Tabellen und Indizes Reorganisieren, aber wann?

Ernst Leber
MT AG
Ratingen

Schlüsselworte:

Performance, Datenbankanalyse, Tabellenreorganisation, Indexreorganisation

Einleitung

Kernpunkt des Vortrages ist die Performance beim Zugriff auf Daten in Oracle Segmenten. An Hand von Tabellen und Indizes sollen die Unterschiede beim Zugriff auf Daten vor und nach der Reorganisation aufgezeigt werden.

Zunächst soll aufgezeigt werden, ob und wann eine Reorganisation von Tabellen und Indizes sinnvoll ist. Da das Diagnostic und Tuning Kit nicht für jede Oracle Version zur Verfügung steht, sollen andere Oracle-Hilfsmittel gezeigt werden, die für eine solche Analyse genutzt werden können. Als letztes werden Mittel und Wege für die Reorganisation selbst aufgezeigt.

Was ist Fragmentierung

Laut Wikipedia steht Fragmentierung unter anderem „bei blockweiser Datenspeicherung für einen Speicherkapazitätsverlust, wenn innerhalb der Speicherblöcke ungenutzter freier Speicher vorhanden ist“

Der Speicherkapazitätsverlust ist eine Seite, es kommt aber auch zu Performance Einbußen, wenn für das Lesen von Daten aus einer Datenbank mehrere, nicht zusammenhängende, Blöcke gelesen werden müssen.

Grundlagen

In einer Oracle Datenbank werden Tabellendaten in Datenblöcken gespeichert. Mehrere dieser Datenblöcke bilden einen Extent. Aus diesen Extents werden Segmente gebildet, die in einem Tablespace zusammengefasst werden.

Aufbau eines Datenblocks

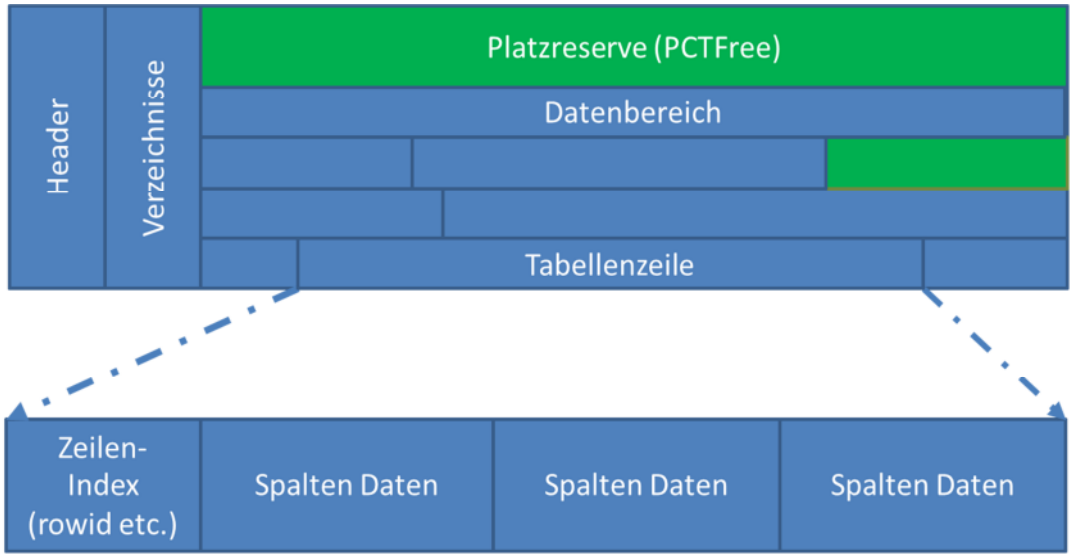

Abb. 1: Schematischer Aufbau eines Datenblocks

Der Header eines Datenblocks beinhaltet unter anderem die Blockadresse. In den Verzeichnisdaten werden Informationen über die Tabellen und Zeilendaten eingetragen. Im Datenbereich werden die eigentlichen Daten der Tabelle bzw. des Indexes gespeichert.

Hier wird betrachtet, wie sich Änderungen in einer Tabelle auf den Datenblock auswirken und welche Folgen diese Änderungen auf den Datenzugriff haben.

Löschen im Datenblock

Beim Löschen von kompletten Zeilen wird der Speicherplatz im Datenblock frei. Der Speicherplatz wird aber erst für das Schreiben von neuen Datenzeilen freigegeben, wenn PCTUsed unterschritten wird. Dieser Parameter kann einmalig beim Anlegen von Tabellen und Indizes angegeben werden.

Update im Datenblock

Bei einem Update einer Tabellenzeile kann es vorkommen, dass diese Zeile zu lang wird, um noch im Datenblock gespeichert zu werden. In diesem Fall wird im Datenblock eine Blockadresse eingetragen, an der der Datensatz gefunden werden kann. In diesem Fall spricht man von „Row Migration“. Dieser Fall kann ebenfalls eintreten, wenn eine Spalte zu einer Tabelle hinzugefügt wird und der Platz im Datenblock für diese Erweiterung nicht ausreicht.

Zeile zu groß für den Datenblock

Beim Speichern von kompletten Dateien in einer Tabelle, z.B. in CLOBs oder BLOBs, reicht unter Umständen die Blockgröße nicht aus, um eine komplette Datei in einem Block zu Speichern. In diesem Fall werden die Daten über mehrere Blocks verteilt

gespeichert. In diesem Fall spricht man von Row-Chaining. Bei Tabellen mit mehr als 255 Spalten wird Row Chaining ebenfalls auftreten, da Oracle diese Tabellen aufteilt.

Folgen der Fragmentierung

Aus den oben gesagten Beispielen ist ersichtlich, dass z.B. beim ‚Aufräumen' einer Tabelle durch Löschen nicht mehr benötigter Daten der frei gewordene Platz im Datenblock nicht immer freigegeben wird.

Das hat zur Folge, dass zum Einen Speicherplatz ‚verschwendet' wird. Zum anderen bedeutet es auch, dass bei einem Full-Table Scan unnötig viele, weil leere Datenblöcke gelesen werden. Der Performancegewinn der durch den Datenbankparameter db_file_multiblock_read_count erreicht werden soll, wird durch das Lesen von halb leeren Datenblöcken ebenfalls verkleinert.

Bei einem Lesezugriff über einen Index hat eine Fragmentierung zur Folge, dass z.B. bei ‚Migrated-Rows' der Index nicht direkt auf den Zieldatenblock zeigt, sondern auf einen Verweis zu einem weiteren Datenblock in dem die zu Lesenden Daten stehen. Hierdurch sind ebenfalls mehrere Lesezugriffe erforderlich.

Feststellen der Fragmentierung

Folgen der Fragmentierung sind, kurz gesagt, sehr viele Lesezugriffe, die sich in den Statistiken der Datenbank durch erhöhte Buffer gets bemerkbar machen.

Falls AWR aus Lizenzgründen nicht eingesetzt werden kann, bietet es sich an, das STATSPACK zu installieren und damit Statistikdaten über die Datenbank zu sammeln.

Außer den vorher genannten Reports kann die Fragmentierung von Tabellen auf verschiedene Arten sichtbar gemacht werden. Im Data Dictionary gibt z.B. die View DBA_TABLES Auskunft über die Eigenschaften einer Tabelle. Voraussetzung hierfür sind gültige Statistiken, die mit DBMS_STATS erstellt wurden.

```
select owner, table_name, pct_free, num_rows,
       blocks, empty_blocks, avg_space, chain_cnt,
       avg_row_len,
       to_char(last_analyzed,'dd.mm.yyyy hh24:mi:ss')
 from dba_tables
order by last_analyzed asc nulls first;
```

Anmerkung: Die Spalte CHAIN_CNT wird nur gefüllt, wenn die Tabelle mit einem Analyze Befehl validiert wurde.

Chained- oder Migrated Rows können ebenfalls mit Hilfe der Statistiken angezeigt werden. Auf Session-Ebene kann hierzu folgendes Statement genutzt werden:

```
SELECT a.name, b.value
  FROM v$statname a, v$mystat b
 WHERE a.statistic# = b.statistic#
   AND lower(a.name) = 'table fetch continued row';
```

Wenn Chained Rows direkt auf Tabellenebene angezeigt werden sollen, kann dies ebenfalls mit einem Analyze-Befehl erfolgen:

```
analyze table ... list chained rows into chained_rows;
```

Die Tabelle 'CHAINED_ROWS' muss vorher z.B. mit dem Script UTLCHAIN.SQL angelegt worden sein.

Für die Analyse von Indizes kann folgendes Script genutzt werden:

```
select owner, index_name, index_type, table_name, blevel,
       leaf_blocks, distinct_keys, avg_leaf_blocks_per_key,
       avg_data_blocks_per_key, clustering_factor
  from dba_indexes;
```

Bei den Indexen sollte auch auf den Clustering Factor geachtet werden. Der Clustering Factor ist ein Maß für die Sortierung der Daten in der Tabelle bezogen auf den Index. D.h. dieser Faktor gibt Auskunft darüber, wieviele Blockzugriffe für das Lesen von allen Daten mit z.B. gleichem Schlüssel erforderlich sind.

Ein hoher Clustering Factor bedeutet, dass bei einem Index Range Scan sehr viele Blockzugriffe nötig sind um die gewünschten Daten zu Lesen, da die Daten in unterschiedlichen Blöcken gespeichert sind.

Eine weitere Möglichkeit ist, einen Index zu validieren und dann in der Tabelle INDEX_STATS das Verhältnis aus den gelöschten Blöcken und den Blöcken des Indexes auszulesen. Ist das Verhältnis größer als 20 % sollte der Index defragmentiert werden.

```
ANALYZE TABLE … VALIDATE STRUCTURE CASCADE;
```

oder

```
ANALYZE INDEX … VALIDATE STRUCTURE.

SELECT * from index_stats;
```

Defragmentieren

Für die Defragmentierung stehen folgende Optionen zu Wahl.

- Alter table … move [online]
  Und danach alter index rebuild [online]
- Export / truncate oder drop / import table oder schema
- Create table as select (CTAS)
- dbms_redefinition
- Alter table… shrink space [compact]

Welche Methode genutzt wird, hängt von mehreren Faktoren ab:

- Erworbene Datenbank Lizenz (siehe unten)
- Vorhandener Speicherplatz

- Tabellengröße
- Größe des Wartungsfensters

Anmerkung: Je nach Datenbanklizenz wird die Wahl der Defragmentierungsmethode weiter eingeschränkt. So sind z.B.: Online Operationen für Indizes und IOT sowie die Online Redefinition von Tabellen mit DBMS_REDEFINITION nur in der Enterprise Edition lizensiert.

Bei einer Reorganisation von Tabellen werden die Indizes in der Regel ebenfalls neu erstellt und sind somit ebenfalls defragmentiert. Nach der Defragmentierung ist es sinnvoll bei Indexen nochmals den Clustering Factor zu betrachten.

Das folgende Listing zeigt den Clustering Factor für identische Tabellen. Tabelle T3 wurde mit ‘`create table t3 as select * from t1 order by col1`’ defragmentiert. Tabelle t2 hingegen durch ‘`create table t2 as select * from t1`’. Nur durch das Sortieren der Daten bei der Neuerstellung wurde bei T3 ein wesentlich besserer Clustering Factor erzielt.

```
INDEX_NAME TABLE_NAME LEAF_BLOCKS DISTINCT_KEYS CLUSTERING NUM_ROWS
                                                FACTOR
---------- ---------- ----------- ------------- ---------- ---------
IDX3       T3         478         71965         624        215895
IDX2       T2         478         71965         215895     215895
```

Kontrolle

Nach der Defragmentierung kann das Ergebnis mit den gleichen Statements geprüft werden, die auch beim Feststellen der Fragmentierung genutzt wurden.

Kontaktadresse:

Ernst Leber
MT AG
Balcke-Dürr-Allee, 9
D-40882 Ratingen

Telefon: +49 (0) 21 02 309 61-0
Fax: +49 (0) 21 02 309 61-101
E-Mail ernst.leber@mt-ag.com
Internet: www.mt-ag.com