Kriterien für Datenpersistenz bei Enterprise Data Warehouse Systemen auf In-Memory Datenbanken

Thorsten Winsemann
Otto-von-Guericke-Universität Magdeburg
Kanalstraße 18
D-22085 Hamburg
+49(0)160/90819410
thorsten.winsemann@t-online.de

Veit Köppen
Otto-von-Guericke-Universität Magdeburg
Universitätsplatz 2
D-39106 Magdeburg
+49(0)391/67-19351
veit.koeppen@ovgu.de

ABSTRACT
Persistente Datenhaltung über mehrere Schichten innerhalb eines Enterprise Data Warehouse Systems ist notwendig, um den dort vorhandenen, sehr großen Datenbestand nutzen zu können, z.B. für Reporting und Analyse. Die Pflege und Wartung solcher meist redundanten Daten ist jedoch sehr komplex und erfordert einen hohen Aufwand an Zeit und Ressourcen. Neueste In-Memory-Technologien ermöglichen gute Performanz beim Datenzugriff, so dass sich die Frage stellt, welche Daten aus welchem Grund bzw. für welchen Zweck überhaupt noch persistent abgelegt werden müssen – und wie sich dies effizient entscheiden lässt. In diesem Papier präsentieren wir eine Übersicht von Gründen für Datenpersistenz, welche als Entscheidungsgrundlage bei der Problematik dient, Daten in Enterprise Data Warehouses auf In-Memory Datenbanken zu speichern.

Kategorien und Themenbeschreibung
H.2.7 [**Database Management**]: Datenbank-Administration – *Data Warehouse und Repository.*

Allgemeine Begriffe
Management, Design.

Schlüsselwörter
Enterprise Data Warehouse, Persistenz, In-Memory Datenbank.

1. EINLEITUNG
Heutige Data Warehouse Systeme (DWS) sind gekennzeichnet durch sehr große Datenvolumina [1]. Der Aufbau und Betrieb solcher Systeme erfordert hohe Anforderungen an die Datenbereitstellung, insbesondere hinsichtlich Performanz, Datengranularität, -flexibilität und -aktualität. Außerdem erfordern solche Einschränkungen die Speicherung zusätzlicher Daten. Verdichtungsebenen werden verwendet, um die Geschwindigkeit des Datenzugriffs zu verbessern, z.B. bei Reporting und Analyse. Diese Datenredundanz wiederum erfordert einen hohen Aufwand an Zeit und Ressourcen, um Datenkonsistenz zu gewährleisten Gleichzeitig wird eine zeitnahe Datenverfügbarkeit eingeschränkt. Neueste Ankündigungen versprechen auf In-Memory Datenbanken (IMDB) basierende Anwendungen, die auf größte Datenbestände – ohne zusätzliche Verdichtungsebenen – performant zugreifen können [2,3].

23rd GI-Workshop on Foundations of Databases (Grundlagen von Datenbanken), 31.05.2011-03.06.2011, Obergurgl, Austria.

Die Verbesserung der Datenzugriffsgeschwindigkeit ist oftmals der Hauptgrund zusätzlicher Datenhaltung. Setzt man voraus, dass dies in einer IMDB weniger wichtig ist, so kommt die Frage auf: Wieviel Persistenz, d.h. nicht-flüchtige Datenspeicherung, ist in IMDB-basierten DWS überhaupt noch notwendig? Dies gilt insbesondere für Enterprise Data Warehouses (EDW). Ist es möglich, jede Art von Analyseanfrage direkt auf dem Rohdatenbestand abzusetzen, welcher „on-the-fly“ transformiert wird? Oder gibt es dennoch gewichtige Gründe der Datenspeicherung? Um diese Fragen zu beantworten, erläutern wir Persistenzgründe in EDW-Systemen und potentielle Konflikte zwischen Datenspeicherung und -verwendung. Zudem definieren wir Indikatoren zur Entscheidungsunterstüzung, ob Daten gespeichert werden sollen oder nicht.

Abschnitt 2 erläutert einführend die Besonderheiten von Enterprise Data Warehouses und einer Schichtenarchitektur. In Abschnitt 3 führen wir Gründe der Datenpersistenz in heutigen EDW auf und beschreiben mögliche Konflikte, welche aufgrund der Anforderungen der Datennutzung und der hierfür notwendigen Aufwände entstehen. Abschnitt 4 erläutert Datenpersistenz auf IMDB-basierten EDW sowie entscheidungsunterstützende Faktoren. Abschnitt 5 fasst die Teile zusammen und gibt einem Ausblick auf zukünftige Arbeiten.

2. EINE SCHICHTENARCHITEKTUR FÜR ENTERPRISE DATA WAREHOUSES
Ein Enterprise Data Warehouse ist ein Business Data Warehouse [4], stellt also entscheidungsunterstützende Informationen für das Management in allen Geschäftsbereichen zur Verfügung. Darüber hinaus stellen EDW eine wichtige Datenbasis für eine Vielzahl von Anwendungen dar, wie zum Beispiel Business Intelligence (BI), Customer Relationship Management (CRM) und die Planung. Innerhalb einer umfassenden Systemlandschaft stellen EDW-Systeme die „Single Source of Truth“ (vgl. [3]) für alle analyse-relevanten Daten des Unternehmens dar. Das heißt, sie ermöglichen eine allgemein gültige Sicht auf einen zentralen, harmonisierten, validen und konsistenten Datenbestand. Ein EDW integriert sehr große Datenbestände aus einer Vielzahl unterschiedlicher Quellsysteme des Konzerns – oftmals weltweit, so dass Daten verschiedener Zeitzonen zusammengeführt werden müssen. Dies erfordert eine fortlaufende Datenverfügbarkeit mit gleichzeitigem Datenladen und -zugriff. Zudem gibt es weitere Anforderungen an den Datenbestand: Ad-hoc-Berichte, „near-real-time“ Verfügbarkeit und Anwendungen, wie beispielsweise CRM, mit einem Bedarf an detaillierten historischen Daten. Ein sich ändernder Informationsbedarf muss schnell und flexibel

gedeckt werden können. Zudem wird ein umfassendes Berechtigungskonzept zur Sicherung sensibler Daten vorausgesetzt. Somit sind verschiedene Gründe von Datenpersistenz spezifisch in einem EDW.

Persistenz in einem Data Warehouse ist eng verbunden mit dessen Architektur. Eine allgemeine Referenzarchitektur (vgl. z.B. [5,6,7,8]) definiert drei Bereiche, welche die drei Arten der Datenverarbeitung darstellen: Datenbeschaffung in der „Staging Area", Datenbearbeitung in der Basisdatenbank, Datenbereitstellung im Data-Mart-Bereich. In diesem eher groben Modell ist Datenspeicherung in jedem Bereich implizit [9]. Die in [10] vorgestellte Schichtenarchitektur (Abb. 1) entwickelt diesen Ansatz hinsichtlich der bereits erwähnten Anforderungen an ein EDW weiter. Die Schichten werden zweckbestimmter; jede der fünf Schichten repräsentiert einen Bereich, in dem der Wert der Daten hinsichtlich ihrer Verwendung gesteigert wird, wenn dies notwendig ist. Eine Schicht bedeutet aber nicht zwangsläufig Datenspeicherung. Wird beispielsweise ein Datenbestand nach der Harmonisierung gespeichert und ist bereits für Analysezwecke verwendbar, so muss er nicht auf die oberste Schicht „durchgereicht" und dort nochmals gespeichert werden. In einem ersten Schritt muss entschieden werden, in welchem Format die Daten wo zu speichern sind. Deshalb ist zunächst der Zweck der Datenverwendung als Grund der Datenspeicherung zu ermitteln.

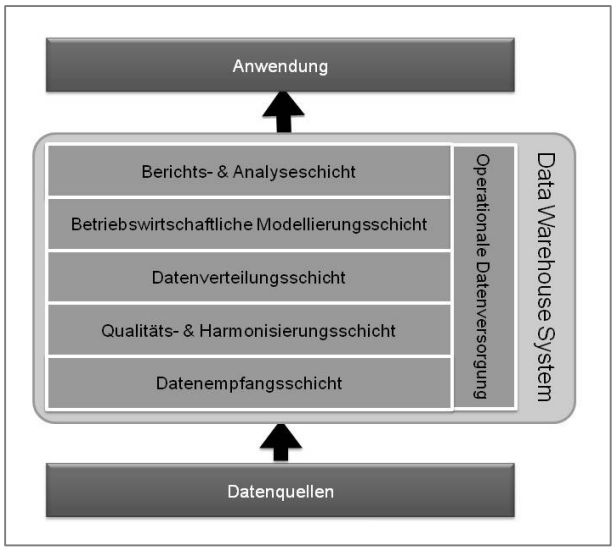

Abb. 1. Schichtenarchitektur für EDW (nach [10])

Die in Abb. 1 dargestellte Schichtenarchitektur für EDW unterteilt sich in die folgenden Bereiche:
Die *Datenempfangsschicht* stellt den „Posteingang" des EDW dar; extrahierte Daten werden ohne oder mit geringen Modifikationen entgegengenommen und abgelegt.
Innerhalb der *Qualitäts- & Harmonisierungschicht* werden die Daten technisch und semantisch integriert. Das beinhaltet Dublettenerkennung, Aspekte der Informationsintegration (vgl. [11]) etc., und entspricht der Transformation des ETL-Prozesses.
Die *Datenverteilungsschicht* enthält harmonisierte und integrierte Unternehmensdaten ohne betriebswirtschaftliche Logik und bildet somit die einheitliche Datenbasis für alle Anwendungen.
In der *Betriebswirtschaftlichen Modellierungsschicht* werden Daten hinsichtlich der Geschäftsanforderungen transformiert; zum Beispiel werden Finanz- mit Logistikdaten verknüpft.
In der *Berichts- & Analyseschicht* werden Daten hauptsächlich verwendungsbezogen transformiert, um performante Zugriffe zum Beispiel beim Reporting oder der Analyse zu gewährleisten.
Innerhalb der *Operationalen Datenversorgung* werden Daten für sehr spezielle Anwendungsfälle und Anforderungen zur Verfügung gestellt, zum Beispiel bei „near real-time Reporting".

Obwohl die Grenzen fliessend sind, können die fünf Schichten den drei Bereichen der Datenverarbeitung wie folgt zugeordnet werden: Datenbeschaffung in der Datenempfangs- und der Qualitäts- & Harmonisierungsschicht, Datenbearbeitung in der Datenverteilungs- und der Betriebswirtschaftlichen Modellierungsschicht, sowie Datenbereitstellung in den Data Marts der Berichts- & Analyseschicht.

3. GRÜNDE FÜR DATENPERSISTENZ

Zwei Gründe von Datenpersistenz im Data Warehouse werden hauptsächlich genannt: Speicherung der bereits transformierten Daten in der Basisdatenbank und Speicherung redundanter, aggregierter Daten im Data-Mart-Bereich. Darüber hinaus gibt es allerdings noch eine Vielzahl von Persistenzgründen im EDW. Hierzu zählen technische Einschränkungen, Governance-Bestimmungen des Unternehmens und Gesetze, Vereinfachung der Datenhandhabung oder ein subjektives Sicherheitsbedürfnis. Sofern uns bekannt, werden Gründe für Datenpersistenz in der Literatur nur selten erwähnt; zudem vermissen wir eine vollständige Auflistung, wie im Folgenden beschrieben.

Entkopplung des Quellsystems: Zur Entlastung des Quellsystems werden Daten direkt nach ihrer erfolgreichen Extraktion im Eingangsbereich des DW gespeichert; hierbei werden die Daten nicht oder nur in geringem Maße verändert (z.B. werden Herkunftsmerkmal oder Zeitstempel angefügt).
Datenverfügbarkeit: Oftmals sind Daten nicht mehr oder nur in einem veränderten Zustand verfügbar; hierzu zählen z.B. Daten aus dem Internet, aus Dateien, welche regelmäßig überschrieben werden, oder aus Altsystemen. Zudem können Netzwerkprobleme dazu führen, dass auf Daten nicht zugegriffen werden kann. Die Speicherung im Warehouse garantiert die Datenverfügbarkeit.
Komplexe Transformationen: Aufgrund ihrer Komplexität sind einige Transformationen sehr zeit- und ressourcenaufwendig, so dass die Daten gespeichert werden, um ein wiederholtes Transformieren zu vermeiden.
Abhängige Transformationen: Unter „abhängige Transformation" verstehen wir solche, deren Durchführung den Zugriff auf weitere Daten erfordert; z.B. erfordert die Verteilung eines Bonus' auf die einzelnen Mitarbeiter die Gesamtanzahl der Mitarbeiter. Diese notwendigen Daten werden im DW gespeichert, um das korrekte Durchlaufen der Transformation zu gewährleisten.
Veränderte Transformationsregeln: Regeln können geändert werden. Besitzen die Daten kein Zeitmerkmal und werden die Transformationen nicht „historisiert", so ist eine identische Transformation nicht mehr möglich.
Aufwendige Datenwiederherstellung: Sind Daten nicht mehr im DWS verfügbar (z.B. weil sie archiviert sind), ist eine Wiederherstellung aufwendig, so dass sie gespeichert werden.
Datenzugriffsgeschwindigkeit: Die redundante Speicherung von Daten in Verdichtungsebenen oder materialisierten Sichten zum Zwecke der Performanzverbesserung beim Datenzugriff stellt einen der häufigsten Gründe für die Einführung einer weiteren Persistenzebene dar.
„En-bloc Datenversorgung": Üblicherweise fliessen neue Daten, aus verschiedenen, gegebenenfalls weltweiten Quellen, zeitlich

verteilt in ein EDW. Nachdem diese syntaktisch und semantisch integiert wurden, werden sie zwischengespeichert und erst zu einem bestimmten Zeitpunkt in die Datenbasis des Warehouse gespielt. Hierdurch wird ein zeitlich definierter, konstanter und in sich plausibler Datenbestand für die darauf aufsetzenden Anwendungen gewährleistet.

Konstante Datenbasis: Einige, auf Daten des DWS aufbauende Applikationen, wie beispielsweise Planung, erfordern eine konstante Datenbasis, welche sich während der Benutzung nicht ändern darf und deswegen separiert gespeichert wird.

„Single Version of Truth“: Transformierte Daten werden nach unternehmensweit gültigen Definitionen, aber ohne spezielle Geschäftslogik gespeichert. Hierdurch wird ein einheitlicher, vergleichbarer Datenbestand geschaffen, auf den die jeweiligen Geschäftsbereiche und Anwendungen zugreifen können [10].

„Corporate Data Memory“: Alle ins EDW extrahierten Daten werden ohne oder nur mit minimaler Veränderung (z.B. durch Anfügen eines Herkunftsmerkmals) gespeichert, um eine größtmögliche Autarkie und Flexibilität von Datenquellen zu ermöglichen. So können Datenbestände (wieder-)hergestellt werden, ohne auf die Quellsysteme zuzugreifen, in denen die Daten möglicherweise schon gelöscht wurden oder nicht mehr zum Zugriff bereitstehen (vgl. [10]).

Komplex-abweichende Daten: Zu integrierende Daten können in Syntax und Semantik sehr von der im EDW üblichen abweichen; eine (zumeist schrittweise) Eingliederung erfolgt erst nach vorheriger Speicherung.

Data-Lineage: Daten in Berichten oder Analysen sind häufig Ergebnis mehrstufiger Transformationsprozesse. Um eine Rückverfolgung zu den Urprungsdaten zu erleichtern oder zu ermöglichen, etwa zur Validierung, können gespeicherte Zwischenergebnisse erforderlich sein (vgl. [12]).

Komplexe Berechtigungen: Anstatt der Definition und Erstellung komplexer Benutzerberechtigungen (z.B. auf Merkmale oder auf Feldinhalte), werden bestimmte Data-Marts mit den Daten erstellt und die Berechtigungen auf dem Data-Mart vergeben.

„Informationsgewährleistung“: Viele EDW haben zu gewährleisten, dass die Daten den Benutzern in einem bestimmten Zeitraum (oftmals sogar 24 Stunden pro Tag) zur Verfügung stehen und für die Anwendungen genutzt werden können. Hierfür werden in der Regel besonders kritische Datenbestände zusätzlich gespeichert.

Corporate Governance: Daten werden gemäß den Compliance-Vorgaben des jeweiligen Unternehmens (Corporate Governance) gespeichert; z.B., um eine aufgrund bestimmter Daten getroffene Managemententscheidung auch im Nachhinein beurteilen zu können.

Gesetze und Bestimmungen: Zudem gibt es auch Gesetze und Bestimmungen, die eine Datenspeicherung begründen; für Deutschland existieren solche beispielsweise im Finanzbereich (Handelsgesetzbuch u.a., [13]) und bei der Produkthaftung [14].

Subjektive Sicherheit: Letzlich kann das sujektive Bedürfnis an Sicherheit ein Grund für Datenspeicherung sein.

Persistenz beinhaltet häufig redundante Datenhaltung, da sowohl Quell- als auch transformierte Zieldaten gespeichert werden; ausschließliches Speichern der Zieldaten bedeutet in aller Regel Datenverlust. Hieraus entstehen hohe Anforderungen, nicht nur an die Hardware (Speicherplatz etc.), sondern auch an die Datenpflege, um etwa die Datenbestände konsistent zu halten.

Der Betrieb produktiver DWS führt zwangsläufig zu Konflikten zwischen den Anforderungen der Datennutzung, wie z.B. Performanz bei Reporting und Analyse, und dem Aufwand an Zeit und Ressourcen, die notwendigen Voraussetzungen hierfür zu schaffen. Im Folgenden beschreiben wir diese Anforderungen und ihre Konsequenzen kurz.

Wie bereits erwähnt, stellt ein EDW häufig die Datenbasis für verschiedene Anwendungen dar; das hohe Datenvolumen resultiert aus den unterschiedlichen Anforderungen dieser Applikationen an die Daten. Der Bedarf an Detailinformationen erfordert viele Daten feinster Granularität. Der Bedarf an historischen Informationen erfordert eine lange Historisierung der Daten. Schließlich wird eine große Bandbreite an Daten gesammelt, beispielsweise für Data-Mining-Szenarios. Dieser große Datenbestand muss für seine Verwendung aufbereitet werden; z.B. ist eine gute Berichtsperformanz sicherzustellen.

Eine hohe Geschwindigkeit beim Datenzugriff wird zumeist durch ein reduziertes Datenvolumen erreicht – durch den Aufbau materialisierter Sichten oder Verdichtungsebenen. Ein einfaches Beispiel hierfür ist die Verdichtung tagesgenauer Daten auf Monat, mit einem Faktor von etwa 30. Pflege und Verwaltung solcher Redundanzen erfordert nicht nur Speicherplatz, sondern auch zusätzlichen Aufwand, die Daten aktuell und konsistent zu halten (vgl. [15]). Da diese Aufwände Zeit kosten, ist die Verfügbarkeit der Daten eingeschränkt. Außerdem beschränken die vordefinierten Datenbestände die Flexibilität der Daten hinsichtlich geänderter und neuer Nutzungsanforderungen.

Ein komplexer Staging-Prozess mit mehreren Schichten persistenter Daten ist einer schnellen Datenverfügbarkeit gegensätzlich. Dies ist insbesondere auch bei Konzepten für „Near-Realtime Reporting“ zu beachten [16].

4. PERSISTENZ BEI IN-MEMORY

EDW-Architekturen basieren gewöhnlich auf relationalen Datenbanken (RDBMS) mit Stern-, Snowflake- oder Galaxy-Schema als Grundlage der Datenmodellierung; siehe z.B. [15,17]. Solche Modelle ermöglichen gute Performanz bei On-line Analytical Processing. Große Datenbestände müssen aber auch hier mittels materialisierter Sichten und Verdichtungsebenen reduziert werden – mit den bereits beschriebenen Konsequenzen.

Spaltenbasierte Datenbanken (vgl. [18,19]) werden aufgrund ihrer Vorteile bei der Datenkomprimierung und dem Lesezugriff [20,21] im Data Warehousing genutzt (z.B. [22,23]). Seit einigen Jahren wird spaltenbasierte In-Memory-Technologie in kommerziellen Data Warehouse Produkten verwendet (z.B. „SAP NetWeaver® Business Warehouse Accelerator“ [24,25], „ParAccel Analytic Database™“ [26]), um verbesserte Antwortzeiten beim Zugriff auf sehr große Datenbeständen zu erzielen. Solche Technologien erlauben das Laden und Abfragen von Datenvolumina im Teradatenbereich mit guter Performanz. Es wurden bereits Installationen angekündigt, die On-Line Transactional und Analytical Processing in einem System mit bis zu 50 TB Daten im Hauptspeicher ermöglichen [27]. In diesem Bereich ist SanssouciDB als ein erstes Produkt zu nennen [3].

Diese technologischen Veränderungen führen zu der Frage, in welchem Maße Datenpersistenz in IMDB-basierten EDW-Systemen noch notwendig ist. Es wird suggeriert, dass bei In-Memory-Technologie keine Daten zusätzlich zu den gespeicherten Urspungsdaten persistent gehalten werden müssen. Alle abgeleiteten Daten, insbesondere die für Analysezwecke aggregierten oder verdichteten, werden „on-the-fly“ ermittelt und zur Verfügung gestellt [3,27]. Dies gilt jedoch nur für einige der o.g. Gründe, wie im folgenden deutlich wird. In diesem Zusammenhang fokussieren wir uns auf Datenbanken, die ACID-fähig sind, inklusive Dauerhaftigkeit (z.B. SanssouciDB, solidDB

von IBM und TimesTen von Oracle [3,28,29]). Persistenz ist hier zu unterscheiden von volatiler Speicherung, bei der die Daten in flüchtigem Speicher gehalten werden und verloren gehen, wenn das System heruntergefahren wird oder abstürzt.

4.1 Notwendigkeit der Datenpersistenz

Eine Entscheidung für Datenpersistenz kann nicht ausschließlich nach einem kostenbasierten Vergleich von „Plattenplatz und Kosten des Updates versus Geschwindigkeitsgewinn der Analyse" getroffen werden. Zunächst ist der Grund der Datenspeicherung (s. Abschnitt 3) zu berücksichtigen. Im RDBMS-basierten DWS ist diese Überlegung weniger ausgeprägt, da die geringere Leistungsfähigkeit der Datenbank und der daraus resultierende Bedarf an aggregierten Daten das Speichern begründet. Um die Notwendigkeit von Datenpersistenz zu ermitteln, führen wir eine Einteilung dieser Gründe ein: die Speicherung der Daten ist nur unterstützend, essentiell oder sogar verpflichtend.

Tab. 1. Persistenzgründe, nach Notwendigkeit gruppiert

Grund/Zweck	Notwendigk.	Gruppierung
Gesetze und Bestimmungen	Verpflichtend	-
Corporate Governance	Verpflichtend	-
Datenverfügbarkeit	Verpflichtend	-
Veränderte Transformationsregeln	Verpflichtend	-
Abhängige Transformationen	Verpflichtend	-
Quellsystem-Entkopplung	Essentiell	Aufwand
Aufwendige Datenwiederherstellung	Essentiell	Aufwand
Komplexe Transformationen	Essentiell	Aufwand
Konstante Datenbasis	Essentiell	Aufwand
"En-bloc Datenversorgung"	Essentiell	Vereinfachung
Komplex-abweichende Daten	Essentiell	Vereinfachung
Data-Lineage	Essentiell	Vereinfachung
Komplexe Berechtigungen	Essentiell	Vereinfachung
„Single Version of Truth"	Essentiell	Design
„Corporate Data Memory"	Essentiell	Design
„Informationsgewähr"	Essentiell	Sicherheit
Zugriffsgeschwindigkeit	Essentiell	Performanz
Subjektive Sicherheit	Unterstützend	-

Verpflichtend zu speichern sind Daten aufgrund von Gesetzen und Bestimmungen sowie Regeln der Corporate Governance. Zudem gilt dies für Daten, welche nicht wieder hergestellt werden können, weil sie nicht mehr oder nur verändert zur Verfügung stehen oder aufgrund geänderter Transformation nicht mehr erstellt werden können. Auch Daten, die bei der Transformation anderer Daten benötigt werden, sind zu speichern, wenn eine gleichzeitige Verfügbarkeit nicht gewährleistet werden kann.
Essentielle Datenpersistenz kann in bestimmte Gruppen unterteilt werden: Zum einen Daten, deren Wiederherstellung nur mit sehr hohem *Aufwand* (an Zeit und Ressourcen) möglich ist, wie z.B. archivierte oder komplex transformierte Daten. Hierbei ist „sehr hoch" allerdings subjektiv und näher zu untersuchen. Eine zweite Gruppe sind Daten, die gespeichert werden, um den Betrieb des Warehouse oder einzelner Anwendungen zu *vereinfachen*; hierzu zählen speziell abgelegte Plandaten oder Data-Marts mit Berechtigungen für besondere Benutzer. Drittens begründet sich Persistenz mit spezieller Konzeption (*Design*) des EDW: „Single Version of Truth", „Corporate Data Memory" zählen u.a. hierzu. *Sicherheit*, etwa zur Gewährleistung der Datenverfügbarkeit, stellt eine weitere Gruppe dar. Letztlich ist Datenspeicherung für eine hohe *Performanz* ein Grund; oftmals der, dem das größte redundante Datenvolumen zugrunde liegt.
Daten, deren Speicherung *unterstützend* ist, beinhalten solche, die wegen subjektiver Sicherheitsüberlegungen abgelegt werden.
Eine komplette Auflistung der Persistenzgründe, gruppiert nach Notwendigkeiten, zeigt Tab. 1.

Abb. 2 zeigt ein vereinfachtes Entscheidungsdiagramm für Datenpersistenz, in dem z.B. unscharfe Begriffe wie „aufwendig", „komplex" und „häufig" abhängig von der Domäne spezifiziert werden müssen. Die ersten drei Abfragen betreffen verpflichtende Gründe, d.h. die Daten sind – auch in IMDB-basierten EDW – zu speichern. Bei den aus anderen Gründen gespeicherten Daten sind die Entscheidungsgrundlagen sehr vielfältig. Stellen die Daten eine „Single Version of Truth" dar oder umfasst das EDW-Design ein „Corporate Data Memory", so sind diese Daten zu speichern. Ist hingegen eine komplexe Reproduktion oder Transformation Grund des Speicherns, so müssen z.B. Zugriffshäufigkeit und Sicherstellung der Verfügbarkeit in Betracht gezogen werden, um entscheiden zu können.

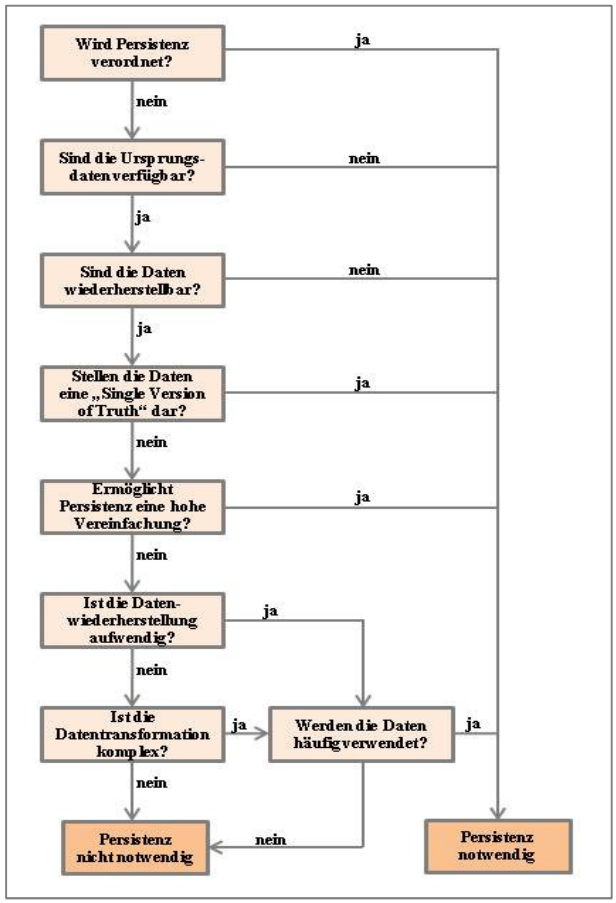

Abb. 2. Entscheidungsdiagramm „Datenpersistenz"

4.2 Bewertung der Persistenz in IMDBs

Alle nicht-verpflichtend gespeicherten Daten sind Gegenstand der Betrachtung bei der Frage nach Persistenz in IMDB-basierten EDW. Insbesondere betrifft dies Daten, die zur Verbesserung der Zugriffsperformanz oder aufgrund komplexer Transformation redundant abgelegt werden. Das bedeutet aber nicht, dass allein die Geschwindigkeit der Datenverarbeitung in einem solchen System jede Art zusätzlicher Speicherung überflüssig machen wird. Dies gilt beispielsweise für die „En-bloc Datenversorgung“ oder beim Aufbau einer konstanten Datenbasis für Planungsläufe. IMDB-Snapshot-Mechanismen, wie in [30] erläutert, halten den Datenbestand zumeist nicht über die benötigte Zeit von Stunden oder Tagen konstant. Hier kommt es nicht auf eine schnelle Versorgung mit neuen Daten an, sondern auf die Herstellung eines über einen definierten Zeitraum unveränderten Datenbestands. Zeitstempelverfahren in In-Memory-Konzepten [3,30] können ein Lösungsszenario sein. Für die Ersetzung eines „Corporate Data Memory“ jedoch sind diese Verfahren nicht geeignet, wenn Daten verschiedener Quellsysteme integriert werden, was insbesondere für ein EDW gilt. Auch werden Persistenzgründe wie komplexe Berechtigungen oder Data-Lineage weiterhin gültig bleiben.

Die Erfahrung zeigt, dass technische Beschränkungen meist früher als erwartet eintreten, so dass die Systemressourcen für die an sie gestellten Aufgaben nicht mehr ausreichen werden. Die Möglichkeit, auf sehr viele Daten mit sehr hoher Performanz zuzugreifen, wird neue Bedürfnisse wecken. Es werden neue Anforderungen aufkommen und die Datenmengen zunehmen. Aufgrund dessen ist auch bei IMDB-basierten Systemen zu betrachten, ob wiederholte, gleichartige Zugriffe und Bearbeitung von Daten „on-the-fly“ nicht durch Vorhalten der Daten im benötigten Format günstiger ist. Dies gilt insbesondere für Datenbestände, auf die häufig zugegriffen wird und die sich nicht oder nur wenig ändern, wie beispielsweise die geschlossenen Jahres-, Quartals- oder Monatsabschlüsse der Finanzbuchhaltung.

Eine weitere Frage in diesem Zusammenhang ist das Datenformat, in dem gespeichert wird, d.h. auf welcher Transformationsstufe die Speicherung optimal ist. Hierbei ist das Format zu ermitteln, welches eine möglichst flexible Verwendung der Daten bei einer größtmöglichen Vermeidung wiederholter, gleichartiger Transformationen darstellt. Dies kann durch kostenbasierte Laufzeitmessungen geschehen, wie folgendes Beispiel erläutert:

Gegeben sei ein Rohdatenbestand (R), der über eine mehrstufige Transformation (Tn; n={1,2,3}) für Analysen (A) abgefragt wird. Zu vergleichen ist, ob es effizienter ist, die Daten nach den einzelnen Transformationen persistent zu speichern (P), sie volatil zu halten (V), oder sie jeweils „on-the-fly” neu zu ermitteln:

(1) R ➔ T1 + P ➔ T2 + P ➔ T3 + A
(2) R ➔ T1 + P ➔ T2 + V ➔ T3 + A
(3) R ➔ T1 + P ➔ T2 ➔ T3 + A
(4) R ➔ T1 + V ➔ T2 ➔ T3 + A
(5) R ➔ T1 ➔ T2 ➔ T3 + A

Weitere Indikatoren, die hier betrachtet werden müssen, sind:

Datenvolumen: Ist die Datenmenge so groß, dass die zur Nutzung notwendige, oftmals sehr komplexe Aufbereitung überhaupt bzw. in einer akzeptablen Zeit „on-the-fly“ durchgeführt werden kann?

Häufigkeit der Datennutzung: Wird auf die Daten so häufig zugegriffen, dass der Nutzen einer zusätzlichen Materialisierung deren Kosten aufwiegt?

Häufigkeit von Datenänderungen: Wird ein Datenbestand so oft geändert (durch Update, Insert, Delete), dass der Aufwand, z.B. für die Konsistenzsicherung der abgeleiteten Verdichtungsebenen, geringer ist als der Geschwindigkeitsgewinn der Anwendung? Und: Wie aufwendig sind diese Änderungen?

Untersuchungen dieser Art sind auch bei RDBMS-basierten DWS valide. Hierbei lässt die Leistungsfähigkeit einer IMDB jedoch als Ergebnis erwarten, dass Transformationen eher „on-the-fly“ als mit redundanter Persistenz durchgeführt werden.

Einige IMDB ermöglichen die Festlegung unterschiedlicher Kriterien zur Dauerhaftigkeit, z.B. durch Definition temporärer Tabellen [29,30]. Hierdurch können Daten, die nicht verpflichtend zu speichern sind, nur in flüchtigem Speicher gehalten werden. Da ein Herunterfahren oder Absturz der Datenbank relativ selten geschieht, sind die Wartungskosten für solche Daten gering. Ein beispielhafter Anwendungsfall hierfür ist die Ermittlung von RFM-Attributen (Recency, Frequency, Monetary) zur Kundenkategorisierung im CRM-Umfeld [31]. Die Emittlung (s. Abb. 3) basiert auf Kundenstamm- und Transaktionsdaten (Kassenbons, Aufträge, Fakturen) und umfasst Selektionen, Kalkulationen, Währungsumrechnungen, Look-Ups zu komplexen Steuerungsdaten etc. Die berechneten Attribute werden zeitnah aktualisiert benötigt, sowohl im DWS, als auch im CRM-System. Zu berücksichtigen ist, dass es sich hierbei um oft sehr große Mengen an Daten handelt, mehrere Millionen Kunden mit jeweils einer zweistelligen Anzahl Transaktionen. Diese Datenbestände ändern sich häufig, so dass auch die RFM-Attribute laufend aktualisiert werden müssen. Da die Ermittlung der Attribute reproduzierbar ist, kann das Vorhalten dieser Daten ausschließlich im flüchtigen Speicher einer Persistierung vorzuziehen sein.

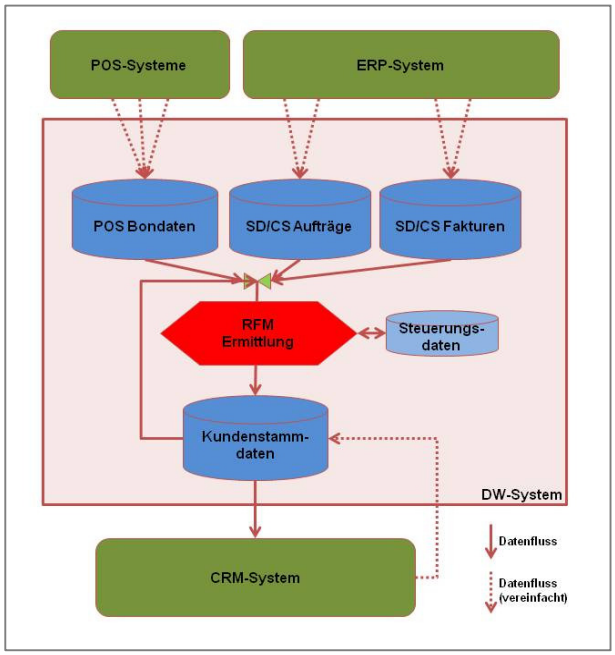

Abb. 3. Ermittlung von RFM-Attributen

Festzuhalten bleibt, dass in einem IMDB-basierten EDW viele Daten nicht mehr gespeichert werden müssen, die in einem RDBMS-basierten aufgrund von Performanzgewinn redundant zu halten sind. Höhere Zugriffsgeschwindigkeiten werden es ermöglichen, Daten „on-the-fly“ für die Nutzung aufzubereiten, insbesondere solche mit relativ einfacher Transformationslogik, wie z.B. Aggregation, Joins etc. Eine Vielzahl materialisierter Sichten wird zu virtuellen Sichten.

5. FAZIT UND AUSBLICK

Enterprise Data Warehouses sind komplexe Systeme mit speziellen Anforderungen an Datenbestand und Datenhaltung, für die eine Architektur dedizierter, zweckbestimmter Schichten geeignet ist. Die Notwendigkeit von Datenpersistenz in solchen Systemen kann nur durch den Zweck der Daten begründet werden. Diese Sichtweise wird bei IMDB-basierten EDW noch entscheidender. Wir beschreiben Gründe der Datenpersistenz und unterteilen sie in verpflichtende, essentielle und unterstützende. Darauf aufbauend nähern wir uns der Entscheidungsfindung, ob Daten in solchen Systemen gespeichert werden.

Persistente Datenhaltung wird es auch in EDW-Systemen auf IMDB geben. Ein großer Anteil heutiger persistierter Daten wird allerdings nur flüchtig gespeichert oder „on-the-fly" berechnet. Zudem wird die Frage aufkommen nach dem Format, in dem die Daten abgelegt werden. Die Antwort hierauf wird nicht einfach zu ermitteln sein; es handelt sich hierbei vielmehr um eine multidimensionale Gewichtung verschiedener Faktoren, wie: Aufwand für Transformation, Speicherung und Updating, Anzahl und Zeit von Datenabfrage und -aktualisierung.

Zukünftige Arbeiten werden eine detaillierte Aufstellung von Persistenzgründen mit ausführlichen Beispielen umfassen. Darüber hinaus werden Indikatoren definiert und beschrieben werden, die die Entscheidungsfindung für/gegen Datenpersistenz unterstützen. Dies umfasst sowohl messbare, wie beispielsweise Vergleiche von Laufzeiten und Wartungsaufwänden zwischen Datenbeständen in verschiedenen Speicherzuständen, als auch nicht-messbare Indikatoren. So wird ermittelt, ob Entscheidungen durch Berechnungen getroffen oder hierdurch zumindest unterstützt werden können.

6. DANKSAGUNG

Diese Arbeit wird teilweise unterstützt vom Bundesministerium für Bildung und Forschung (BMBF) innerhalb des ViERforES-II-Projekts (Nr. 01IM10002B).

7. LITERATUR

[1] R. Winter: „Why Are Data Warehouses Growing So Fast?"; www.b-eye-network.com/print/7188 {03.05.2011}; 2008.

[2] H. Plattner et al.: „ETL-less Zero Redundancy System and Method for Reporting OLTP Data" (US 2009/0240663 A1); US Patent Application Publication; 2009.

[3] H. Plattner, A. Zeier: „In-Memory Data Management"; Springer-Verlag, Berlin; 2011.

[4] B.A. Devlin, P.T. Murphy: „An architecture for a business and information system"; in: IBM Systems Journal 27(1), S.60-80; 1988.

[5] V. Poe: „Building a data warehouse for decision support"; Prentice Hall PTR, Upper Saddle River; 1996.

[6] H. Muksch, W. Behme (Hrsg.): „Das Data Warehouse-Konzept"; Gabler-Verlag, Wiesbaden, 4.Auflage; 2000.

[7] P. Gluchowski; P. Chamoni: „Entwicklungslinien und Architekturkonzepte des On-Line Analytical Processing"; in: Analytische Informationssysteme, Springer-Verlag, 3.Auflage, S.143-176; 2006.

[8] T. Zeh: „Referenzmodell für die Architektur von Data-Warehouse-Systemen (Referenzarchitektur)"; www.tzeh.de/doc/gse-ra.ppt {03.05.2011}; 2008.

[9] B.A. Devlin: „Business Integrated Insight (BI²)"; www.9sight.com/bi2_white_paper.pdf {03.05.2011}; 2009.

[10] SAP: „PDEBW1 - Layered Scalable Architecture (LSA) for BW"; Schulungsunterlagen, SAP AG; 2009.

[11] U. Leser, F. Naumann: „Informationsintegration"; dpunkt-Verlag, Heidelberg; 2007.

[12] Y. Cui, J. Widom: „Lineage Tracing for General Data Warehouse Transformations"; in: The VLDB Journal 12(1), S.41-58; 2003.

[13] §§239,257 HGB (Stand: 01.03.2011); §25a KWG (Stand: 01.03.2011); §147 AO (Stand: 08.12.2010).

[14] §13 ProdHaftG (Stand: 19.07.2002).

[15] W. Lehner: „Datenbanktechnologie für Data-Warehouse-Systeme"; dpunkt-Verlag, Heidelberg; 2003.

[16] J. Langseth: „Real-Time Data Warehouses: Challenges and Solutions"; on: www.dssresources.com {03.05.2011}; 2004.

[17] R. Kimball, M. Ross: „The Data Warehouse Toolkit"; Wiley Publishing Inc., Indianapolis, 2.Auflage; 2002.

[18] G.P. Copeland, S.N. Khoshafian: „A Decomposition Storage Model"; in: SIGMOD`85, S.268-279; 1985.

[19] M.J. Turner et al.: „A DBMS for large statistical databases"; in: 5th VLDB`79, S.319-327; 1979.

[20] D.J. Abadi et al.: „Integrating Compression and Execution in Column-Oriented Database Systems"; in: SIGMOD`06, S.671-682; 2006.

[21] D.J. Abadi: „Query Execution in Column-Oriented Database Systems"; Dissertation, MIT; 2008.

[22] M. Stonebraker et al.: „C-Store: A Column-oriented DBMS"; in: 31st VLDB`05, S.553-564; 2005.

[23] D. Slezak et al.: „Brighthouse: An Analytic Data Warehouse for Ad-hoc Queries"; in: PVLDB 1(2), S.1337-1345; 2008.

[24] T. Legler et al.: „Data Mining with the SAP NetWeaver BI Accelerator"; in: 32nd VLDB`06, S.1059-1068; 2006.

[25] J.A. Ross: „SAP NetWeaver® BI Accelerator"; Galileo Press Inc., Boston; 2009.

[26] ParAccel: „PARACCEL ANALYTIC DATABASE™"; www.paraccel.com/wp-content/uploads/2010/07/PA_DS.pdf {03.05.2011}; 2011.

[27] H. Plattner: „A Common Database Approach for OLTP and OLAP Using an In-Memory Column Database"; in: SIGMOD`09, S. 1-2; 2009.

[28] IBM: IBM solidDB™; www.ibm.com/software/data/soliddb {03.05.2011}; 2010.

[29] Oracle: „Extreme Performance Using Oracle TimesTen In-Memory Database"; www.oracle.com/technetwork/database/timesten/overview/wp-timesten-tech-132016.pdf {03.05.2011}; 2009.

[30] A. Kemper, T. Neumann: „HyPer: Hybrid OLTP&OLAP High PERformance Database System"; www3.in.tum.de/research/projects/HyPer/HyperTechReport.pdf {03.05.2011}; 2010.

[31] J. Stafford: „RFM: A Precursor of Data Mining"; www.b-eye-network.com/view/10256 {03.05.2011}; 2009