Effiziente Speicherung für SAP

Jörn Bartels
Oracle Deutschland
München

Schlüsselworte

IOT, Index, Performance.

Einleitung

Dieser Vortrag beschäftigt sich mit einem der wichtigsten Themen der Datenbanktechnik, der optimalen Performance und dem minimalen Plattenplatz Verbrauch. Es wird untersucht, in wie weit der Einsatz von komprimierten Index Organized Tables (IOT) in einem SAP Umfeld Vorteile bringt.

Index Organized Tables (IOT)

Daten und Indexinformationen werden in Oracle normalerweise separat gespeichert. Die Daten im Daten Segment, und die Indizes in Index Segmenten. Von den Indices wird dann durch die ROWID auf die entsprechende Zeile im Daten Segment verwiesen. Um auf einen Datesatz zuzugreifen, müssen daher immer mindestens zwei Blöcke gelesen werden.

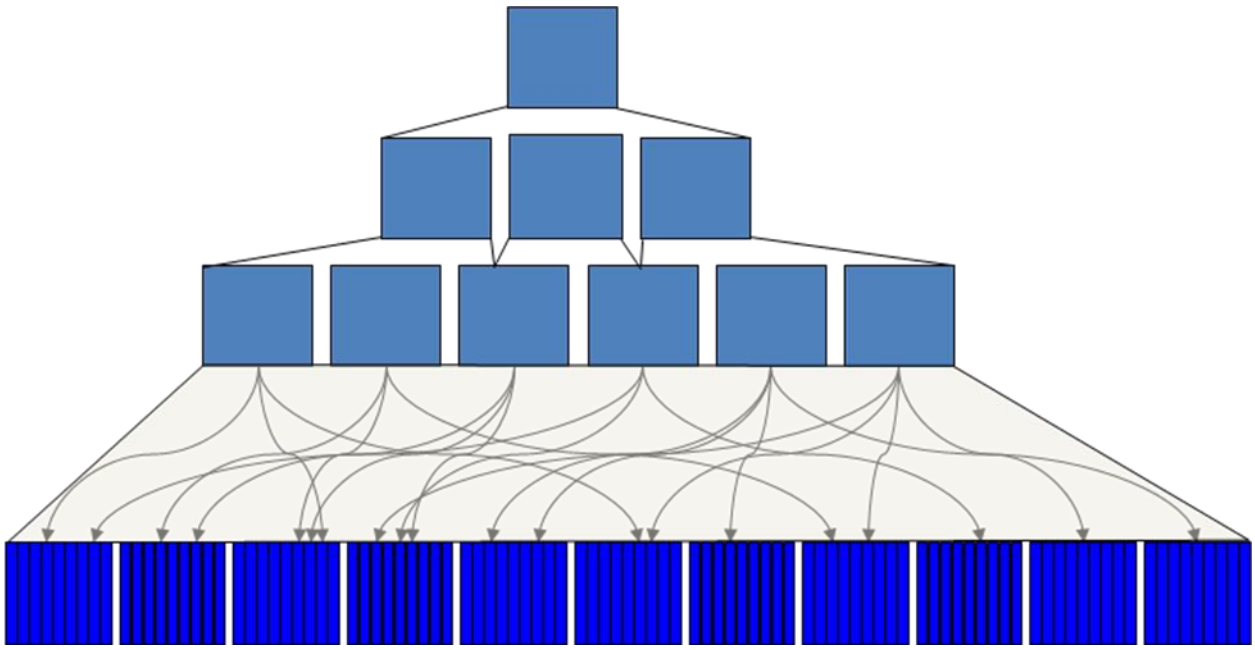

Abb. 1: Speicherung in Daten und Index Segment

Ein Indexblatt kann viele viele Blöcke adressieren. Bei einem indexsequentiellen Scan der Tabelle müssen häufig sehr viele Datenblöcke gelesen werden, da die Datensätze bunt gemischt über viele Datenblöcke verteilt sein können. In der folgenden Grafik sieht man als Beispiel einen einzelnen Blatt eines Indexbaumes

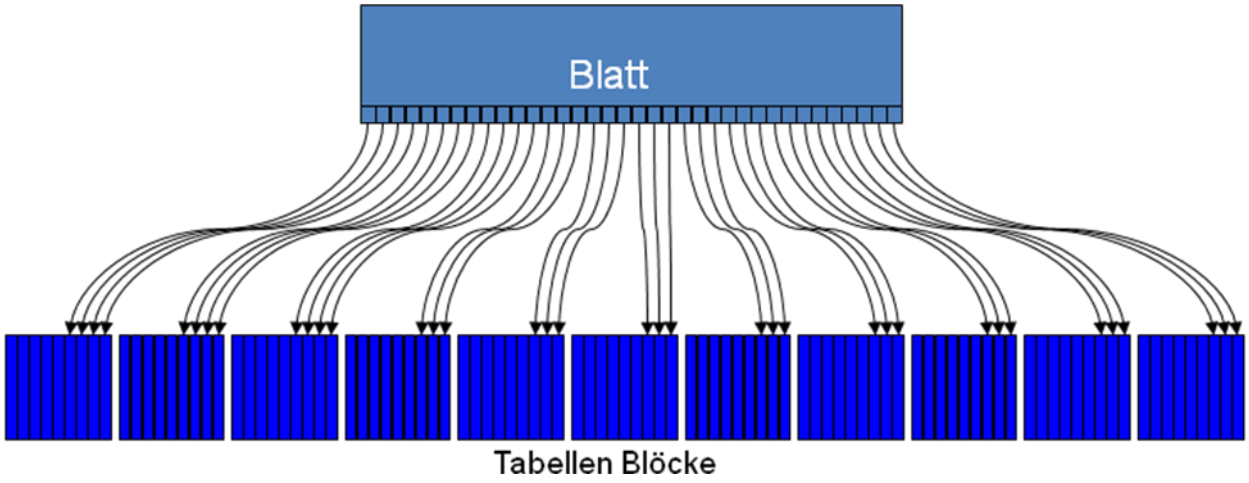

Abb. 2: Einzelnes Blatt einer Tabelle

In einem IOT werden die Daten gemeinsam mit dem Primär Schlüssel (Primary Key) gespeichert. Dies hat drei Vorteile. Für den Zugriff auf einen Datensatz kann man mit einem IO auskommen, der Schlüssel muss nicht doppelt gespeichert werden, und die Rowid muss gar nicht gespeichert werden.

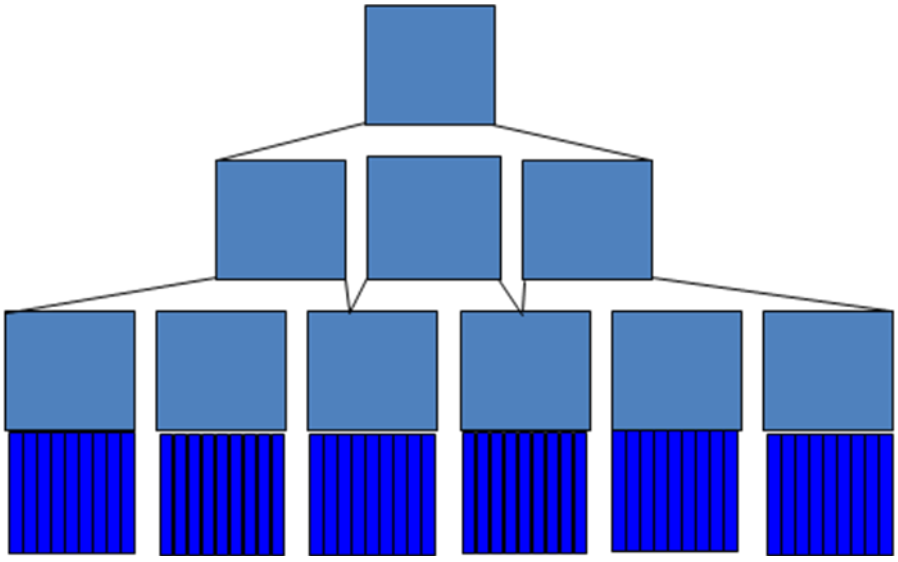

Abb. 3: Speicherung in einem IOT

Index komprimierung

Die Index Komprimierung kann auch für IOTs benutzt werden. Im Indexschlüssel werden dafür identische Präfix Teile durch einen Verweis auf den vorherigen Eintrag im Index ersetzt. Die Länge des Präfix muss bei Anlage des Index definiert werden. Die Länge des Präfix sollte so definiert werden, dass eine optimale Komprimierung erreicht wird.

Im folgenden Beispiel wird eine optimale Komprimierung bei einer Präfix Länge von 4 erreicht.

1	A	X	1	A	1	ROWID1
				B	2	ROWID2
1	A	X	2	A	3	ROWID3
1	A	Y	1	B	4	ROWID4
1	A	Y	3	C	5	ROWID5
				C	6	ROWID6
				D	7	ROWID7
1	B	X	1	A	1	ROWID8
				A	2	ROWID9
				C	3	ROWID10
1	B	X	3	A	4	ROWID11
				C	5	ROWID12
				C	6	ROWID13

Abb. 4: Index Komprimierung

Die optimale Länge für den Präfix kann auf verschiedenen Wegen berechnet werden:

- Offline
  - ANALYZE INDEX <xxx> VALIDATE STRUCTURE
  - SELECT opt_cmpr_count, opt_cmpr_pctsave
    FROM index_stats
- Online
  - Package IND_COMP aus Hinweis 1109743
  - EXEC ind_comp.get_column('SAPR3' , 'GLPCA')
  - BRSPACE

Ergebnisse der Indexkomprimierung bei ausgewählten SAP Tabellen

- Index GLPCA~1 von 18 GB auf 4,5 GB reduziert.
  - 75% kleinerer Index
- Indices der 10 größten Tabellen von 224 GB auf 138 GB reduziert.
  - 40% kleinere Indices

Weitere Informationen zur Index Komprimierung finden sich im SAP Hinweis 1109743.

Typische Index Nutzung in SAP Datenbanken.

Viele Tabellen und Indizes in einer SAP Datenbank lassen sich in Nutzungsklassen einteilen. Diese Klassen eignen sich mehr oder weniger gut für eine Optimierung mit IOTs.

Monoton wachsende Schlüssel

Häufig werden die Schlüssel für eine Tabelle monoton wachsend erzeugt, und auch so in die Datenbank eingefügt.

Die Datensätze werde dafür auf viele Blöcke verteilt, die jeweils noch Platz haben. Eine zusammen eingefügte Gruppe von Datensätzen wir daher über mehr Blöcke verteilt, als notwendig. In eine IOT würden die Daten dagegen in der minimal möglichen Zahl von Blöcken gespeichert.

Die Vorteile eines IOT sind hier:

1. Weniger Platz Verbrauch
2. Weniger IO
3. Bessere Cache Nutzung.

Buchungen pro Konto

Werden die Schlüssel in einer vollständig chaotischen Folge eingefügt, wie sie zum Beispiel ein Konto in einem Buchungssatz darstellt, werden die Daten über alle Datenblöcke der Tabelle verteilt.

Ein indexsequentielles Lesen muss daher fast so viele Blöcke lesen, wie es es Zeilen liest.

Die Vorteile eines IOT sind hier:

1. Weniger Platz Verbrauch
2. Viel weniger IO
3. Viel bessere Cache Nutzung

GUIDs im Schlüssel

Werden GUIDs für den Schlüssel genommen, haben wir auch eine vollständig chaotische Verteilung der Werte, und Schlüssel werden über alle Datenblöcke verteilt. Es gibt noch nicht einmal eine beieinanderliegende Speicherung der Index Werte, sondern die Schlüssel sind über den ganzen Index verteilt.

Da es ein index sequentielles Lesen hier nicht gibt, bleibt nur ein einziger Vorteil:

1. Weniger Platz Verbrauch.

Zusätzliche Indizes für ein IOT

Ein IOT hat natürlich auch Nachteile. Sobald auf der Tabelle ein zweiter Index definiert wird, muss in diesem Index nicht nur die Rowid gespeichert werden, sondern auch der vollständige Primär Schlüssel. Dies ist notwendig, damit auch bei einem Update des Primär Schlüssels, und einer damit verbundenen Änderung der Rowid, der Satz immer noch gefunden werden kann.

Der Platzvorteil, der durch den Einsatz von komprimierten IOTs erzielt wird, wird duch die zusätzliche Speicherung des Primärschlüssels in einem zweiten Index meist wieder aufgebraucht. Die anderen Vorteile bleiben natürlich erhalten.

Durch die Speicherung der sogenannten „Guess Rowid“ kann aber auch auf ein IOT genau so schnell zugegriffen werden, wie auf eine normale Tabelle. Die Guess Rowid wird bei der Einfügung des Satzes mit in den Index aufgenommen, dann aber nicht weiter gepflegt.

Beispiele

CDHDR

Tabelle	31 GB
Index CDHDR~0	18 GB
Summe	49 GB
IOT	28 GB
Faktor:	1,75

VBOX

Tabelle:	122 GB
Index VBOX~0	135 GB
Index VBOX~A	57 GB
Summe	364 GB
IOT	72 GB
Index VBOX~A	139 GB
Summe	211 GB
Faktor	1,7

Abfrage

Tabelle /SAPAPO/MATLSPP

Index auf (MANDT, MATID, LOCID, SIMID)

Die Tabelle hat 112 weitere Spalten.

```
SELECT      *
FROM     "/SAPAPO/MATLSPP"
WHERE  "MANDT" = :A0 AND "SIMID" = :A1 AND "MATID" = :A2 AND "LOCID" = :A3
OR  "MANDT" = :A4 AND "SIMID" = :A5 AND "MATID" = :A6 AND "LOCID" = :A7
OR…. (47)
OR  "MANDT" = :A196 AND "SIMID" = :A197 AND "MATID" = :A198 AND "LOCID" = :A199
```

	Tabelle mit Index	IOT
Elapsed time per Exec (s)	0,4	0,04
Reads per Exec	17,91	5,36
Executions	429.564	429.705

In einem Simulations Job mit einer Mischung aus select, update, insert und delete wurden die folgenden Zeiten gemessen:

Elapsed Zeit für Tabelle mit Index war 44 Minuten, und bei Speicherung als Komprimiertes IOT waren es dann nur noch 21 Minuten.

Erstellung von IOTs

Mit dem Packaget „IOTC“ können Tabellen automatisch in ein IOT umgewandelt werden.

Der Aufruf des Packages erfolgt wie folgt:

EXEC IOTC.CONVERT (<n>, <parallel>)
EXEC IOTC.CONVERT (<tab>, <parallel>)

Die erste Form erzeugt ein Umsetzungsscript für die grössten <n> Tabellen, während die zweite Form ein Script für genau eine Tabelle erzeugt.

Weitere Informationen dazu kann man dem SAP Hinweis 1856270 entnehmen.

Zusammenfassung

Durch den Einsatz von komprimierten IOTs können Queries bis zu 5 mal schneller werden. Inserts und Deletes werden bis zu doppelt so schnell. Die durchschnittliche Platzersparnis beträgt 25%. Durch die bessere Platzausnutzung sind wesentlich mehr „Hot“ Daten im Datenbank Cache.

Die Speicherung als IOTs eignet sich für SAP Datenbanken besonders gut, da alle Tabellen einen Primär Schlüssel (UNIQUE INDEX) haben, und 70% aller Tabellen keinen weiteren Index haben.

IOTs sind in der Version 8 eingeführt worden, und daher schon seit vielen Jahren im Einsatz. Um diese zu benutzen sind auch keinerlei zusätzlich Optionen nötig. Sie sind in der Datenbank Lizenz schon enthalten.

Kontaktadresse:
Jörn Bartels
Oracle Deutschland B. V. & Co. KG
Riesstr. 25
D-80992 München

Telefon: +49 (0) 89-1430 1120
E-Mail Joern.Bartels@oracle.com
Internet: www.oracle.com