Partitionierung – Indizes und Statistiken

Klaus Reimers
ORDIX AG
Paderborn

Schlüsselworte

Local index, global index, prefixed index, nonprefix index, hash partitioned index, unusable index, orphaned index, granularity, dbms_stats, incremental, incremental_stale

Einleitung

Nachdem ich vor einigen Jahren im Rahmen der DOAG Konferenz einen Überblick über die verschiedenen Methoden der Partitionierung gegeben habe und auf der DOAG Database 2013 die Neuerungen in Oracle 12c (Partitionierung) erläutert habe, möchte ich in diesem Vortrag zwei Themenbereiche genauer beleuchten.

Die Themen Indizierung innerhalb von partitionierten Tabellen und die Vorgehensweise bei der Erstellung von Statistiken auf partitionierten Tabellen stehen daher dieses Mal im Mittelpunkt.

Die Partitionierung von Tabellen und Indizes bedeutet die Aufteilung von großen Tabellen und Indizes in kleinere Einheiten, die separat verwaltet und angesprochen werden können.

Die Partitionierung kann zu folgenden Vorteilen führen:

- Reduzierung der Zeit für Verwaltungsaufgaben (kleinere Verwaltungseinheiten)
- Verbesserung der Performance (durch Erhöhung der Parallelität)
- Verbesserung der Verfügbarkeit (durch Reduzierung der Folgen eines Ausfalls)
- Verbesserte Unterstützung für sehr große Datenbanken

Objektiv muss angemerkt werden, dass diese Vorteile insgesamt durch einen erhöhten Verwaltungsaufwand erkauft werden, denn die einzelnen Partitionen müssen definiert, angelegt und verwaltet werden.

Alle Partitionen einer Tabelle oder eines Index besitzen die gleichen logischen, aber unter Umständen andere physikalische Attribute.

Man unterscheidet grundsätzlich vier verschiedene Methoden der Partitionierung:

- Range-Partitionierung
- Hash-Partitionierung
- List-Partitionierung
- System-Partitionierung

Zudem existiert die Composite-Partitionierung oder auch Sub-Partitionierung genannt.

Mit Oracle 11g sind drei erweiterte Verfahren hinzugekommen:

- Interval-Partitionierung
- REF-Partitionierung
- Virtual-Based-Column-Partitionierung

Bei allen Methoden oder erweiterten Verfahren ist sowohl das Thema Indizierung, als auch das Thema Statistiken jeweils nahezu identisch zu betrachten. Daher gehe ich in meinem Vortrag kaum noch auf die unterschiedlichen Methoden ein.

Indizierung

Dieser Abschnitt erläutert das Basismodell für partitionierte Indizes. Ähnlich wie bei Tabellen existiert auch für Indizes die Möglichkeit, Partitionen zu bilden. Grundsätzlich erfolgt die Definition von Partitionen ebenfalls anhand von Syntaxerweiterungen, in diesem Fall als Erweiterung des Kommandos CREATE INDEX.

Indextypen

Eine Indizierung kann sowohl auf partitionierte als auch auf normale Tabellen vorgenommen werden. Seit der Version 8 unterstützt Oracle vier Indextypen, mit deren Hilfe auf die eigenen Anforderungen eingegangen werden kann:

- Global nonpartitioned prefixed Indexes
- Global partitioned prefixed Indexes
- Local prefixed Indexes
- Local nonprefixed Indexes

Hier sind grundsätzlich alle Indexarten (normal / reverse / function based / bitmap) verwendbar. Natürlich kann auch die Kombination aus Tabelle und Index (IOT) verwendet werden.

Global Index

Bei einem Global Index können die Partition Keys einer bestimmten Indexpartition Datensätze referenzieren, die in mehr als einer Tabellen-Partitionierung abgelegt sind, d.h. es existiert keine 1:1-Beziehung zwischen den Partitionen der Tabelle und des Indizes. Im Normalfall ist also ein globaler Index nicht equi-partitioned mit der zugrundeliegenden Tabelle.

Sollte „zufällig” doch eine Equi-Partitionierung vorliegen, wird dies von keinem Oracle-Mechanismus ausgenutzt (Bestimmung eines Ausführungsplans, Partitionsoperationen). Bei einem Global Index müssen nicht einmal die Partition Keys übereinstimmen, d.h. ist die Tabelle partitioniert basierend auf Spalte A, kann der Index partitioniert sein bzgl. der Spalte B. Der nonpartitioned Index ist der normale Index, wie er auch auf einer nicht partitionierten Tabelle auftritt.

Es muss darauf geachtet werden, dass die letzte bzw. höchste Partition auch die entsprechend maximalen Werte der Tabelle aufnehmen kann, alle Werte der Tabelle müssen auch im Index

untergebracht werden. So reagiert Oracle direkt mit einer Fehlermeldung, wenn kein MAXVALUE für die größte Partition deklariert worden ist.

Die Administration eines globalen Index gestaltet sich sehr komplex, da ein Wartungseingriff auf den Index oder auf die Tabelle nicht 1:1 den jeweils anderen Teil betrifft, sondern eine Reorganisation einer Indexpartition mehrere Tabellenpartitionen betreffen kann.

Local Index

Bei einem Local Index beziehen sich alle Partition Keys einer bestimmten Indexpartition immer auf solche Datensätze, die in genau einer Tabellen-Partitionierung abgelegt sind, d.h. es existiert eine 1:1-Beziehung zwischen den Partitionen der Tabelle und des Index. Die Objekte sind damit equi-partitioned.

Oracle konstruiert die Local Indizes so, dass die Partitionsgrenzen mit der zugrundeliegenden Tabelle übereinstimmen. Verwaltungstechnische Maßnahmen wie das Hinzufügen, Splitten oder Löschen von Partitionen werden automatisch auch für die Index-Partitionen vorgenommen. Diese Vorgehensweise stellt sicher, dass die Eigenschaft des Equi-Partitioning aufrechterhalten bleibt.

Local Indizes bieten unter anderem die folgenden Vorteile:

- Die Durchführung einer Verwaltungsaufgabe gegenüber einer Tabellen-Partition beeinflusst nur eine Index-Partition. Die Zeit, die zum Hinzufügen eines Local Index verbraucht wird, ist proportional zur Partitionsgröße.
- Der Cost-Based Optimizer kann geeignete Ausführungspläne auswählen, die ein Minimum an Partitionen betrifft (Pruning).

Prefixed und Nonprefixed Index

Ein Index wird als Prefixed Index bezeichnet, wenn die Indexspalten genau mit dem Partition Key beginnen, danach können noch beliebig andere Spalten folgen. Ein Index wird als Nonprefixed Index bezeichnet, wenn die führenden Spalten vom Index nicht auf den gleichen Spalten des Partition Keys basieren.

Wichtig ist diese Unterscheidung eigentlich nur bei der Definition von Eindeutigkeiten (Uniqueness). Ein Index kann nur dann UNIQUE sein, wenn er PREFIXED ist.

Partitionsverwaltung

Mit der Einführung von Partitionen mussten auch Mechanismen zur Partitionierungsverwaltung geschaffen werden, um den Vorteil der Behandlung kleinerer, überschaubarer Einheiten optimal nutzen zu können. Eine Verwaltungsaufgabe manipuliert eine Partition einer Tabelle oder eines Index. So kann beispielsweise eine neue Partition zu einer Tabelle hinzugefügt werden oder es wird eine Partition aus I/O-Gründen in einen anderen Tablespace verschoben.

Die meisten Verwaltungsaufgaben hinsichtlich der Partitionierungen finden in regelmäßigen Zeitintervallen statt, weil sie vorhersehbar und damit planbar sind. In einer historischen Datenbank

werden gewöhnlich alte Datenbestände, die in einer Partition zusammengefasst sind, beispielsweise monatlich gelöscht. Gleichzeitig wird ein neuer Datencontainer (Partition) bereitgestellt, um die aktuellsten Daten aufzunehmen. Eine andere Klasse von Aufgaben lässt sich nicht vorhersehen, wenn aufgrund von Anwendungs- oder Systemproblemen eine unverzügliche Maßnahme ergriffen werden muss. Beispielsweise kann eine rasche Teilung einer Partition notwendig werden, wenn die Anwendung eine sehr hohe I/O-Last auf einen bestimmten Datenbestand erzeugt.

Man unterscheidet zwischen sogenannten „One Step" und „Three Step" Operationen, um das Sperrverhalten für DDL-Statements auf ein Objekt sowohl untereinander als auch gegenüber DML-Statements festzulegen. One Step Operationen setzen eine DML-Sperre im Exclusive Mode auf die betroffene Tabelle. Diese Operationen, die alle DDL-Statements außer CREATE INDEX beinhalten, werden sehr schnell durchgeführt, d. h. sie halten die DML-Sperre und die entsprechende Data-Dictionary-Sperre nur für einen sehr kurzen Zeitraum. Im Gegensatz dazu stehen die Three Step Operationen, die weniger restriktive DML-Sperren auf die Tabelle anfordern oder nur die betroffene Partition exklusiv sperren. Diese Operationen sind meist sehr lang andauernd (Minuten bis Stunden).

Folgende Operationen gelten beispielsweise als Three Step Operationen:

- ALTER TABLE MOVE PARTITION
- ALTER TABLE SPLIT PARTITION
- ALTER TABLE EXCHANGE PARTITION
- DIRECT PATH LOAD
- CREATE INDEX
- ALTER INDEX REBUILD PARTITION

Die Kommandos laufen in der Regel recht lange, lassen aber zu, dass parallel andere Operationen auf dasselbe Objekt zu.

Daneben existieren noch zwei Kommandos

- ALTER TABLE DROP PARTITION
- ALTER TABLE TRUNCATE PARTITION

Diese Befehle werden nach dem „One Step" Protokoll abgearbeitet, wenn kein globaler Index auf der zugrundeliegenden Tabelle definiert ist, anderenfalls wird nach dem „Three Step" Protokoll verfahren.

Unsable Indizes

Alle Aktionen bei denen innerhalb der Partitionen Datensätze verschoben (ROW MOVEMENT) oder gelöscht werden, können zu UNUSABLE Indizes führen. Dieses Problem tritt vor allem bei globalen Indizes auf, da bei der Bearbeitung von nur einer Partition sofort alle globalen Indizes betroffen sind. Lokale Indizes werden nur auf der bearbeiteten Tabellenpartition unbenutzbar.

Mit Oracle 10g ist eine Syntaxerweiterung erfolgt (UPDATE INDIZES) mit deren Hilfe die Indizes innerhalb des 3-Step-Kommandos direkt rebuildet worden sind. Dieses hat zwar Skripting eingespart, nicht aber Zeit.

Mit Oracle 12c ist dieses Problem bei TRUNCATE und DROP PARTITION etwas entzerrt worden, die Funktionalität der ORPHANED Indizes erleichtert hier die Arbeit des Administrators und die Verfügbarkeit der Indizes. Diese Indizes werden dann nicht mehr UNUSABLE, sondern sind sofort

verwendbar. Die eigentliche Nachpflege erfolgt dann in einem nächtlichen Wartungsfenster automatisch.

Mit Oracle 11g Release 2 wurde die Nutzung des Speicherplatzes für UNUSABLE Indizes oder Indexpartitionen optimiert. Wenn diese Objekte auf den Status UNUSABLE wechseln, wird der gesamte Speicherplatz wieder freigegeben. Zudem werden diese Objekte nicht mehr durch DML-Operationen gepflegt, was bei Massenladevorgängen von Vorteil sein kann.

Statistiken

Der Optimizer versucht den möglichst besten Ausführungsplan für ein Statement zu generieren.

Auf das Rechenmodell des Optimizer wirken diverse Rahmenbedingungen ein:

- Parameter (init.ora / spfile)
- Statistiken in der Datenbank

Der Optimizer ermittelt alle möglichen Ausführungspläne, errechnet dann jeweils einen internen Kostenfaktor und nimmt anschließend den „günstigsten/billigsten“ Ausführungsplan.

In diesem Vortrag werden die Möglichkeiten zur Erstellung möglichst zielführender Statistiken betrachtet. Mit dem PL/SQL-Package DBMS_STATS können Statistiken für den kostenbasierenden Optimizer generiert und verwaltet werden. Diese Statistiken können sowohl im Data Dictionary, als auch im Benutzerschema gespeichert werden.

Mit dem Package DBMS_STATS werden die Statistiken berechnet oder gesetzt. Dabei gibt es einige Parametrisierungen, die auf das Ergebnis und auf die Art der Generierung Einfluss nehmen können.

Granularität

Die Granularität bestimmt die Ebene, auf deren Basis die Statistiken für partitionierte Tabellen und Indizes generiert werden sollen:

- ALL – alle Statistiken auf allen Ebenen werden berechnet
- AUTO – Oracle entscheidet intern über die Art der Erstellung der Statistiken
- GLOBAL – nur die globalen Statistiken werden erzeugt
- PARTITION – nur die lokalen Statistiken werden erzeugt
- GLOBAL and PARTITION – alle Statistiken werden erzeugt (keine Sub-Partitionen)
- SUBPARTITION – nur auf der Sub-Ebene werden Statistiken erzeugt

Incremental

Der Parameter INCREMENTAL dient zur Optimierung der Laufzeit bei der Erstellung von globalen Statistiken. Die Statistiken werden nicht jeweils neu berechnet, sondern die Neuerungen werden eingearbeitet. Dafür muss sich Oracle viele Zwischenergebnisse merken, um diese später wieder verwenden zu können. Dieses geschieht in einer Tabelle des Users SYS im SYSAUX Tablespace.

Incremental Staleness

Die Erzeugung von inkrementellen Statistiken kann trotz der Optimierung in Oracle 11g sehr lange dauern. Mit Oracle 12c kann über diesen Parameter eine weitere Optimierung erreicht werden. Die Optimierung funktioniert auch, wenn einzelne Partitionen über Status STALE sind.

Fazit

In diesem Vortrag werden viele der hier bereits angesprochenen Aspekte in einer zusätzlichen Demo gezeigt und erläutert, welche Verfahren unter welchen Bedingungen in der Praxis zu empfehlen sind.

Kontaktadresse:

Klaus Reimers
ORDIX AG
Westernmauer 12-16
D-33098 Paderborn

Telefon: +49 (0) 5251 / 1063 - 0
Fax: +49 (0) 180 / 1673 490
E-Mail info@ordix.de
Internet: www.ordix.de