Sind wir eigentlich ganz dicht?

Eero Mattila
Quest Software GmbH
Köln

Schlüsselworte
Komprimierung, Datenexplosion, Datenverdichtung, Advanced Compression Option

Einleitung

Datenmengen explodieren und stellen die Betreiber von Datenbanken vor große Herausforderungen. Im Laufe der Jahre hat Oracle unterschiedliche Möglichkeiten zur Verdichtung von Daten eingeführt, angefangen mit der Index Key Komprimierung in der Version 8.1.5 über einfache Tabellenkomprimierung in 9.2, Backup-Komprimierung in 10g, erweiterte Komprimierung in 11.1 bis hin zur hybriden Spaltenkomprimierung im Zusammenhang mit Exadata in 11g Release 2. Diese Techniken zur Verdichtung stellen einerseits Chancen zur Speicherplatzeinsparung und Performance-Vorteile dar, können aber auch Zusatzkosten – monetär wie performancebedingt – verursachen.
Dieser Vortrag bietet einen Überblick über die verschiedenen Einsatzgebiete und Techniken zur Verdichtung von Daten in einer Oracle Datenbank.

Datenexplosion und ihre Folgen

Das kollektive Wissen der Menschheit nimmt täglich zu und wird immer eifriger nicht nur Dokumentiert, sondern auch über verschiedenste Medien verbreitet. Gesetzliche Vorschriften verpflichten uns, bestimmte Daten sogar über Jahrzehnte aufzubewahren. Dies führt zu einem explosionsartigen Wachstum des Bedarfs an Speicherplatz, der wiederum schlicht Geld kostet. Hinzu kommt, dass die Verarbeitung immer größerer Datenmengen auch mehr Zeit kostet – die Performance der Anwendungen nimmt ab. Das grundsätzliche Problem – die Datenexplosion an sich – kann nicht bekämpft werden, also müssen wir andere Möglichkeiten finden, damit umzugehen. Komprimierung von Daten an verschiedenen Stellen und auf verschiedenen Arten und Weisen kann oft gute Dienste zu diesem Zweck leisten: Sie kann einerseits helfen, Speicherplatz zu sparen, andererseits aber auch, Zugriffszeiten auf Datenbankinhalte zu optimieren.

Index Key Compression

Bereits in der Version 8.1.5 der Oracle-Datenbank wurde die Möglichkeit eingeführt, Indizes durch Komprimierung zu optimieren. Hierbei handelt es sich um ein Feature, das in jeder Edition der Datenbank zur Verfügung steht und nicht separat lizensiert werden muss. Bei Index Key Compression werden sich wiederholende Werte in führenden Spalten von Indizes dedupliziert, indem sie nur einmal in einer besonderen Präfix-Tabelle im Leaf-Block gespeichert werden. Die eigentlichen Index-Einträge referenzieren die Präfix-Tabelle. Je nach Selektivität der zu komprimierenden Schlüssel kann der benötigte Speicherplatz des Indexes erheblich reduziert und dadurch deutlich geringerer I/O bei Indexzugriffen erzielt werden. Neben B*Tree-Indizes kann die Komprimierung auch für Index Organized Tables eingesetzt werden.

Die Komprimierung kann sowohl beim Erstellen eines neuen Indexes als auch nachträglich definiert werden:

```
CREATE INDEX ON tabelle (spalte1, spalte23, …) COMPRESS x;
```

```
ALTER INDEX index REBUILD COMPRESS x;
```

Mit x wird die Anzahl der zu komprimierenden führenden Spalten angegeben.

Ob und wie viel ein Index komprimiert werden sollte, kann mit dem folgenden Befehl ermittelt werden:

```
ANALYZE INDEX index VALIDATE STRUCTURE;
```

Anschließend gibt die View INDEX_STATS Auskunft über die zu erzielende Ersparnis und die optimale Komprimierungseinstellung:

```
SELECT blocks, opt_cmpr_pctsave, opt_cmpr_count
FROM index_stats;

    BLOCKS OPT_CMPR_COUNT OPT_CMPR_PCTSAVE
---------- -------------- ----------------
       988              2               37
```

Demnach empfiehlt sich in diesem Fall eine Komprimierung des Indexes mit COMPRESS 2 (Wert in der Spalte OPT_CMPR_COUNT), und es ist mit einer Reduzierung der Indexblöcke um ca. 37 Prozent zu rechnen.

```
ALTER INDEX comp_test REBUILD COMPRESS 2;
```

Die erneute Abfrage gegen INDEX_STATS ergibt folgende Ergebnisse:

```
SELECT blocks, opt_cmpr_pctsave, opt_cmpr_count
FROM index_stats;

    BLOCKS OPT_CMPR_COUNT OPT_CMPR_PCTSAVE
---------- -------------- ----------------
       621              2                0
```

Wie erwartet, hat sich die Anzahl der Blöcke in unserem Index um ein gutes Drittel reduziert.

Bei ANALYZE INDEX … VALIDATE STRUCTURE ist zu beachten, dass der Befehl einen DML-Lock auf den Index erzeugt, so dass insbesondere bei großen Indizes die Performance von Inserts, Updates und Deletes während des Analyze-Vorgangs beeinträchtigt werden kann.

Tabellenkomprimierung

Oracle 9i Release 2 führte erstmals die Komprimierung für Tabellendaten ein, zunächst allerdings nur für ausgewählte, sogenannte Bulk-Load-Operationen (Direct Path SQL*Load, CREATE TABLE AS SELECT, paralleles DML und INSERT mit APPEND-Hint). Dies bedeutete, dass mittels normaler DML-Befehle eingefügte oder geänderte Daten nicht komprimiert wurden. Somit war dieses Feature vor allem interessant für Datawarehouse-Umgebungen. Allerdings gab (und gibt) es dort häufig Einschränkungen – sei es, dass der ETL-Vorgang keinen Direct Load ermöglicht, oder dass ein CTAS aufgrund des Datenvolums zu viel Zeit in Anspruch nehmen würde.

Mit Oracle Database 11g wurde die (separat zu lizensierende) Advanced Compression Option eingeführt. Sie erweitert die Tabellenkomprimierung auf INSERT- und UPDATE-Kommandos, so dass die Datenverdichtung auch in OLTP-Anwendungen zum Tragen kommt. Die oben beschriebene ursprüngliche Funktionalität für Bulk-Operationen ist weiterhin ohne Zusatzkosten verwendbar und wird nun Basic Compression genannt.

Die Komprimierung der Tabellendaten erfolgt auf Blockebene: Wiederholte Spaltenwerte werden – wie bei der Index Key Komprimierung – einmalig in einer Symboltabelle im Blockheader abgelegt und im Block selbst lediglich referenziert. Je nach Charakter der Daten können dadurch erheblich mehr Datensätze in einem Block untergebracht werden, was wiederum die Anzahl der physikalischen und logischen I/O-Operationen reduziert.

Beim Erzeugen einer neuen Tabelle wird die Komprimierung durch die Klause COMPRESS FOR definiert, gefolgt durch die Operationen, für die die Komprimierung gelten soll. Gültige Optionen sind BASIC (nur Bulk Inserts und CTAS) und OLTP (alle Operationen). Bei Exadata gibt es noch weitere Optionen – dazu mehr weiter unten.

```
CREATE TABLE […] COMPRESS FOR OLTP;
```

Bereits bei Tabellen recht übersichtlicher Größe lässt sich die Platzersparnis feststellen. Im folgenden Beispiel werden zwei identische Tabellen erstellt, einmal komprimiert, einmal nicht. Anschließend wird die Anzahl der Blöcke anhand der View USER_SEGMENTS kontrolliert:

```
CREATE TABLE doag_normal
AS SELECT * FROM dba_objects;

CREATE TABLE doag_compr COMPRESS FOR OLTP
AS SELECT * FROM dba_objects;

SELECT segment_name, blocks
FROM user_segments
WHERE segment_name LIKE 'DOAG%';
```

Das Ergebnis ist eindeutig:

```
SEGMENT_NAME             BLOCKS
-------------------- ----------
DOAG_COMPR                  384
DOAG_NORMAL                1152
```

Auch nachträglich lässt sich die Verdichtung einstellen:

```
ALTER TABLE tabelle COMPRESS FOR OLTP;
```

Dabei ist jedoch zu beachten, dass bereits vorhandene, vollständig ausgefüllte Blöcke nicht automatisch komprimiert werden. Damit diese Blöcke verdichtet werden können, müssten zunächst Datensätze gelöscht werden. Wird die Tabelle jedoch gleichzeitig reorganisiert, greift die Komprimierung sofort:

```
ALTER TABLE tabelle MOVE COMPRESS FOR OLTP;
```

Die Komprimierung kann außerdem als Standardeinstellung für alle neuen Tabellen in einem Tablespace definiert werden, sowohl bei der Erstellung des Tablespaces als auch nachträglich:

```
CREATE TABLESPACE […] DEFAULT COMPRESS FOR OLTP […];
ALTER TABLESPACE DEFAULT COMPRESS FOR OLTP;
```

Welche Tabellen sollten komprimiert werden?

Typische Kandidaten sind natürlich große Tabellen, die sich wiederholende Daten enthalten. Um festzustellen, ob und wie stark sich eine Tabelle verdichten lässt, steht ab Oracle 11g R2 das mitgelieferte Package DBMS_COMPRESSION zur Verfügung. Für ältere Versionen hält OTN das Package DBMS_COMP_ADVISOR zum Herunterladen bereit. Die Prozedur GET_COMPRESSION_RATIO (bzw. GETRATIO in der älteren Fassung) berechnet für die angegebene Tabelle den zu erwartenden Komprimierungsgrad für die gewünschte Komprimierungsart. Derzeit beschränkt sich die Unterstützung auf die BASIC, OLTP sowie die verschiedenen Varianten der Exadata Hybrid Columnar Compression (s. weiter unten). Hinweise für Indizes oder unstrukturierte Daten gibt es leider nicht.

Was kostet mich die Index- oder Tabellenkomprimierung?

Der reine Lizenzkostenaspekt der Advanced Compression Option soll hier nicht weiter vertieft werden. Man muss sich dessen bewusst sein, wie auch der Tatsache, dass die Funktionalität in jeder Installation einer Enterprise Edition vorhanden und ohne besondere Freischaltung einsetzbar ist. Sobald das Zauberwort COMPRESS in einem CREATE- oder ALTER-Statement erscheint, ist Lizensierung fällig. Wer die Option nicht besitzt und seinen Anwendern die Verwendung unterbinden möchte, kann auf einen nicht dokumentierten init.ora-Parameter (_OLTP_COMPRESSION=FALSE) zurückgreifen – aber auf eigene Gefahr.

Komprimierung bedeutet selbstverständlich Umrechnung von Daten. Heißt das also, dass wir uns die Platzersparnis und I/O-Vorteile auf Kosten von CPU-Verbrauch erkaufen? Kann jedes INSERT oder UPDATE unmittelbar ein Umschreiben von Blockheadern und –inhalten verursachen? Müssen bestehende Anwendungen angepasst oder darauf vorbereitet werden?

Für lesende Zugriffe bedeutet die Komprimierung – oder vielmehr Dekomprimierung – meist keinen messbaren Overhead. Dadurch, dass die Symboltabelle auf Blockebene geführt wird, geschieht die Umrechnung extrem schnell. Im Gegenteil kann die Leseperformance in vielen Fällen sogar erheblich gesteigert werden, weil weniger Blöcke gelesen werden müssen, so dass die I/O-Vorteile den ohnehin geringen CPU-Overhead deutlich überwiegen.

Interessanter ist die Frage nach den DML-Vorgängen in schreibintensiven OLTP-Systemen. Glücklicherweise lösen einzelne INSERTs oder DELETEs keine Komprimierungsvorgänge aus, sondern die Komprimierung findet asynchron statt, sobald ein bestimmter Füllgrad des Blocks erreicht wird. Umfangreiche UPDATEs können jedoch zu *Migrated Rows* und damit Performance-Einbußen führen. Gründliche Tests sind daher in jedem Fall vonnöten, ehe bestehende Daten in Produktionsumgebungen umgestellt werden können.

Der Einsatz von Index- und Tabellenkomprimierung ist aus der Sicht der Datenbankanwendungen vollkommen transparent. Anpassungen bestehender oder Vorbereitung neuer Applikationen ist nicht erforderlich.

Komprimierung von unstrukturierten Daten

Besonders unstrukturierte Daten nehmen häufig sehr viel Platz in Anspruch und somit prädestinierte Kandidaten für Komprimierung, sofern sie nicht von vornherein in einem komprimierten Format vorliegen. In Oracle 10g wurde das Package UTL_COMPRESS vorgestellt, mit dem große binäre Objekte (RAW, BLOB und BFILE) komprimiert werden können.

SecureFiles in Oracle 11g werden gern als die „nächste Generation von LOBs“ genannt. Neben verschiedenen Vorteilen bezüglich Performance und Administration bieten sie – auch wieder unter Nutzung der Advanced Compression Option – erweiterte Möglichkeiten zur Datenverdichtung. Als SecureFiles gespeicherte große Objekte können einerseits Komprimiert werden, wobei drei Komprimierungsstufen, LOW, MEDIUM und HIGH zur Verfügung stehen. Zu beachten ist, dass die Komprimierung der LOBs unabhängig von der regulären Tabellenkomprimierung ist: Tabellenkomprimierung komprimiert keine LOBs und umgekehrt. Neben der Komprimierung lassen SecureFiles auch die Deduplizierung der Objekte zu: Identische Dateien werden einmal gespeichert und von weiteren Datensätzen referenziert. Diese beiden Methoden können zu drastischen Einsparungen an Speicherplatz führen.

Hybrid Columnar Compression

Die oben beschriebenen Tabellenkomprimierungsarten können unabhängig vom verwendeten Speichersystem eingesetzt werden. Die hybride Spaltenkomprimierung, oder Hybrid Columnar Compression, kurz HCC, setzt dagegen einen Exadata Storage Server bzw. Oracle SAN oder NAS Storage Server voraus. HCC unterscheidet zwischen Datawarehouse- und Archivsystemen und stellt für beide optimierte Komprimierungsalgorithmen zur Verfügung.

Bei HCC handelt es sich um eine Kombination aus spalten- und zeilenbasierter Datenspeicherung. Die Datensätze werden in sog. logische Komprimierungsgruppen organisiert, die mehrere Blöcke umfasst und die Daten spaltenweise in verschiedenen Blöcken abspeichert. Dadurch werden sich wiederholende Daten in die gleichen Blöcke konzentriert, so dass eine extrem effiziente Komprimierung möglich ist.

Warehouse-Anwendungen sind dadurch geprägt, dass extrem große Datenvolumina mittels SELECT-Statements abgefragt und meist nie oder sehr selten verändert werden. Die Warehouse Compression verwendet einen speziell für Abfragen optimierten Algorithmus, wobei unter zwei Komprimierungsstufen gewählt werden kann. Typische Komprimierungsraten sind hier 6 bis 10 im Vergleich zu nicht komprimierten Daten.

Archivsysteme dienen zur Speicherung von historischen Daten, die aufgrund von gesetzlichen Vorschriften sogar jahrzehntelang vorgehalten werden müssen. Häufig werden solche Daten auf Bandarchiven abgelegt. Bei Bedarf müssen sie jedoch für den Zugriff wieder hergestellt werden, was sehr aufwändig sein kann. Die Archive Compression von HCC ermöglicht es, historische Daten auf Platzersparnis zu optimieren und trotzdem jederzeit im Zugriff zu behalten. Der Komprimierungsfaktor liegt bei Archive Compression häufig bei mind. 15 – aus einem Terabyte Daten werden ca. 65 GB.

Backup-Komprimierung

Das Wachstum der Datenmengen führt zwangsläufig auch zu größeren und damit langsameren Datensicherungen. Nicht überraschend, kann auch hier die Komprimierung wertvolle Vorteile verschaffen. Oracle Recovery Manager sichert Daten blockweise, „physikalisch“, und ermöglicht die

Wiederherstellung einer Datenbank, eines Tablespaces, oder auch einzelner Datenblöcke. Für Exports, oder „logische“ Backups einzelner Schemata oder Objekte, bietet sich Data Pump an. Beide Werkzeuge sind in der Lage, Daten unter Nutzung von Advanced Compression Option zu komprimieren, RMAN auch ohne das kostenpflichtige Add-on.

Bei RMAN wird die Komprimierung mit dem folgenden Befehl eingeschaltet:

```
CONFIGURE DEVICE TYPE [DISK|SBT] BACKUP TYPE TO COMPRESSED BACKUPSET;
```

Anschließend kann der Komprimierungsalgorithmus ausgewählt werden:

```
CONFIGURE COMPRESSION ALGORITHM '[BASIC|LOW|MEDIUM|HIGH]';
```

Die Komprimierungsstufe BASIC ist ohne die Advanced Compression Option verwendbar.

Da die Performance der verschiedenen Komprimierungsstufen stark von zahlreichen Faktoren (Charakter der Daten, Netzwerkkonfiguration, CPU-Ressourcen usw.) abhängt, können keine allgemein gültige Empfehlungen abgegeben werden. Höhere Komprimierung ist meist mit höherem CPU-Verbrauch verbunden, kann dafür durch geringere Netzwerkbelastung von Vorteil sein. Ausführliche Tests in der jeweiligen Umgebung sind letztendlich ausschlaggebend für die optimale Auswahl der Parameter. Besonders wichtig dabei ist, das Restore gründlich zu testen: Die kleinste Sicherung ist nichts wert, wenn die Wiederherstellung ewig dauert.

Data Pump Exports kennen ebenfalls den Parameter COMPRESSION, der folgende Werte akzeptiert: ALL, METADATA_ONLY und DATA_ONLY.

Sowohl RMAN als auch Data Pump dekomprimieren die Daten transparent beim Wiederherstellen bzw. Importieren.

Zusammenfassung

Oracle Datenbank bietet zahlreiche Mechanismen zur Komprimierung von Daten und dadurch zur Optimierung von Speicherplatz und Anwendungsperformance. Während einzelne dieser Methoden ohne Extrakosten eingesetzt werden können, sind bei vielen verlockenden Funktionen zusätzliche Lizenzen fällig, so dass deren Einsatz wohl überlegt werden muss. Performance ist ein wichtiger Aspekt bei allen Anwendungen, und die Komprimierung kann sich in gewissen Konstellationen auch negativ darauf auswirken. Daher ist ein umfangreiches Testen unbedingt notwendig, bevor Komprimierung produktiv eingesetzt wird. Wer aber mit stetig wachsenden Datenmengen, Performanceverlust und limitierter Speicherkapazität zu kämpfen hat, tut gut daran, sich die verschiedenen Komprimierungsmöglichkeiten genauer anzuschauen.

Kontaktadresse:
Eero Mattila
Quest Software GmbH
Im Mediapark 4e
DE-50670 Köln

Telefon: +49 (0) 221-5777 4169
Fax: +49 (0) 221-5777 450
E-Mail eero.mattila@quest.com
Internet: www.questsoftware.de