Datenbankstatistiken im Griff mit DBMS_STATS

Dierk Lenz
Herrmann & Lenz Services GmbH
Burscheid

Schlüsselworte
Oracle Database, Datenbankstatistiken, Optimizer, Autotask-Infrastruktur

Einleitung

Betrachtet man die Performance von Oracle-Datenbanken, so ist der Optimizer die Kernkomponente, die bei fehlerhaften Ergebnissen den größten Schaden anrichten kann. Als Basis für seine Arbeit steht dem Optimizer eine Sammlung von Objektstatistiken zur Verfügung, die – seit Oracle 10g – automatisch auf den aktuellen Stand gebracht wird.

Mit Oracle 11g ist die Möglichkeit, das Verhalten des Analyseprozesses zu verändern, stark verbessert worden. Damit ist die aufwändige Strategie, das Sammeln der Statistiken über selbst entwickelte Mechanismen durchzuführen, nicht mehr notwendig.

Im Vortrag wird aufgezeigt, wie man die Sammlung der Statistiken kontrollieren und manipulieren kann, falls es notwendig wird, vom Standardschema abzuweichen. Hierbei fließen praktische Erfahrungen aus diversen Projekten ein.

Die Rolle von Statistiken in der Datenbank

Mit Oracle7 wurde erstmals der cost-based Optimizer (CBO) eingeführt. In den ersten Versionen gab es viele Fehloptimierungen uns sogar falsche Abfrageergebnisse. Zum Glück war Oracle mit diesem neuen Feature wie mit vielen anderen auch verfahren: Die damals bewährte Methode, der rule-based Optimizer (RBO), war ebenfalls noch vorhanden.

Mit der Zeit ist sowohl der CBO besser geworden als auch die Anwendungslandschaft komplexer – somit wurde die Ablösung des RBO immer dringender. Allerdings hat es lange gedauert, bis der CBO auf eine breite Akzeptanz gestoßen ist. Dafür gibt es im Wesentlichen zwei Gründe: Einerseits gibt es bei der Einführung einer neuen Technik immer wieder das ein oder andere Negativbeispiel (das zum sofortigen Rückfall auf die alte Technologie geführt hat); andererseits ist ein CBO immer nur so gut wie seine Basisdaten – und die hat die Datenbank zunächst einmal nicht automatisch ermittelt!

Bis einschließlich Oracle9i R2 war der CBO darauf angewiesen, dass der DBA sich irgendwie um das Erstellen der Statistiken gekümmert hat. Hinzu kam, dass auch die Methodik sich geändert hat: Wurde zunächst das `ANALYZE`-Kommando genutzt, so kam mit Oracle8i das PL/SQL-Package `DBMS_STATS` hinzu; die Verwirrung wurde also immer größer.

Erst mit Oracle 10g wurde ein `GATHER_STATS_JOB` im neuen Scheduler automatisch hinterlegt. (Man beachte: wieder eine neue Komponente, parallel zur „gewohnten“ Job-Steuerung mit DBMS_JOB eingeführt!) Für viele DBAs war die größte Schwierigkeit, diesen Job überhaupt zu entdecken – für den neuen Scheduler gab es immerhin einen komplett neuen Satz an DD-Views.

Allerdings war dies der eigentliche Durchbruch des CBO, da nun endlich für jede Datenbank sichergestellt war, dass regelmäßig Statistiken gesammelt wurden.

Um die Serie der Neuerungen zu vervollständigen, wurde mit Oracle 11g der fest eingebundene Scheduler Job wieder abgeschafft und das Sammeln der Statistiken als eine von mehreren Tasks in die Autotask-Infrastruktur verlagert. Diese Komponente ist komplett neu hinzugekommen; im Hintergrund aber „verzahnt“ mit dem Scheduler.

Die aktuelle Version 11 gegeben, stellen sich nun einige Fragen:

Wie aktiviere und deaktiviere ich die Autotask-Infrastruktur?

Welche Parameter kann ich beeinflussen und wie mache ich das?

Welche sonstigen Parameter sind für den Optimizer relevant?

Die Autotask-Infrastruktur

Die Autotask-Infrastruktur bietet die Möglichkeit, vorgegebene Standardaufgaben kontrolliert und wiederkehrend auszuführen. Die Aufgaben sind: „auto optimizer stats collection“, „auto space advisor“ und „sql tuning advisor“; letztere ist standardmäßig deaktiviert. Die passende DD-View hierzu ist `DBA_AUTOTASK_CLIENT`.

Dahinter steht, dass Oracle für diese Tasks, die ausschließlich der Oracle-internen Logistik zuzuordnen sind, einen eigenen Platz bietet. Falls man den kompletten Bereich nicht aktiviert haben möchte, kann man dies mit den Prozeduren `DBMS_AUTO_TASK_ADMIN.ENABLE` und `DBMS_AUTO_TASK_ADMIN.DISABLE` ändern. Letztere Prozedur wird z.B. ausgeführt, wenn man eine Datenbank mit dem DBCA konfiguriert und das Häkchen bei „Enable automatic maintenance tasks“ wegnimmt.

Autotask-Infrastruktur, Scheduler-Windows und -Jobs

Dass man den bei Oracle 10g üblichen `GATHER_STATS_JOB` nicht mehr findet, heißt nicht, dass der Scheduler für Autotasks nicht verwendet wird. Allerdings wird der Job lediglich erstellt, wenn er benötigt wird. Nach Beendigung findet man den Job nicht mehr in `DBA_SCHEDULER_JOBS`, allerdings in `DBA_SCHEDULER_JOB_LOG` bzw. `DBA_SCHEDULER_JOB_RUN_DETAILS`.

Wann und wie laufen nun die einzelnen Tasks? Hierzu ist zunächst ein Blick in die View `DBA_AUTOTASK_WINDOW_CLIENTS` angesagt:

WINDOW_NAME	WINDOW_NEXT_TIME	WINDOW_ACTIVE	AUTOTASK_STATUS	OPTIMIZER_STATS	SEGMENT_ADVISOR	SQL_TUNE_ADVISOR	HEALTH_MONITOR
MONDAY_WINDOW	17.09.12 22:00:00,000000000 EUROPE/VIENNA	FALSE	ENABLED	ENABLED	ENABLED	DISABLED	DISABLED
TUESDAY_WINDOW	18.09.12 22:00:00,000000000 EUROPE/VIENNA	FALSE	ENABLED	ENABLED	ENABLED	DISABLED	DISABLED
WEDNESDAY_WINDOW	19.09.12 22:00:00,000000000 EUROPE/VIENNA	FALSE	ENABLED	ENABLED	ENABLED	DISABLED	DISABLED
THURSDAY_WINDOW	20.09.12 22:00:00,000000000 EUROPE/VIENNA	FALSE	ENABLED	ENABLED	ENABLED	DISABLED	DISABLED
FRIDAY_WINDOW	14.09.12 22:00:00,000000000 EUROPE/VIENNA	FALSE	ENABLED	ENABLED	ENABLED	DISABLED	DISABLED
SATURDAY_WINDOW	15.09.12 06:00:00,000000000 EUROPE/VIENNA	FALSE	ENABLED	ENABLED	ENABLED	DISABLED	DISABLED
SUNDAY_WINDOW	16.09.12 06:00:00,000000000 EUROPE/VIENNA	FALSE	ENABLED	ENABLED	ENABLED	DISABLED	DISABLED

Hier ist für jedes Scheduler-Window, das der Autotask-Infrastruktur zugeordnet ist (Window-Gruppe `MAINTENANCE_WINDOW_GROUP`: `MONDAY_WINDOW`, `TUESDAY_WINDOW`, …), angegeben, ob der entsprechende Client im Window laufen soll oder nicht. Diese Zuordnung wird über eine

überladene Variante der o.a. Prozeduren `DBMS_AUTO_TASK_ADMIN.ENABLE` und `DBMS_AUTO_TASK_ADMIN.DISABLE` spezifiziert.

Als Unterschied zwischen den Autotasks in 11g und dem Scheduler Job in 10g ist also auch die Verwendung unterschiedlicher Windows festzuhalten (`WEEKNIGHT_WINDOW` und `WEEKEND_WINDOW` in 10g; ein Window pro Tag in 11g).

Die einzelnen Windows können den Gegebenheiten des Datenbankbetriebs angepasst werden; es ist aber auch die Verwendung eigener Windows möglich. Diese müssen mit dem API `DBMS_SCHEDULER` der Window-Gruppe `MAINTENANCE_WINDOW_GROUP` zugeordnet werden. Die Zuordnung findet sich in der DD-View `DBA_SCHEDULER_WINGROUP_MEMBERS`:

WINDOW_GROUP_NAME	WINDOW_NAME
MAINTENANCE_WINDOW_GROUP	MONDAY_WINDOW
MAINTENANCE_WINDOW_GROUP	TUESDAY_WINDOW
MAINTENANCE_WINDOW_GROUP	WEDNESDAY_WINDOW
MAINTENANCE_WINDOW_GROUP	THURSDAY_WINDOW
MAINTENANCE_WINDOW_GROUP	FRIDAY_WINDOW
MAINTENANCE_WINDOW_GROUP	SATURDAY_WINDOW
MAINTENANCE_WINDOW_GROUP	SUNDAY_WINDOW

Mit Oracle 11g R2 ist eine weitere PL/SQL-Prozedur hinzugekommen, mit der man das Sammeln der Statistiken aus der Autotask-Umgebung ad hoc vornehmen kann: `DBMS_AUTO_TASK_IMMEDIATE.GATHER_OPTIMIZER_STATS`.

Was passiert beim Automatischen Sammeln der Statistiken?

Die Autotask-Infrastruktur erzeugt einen Scheduler-Job, dessen Name mit `ORA$AT_OS_OPT` beginnt und startet diesen. In der View `DBA_SCHEDULER_JOB_RUN_DETAILS` kann dies nachvollzogen werden:

OWNER	JOB_NAME	ACTUAL_START_DATE
SYS	ORA$AT_OS_OPT_SY_211	13.09.12 22:00:02,134748000 EUROPE/VIENNA
SYS	ORA$AT_OS_OPT_SY_209	12.09.12 22:00:02,282518000 EUROPE/VIENNA
SYS	ORA$AT_OS_OPT_SY_207	11.09.12 22:00:00,965516000 EUROPE/VIENNA
SYS	ORA$AT_OS_OPT_SY_205	10.09.12 22:00:02,138566000 EUROPE/VIENNA
SYS	ORA$AT_OS_OPT_SY_203	09.09.12 22:01:26,779151000 EUROPE/VIENNA

Der Job führt die Prozedur `DBMS_STATS.GATHER_DATABASE_STATS_PROC_JOB` aus. Dabei werden für alle Tabellen in der Datenbank Statistiken gesammelt, bei denen sich mehr als 10% der Daten geändert haben. Oracle führt eine Statistik über die durchgeführten DMLs (`INSERT`s, `UPDATE`s und `DELETE`s), die über die View `DBA_TAB_MODIFICATIONS` abgerufen werden können.

Für diverse Parameter gelten dabei zunächst die Standardeinstellungen. Z.B. gilt für ESTIMATE_PERCENT der Wert DBMS_STATS.AUTO_SAMPLE_SIZE. Hier verwendet Oracle eine Methode, zunächst nur wenige Datensätze zu analysieren (5500) und zu beurteilen, ob die Statistiken sinnvoll erscheinen. Ist dies nicht der Fall, wird die Anzahl der Datensätze schrittweise erhöht (55000, …). Wenn 25% oder mehr Datensätze analysiert werden müssen, wird auf COMPUTE (100%) umgestellt.

So sinnvoll dies zu sein scheint, so gibt es hier und da Tabellen, bei dem der DBA lieber einen Wert für ESTIMATE_PERCENT fest vorgeben möchte. Dies kann man über die Prozedur DBMS_STATS.SET_TABLE_PREFS erreichen. Neben SET_TABLE_PREFS gibt es SET_SCHEMA_PREFS (als Voreinstellung für ein Schema), SET_DATABASE_PREFS (für alle Tabellen der Datenbank) sowie SET_GLOBAL_PREFS (globale Voresinstellungen). Hierbei ist zu beachten, dass die Werte bei SET_SCHEMA_PREFS und SET_DATABASE_PREFS für alle aktuell existierenden Tabellen hinterlegt werden. Kommen neue Tabellen hinzu, gelten für diese die globalen Einstellungen.

Die Abfrage der Werte kann über die Funktion DBMS_STATS.GET_PREFS erfolgen. Werden die Parameter OWNNAME und TABNAME übergeben, so wird der Wert für die angegebene Tabelle ermittelt; werden die Parameter nicht übergeben, so ist es der globale Wert.

Abgelegt werden die Werte in den Tabellen SYS.OPTSTAT_USER_PREF$ und SYS.OPTSTAT_HIST_CONTROL$.

Welche Parameter können modifiziert werden?
Im vorigen Abschnitt wurde bereits ESTIMATE_PERCENT diskutiert. CASCADE bestimmt, ob abhängige Objekte (Indizes) analysiert werden sollen. Mit DEGREE wird der Parallelisierungsgrad bestimmt. METHOD_OPT steuert die Erstellung von Histogrammen.

Mit STALE_PERCENT kann der Schwellwert für die Statistikberechnung angepasst werden. NO_INVALIDATE bietet die Möglichkeit, bestehende Cursor bei neu berechneten Statistiken nicht zu invalidieren, um einen „Compile-Sturm“ zu vermeiden. Mit GRANULARITY kann bei partitionierten Objekten der Umfang der Statistikberechnung bestimmt werden. Mit PUBLISH ist die Kontrolle über den Zeitpunkt des Einspielens der neu berechneten Statistiken ins System möglich.

Als globale Voreinstellung kann mit AUTOSTATS_TARGET bestimmt werden, welche Objekte bei der automatischen Statistikerstellung analysiert werden. Die Standardeinstellung ist AUTO; weitere Möglichkeiten sind ALL und ORACLE.

Optimizer-relevante Serverparameter
Es gibt einige dokumentierte und sehr viele undokumentierte Serverparameter, die das Verhalten des Optimizers steuern. Einige davon werden hier kurz vorgestellt.

Der Parameter db_file_multiblock_read_count steuert die Anzahl der Datenbankblöcke, die maximal bei Full Table Scans (FTS) „am Stück“ gelesen werden. Die Empfehlung (für Versionen 10g und höher) ist, diesen Parameter nicht explizit zu setzen. Intern wird dann mit einem Wert von 128 gearbeitet, allerdings ohne den Optimizer mit der Schätzung von wenig IOs für FTS in die FTS-Ecke zu drängen.

Die Parameter `optimizer_index_caching` und `optimizer_index_cost_adj` helfen ebenfalls bei der Benutzung von Indizes. Der erste gibt den Prozentsatz an, den der Optimizer für Indexblockzugriffe als Cache-Treffer annehmen soll; der zweite kann die Kostenbewertung des Optimizers für FTS- und Indexpläne gegeneinander verschieben. Die Kombination von 90 (0) für `optimizer_index_caching` und 25 (100) für `optimizer_index_cost_adj` ist für viele OLTP-artige Systeme eine gute Grundeinstellung.

Kontaktadresse:
Dierk Lenz
Herrmann & Lenz Services GmbH
Höhestraße 37
51399 Burscheid

Telefon:	+49 2174 6712 0
Fax:	+49 2174 6712 22
E-Mail	dierk.lenz@hl-services.de
Internet:	www.hl-services.de