th International Conference on Data Engineering, pp. 456-465. IEEE Computer Society, Washington, DC, USA (2008).

[TVS07] Tziovara, V., Vassiliadis, P., Simitsis, A.: Deciding the physical implementation of ELT workflows. In: DOLAP '07: Proceedings of the ACM tenth international workshop on Data warehousing and OLAP, pp. 49-56. ACM, New York, NY, USA (2007).

[WK11] Winsemann, T., Köppen, V.: Kriterien für Datenpersistenz bei Enterprise Data Warehouse Systemen auf In-Memory Datenbanken. In: In Proceedings of the 23. GI-Workshop on Foundations of Databases, pp. 97-102. GI (2011).