6 HANA-optimierte InfoCubes

HANA-optimierte InfoCubes bilden im »SAP BW powered by SAP HANA« das Pendant zu relationalen InfoCubes in BW-Systemen mit relationalen Datenbanksystemen. Obwohl ihr Modell wesentlich auf die spaltenorientierte Speicherstruktur der HANA-Datenbank ausgerichtet ist, leben einige Aspekte ihrer relationalen Verwandten in ihnen weiter, sodass das Verständnis der relationalen InfoCubes selbst beim Einsatz von HANA vorteilhaft ist.

Vor dem Einsatz von HANA-optimierten InfoCubes ist zu beachten, dass bei der Verwendung von HANA auch DataStore-Objekte eine sinnvolle Datenbasis zur Analyse darstellen können – dies war bei relationalen Datenbanken nicht der Fall. Da DataStore-Objekte in der Regel die Datenlieferanten für InfoCubes darstellen[1], kann auf diese Weise unter Umständen die redundante Speicherung von Daten in InfoCubes vollständig entfallen. Der *Einsatz von HANA-optimierten InfoCubes ist unter diesen Umständen vor allem eine Designentscheidung* und wird in Kapitel 34 näher erläutert.

Eine Ausnahme stellen lediglich die *Behandlung von Bestandskennzahlen* und die *Beachtung der Requeststatus*[2] dar, die durch DataStore-Objekte nicht unterstützt werden, sodass der Einsatz von HANA-optimierten InfoCubes alternativlos ist.

Das Anlegen eines HANA-optimierten InfoCubes erfolgt ebenso wie bei relationalen InfoCubes in der Data Warehousing Workbench im Kontextmenü der InfoArea, der ein InfoCube untergeordnet sein soll (siehe Abb. 6–1).

1. Detaillierte Informationen hierzu werden in Teil V (BW-Design) gegeben.
2. Siehe Kapitel 28.

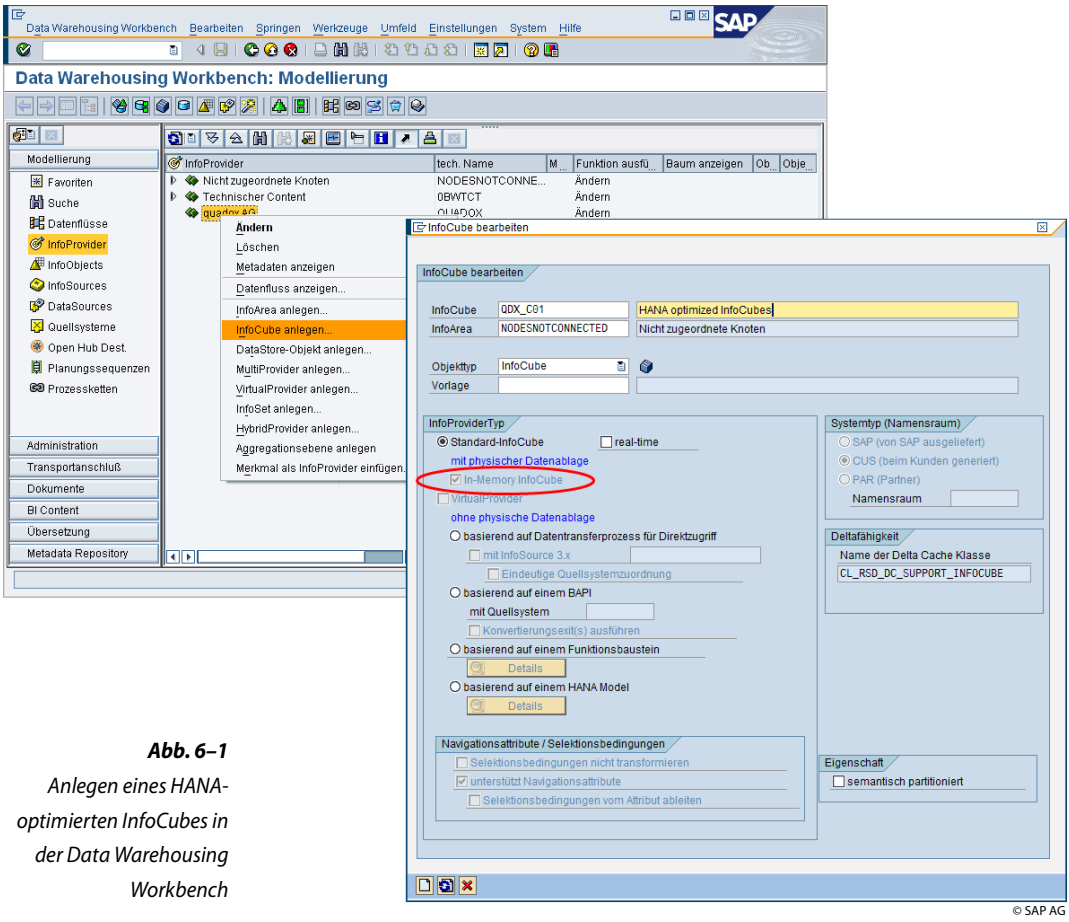


Abb. 6–1

Anlegen eines HANA-optimierten InfoCubes in der Data Warehousing Workbench

Neu angelegte InfoCubes werden grundsätzlich als HANA-optimierte InfoCubes[3] angelegt und im spaltenbasierten Speicher der HANA-Datenbank abgelegt. Eine Ausnahme bilden lediglich Realtime-InfoCubes für die integrierte Planung und die Datenbewirtschaftung nahe Echtzeit; diese profitieren zwar ebenfalls von der In-Memory-Technologie der HANA-Datenbank, jedoch werden sie wie ein relationaler InfoCube im zeilenbasierten Speicher abgelegt.

Auch nach einer Migration auf HANA können InfoCubes im zeilenbasierten Speicher liegen. Solche InfoCubes können (und müssen) mithilfe der Transaktion `RSMIGRHANADB` in HANA-optimierte InfoCubes konvertiert werden (siehe Abb. 6–2).

3. HANA-optimierte InfoCubes werden an einigen Stellen im BW auch als In-Memory InfoCube bezeichnet. Dies ist vor allem dann der Fall, wenn an der jeweiligen Stelle nicht zwingend festgeschrieben ist, ob der InfoCube zeilen- oder spaltenbasiert gespeichert wird.

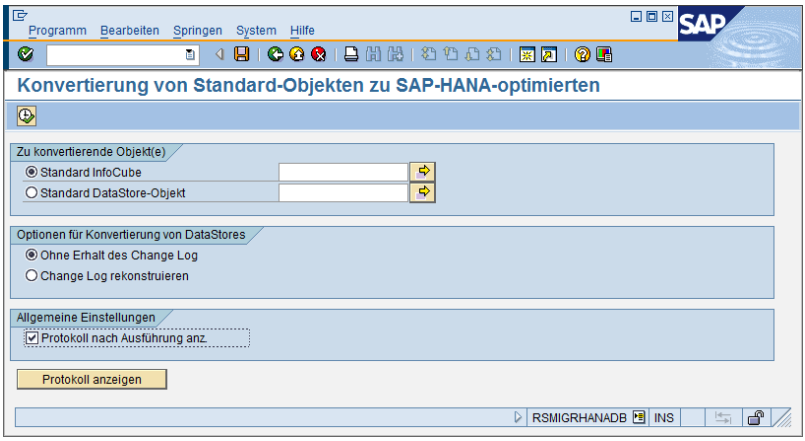


Abb. 6–2
Konvertierung von relationalen Modellen in HANA-optimierte Modelle

Die Modellierung eines HANA-optimierten InfoCubes erfolgt analog zu relationalen InfoCubes, d.h., es sind Kennzahlen und Dimensionen in das Modell aufzunehmen, als würde ein Star-Schema modelliert. Auch die Restriktionen, die von relationalen InfoCubes bekannt sind, gelten für HANA-optimierte InfoCubes, d.h., es können maximal 233 Kennzahlen und maximal 13 Dimensionen frei definiert werden und je maximal 248 InfoObjekte enthalten sein – ungeachtet dessen, dass HANA diese Restriktion selbst nicht verlangt.

Dimensionstabellen

Auf der Datenbank wird jedoch lediglich die Paketdimension als Dimensionstabelle angelegt, während die Stammdaten-IDs aller anderen Merkmale direkt in der Faktentabelle abgelegt werden. Die Faktentabelle eines HANA-optimierten InfoCubes gleicht insofern eher einer Faktentabelle, in der ausschließlich Line-Item-Dimensionen modelliert sind – jedoch ohne die Begrenzung auf maximal 16 Schlüsselfelder, die aus dem ABAP Dictionary bekannt ist[4].

Abbildung 6–3 skizziert diese Besonderheit der HANA-optimierten InfoCubes und stellt sie dem Modell relationaler InfoCubes gegenüber. Dabei wird deutlich, dass HANA-optimierte InfoCubes kein vollständig neues Modell abbilden, sondern lediglich das Konzept der Line-Item-Dimensionen, das auch von relationalen InfoCubes bekannt ist, durchgängig anwenden.

4. Im ABAP Dictionary wird die unkomprimierte Faktentabelle eines InfoCubes beschrieben. In dieser wird die Dimensions-ID für die Paketdimension als Primärschlüssel und alle Stammdaten-IDs als Nicht-Schlüsselfelder dargestellt. Hierbei handelt es sich jedoch lediglich um die Präsentation der Faktentabelle im ABAP Dictionary und nicht um die tatsächliche Definition in HANA.

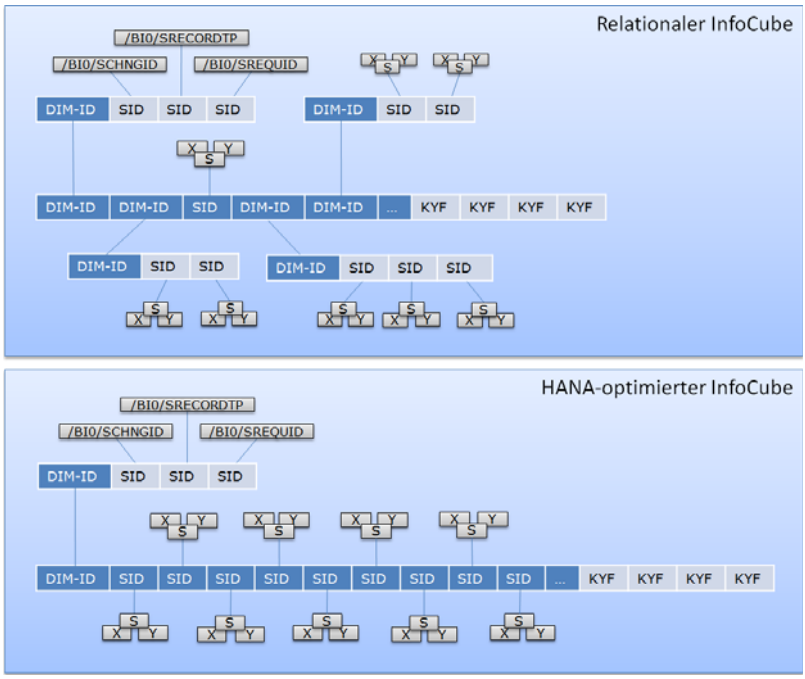


Abb. 6–3
InfoCube-Modell in RDBMS vs. HANA

Der Vorteil der flachen Tabellenstrukturen bei HANA-optimierten InfoCubes liegt unter anderem darin, dass die Gestaltung der Dimensionen und die Verteilung von Merkmalen auf Dimensionen ausschließlich der Gruppierung/Ordnung dienen und keinerlei Einfluss auf die Performance des Datenmodells hat. Es entstehen keinerlei große oder kleine Dimensionstabellen, die unter Umständen zeitaufwendig optimiert werden müssen, wodurch die Modellierung eines HANA-optimierten InfoCubes schneller und einfacher ist als ein vergleichbarer relationaler InfoCube.

Faktentabelle

Die Faktentabelle eines HANA-optimierten InfoCubes wird im ABAP Dictionary lediglich durch eine Tabelle repräsentiert[5]. Tatsächlich besteht die Tabelle jedoch aus mehreren Indexpartitionen[6], deren Zusammenspiel zu betrachten ist:

- eine Partition für unkomprimierte Requests
- eine Partition für komprimierte Requests
- eine Partition für Bestandsinitialisierungen
- eine Partition für historische Bestände

5. Der Tabellenname entspricht der Tabelle für die unkomprimierte Faktentabelle von relationalen InfoCubes.
6. Die Partitionen sind im HANA Studio ersichtlich.

Die beiden letzteren Indexpartitionen werden selbst dann angelegt, wenn es sich nicht um einen InfoCube mit Beständen handelt.

Der Zweck der Partitionen ist teilweise mit ihren relationalen Pendants vergleichbar. In Details sind jedoch wesentliche Unterschiede zu erkennen, die in der nachfolgenden Beschreibung hervorgehoben werden. Im Zusammenspiel der Partitionen sind ferner weitere Indizes wie der *Delta-Index* zur Faktentabelle sowie u.U. die *Gültigkeitstabelle für Bestände* relevant. Nachfolgend wird die Beschreibung in die beiden folgenden Themenkomplexe aufgeteilt:

- Verbuchung in den Delta-Index, der *Delta-Merge* und das Zusammenspiel der Partitionen für unkomprimierte und komprimierte Requests im Rahmen der *Komprimierung* (Abschnitt 6.1)
- Umgang mit *Bestandskennzahlen* (Abschnitt 6.2)

Ebenso wie bei relationalen InfoCubes unterliegt die Modellierung von HANA-optimierten InfoCubes Einschränkungen, sobald Daten in einem Cube enthalten sind. Diese Einschränkungen werden in Abschnitt 6.3 behandelt.

Den Abschluss der Beschreibung von HANA-optimierten InfoCubes bildet Abschnitt 6.4 mit einer Auflistung der Strukturen, die beim Aktivieren der Cubes im ABAP Dictionary generiert werden.

6.1 Delta-Merge und Komprimierung

Im spaltenbasierten Speicher der HANA-Datenbank besteht jede Spalte aus einem *Hauptindex* und einem sogenannten *Delta-Index*[7]. Dies gilt für jede Tabelle in der spaltenbasierten Ablage und trägt dem Problem Rechnung, dass Änderungen an bestehenden Indizes (also auch das Hinzufügen neuer Datensätze) bei der spaltenbasierten Speicherung ressourcenintensiv sind. Dementsprechend puffert der Delta-Index das Schreiben neuer Requests in der Faktentabelle ab und ist dementsprechend für Schreibzugriffe optimiert, während der Hauptindex für Lesezugriffe optimiert ist.

In technischer Hinsicht (aber nicht in inhaltlicher!) ist das Zusammenwirken von Delta-Index und Hauptindex vergleichbar mit unkomprimierter und komprimierter Faktentabelle bei relationalen InfoCubes. Bei Lesezugriffen auf den Index werden auch die Daten von Delta-Index und Hauptindex zusammengeführt, sodass die Ablage von Daten in einem der Indizes für die Anwendung transparent ist.

7. Die Bezeichnung »Delta« steht dabei in keinerlei Beziehung zu den gleichnamigen Verarbeitungsverfahren im Rahmen der Datenbewirtschaftung.

Delta-Merge

Die Überführung der Daten aus dem Delta-Index in den Hauptindex erfolgt seitens der HANA-Datenbank durch einen sogenannten *Delta-Merge*. Hierbei werden Delta-Index und die (Haupt-)Indexpartition für unkomprimierte Requests zu einer neuen Indexpartition für unkomprimierte Requests zusammengefasst, die die alte Partition ersetzt. Schreiboperationen werden mit dem Start des Delta-Merge in einen neuen Delta-Index gelenkt, der nach dem Merge den alten Delta-Index ersetzt.

Durch dieses Vorgehen steht der Index durchgängig für Schreib- und Leseoperationen zur Verfügung, jedoch werden im Hauptspeicher der HANA-Datenbank während des Delta-Merge die alte und neue Indexpartition für unkomprimierte Requests sowie der alte und neue Delta-Index abgelegt. Der Vorgang des Delta-Merge kann daher sehr ressourcenintensiv sein, sodass es sinnvoll sein kann, auf den Zeitpunkt des Delta-Merge Einfluss zu nehmen.

Zeitpunkt des Delta-Merge

Die HANA-Datenbank bietet eine Vielzahl von Merge-Typen, die mit spezifischen Zeitpunkten verbunden sind, zu denen sie durchgeführt werden. Für den Betrieb mit dem BW sind dabei nur der sogenannte *Auto Merge* und der *Smart Merge* relevant. Im Falle des Auto Merge überprüft die HANA-Datenbank selbstständig die Größe der Delta-Tabellen und löst beim Überschreiten bestimmter Größen den Delta-Merge aus. Im Falle des Smart Merge wird die Entscheidung für oder gegen die Durchführung eines Delta-Merge nach denselben Kriterien getroffen wie beim Auto Merge; der Zeitpunkt der Prüfung wird jedoch nicht durch die HANA-Datenbank bestimmt, sondern durch das BW getriggert.

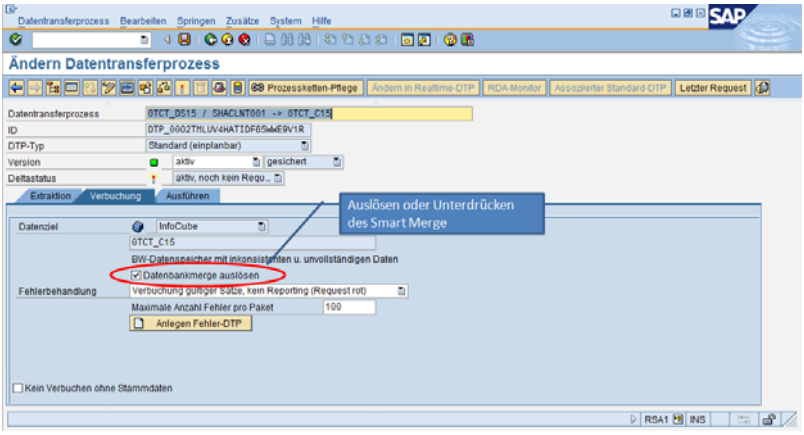
Während die Durchführung von Merge-Vorgängen in der Regel durch BW und HANA automatisiert erfolgt, kann im Falle von HANA-optimierten InfoCubes und schreiboptimierten DataStore-Objekten[8] Einfluss auf die Verwaltung genommen werden.

Per Default wird bei diesen Datenzielen nach der Ausführung jedes Requests ein Smart Merge ausgelöst – was jedoch nicht zwingend bedeutet, dass er tatsächlich ausgeführt wird, da die HANA-Datenbank die Entscheidung anhand der Größe der Delta-Tabelle selbst trifft.

Insbesondere wenn sehr große Datenmengen geladen wurden, kann es dennoch sinnvoll sein, den Merge zu unterdrücken, um beispielsweise folgende Ladevorgänge in dasselbe Datenziel oder Analysen nicht zu beeinträchtigen. Hierfür besteht in den Datentransferprozessen[9] zur

8. Siehe Abschnitt 14.11.
9. Eine detaillierte Beschreibung von Datentransferprozessen erfolgt im Rahmen der Datenbewirtschaftung in Kapitel 18.

Bewirtschaftung der HANA-optimierten InfoCubes die Möglichkeit, die Option *Datenbankmerge auslösen* zu deaktivieren (siehe Abb. 6–4).



© SAP AG

Abb. 6–4
Auslösen und Unterdrücken des Smart Merge

Soll ein Smart Merge außerhalb von Datentransferprozessen getriggert werden, so kann hierfür der Prozesstyp *Delta-Merge auslösen* in Prozessketten genutzt werden[10].

Die Indexpartition für unkomprimierte Requests sowie der Delta-Index verfügen über die Request-ID und sind Insert-Only, d.h., sie sind unter Umständen sehr detailliert und darauf ausgelegt, nur neue Daten zu speichern.

Das Löschen und Ändern bestehender Daten wird auf dieser Ebene vor allem in Form inverser Datensätze zusätzlich gespeichert. Eine Ausnahme stellt das Löschen der Daten eines vollständigen Ladevorgangs (Requests) dar: Hierbei bleiben die Daten im Index unverändert bestehen und es wird lediglich der entsprechende Request aus der Paketdimension der Faktentabelle gelöscht, sodass die Relation zur Faktentabelle nicht mehr integer ist und der Request für die Datenanalyse logisch gelöscht ist.

Komprimierung

Erst im Zuge der Komprimierung wird die Request-ID aus den Daten entfernt[11] und Datensätze mit gleichen Merkmalswerten aggregiert. Ebenso wie bei relationalen InfoCubes können dabei auch entstehende Datensätze mit Nullwerten gelöscht werden.

10. Siehe Abschnitt 25.2.
11. Genauer: Die Request-ID wird nicht entfernt, sondern mit dem Initialwert versehen, sodass sich die Daten unterschiedlicher Ladevorgänge nicht mehr über die Request-ID differenzieren lassen.

Technisch werden im Rahmen der Komprimierung Daten aus der Partition für unkomprimierte Requests in die Partition für komprimierte Requests übernommen.

Da bereits die Indexpartition für unkomprimierte Request uneingeschränkt für die Datenanalyse geeignet ist, dient die Komprimierung – anders als bei relationalen InfoCubes – ausschließlich zur Reduktion des Datenvolumens im Allgemeinen und in der Partition für unkomprimierte Requests im Speziellen. Letzteres ist von Bedeutung, da hierdurch der Prozess des Delta-Merge mit einem geringeren Ressourcenverbrauch verbunden ist. Die Komprimierung sollte daher in jedem Fall durchgeführt werden – selbst dann, wenn sich das Datenvolumen dadurch insgesamt nicht verringert.

6.2 InfoCubes mit Bestandskennzahlen

Bei HANA-optimierten InfoCubes mit Bestandskennzahlen werden grundsätzlich die Konzepte aufgegriffen, die bereits von relationalen InfoCubes bekannt sind. Dieser Abschnitt baut auf dem in Abschnitt 5.2.3 Beschriebenen auf und soll vor allem vergleichende Informationen für den Umgang mit Beständen in HANA-optimierten InfoCubes liefern.

Bestandsinitialisierung

Ein grundlegender Unterschied zwischen HANA-optimierten InfoCubes und relationalen InfoCubes besteht darin, dass bei HANA-optimierten InfoCubes *keine Rückrechnung von Beständen* aus der Stützstelle mehr durchgeführt wird, um dem Bestand zu einem gegebenen Zeitpunkt zu ermitteln. Stattdessen wird jeder Bestand aus der Bestandsinitialisierung und den danach erfolgten Bestandsveränderungen ermittelt.

Eine Stützstelle mit den aktuellen Bestandsdaten ist nicht mehr erforderlich, da mit HANA auch die Vorwärtsrechnung aus der Bestandsinitialisierung möglich ist – was bei relationalen InfoCubes aus Performance-Gründen vermieden wurde. Daraus resultiert, dass die Bestandsinitialisierung unverändert bestehen bleibt und sogar in einer eigenen Indexpartition abgelegt wird.

Inhaltlich werden Daten der Bestandsinitialisierung wie bei relationalen InfoCubes durch den Wert `1` im InfoObjekt `ORECORDTP` gekennzeichnet, während Bestandsveränderungen den Wert `0` erhalten und in den normalen Indexpartitionen für unkomprimierte und komprimierte Requests abgelegt werden.

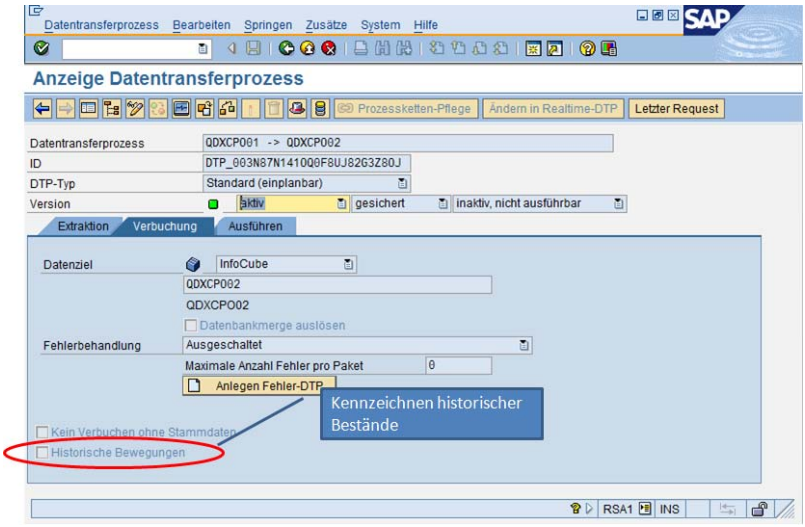
Historische Bestände

Eine besondere Behandlung erfordern auch bei HANA-optimierten InfoCubes die sogenannten *historischen Bestände*, also Bestandsveränderungen, die zeitlich vor der Bestandsinitialisierung liegen.

Zwar werden keine Stützstellen mehr aktualisiert, wovon historische Bestände ausgenommen werden müssten; jedoch ist es auch ohne Stützstellen erforderlich, Bestände, die zeitlich vor der Bestandsinitialisierung liegen, mithilfe der historischen Bestände rückzurechnen.

Hierfür werden historische Bestände in HANA-optimierten InfoCubes explizit als solche gekennzeichnet und mit dem Wert 2 im Info-Objekt ORECORDTP versehen. Zusätzlich werden historische Bestände in einer eigenen speziell hierfür vorgesehenen Indexpartition gespeichert.

Ob es sich bei Bestandsveränderungen um historische Bestände handelt, wird bei HANA-optimierten InfoCubes nicht mehr im Zuge der Komprimierung festgelegt, sondern über eine entsprechende Einstellung im jeweiligen Datentransferprozess, durch den die historischen Bestände in den InfoCube gelangen (siehe Abb. 6–5).



© SAP AG

Abb. 6–5
Kennzeichnen historischer Bestände in HANA-optimierten InfoCubes

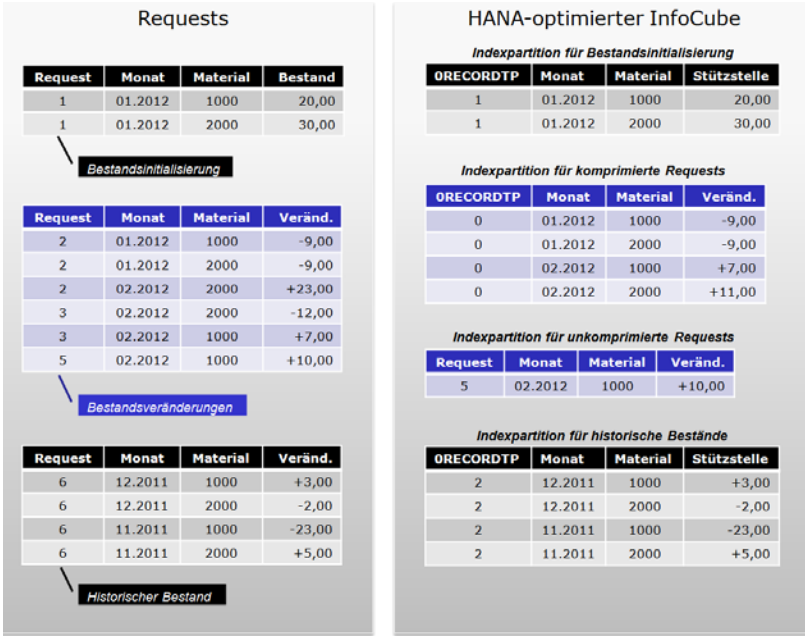
Dieses geänderte Vorgehen verringert vor allem die Fehleranfälligkeit, die die Komprimierung ohne Stützstellenfortschreibung mit sich brachte, und macht das Laden von historischen Beständen bei Bedarf reversibel.

Eine Ausnahme bilden HANA-optimierte InfoCubes mit Bestandskennzahlen, die mithilfe der alten Fortschreibungsregeln der BW-Releases bis 3.x fortgeschrieben werden: Hierbei müssen historische Bestände mangels eines entsprechenden Datentransferprozesses weiterhin ohne Stützstellenfortschreibung komprimiert verwenden.

Abbildung 6–6 skizziert das Zusammenspiel der Indexpartitionen anhand eines Beispiels, in dem neben der Bestandsinitialisierung (Request 1) auch komprimierte (Requests 2+3) wie unkomprimierte

(Request 5) Requests mit Bestandsveränderung vorliegen und durch historische Bestände (Request 6) ergänzt wurden.

Abb. 6–6
Komprimierung der Faktentabelle bei HANA-optimierten InfoCubes mit Bestandskennzahlen



Gültigkeitstabelle

Wie bei ihren relationalen Pendants ist auch bei HANA-optimierten InfoCubes mit Bestandskennzahlen der Einsatz einer Gültigkeitstabelle mit einer zeitlichen Gültigkeit und ggf. weiteren Bezugsmerkmalen erforderlich. Arbeitsweise und Umgang mit der Gültigkeitstabelle und den Bestandsparametern ist bei beiden InfoCube-Typen identisch und wird ab Seite 84 erläutert.

6.3 Remodellierung von HANA-optimierten InfoCubes

HANA-optimierte InfoCubes weisen aufgrund ihrer spaltenbasierten Speicherstruktur gänzlich andere Verhaltensweisen bei Änderungen auf, als dies bei ihren relationalen Pendants der Fall ist. Insbesondere das Hinzufügen oder Entfernen von Feldern stellt in der spaltenbasierten Ablage lediglich eine Anpassung von Metadaten dar, um eine transponierte Spalte hinzuzufügen oder zu entfernen.

Diese besonderen Fähigkeiten der HANA-Datenbank (sowie auch die flache Struktur, die das Star-Schema weitgehend ersetzt) werden bei der Modellierung von HANA-optimierten InfoCubes unterstützt; dennoch sind Anpassungen auch bei diesem Datenbanksystem an Einschränkungen gebunden, sobald Daten in einem InfoCube gespeichert sind.

In der nachfolgenden Tabelle sind die Möglichkeiten zur Änderung von HANA-optimierten InfoCubes und relationalen InfoCubes gegenübergestellt. Im Vergleich zu der Tabelle auf Seite 104 fehlen solche Änderungsoperationen, die bei HANA-optimierten InfoCubes nicht relevant sind (bspw. die Kennzeichnung von Line-Item-Dimensionen oder das Ändern des Partitionierungsschemas).

Modellierung eines gefüllten InfoCubes	HANA-optimiert	Relational
Merkmal mit Initialwert aufnehmen	✓	✓
Merkmal mit definiertem Wert aufnehmen		(✓)[a]
Merkmal löschen	✓	(✓)[b]
Navigationsattribute an-/ausschalten	✓	✓
Kennzahl mit Initialwert aufnehmen	✓	✓
Kennzahl mit definiertem Wert aufnehmen		(✓)[c]
Kennzahlen löschen	✓	(✓)[d]
Dimensionen anlegen	✓	✓
Dimensionen löschen	✓	
Dimensionszuordnung eines Merkmals ändern	✓	
Providerspezifische Konstanten ändern		
Bestandsparameter ändern	✓[e]	✓

a. Mithilfe des Remodellierungstools eingeschränkt möglich.
b. Nur mithilfe des Remodellierungstools möglich.
c. Dito.
d. Dito.
e. Das Verfahren ist identisch mit dem Verfahren für relationale InfoCubes (siehe Abschnitt 5.4.3).

Die Modellierung von InfoCubes in der Data Warehousing Workbench bietet damit bereits Funktionalität, die bei relationalen InfoCubes nur durch den Einsatz des Remodellierungstools bereitgestellt wird. Letzteres kann bei HANA-optimierten InfoCubes nicht eingesetzt werden, wodurch bei diesen InfoCubes die Möglichkeit fehlt, Merkmale oder Kennzahlen mit einem definierten Wert in bestehende InfoCubes aufzunehmen. Das Füllen von Merkmalen und Kennzahlen mit Werten kann ausschließlich über den Weg der normalen Datenbewirtschaftung erfolgen.

6.4 Generierte Tabellen

Bei der Aktivierung der Metadaten eines InfoCubes werden durch das BW automatisch alle erforderlichen Tabellen im ABAP Dictionary und in der HANA-Datenbank angelegt.

Nachfolgend werden die im ABAP Dictionary generierten Tabellen tabellarisch dargestellt. Die in der HANA-Datenbank angelegten Partitionen für die Faktentabelle sind aus Sicht des ABAP Dictionary transparent. Die darin enthaltenen Daten können lediglich durch den Inhalt der Paketdimension differenziert werden[12].

Aufgrund der unterschiedlichen Namensräume wird in der Tabelle zwischen InfoCubes unterschieden, die aus dem BI Content entnommen sind, und solchen, die selbst definiert sind. Bei Letzteren wird ferner auch der Fall angeführt, dass die Cubes als semantisch partitionierte Objekte definiert sind.

	InfoCube (BI Content)	InfoCube (selbst definiert)	InfoCube als SPO
Technischer Name	`Otttttttt`	`{A-Z}tttttttt`	`{A-Z}tttttt`
Faktentabelle	`/BIO/FOtttttttt`	`/BIC/F{A-Z}tttttttt`	`/BIC/F{A-Z}tttttt`*nn*
Gültigkeitstabelle[a]	`/BIO/LOtttttttt`	`/BIC/L{A-Z}tttttttt`	`/BIC/L{A-Z}tttttt`*nn*
Paketdimension	`/BIO/DOttttttttP`	`/BIC/D{A-Z}ttttttttP`	`/BIC/D{A-Z}tttttt`*nn*`P`

a. Nur bei Bestands-InfoCubes.

In der Tabelle wird `t` als Platzhalter für beliebige alphanumerische Zeichen im technischen Namen des Cubes und *nn* für die vom System vergebene ID der semantischen Partition (nur bei semantisch partitionierten Objekten) verwendet.

12. `OREQUEST` zur Unterscheidung zwischen unkomprimierter und komprimierter Partition bzw. `ORECORDTP` zur Unterscheidung zwischen Bestandsveränderungen, historischen Beständen oder Bestandsinitialisierung.