Oracle 11g – Zehn weniger bekannte Features

Martin Frauendorfer
SAP AG
Walldorf

Schlüsselworte:

11g, Delta, Features, SQL, Monitoring, Statistiken, Kalibrierung, Advisor, DDL, SCN, Nested Loop Join, Full Table Scan, Maintenance Window, Resource Manager

Einleitung

Neben populären Oracle 11g-Features wie Advanced Compression stehen mit diesem Datenbankrelease auch eine Reihe kleinerer Neuigkeiten und Verbesserungen zur Verfügung, die sich im Rahmen der Datenbankadministration und Performanceoptimierung sinnvoll verwenden lassen. Zehn solcher Features sollen im Rahmen dieser Session angesprochen werden.

SAP-Hinweis 1438410 stellt eine Sammlung von SQL-Kommandos zur Verfügung. Im Rahmen dieses Dokuments hier wird bei Bedarf mittels *SQL: „<script_name>.txt“* auf Skripten dieser Sammlung verwiesen.

SQL Monitoring

SQL Monitoring steht im Rahmen des Oracle Tuning Packs zur Verfügung und kann genutzt werden, aktuell langlaufende SQL-Statements zu analysieren. Über klassische Analysemethoden wie Shared Cursor Cache oder die Session-Übersicht hinaus bietet es folgende Alleinstellungsmerkmale:

- Jedes lange laufende SQL-Statement kann über SID, SQL_ID und SQL_EXEC_ID eindeutig identifiziert und analysiert werden.
- Für jeden einzelnen Langläufer werden die Bind-Variablen-Inhalte gespeichert.
- Für jeden Schritt des Ausführungsplans werden bearbeitete Rows, Disk Reads, Buffer Gets, PGA- und PSAPTEMP-Verbrauch ermittelt.
- ASH-Samples können automatisch den jeweiligen Schritten des Zugriffsplans zugewiesen werden.
- Zum Abbruch führende Fehler und PX-Downgrades werden mitprotokolliert.

Die SQL Monitoring-Informationen stehen im wesentlichen über die Views GV$SQL_MONITOR und GV$SQL_PLAN_MONITOR zur Verfügung. Außerdem wurde GV$ACTIVE_SESSION_HISTORY um Spalten wie die SQL_EXEC_ID (eindeutiger Zähler, wenn das selbe SQL-Statement von der selben Session mehrmals ausgeführt wurde) und die SQL_PLAN_LINE_ID erweitert, um SQL Monitoring sinnvoll zu unterstützen. Standardmäßig werden SQL-Statements mit Laufzeiten von mehr als 5 Sekunden in das Monitoring aufgenommen und frühestens eine Minute nach deren Beendigung wieder aus der Übersicht entfernt.

Die Skripten *SQL: “SQL_SQLMonitoring_Overview.txt”, SQL: “SQL_Monitoring_BindVariableContent.txt” und SQL: “SQL_Monitoring_ExecutionPlan.txt”* stellen Auswertemöglichkeiten der SQL Monitoring-Information zur Verfügung. Außerdem kann mittels Oracle-Prozedur DBMS_SQLTUNE.REPORT_SQL_MONITOR ein umfassender Übersichtsreport zu einem gemonitorten SQL-Statement erzeugt werden.

Extended Statistics

Neben ungleichmäßiger Werteverteilung sind korrelierte Spalten ein Hauptgrund für Fehlentscheidungen des Oracle Cost Based Optimizers (CBO) und damit verbundenen langen Laufzeiten von SQL-Statements. Durch spezielle Extended Statistics, sogenannte Column Group Statistics, können kritische Spaltenkorrelationen dem CBO bekannt gemacht werden. Eine solche Spaltengruppe kann mit folgendem Kommando bekannt gemacht werden:

```
SELECT DBMS_STATS.CREATE_EXTENDED_STATS('<owner>', '"<table_name>"', '("<col1>", ..., "<colN>")') FROM DUAL;
```

Bei nachfolgenden Statistikläufen erhält die Spaltengruppe automatisch Statistiken wie jede normale Spalte auch. Durch den NUM_DISTINCT-Wert der Spaltengruppe kann der CBO die Korrelation bestimmen.

Alternativ kann die Definition einer Spaltengruppe und die Statistikerstellung durch folgendes Kommando kombiniert werden:

```
EXEC DBMS_STATS.GATHER_TABLE_STATS('<owner>', '"<table_name>"', METHOD_OPT => 'FOR
COLUMNS ("<col1>", ..., "<colN>") SIZE 1');
```

Kritische Spaltengruppen können in der Regel nur im Rahmen von SQL-Optimierungen bestimmt werden. Ein proaktives Anlegen ist angesichts der Vielzahl der Kombinationsmöglichkeiten von Spalten schwer möglich.

Existierende Extended Statistics können dem View DBA_STAT_EXTENSIONS entnommen werden.

Das Dokument CaseStudy_ExtendedStatistics.txt enthält eine Case Study zu Extended Statistics: Optimierung von Zugriffen auf die Tabelle AUSP durch Multi Column Statistics für die Spalten KLART and ATINN.

SQL Repair Advisor

SQL-Statements können aus verschiedenen Gründen problematisch sein: Sie können falsche Ergebnisse liefern, mit einem Fehler abbrechen oder eine sehr lange Laufzeit aufweisen. Der SQL Repair Advisor kann genutzt werden, einen „Patch“ für das Problem zu erstellen. Dabei handelt es sich um keinen Software-Patch im herkömmlichen Sinn, sondern um eine den Zugriffsplan beeinflussende SQL Plan Baseline.

Der SQL Repair Advisor ist durch folgende Prozeduren des DBMS_SQLDIAG-Pakets implementiert:

- CREATE_DIAGNOSIS_TASK
- EXECUTE_DIAGNOSIS_TASK
- REPORT_DIAGNOSIS_TASK
- ACCEPT_SQL_PATCH
- DROP_SQL_PATCH

Im View DBA_SQL_PATCHES findet man vorgeschlagene und implementierte Patches.

Die Ausführung des SQL Repair Advisors kann einige Zeit in Anspruch nehmen, da er das fragliche SQL-Statement mit zahlreichen unterschiedlichen CBO-Settings ausführt. Bei Bedarf kann der Standard-Timeout von 1800 Sekunden durch Angabe eines höheren TIME_LIMIT-Parameters angepaßt werden.

Das Dokument CaseStudy_SQLRepairAdvisor.txt enthält eine Case Study zum SQL Repair Advisor: Die Ermittlung eines Patches zur Behebung einer falschen Ergebnismenge bei einem kartesischen Merge Join in Verbindung mit einer STAR-Transformation.

I/O Calibration

Die Bestimmung eines optimalen Parallelitätsgrad für I/O-intensive Aktionen wie Datenbankreorganisationen ist oft eine schwierige Aufgabe. Als einfaches Hilfsmittel wurde bislang oft die Anzahl der CPUs in Betracht gezogen. Dies war allerdings nur bedingt hilfreich, da massive Datenbankoperationen meist I/O- und nicht CPU-gebunden sind. Mit Oracle 11g steht nun im Rahmen der Prozedur DBMS_RESOURCE_MANAGER.CALIBRATE_IO eine Möglichkeit zur Verfügung, den möglichen I/O-Durchsatz zu messen.

Es werden unter anderem der maximale Gesamtsystem-Durchsatz für lesendes I/O (max_mbps) und der Einzelprozess-Durchsatz für lesendes I/O (max_pmbps) ermittelt. Auf Basis dieser Werte läßt sich abschätzen, wie hoch eine Parallisierung sein kann, ohne den maximal möglichen I/O-Durchsatz deutlich zu überschreiten.

Auch Oracle Parallel Execution kann das Ergebnis der Kalibierung für eine optimale Parallelisierung nutzen. Dazu muss der Parameter PARAMETER_DEGREE_LIMIT = IO gesetzt sein. Zusätzlich muss PARALLEL_DEGREE_POLICY auf LIMITED oder AUTO stehen.

Automatic Shared Pool Extensions

Während allgemein bekannt sein dürfte, dass die Features Automatic Shared Memory Management (ASMM, SGA_TARGET-Parameter) und Automatic Memory Management (AMM, MEMORY_TARGET-Parameter) zu Größenveränderungen von Shared Pool und Buffer Pool führen können, ist relativ schlecht dokumentiert, dass auch ohne dieses Features Größenveränderungen stattfinden können, um Fehler wie ORA-04031 (Überlauf des Shared Pools) zu vermeiden. Dieses Feature kann nur durch explizites Setzen von _MEMORY_IMM_MODE_WITHOUT_AUTOSGA = FALSE ausgeschaltet werden. Da die Vermeidung von ORA-04031-Fehlern aber auch sinnvoll sein kann, ist es im allgemeinen sinnvoller, die möglichen Größenänderungen zu erlauben und den Underscore-Parameter nicht zu setzen.

Das Dokument CaseStudy_SharedPoolExtensions.txt enthält eine Analyse einer umfangreichen Größenveränderung von Shared Pool und Buffer Pool aufgrund von _MEMORY_IMM_MODE_WITHOUT_AUTOSGA = TRUE.

DDL Lock Timeout

In änderungsintensiven Systemen existieren für manche Tabellen wie BALHDR oder EDIDC oft nahezu kontinuierlich offene, nicht committete Änderungen. Dies führt dazu, dass DDL-Operationen keine exklusive Tabellensperre erhalten können und mit ORA-00054-Fehlern abbrechen. Die BR*TOOLS von SAP wiederholen in einem solchen Fall die Operation mehrere Male mit einem gewissen zeitlichen Abstand, im ungünstigsten Fall brechen aber auch alle nachfolgenden Versuche mit ORA-00054 ab.

Mit Hilfe des Parameters DDL_LOCK_TIMEOUT läßt sich nun eine Wartezeit in Sekunden festlegen, während der auf die Verfügbarkeit des Tabellenlocks gewartet wird. Neue Änderungsanfragen werden dabei blockiert, es kann während der Wartezeit also zu Enqueue-Wartesituationen von anderen Prozessen kommen. Daher sollte DDL_LOCK_TIMEOUT immer nur so hoch gewählt werden wie eine Verzögerung durch Enqueue-Waits für das laufende System tolerabel ist. Ist die Tabellensperre bis zum Ablauf der Wartezeit nicht verfügbar, bricht die DDL-Operation mit ORA-00054 ab. Die Wahrscheinlichkeit für einen ORA-00054 ist aber deutlich geriner als ohne die DDL_LOCK_TIMEOUT-Wartezeit.

Unrecoverable SCN Tracking

NOLOGGING-Operationen, wie sie z.B. häufig beim Anlegen von Indizes in BW-Systemen verwendet werden, werden standardmäßig in den Controlfiles verzeichnet. Grund dafür ist, dass NOLOGGING-Operationen nicht recovert werden können und durch den Eintrag im Controlfile diese „unrecoverable SCN“ für Tools wie RMAN hinterlegt wird. Bei massiv parallelen NOLOGGING-Operationen können die Controlfile-Zugriffe zum limitierenden Faktor werden. Da die „unrecoverable SCN“ für einen sicheren Systembetrieb nicht nötig ist, kann auf den Update des Controlfiles verzichtet werden. Ab Oracle 11.2.0.2 existiert für diesen Zweck der Parameter DB_UNRECOVERABLE_SCN_TRACKING = FALSE. Alternativ kann in allen aktuellen Oracle-Releases auch Event 10359 auf Level 1 gesetzt werden. Der neue Parameter ist aber in jedem Fall sprechender.

Nested Loop Join Implementation

Seit Oracle 11g werden standardmäßig zwei Nested Loop Join-Optimierungen verwende.

Das sogenannte Nested Loop Join Batching sorgt dafür, dass der Absprung vom inneren Index auf die innere Tabelle des Joins nicht bei jedem Datensatz erfolgt, sondern erst dann, wenn mehrere ROWIDs gesammelt wurden. Das kann durch Multiblock-Lesezugriffe zu deutlichen Performancevorteilen im Vergleich zu normalen Single Block Reads führen. Dieses Feature äußert sich im Ausführungsplan dadurch, dass zwei Zeilen mit „NESTED LOOPS“ zu finden sind, obwohl man beim Joinen von zwei Tabellen nur einen dieser Schritte erwarten würde. Ein „NESTED LOOPS“ bezieht sich auf den inneren Index, der andere auf die innere Tabelle.

Eine zweite Optimierung stellt sicher, dass die Position im inneren Index des Joins gemerkt wird und nachfolgende Zugriffe keinen kompletten INDEX SCAN mehr durchführen müssen, sondern direkt im fraglichen Leaf Block landen. Das ist nur dann hilfreich, wenn aufeinanderfolgende Abstiege auf den inneren Index auch tatsächlich im gleichen Bereich des Indexes landen. Liegt eine solche Situation vor, kann sich die Anzahl der Buffer Gets des Nested Loop-Joins im Vergleich zu Oracle 10g deutlich reduzieren.

Full Table Scan Implementation

Blöcke, die mit Oracle <= 10g im Rahmen von Full Table Scans von Platte gelesen werden, werden normalerweise in den Oracle Buffer Pool geladen („db file scattered read“), dann allerdings oft schnell wieder verdrängt.

Mit Oracle 11g werden die Blöcke aus Full Table Scans dagegen vom Schattenprozess direkt – also unter Umgehung des Buffer Pools - von Platte gelesen („direct path read“).

Dieses Verhalten ist meist wünschenswert, weil es den unnötigen Buffer Pool-Overhead vermeidet. Infolgedessen werden „db file scattered read“-Waits im Vergleich zu Oracle 10g seltener, „direct path read“-Waits tauchen häufiger auf.

Maintenance Windows and Resource Manager

Sowohl Maintenance Windows als auch der Resource Manager existieren bereits seit Oracle 10g, allerdings ist neu, dass alle Maintenance Windows standardmäßig einen Resource Manager-Plan zugeordnet haben, der zur ungewollten Aktivierung des Resource Managers während solcher Zeitfenster führt. Im Alert Log wird dies durch Einträge der folgenden Art protokolliert:

```
Setting Resource Manager plan SCHEDULER[0x124EED6]:DEFAULT_MAINTENANCE_PLAN via scheduler window
```

Der aktivierte Resource Manager kann nun aus verschiedenen Gründen für Wartesituationen verantwortlich sein. Zu erkennen sind diese an Wait Events wie „resmgr:cpu quantum“ und „resmgr:resource group CPU method“. Um diese unnötigen Wartesituationen zu vermeiden, kann die Zuweisung des Resource Manager-Plans zu den Maintenance Windows mit Kommandos der folgenden Art entfernt werden (SAP-Hinweis 1579946):

```
EXECUTE DBMS_SCHEDULER.SET_ATTRIBUTE('<maintenance_window_name>', 'RESOURCE_PLAN', NULL);
```

Im Dokument CaseStudy_ResourceManagerWaits.txt ist die Analyse eines Auftretens von “resmgr:cpu quantum” enthalten.

Kontaktadresse:

Martin Frauendorfer
SAP AG
Dietmar-Hopp-Allee 16
D-69190 Walldorf

Telefon: +49 (0) 6227 – 60 50 70
E-Mail martin.frauendorfer@sap.com
Internet: www.sap.de