

Marcus Heinzl ist BI-Berater bei Inforte Deutschland GmbH, Hamburg, einem Unternehmen der Business & Decision Group.
E-Mail: marcus.heinzl@inforte.com

Data Life Cycle Management

Mit Nearline Storage Daten parat halten

Unter Data Life Cycle Management (DLM) versteht man die Verwaltung von Daten bezogen auf den Zeitraum ihrer Nutzung. Dabei ist das Ziel von DLM, Kosten und Performance eines Anwendungssystems zu optimieren. Typische Phasen des DLM-Zyklus sind die Erfassung, Verteilung, Nutzung, Verwaltung und letztlich die Archivierung von Daten. Ein Nearline Storage (NLS) ist eine Komponente im Rahmen eines DLM. Es ist ein Speichersystem, das im Wesentlichen die Eigenschaften eines Archivsystems erfüllt, also die Entlastung des Online-Systems und die kostengünstige Auslagerung selten genutzter Daten. Ein NLS hält jedoch im Gegensatz zu einem Archivsystem die Daten weiterhin für den Online-Zugriff bereit, so dass die Grenze zwischen online genutzten Datenspeichern und konventionellen Archivsystemen verschwindet. Somit stehen insbesondere für BI-Anwendungen historische Daten, zum Beispiel für Trendanalysen, mit unbegrenzter Historie für einen Online-Zugriff zur Verfügung. NLS bietet damit die Möglichkeit, die Phasen Nutzung, Verwaltung und Archivierung des DLM für ein Enterprise Data Warehouse (EDW) zu integrieren, bei gleichzeitiger Optimierung von Kosten und Performance. Basierend auf Best-Practice-Erfahrungen wird nachfolgend die Funktionsweise einer Nearline-Storage-Lösung beschrieben und exemplarisch am Beispiel von SAND/DNA von SAND Technologies in Verbindung mit SAP BI 7.0 dargestellt.

Relevanz einer Archivierungslösung

Betrieb und Ausbau einer unternehmensweiten EDW-Plattform führen vor allem zu steigenden Betriebskosten aufgrund eines stetigen Wachstums der gespeicherten Datenmenge. Ein Kernaspekt eines EDW ist die Speicherung historischer Daten. Im Gegensatz zu operativen Systemen werden historisierte Daten im EDW oft mit langfristiger Historie aufbewahrt. Neben dem stetigen Wachstum der Datenmenge kommt weiterer Speicherplatzbedarf durch die Anbindung neuer Quellsysteme

und damit verbundenen neuen Anwendungen hinzu. Eine Archivierungslösung, also die Auslagerung selten oder nicht mehr verwendeter Daten auf kostengünstige Speichermedien, ist für viele Unternehmen früher oder später unumgänglich, da das steigende Datenvolumen verschiedene negative Seiteneffekte hat:

- ◆ Die Datenbasis für OLAP-fähige Plattformen (Online Analytical Processing) setzt teure, schnelle Speichersysteme für hoch performante Zugriffe aus analytischen Anwendungen voraus. Aus technischen Gründen kommt üblicherweise eine redundante Datenhaltung im Rahmen einer RAID-Lösung (Redundant Array of Independent Disks) zum Einsatz, die das notwendige Datenvolumen und damit die Kosten pro gespeicherter Information vergrößert.
- ◆ Die Performance der Datenladeprozesse sinkt, da datenbankseitig große Tabellen gelesen und geschrieben werden müssen, wodurch im Allgemeinen die Zugriffperformance sinkt. Die Lese- und Schreibprozesse können zwar durch entsprechende Technologien wie zum Beispiel parallele Zugriffe oder Indizierung beschleunigt werden, dies führt aber auf der anderen Seite zu Nachteilen: Erhöhte Kosten oder Performanceeinbußen beim Schreiben und die damit verbundene Generierung und Pflege von Datenbankindizes.
- ◆ Eine Abnahme der Performance von Lesezugriffen auf sehr großen Datenstrukturen führt letztlich auch dazu, dass die Performance in Reportinganwendungen stark abnimmt und deren Benutzbarkeit negativ beeinflusst. Die Akzeptanz von Anwendungen verringert sich, insbesondere wenn diese zeitkritisch sind.

Anforderungen an eine Archivierungslösung

Aus den genannten Nachteilen großer Datenmengen in Bezug auf Kosten und Performance lassen sich direkt die Anforderungen an eine Archivierungslösung ableiten:

- ◆ Die Kosten pro gespeicherter Informationseinheit müssen deutlich unter den Kosten der Speicherung im OLAP-System liegen. Dazu muss die Archivierungslösung hohe Kompressionsraten erreichen, um die Kosten pro archivierter Informationseinheit zu senken.
- ◆ Das OLAP-System muss in Bezug auf die beschriebenen Nachteile durch große Datenmengen entlastet werden.

Archivierungsproblematik im EDW

In den Aufbau eines EDW wird viel Geld und Zeit investiert, um eine integrierte Unternehmensplattform mit allen Daten zu erhalten, die sowohl aktuelle, als auch historische Daten zur Verfügung stellt. Dies steht im Widerspruch zur Archivierung, die schließlich Teile der Daten

dem aktiven Zugriff entzieht. Somit ergibt sich für den Einsatz einer Archivierungslösung die entscheidende Frage, welche Daten eigentlich archiviert werden sollen und wie auf diese Daten im Bedarfsfall wieder zugegriffen werden kann.

Gerade historische Daten werden üblicherweise nur sehr selten oder gar nicht mehr in Berichten oder zum Nachlesen benutzt. Es wäre somit konsequent, diese nicht mehr benutzten Daten aus dem EDW zu entfernen und zu archivieren. Manche Daten müssen jedoch gegebenenfalls von Zeit zu Zeit zur Verfügung stehen, auch wenn es sich dabei nur um seltene Zugriffe handelt. Dementsprechend sollten genau diese Daten auf keinen Fall aus dem Zugriff von Anwendungen entfernt werden, sondern auch nach einer Archivierung über ein entsprechendes automatisches Zugriffsverfahren zur Verfügung stehen.

Herkömmlichen Archivierungslösungen im Rahmen des DLM für EDWs sind an dieser Stelle Grenzen gesetzt, da nie genau bestimmt werden kann, welche Daten archiviert werden sollen und welche nicht. Dies führt zu der Idee eines so genannten NLS, das die bereits beschriebenen Anforderungen an eine Archivierungslösung durch Auslagerung der Daten aus dem operativen System erfüllt und dennoch einen OLAP-Zugriff auf die Daten zulässt. Für Reportinganwendungen führt dies letztlich zu einer integrierten Betrachtung der Phasen Nutzung, Verwaltung und Archivierung im DLM. Im Folgenden wird zunächst beschrieben, welche verschiedenen Datenzugriffe in einem EDW üblich sind, um darauf aufbauend die Idee eines NLS zu präzisieren.

Datenzugriffe im EDW

In EDW-Architekturen und darauf basierenden Reporting-Anwendungen gibt es, unabhängig vom Hersteller der zugrunde liegenden Systemplattform, typische Arten von Datenzugriffen, die potenziell für das Nachlesen archivierter Daten in Frage kommen.

- ◆ Zunächst gibt es natürlich OLAP-Zugriffe, die die Objekte direkt im EDW nutzen. Sämtliche ausgeführten Berichte basieren zum Beispiel auf OLAP-Zugriffen (oder einer vergleichbaren Technologie).
- ◆ Weiterhin ist es innerhalb von Datenladeprozessen üblich, dass Datenstrukturen denormalisiert und um zusätzliche Information angereichert werden, damit später eine hohe Reporting-Performance auf entsprechend vorbereiteten Daten erreicht wird. Ansonsten müssten die Daten zur Ausführungszeit eines Berichtes miteinander verknüpft werden, was viel Zeit beansprucht. Für die Anreicherung von Datensätzen werden in der

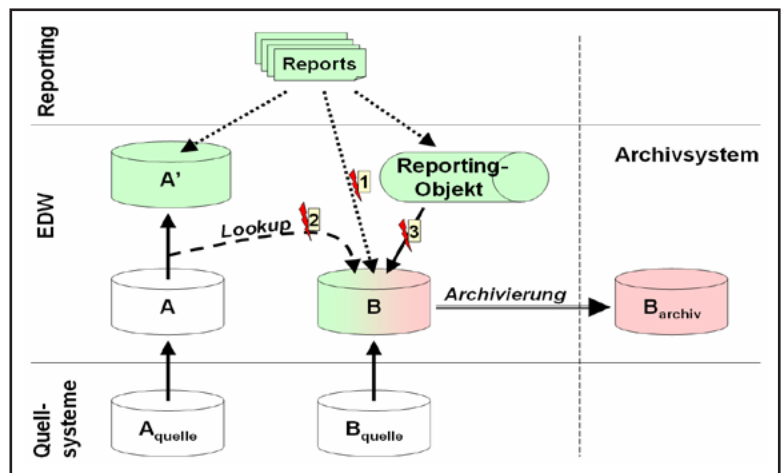


Abb. 1: Zugriffsarten in EDWs

Regel Daten aus anderen Datenspeichern nachgelesen, man spricht hier von einem *Lookup*. Speziell bei zeitabhängigen Daten kann es notwendig sein, historische Daten nachzulesen, um eine historisch korrekte Anreicherung zu erzielen.

- ◆ Letztlich gibt es in der Regel noch andere direkte oder indirekte Zugriffe auf EDW-Objekte, wie zum Beispiel simple Datenbankviews oder aber auch komplexe Analyseprozesse, die ihrerseits von analytischen Anwendungen benutzt werden können.

Abbildung 1 veranschaulicht die unterschiedlichen Zugriffsarten in einem EDW. Allen Zugriffsarten ist gemeinsam, dass bei einem herkömmlichen Archivierungsverfahren Daten (rot dargestellt) dem aktiven Zugriff entzogen werden.

NLS-Architektur

Die Grundidee für eine Architektur zugreifbarer und somit integrierter Archivdaten liegt auf der Hand: Selektionsanfragen, die partiell auch archivierte Daten betreffen, müssen parallel an das EDW und das Archiv verteilt werden. Bezogen auf das grafisch illustrierte Beispiel muss die Ergebnismenge einer Datenanfrage auf das Datenobjekt B die Vereinigungsmenge aus den Datenmengen aus B und B_{archiv} sein. Dafür ist eine Schnittstelle notwendig, die in der Lage ist, eine Datenanfrage passend auf die Datenobjekte B und B_{archiv} zu verteilen, d.h. die Anfrage muss in zwei Anfragen aufgeteilt werden, deren Ergebnismengen anschließend wieder zusammengeführt werden. Eine passende Verteilung ergibt sich durch ein Zeitmerkmal, an dem festgemacht wird, welche Daten archiviert werden. Dazu wird in Bezug auf das gegenwärtige Datum ein Intervallabstand definiert. Ist das Zeitmerkmal zum Beispiel der Kalendertag und beträgt der Intervallabstand sechs Monate, dann werden mit dem nächsten Archivierungslauf alle Daten, die – bezogen auf das Datum an dem der Archivierungsvorgang durchge-

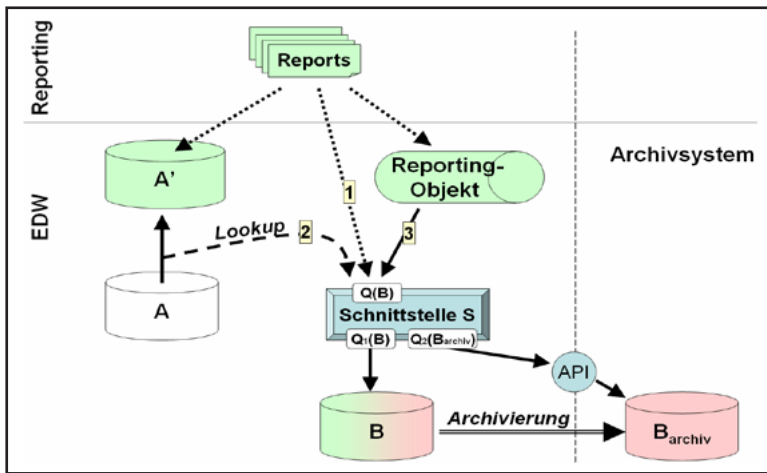


Abb. 2: Integration von Archivdaten

führt wird – älter als sechs Monate sind, archiviert. Bezüglich des Datums des letzten Archivierungslaufs ist für die Schnittstelle somit eindeutig abgegrenzt, welche Daten aus dem Archiv gelesen werden müssen und welche online verfügbar sind.

Abbildung 2 veranschaulicht die Zugriffsschicht einer integrierenden Schnittstelle, die unmittelbar oberhalb des gelesenen Datenobjektes platziert ist und somit den verschiedenen Zugriffsarten zentral zur Verfügung steht. $Q(B)$ stellt dabei eine Datenanfrage auf das Datenobjekt B dar, die auf die Anfragen $Q_1(B)$ und $Q_2(B_{archiv})$ verteilt wird.

Die in der Regel begrenzte Menge von Zugriffsmechanismen in einem EDW muss um die Benutzung der Schnittstelle erweitert werden, um einen integrierten Archivzugriff zu ermöglichen. Neben den einleitend beschriebenen Anforderungen an ein Archivierungssystem ergibt sich für NLS-Systeme die zusätzliche Anforderung, dass das Archivierungssystem vom BI-System aus über eine Anfrageschnittstelle (API) online erreichbar sein muss, bei geringer Zugriffszeit- und hoher Leseperformance. Ansonsten wären Reporting-Zugriffe (OLAP), die auf Benutzerinteraktion basieren, nicht mehr sinnvoll durchführbar.

Die Anforderungen an eine NLS-Lösung, hohe Kompression bei gleichzeitig hoher Leseperformance zu ermöglichen, scheinen dabei zunächst im Widerspruch zu stehen, denn sobald Daten komprimiert werden, muss beim Lesen der Daten eine Dekompression stattfinden, die Zeit beansprucht. Durch geschickte Wahl des Archivierungsintervalls kann man diesen Widerspruch jedoch kompensieren, indem man in Bezug auf ein praxisrelevantes Szenario das Intervall so wählt, dass Zugriffe auf die archivierten Daten minimiert werden und die Ausnahme darstellen. Wird zum Beispiel nur auf der aktuellen und der letzten Geschäftsperiode berichtet, dann wären die Zugriffe auf das

NLS-System die Ausnahme, wenn das Intervall so gewählt ist, dass die letzten beiden Perioden nicht archiviert werden. In der Regel wird bei geschickter Wahl des Intervalls also gar nicht auf dem Archiv gelesen, womit die Performance auch nicht beeinträchtigt wird. Die Anforderungen an die Performance einer API für das Archivierungssystem sind daher längst nicht so restriktiv wie die Anforderungen an die Performance einer OLAP-Engine. Der Widerspruch aus hoher Kompression und hoher Performance verschiebt sich in einen akzeptablen Toleranzbereich.

NLS am Beispiel von SAP BI 7.0 mit SAND DNA

Mit SAP BI 7.0 unterstützt SAP neue Funktionalitäten für Data Life Cycle Management durch die Integration von NLS-Funktionalität. In vorherigen Releases konnten Daten nur vollständig archiviert werden, mit der Konsequenz, dass die Daten für Online-Zugriffe durch Queries oder programmierte Lookups nicht mehr verfügbar waren. Mit der Integration der NLS-Funktionalität wurde dies geändert. SAP selbst liefert dabei nicht das Archivierungssystem, sondern hat lediglich eine API spezifiziert und in SAP BI 7.0 integriert. Die API unterstützt alle notwendigen Schnittstellen für Lese- und Schreibzugriffe auf dem Archiv. Prinzipiell kann somit eine beliebige Archivierungslösung als NLS dienen, solange die SAP-API auf der Gegenseite entsprechend unterstützt wird. Ein Beispiel für eine solche Archivierungslösung ist das dateibasierte Archivsystem SAND DNA von SAND Technologies.

Über die API können innerhalb von SAP BI 7.0 die Datenobjekte *InfoCubes* und *DataStore-Objekte* archiviert

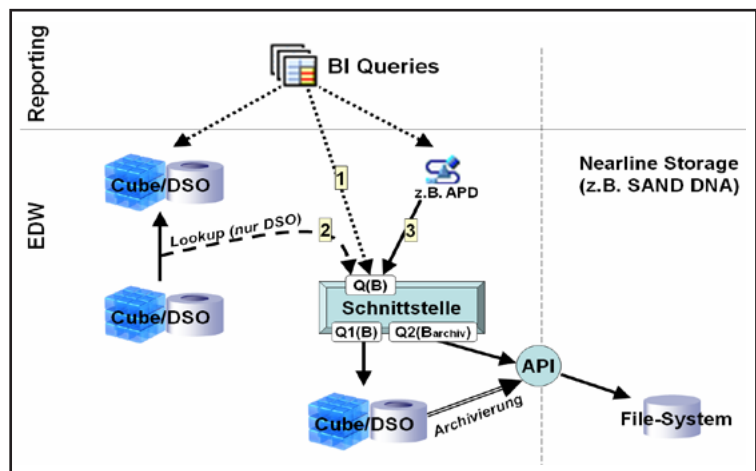


Abb. 3: NLS in SAP BI in Netweaver 2004

werden. Damit ergibt sich im Rahmen von SAP BI 7.0 die in Abbildung 3 dargestellte Architektur. Nachfolgend wird exemplarisch für ein SAP-DataStore-Objekt die Umsetzung der NLS-Funktionalität in SAP BI 7.0 erläutert.

Implementierungsumfeld eines SAP-DataStore-Objekts

In SAP BI gibt es vielfältige Möglichkeiten, um auf ein DataStore Objekt (DSO) zuzugreifen. Abbildung 4 stellt die möglichen Zugriffsarten schematisch dar. Grün markierte Zugriffsarten nutzen das NLS automatisch, gelb markierte müssen entsprechend angepasst werden, rot markierte können das NLS nicht nutzen und für die blau markierten Zugriffsarten wird in einem der folgenden Service Packs eine Unterstützung von SAP ausgeliefert. Für einen SAP-InfoCube ist das Umfeld im Prinzip identisch, mit der Ausnahme, dass Lookups auf InfoCubes in der Regel nicht durchgeführt werden und daher auch nicht durch eine NLS Schnittstelle unterstützt werden.

Queries für *OLAP-Reporting*, die auf InfoCubes oder DSOs aufsetzen, unterstützen den Zugriff auf ein NLS, welcher lediglich in den Einstellungen einer Query eingeschaltet werden muss. *MultiProvider* und *InfoSets* verfügen über keinen eigenen Datenspeicher, sondern kapseln den Zugriff auf ein oder mehrere DSOs oder InfoCubes. Queries auf MultiProvidern werden in einem der nächsten Service Packs für SAP BI 7.0 durch einen Zugriff auf das NLS unterstützt. InfoSets werden dagegen nicht unterstützt.

Das *Data Staging* in SAP BI 7.0 erfolgt durch die in SAP BI 7.0 neue Technologie der *Transformation*. Diese

unterstützt automatisch den Zugriff auf das NLS. Die alte Technologie der *Fortschreibungen* wird aus Gründen der Abwärtskompatibilität weiterhin in SAP BI 7.0 angeboten, unterstützt jedoch nicht den Zugriff auf ein NLS. Für programmierte Lookups, die im Rahmen des Data Staging durchgeführt werden, bietet SAP eine ABAP-basierte Schnittstelle an, die eine automatische Zugriffsverteilung auf InfoProvider und zugehöriges NLS vornimmt. Bestehende Lookups müssen dementsprechend angepasst werden.

Sonstige Datenzugriffe werden nur partiell unterstützt. Der SAP Analysis Process Designer (APD) verhält sich je nach Objekttyp unterschiedlich. Ein Knoten vom Typ ABAP kann wie ein herkömmlicher ABAP-Lookup durch Nutzung der ABAP-Schnittstelle erweitert werden. Alle anderen Knoten unterstützen das NLS nicht. *Datenbank-views*, die direkt auf die Tabellen eines DSO zugreifen, unterstützen den Zugriff auf das NLS aus technischen Gründen nicht, da ein Datenbankview ein Objekt der Datenbankschnittstelle ist, die Schnittstelle zum Zugriff auf das NLS aber in der OLAP-Engine liegt.

Einführung der SAP-NLS-Lösung

Der Einführungsprozess gliedert sich in die typischen Phasen der Entwicklung einer Software. Zunächst erfolgt eine *Analysephase*, in der im Kontext von NLS insbesondere das Umfeld von InfoProvidern sowie deren Datenbankeigenschaften untersucht werden. Die Umfeldanalyse lässt Rückschlüsse auf den Impact einer Archivbindung zu. Im Kontext der Datenbankeigenschaften sind insbesondere die Breite und die Anzahl von Datensätzen in Bezug auf eine vorgegebene zeitliche Granularität, bei-

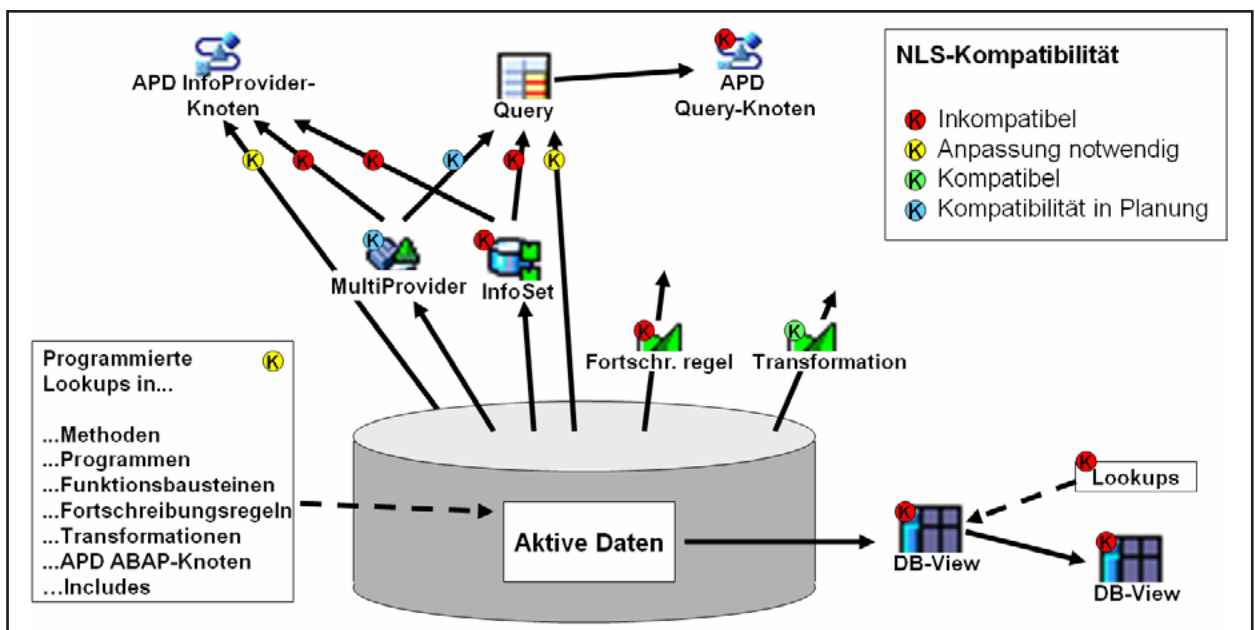


Abb. 4: Zugriffsart eines SAP-DataStore-Objekts

spielsweise einer monatsbasierten Betrachtung, interessant. Dies ermöglicht statistische Rückschlüsse auf die zu archivierende Datenmenge in einem zu archivierenden Zeitraum. Darüber hinaus lässt dies Rückschlüsse auf die Datenmengen zu, die für einen betrachteten Zeitraum effektiv im Archiv ausgelagert werden.

Als nächstes folgt die *Spezifikationsphase* für die notwendigen Anpassungen an bestehenden InfoProvidern sowie für Neuentwicklungen. Die Archivierungsobjekte selbst und die Anpassungen an Objekten im Umfeld eines InfoProviders werden in einem Blueprint beschrieben.

Auf Basis der Spezifikation erfolgen entsprechend die *Implementierungsphase* und eine nachgelagerte *Testphase*. Werden bestehende InfoProvider durch eine Archivianbindung erweitert, so erfolgen üblicherweise zusätzliche *Integrationstests*.

Schließlich erfolgt die *GoLive-Phase* für die Archivierung eines InfoProviders. Basierend auf einem Vorgehensprozess und den in der Analysephase gefundenen Datenbankeigenschaften wird das Vorgehen für die Initialisierung und den stetigen Betrieb eines Archivs ausgearbeitet. Insbesondere die Initialisierung sehr großer Objekte muss unter Umständen speziell behandelt werden, um die Performance des laufenden Reportingbetriebs nicht zu stören.

Begleitend zum Einführungsprozess hat sich die Entwicklung unterstützender Tools bewährt, um den Prozess zu vereinfachen und zu automatisieren. Ein Tool zur *Analyse des Umfelds eines InfoProviders*, das automatisch sämtliche Abhängigkeiten inklusive programmierter Lookups ermittelt, ist ein sehr hilfreiches und Zeit sparendes Mittel in der Analysephase. Ergänzend dazu ist ein Tool zur *automatischen Analyse der Datenbankeigenschaften eines InfoProviders* hilfreich, um später auf das Archivierungsverhalten schließen zu können. Für die Initialisierung der Archivierung eines InfoProviders empfiehlt sich die Entwicklung eines speziellen *Initialisierungsprogramms*, das in SAP-Datenladeprozesse integriert werden kann. Das Programm sollte in der Lage sein, Eigenschaften der Initialisierung auszusteuern wie zum Beispiel das Zeitfenster, in dem archiviert werden darf, oder mit welcher Granularität einzelne Archivierungsrequests durchgeführt werden (Monats- oder Jahreszeitscheiben).

Performanceeigenschaften im SAND/DNA- und SAP-Umfeld

Mit SAND/DNA von SAND Technologies in Verbindung mit SAP BI 7.0 werden aufgrund eines spaltenbasierten Kompressionsmechanismus im SAND-System in der Regel Kompressionsraten weit über 80 Prozent erreicht. Ebenso führt der spaltenbasierte Kompressionsmechanismus zu guten Performancewerten bei der Ausführung von SAP-Queries, die auf das Archiv zugreifen. Durchschnittlich liegt die Ausführungszeit von SAP-Queries, die auf das Archiv zugreifen, bei 2,5-

facher Ausführungszeit im Vergleich zu einer Query ohne Archivzugriff. In Anbetracht der Tatsache, dass bei passend gewähltem Archivierungsintervall Zugriffe auf das Archiv die Ausnahme darstellen, sind im laufenden Betrieb von SAND/DNA in Verbindung mit SAP BI 7.0 trotz Archivierung mit Vollzugriff keine wesentlichen Performanceeinbußen zu erwarten.

Zusammenfassung

Der Entzug von Daten aus dem Online-Zugriff von Reportinganwendungen aufgrund konventioneller Archivierungsverfahren stellt insbesondere in EDW-Architekturen ein essentielles Problem dar. Das Archivieren von Daten steht dem Grundprinzip eines EDW, der nahezu unbegrenzten Datenverfügbarkeit, entgegen. Nearline Storage liefert das Potenzial, diese Einschränkung aufzulösen und EDW-Architekturen um eine nachhaltige Datenverfügbarkeit bei gleichzeitiger Kosten- und Performanceeffizienz zu erweitern. Dies bedeutet, dass die Phasen der Nutzung, Verwaltung und Archivierung im Data Life Cycle Management integriert werden. Mit Nearline Storage wird damit die nächste Stufe der Evolution im Data Life Cycle Management von EDW-Anwendungen erreicht.

Weiterführende Quellen

- [BOS01] Stefan Born et al. Leitfaden zum Thema "Information Lifecycle Management". Bitkom, April 2004. www.bitkom.org/files/documents/BITKOM_Leitfaden_ILM_Stand_21-04-2004.pdf
- [GIM01] Michael Gießelbach. Return-on-Investment durch Information Lifecycle Management. Storage Technologie Corporation, 2005. www.decus.de/slides/sy2003/10_04/3g02.pdf
- [INM01] Bill Inmon. The Role of Nearline Storage in Data Warehouse – Part 1. Publiziert in DM Direct, Oktober 1999. www.dmreview.com/editorial/dmreview/print_action.cfm?articleId=1502
- [INM02] Bill Inmon. The Role of Nearline Storage in Data Warehouse – Part 2. Publiziert in DM Direct, Oktober 1999. www.dmreview.com/editorial/dmreview/print_action.cfm?articleId=1529
- [MRU01] Roland Markowski, Rainer Uhle. Mehr „Ordnung“ im Data Warehouse. www.sand.com, Februar 2007. www.sand.com/downloads/E-3_Feb_07_72dpi.pdf
- [SST01] SearchStorage.com. Definition - Data Life Cycle Management. November 2004. searchstorage.techtarget.com/sDefinition/0,,sid5_gci963642,00.html
- [WIL01] Andreas Wilmsmeier et al. Mastering the SAP Business Information Warehouse, Second Edition. Wiley Publishing Inc., 2006.