Vorstellung

DIE QSC AG

Datum: 09.06.2015

Daten & Fakten

12 Standorte
In Deutschland

30 Jahre
Erfahrung

Hochmodernes IP-Basiertes
Sprach-Daten-Netz (NGN)

455,5
Mio. € Umsatz in 2013

Support für
15.000 Endanwender

1700
Mitarbeiter

Im **TEC DAX**
seit 2004

Über **10.000** m2
TIER 3+ Rechen-
zentrumsfläche

Zertifizierungen
ISO 27001:2005
Experton Cloud Leader 2013
TÜV Service tested: sehr gut
IT-Betrieb nach ITIL
Projekte nach PMI®

QSC AG

Der SQL Tuning Advisor

AUTOMATISCHE AUSWERTUNG

Datum: 09.06.2015

Benjamin Kurschies

QSC AG

About me

- Benjamin Kurschies
- - Seit 7 Jahren Oracle DBA
- - spezialisiert auf HA & Performance Tuning
- - Oracle Certified Professional
- - Oracle Certified Expert: Performance Tuning
- - Oracle Certified Expert: RAC & GI

Automatic SQL Tuning Advisor

1. Jede Nacht analysiert die Oracle Datenbank den Workflow des Tages
2. Alle Empfehlungen können sehr einfach automatisch angewendet werden:

```
BEGIN
DBMS_SQLTUNE.SET_TUNING_TASK_PARAMETER(
    task_name => 'SYS_AUTO_SQL_TUNING_TASK',
    parameter => 'ACCEPT_SQL_PROFILES', value => 'TRUE');
END;
/
```

ENDE

Vielen Danke für ihre Aufmerksamkeit

. . .

Oder doch nicht?

Agenda

1. Voraussetzungen
2. Warum nicht "automatic accept SQL profiles"
3. Findings anzeigen
4. Findings analysieren

Automatic SQL Tuning Advisor: Voraussetzung

1. Oracle EE & Diagnostic Pack & Tuning Pack

2. Automatic Maintenance Tasks sind aktiv
   - select operation_name, status from DBA_AUTOTASK_TASK

Mehr Informationen: z.B.
http://www.gokhanatil.com/2010/02/automated-maintenance-tasks-in-oracle-11g.html

Agenda

1. Voraussetzungen
2. Warum nicht "automatic accept SQL profiles"
3. Findings anzeigen
4. Findings analysieren

Warum nicht "automatic accept SQL profiles"

Folgendes Szenario:

Alle Maintenance Tasks sind aktiv, SQL Profile werden automatisch akzeptiert.

Es läuft täglich ein Report, der die Umsätze des vergangenen Tages anzeigt:

```
Select sum(costs), sum(gains) from sales where timestamp between :b1 and :b2
```

Alle notwendigen Indizes sind angelegt, der Report ist zügig erstellt.

Am 02.01. läuft der Jahresabschluss

```
Select sum(costs), sum(gains) from sales where timestamp between :b1 and :b2
```

Dieser benötigt – natürlich – deutlich länger.

Am darauffolgendem Tag benötigt der Tagesreport viel länger… was ist passiert?

Agenda

1. Voraussetzungen
2. Warum nicht "automatic accept SQL profiles"
3. Findings anzeigen
4. Findings analysieren

Automatis SQL Advisor: Findings anzeigen

Der "normale" Weg

```
SELECT DBMS_SQLTUNE.REPORT_AUTO_TUNING_TASK FROM dual;
```

Vorteil:

Schnelle Möglichkeit um zu prüfen, ob es der DB "gut geht" – z.B. wenn gerade Performanceprobleme von Anwendern gemeldet werden

Nachteil:

Viel Aufwand wenn dies für viele Datenbanken immer wieder gemacht werden soll.

Automatis SQL Advisor: Findings anzeigen

Manuell – notwendige Tabellen

- DBA_ADVISOR_FINDINGS
  - Alle findings (nicht nur vom Automatic SQL Tuning Advisor) werden hier zusammengefasst
  - z.B. MESSAGE -> "Some alternative execution plans for this statement were found…"
- DBA_ADVISOR_RECOMMENDATIONS
  - Die zu den Findings gehörenden Empfehlungen werden hier gespeichert
  - z.B. TYPE -> "ALTERNATIVE PLAN", BENEFIT -> "3000"
- DBA_ADVISOR_ACTIONS
  - Hier steht welche Aktionen genau empfohlen werden.
  - z.B. COMMAND -> "CREATE SQL PLAN BASELINE"
- DBA_ADVISOR_EXECUTIONS
  - Hier steht, wann welche Analyse gemacht wurde
- DBA_ADVISOR_OBJECTS
  - Hier sind alle Objekte dokumentiert. Objekte sind z.B. SQL Statements, Tabelle, Indizes, usw.

Automatis SQL Advisor: Findings anzeigen

Manuell – SQL Statement (Beispiel)

```
Select ar.type,
       ao.attr1 as sql_id,
       ar.benefit/100||'%' as benefit,
       ae.execution_end as finding_date
       CASE ar.TYPE
            WHEN 'SQL PROFILE' then …
            WHEN 'INDEX' then …
       END as accept_command
from DBA_ADVISOR_FINDINGS af
inner join DBA_ADVISOR_RECOMMENDATIONS ar on af.finding_id=ar.finding_id
inner join DBA_ADVISOR_ACTIONS aa on ar.rec_id=aa.rec_id
inner join DBA_ADVISOR_EXECUTIONS ae on ae.execution_name=af.execution_name
inner join DBA_ADVISOR_OBJECTS ao on ae.execution_name=ao.execution_name and
aa.object_id=ao.object_id
where af.task_name ='SYS_AUTO_SQL_TUNING_TASK' and ar.benefit > 3000
```

Automatis SQL Advisor: Findings anzeigen

Manuell – SQL Statement (Ausgabe)

Automatic SQL
ing Advisor - Findi

Automatis SQL Advisor: Findings anzeigen

Manuell – Vorteile bisher:

- Ausgabe ist übersichtlicher - Eine Zeile pro Finding
- Filterung möglich, z.B. nur ab 30% Benefit, nur Index Erstellung usw.
- Ausgaben sind von einem Monitoring auswertbar (!)

Agenda

1. Voraussetzungen
2. Warum nicht "automatic accept SQL profiles"
3. Findings anzeigen
4. Findings analysieren

Automatis SQL Advisor: Findings analysieren

Es gibt 4 Arten von Findings

- RESTRUCTURE SQL
  - z.B. select * from table where to_number(a)=4 -> select * from table where a='4'
  - Muss in der entsprechenden Anwendung umgesetzt werden (alternativ: function based index)
  - Es muss normalerweise nicht geprüft werden, ob es sinnvoll ist
- INDEX
  - Nur jemand, der die Applikationslogik kennt (z.B. Applikationshersteller) kann beurteilen, ob der zusätzliche Index sinnvoll ist oder die anderen Operationen auf dieser Tabelle zu sehr verlangsamen.
- SQL PROFILE
  - Nur jemand, der die Applikationslogik kennt (z.B. Applikationshersteller) kann beurteilen, ob der neue Ausführungsplan in allen Szenarien passt – siehe "automatic accept SQL profiles"
- PARALLEL EXECUTION
  - Verursachen in Summe mehr CPU aufwand
  - Sollte nur verwendet werden, wenn vom Systemadministrator freigegeben (ausreichend CPU Ressourcen) ist.

Automatis SQL Advisor: Findings analysieren

SQL Profile anzeigen

- Die Ausführungspläne sind in der Tabelle "dba_advisor_sqlplans" gespeichert
- Alle Zeilen mit der selben PLAN_ID gehören zu dem selben Ausführungsplan
- Die ID gibt die Reihenfolge innerhalb eines Ausführungsplans an

Automatis SQL Advisor: Findings analysieren

SQL Profile anzeigen: Beispiel SQL

```
SELECT
  '|' "|", LPAD(id,4,' ') "  ID"
 ,'|' "|", LPAD(' ', 2*DEPTH) || operation OPERATION
 ,'|' "|", object_name NAME
 ,'|' "|", TO_CHAR(NVL(CARDINALITY,0), '999G999G999') as "ROWS"
 ,'|' "|", TO_CHAR(NVL2(bytes,round(bytes/1024),0),'999G999G999')  kb
 ,'|' "|", TO_CHAR(NVL2(temp_space,round(temp_space/1024),0),'999G999G999') "TempSpc (kb)"
 ,'|' "|", TO_CHAR(NVL(COST,0),'999G999G999') COST
 ,'|' "|", TO_CHAR(NVL(IO_COST,0),'999G999G999') IO_COST
 ,'|' "|", LPAD(NVL2(COST,CASE WHEN COST != 0 THEN round((COST-io_cost)/COST*100) ELSE 0 END,0)||' %',8,' ') PCT_CPU
 ,'|' "|", NVL2(time,TO_CHAR(trunc(time/3600),'FM9900')||':'||TO_CHAR(TRUNC(MOD(time,3600)/60),'FM00') || ':'||TO_CHAR(MOD(time,60),'FM00'),NULL) TIME
 ,'|' "|"
FROM dba_advisor_sqlplans
WHERE SQL_ID=&SQL_ID
AND attribute='Original with adjusted cost'
ORDER BY plan_id, id, parent_id;
```

Automatis SQL Advisor: Findings analysieren

SQL Profile anzeigen: Ausgabe

ATTRIBUTE	|	ID	|_1	OPERATION	|_2	NAME	|_3	ROWS	|_4	KB	|_5	TempSpc (kb)	|_6	COST	|_7	IO_COST	|_8	PCT_CPU	|_9	TIME	|_10
Original with adjusted cost	|	0	|	SELECT STATEMENT	|		|	1	|	0	|	0	|	6	|	5	|	17 %	|	00:00:01	|
Original with adjusted cost	|	1	|	COUNT	|		|	0	|	0	|	0	|	0	|	0	|	0 %	|		|
Original with adjusted cost	|	2	|	VIEW	|		|	1	|	0	|	0	|	6	|	5	|	17 %	|	00:00:01	|
Original with adjusted cost	|	3	|	SORT	|		|	1	|	0	|	0	|	6	|	5	|	17 %	|	00:00:01	|
Original with adjusted cost	|	4	|	TABLE ACCESS	|	BKN_BILANZ_JOB	|	1	|	0	|	0	|	5	|	5	|	0 %	|	00:00:01	|
Original with adjusted cost	|	5	|	INDEX	|	X6_BKN_BILANZ_JOB	|	1	|	0	|	0	|	4	|	4	|	0 %	|	00:00:01	|

Automatis SQL Advisor: Findings analysieren

Weitere Automatisierung: SQL Function

```
create function asta_plan (
   v_type_id in number,
   v_object_id in number
   )
return type_asta_plan_table PIPELINED
AS
…
BEGIN
        CASE v_type_id
                WHEN 1 then V_TYPE:='SQL PROFILE'
                WHEN 2 then V_TYPE:='PARALLEL EXECUTION'
                WHEN 3 then V_TYPE:='INDEX'
                ELSE V_TYPE:='N/A';
        END CASE;
        select ATTR1 into V_SQL_ID from dba_advisor_objects where object_id=V_OBJECT_ID;
        select distinct SQL_TEXT into V_SQL from dba_hist_sqltext where sql_id=V_SQL_ID;
        PIPE ROW(type_asta_plan('Analyze von SQL_ID '||V_SQL_ID));
…
```

Q & A

Fragen?

QSC AG

Besuchen Sie uns

qsc.de
blog.qsc.de

xing.com/companies/qscag
facebook.com/QSCAG

twitter.com/QSCAGPresse
youtube.com/qscgermany