OPTIMIZAREA INSTANȚEI ȘI TEHNICI DE DIAGNOZĂ

Suport Laborator Saptămâna 06

FEAA Master SIA/SDBIS

Cuprins

OPTIMIZAREA INSTANȚEI ȘI TEHNICI DE DIAGNOZĂ

În laboratorul trecut am discutat despre instanţa Oracle, despre felul în care aceasta poate fi pornită sau oprită, dar şi despre alocarea spaţiului pentru diferite tipuri de segmente: tabele, indecşi etc. Ne propunem în acest laborator să acoperim câteva concepte şi tehnici de optimizare de bază. Domeniul este extrem de vast, prin urmare vom face doar o introducere în ceea ce înseamnă optimizarea unei baze de date Oracle, suficient însă pentru a pregăti terenul unei aprofundări ulterioare, desigur, pentru cei ce vor dori acest lucru.

Vom aborda optimizarea bazei de date din perspectiva a doua direcţii mari: optimizarea instanţei Oracle şi optimizarea interogărilor SQL.

Optimizarea instanței Oracle

Ne amintim că instanţa Oracle nu era nimic altceva decât o colecţie de procese şi structuri de memorie. Ei bine, dacă acestea nu sunt configurate corespunzător, atunci e posibil ca aplicaţiile care folosesc baza de date să nu funcţioneze optim. De asemenea, pentru aplicaţii de tip OLTP, puternic tranzacţionate, este foarte importantă dimenzionarea corectă a dimensiunii şi numărului de grupuri de tip redolog.

Optimizarea structurilor de memorie

Să începem cu memoria alocată instanţei. Stim deja de SGA (System/Shared Memory Area) şi PGA (Program Global Area). Le putem configura individual sau putem să lăsăm serverul Oracle să facă automat optimizarea acestor structuri de memorie, dar doar dacă-l configurăm cu ceea ce se numeşte AMM (Automatic Memory Management). Oracle va urmări gradul de încărcare a serverului şi va decide care e dimensiunea optimă pentru zonele de memorie asociate. Configurăm AMM prin intermediul parametrului "memory_target".

Cât e "memory_target" pe sistemul dumneavoastră? Dar "memory_max_target"? Care e diferența dintre ele?

```
SYS@SQL> show parameter memory_target
NAME                                 TYPE        VALUE
------------------------------------ ----------- -------------
memory_target                        big integer 352M

SYS@SQL> show parameter memory_max_target
NAME                                 TYPE        VALUE
------------------------------------ ----------- ------------
memory_max_target                    big integer 352M
```

Cum ar fi să configurăm "memory_max_target" cu încă 200MB? Folosim comanda de mai jos (va trebui să faceţi calculul în funcţie de valoarea curentă a parametrului "memory_max_target" de pe calculatorul dumneavoastră):

```
SYS@SQL> alter system set memory_max_target=552M scope=spfile;
```

```
System altered.
```

*Comanda **ALTER SYSTEM SET** este folosită pentru a configura diverşi parametri ai instanţei Oracle. Totuşi, există două mari categori: parametri **statici** şi parametri **dinamici**. Cei statici, aşa cum este şi "memory_max_target", nu pot fi modificaţi direct, ci doar în fişierul de parametri, urmând ca noua valoare să fie luată în considerare doar după restartarea instanţei Oracle. Observaţi clauza "scope=spfile". Cei dinamici poti fi modificaţi direct, fără a fi necesară repornirea instanţei.*

Dăm restart la instanţa Oracle şi inspectăm noua valoare a parametrului "memory_max_target":

```
SYS@SQL> shutdown immediate
Database closed.
Database dismounted.
ORACLE instance shut down.

SYS@SQL> startup
ORACLE instance started.

Total System Global Area  578813952 bytes
Fixed Size                  3048200 bytes
Variable Size             494931192 bytes
Database Buffers           75497472 bytes
Redo Buffers                5337088 bytes
Database mounted.
Database opened.

SYS@SQL> show parameter memory_max_target

NAME                                 TYPE        VALUE
------------------------------------ ----------- -----------
memory_max_target                    big integer 552M
```

Verificăm valoarea parametrului "memory_target" şi-i mai adăugăm 100MB:

```
SYS@SQL> show parameter memory_target

NAME                                 TYPE        VALUE
------------------------------------ ----------- -----------
memory_target                        big integer 352M

SYS@SQL> alter system set memory_target = 452M;

System altered.
```

Observaţi felul în care a fost setat parametrul "memory_target". Dat fiind că acesta este un parametru dinamic, nu a mai fost necesară repornirea instanţei.

Pentru a şti dacă memoria configurată e suficientă se poate inspecta view-ul sistem *V$MEMORY_TARGET_ADVICE*, dar doar dupa ce baza de date a fost deschisă şi folosită o perioadă mai lungă, astfel încât Oracle să poată face o recomandare pertinentă, în funcţie de gradul de încărcare şi de tiparul tranzacţiilor rulate.

Corelaţi întodeauna memoria alocată serverului Oracle cu cea disponibilă în sistemul de operare! Altfel, performanţa server-ului se va degrada considerabil datorită procesului de "swapping", adica a alocării zonelor de memorie care nu încap în memoria fizică, pe disc.

Optimizarea numărului de procese

Laboratorul trecut am discutat despre diferenţa dintre procesele "background" şi "foreground". Am văzut că în arhitecturile server dedicate, pentru fiecare conexiune, un nou proces "foreground" este alocat. În funcţie de felul în care aplicaţiile se conectează la baza de date sau de numărul lor e posibil să trebuiască să măriţi limita maximă de procese ce pot fi create de instanţa Oracle. Acest lucru se face prin intermediul parametrului "processes".

Temă pentru acasă: *identificaţi valoarea curentă a parametrului "processes" de pe instalarea Oracle pe care o aveţi disponibilă. Măriţi limita cu o sută. Ce puteţi spune despre acest parametru?*

Adăugarea/dimensionarea grupurilor redolog

Atunci când au loc foarte multe modificări în baza de date (INSERT-uri, UPDATE-uri, DELETE-uri etc.), toate acestea sunt jurnalizate în fişierele redolog. Daca acestea sunt scrise foarte rapid, trecerea de la un grup redolog la altul s-ar putea să aiba un impact negativ asupra performanţei server-ului. În plus, dacă baza de date rulează în modul arhivat (vom aprofunda ce înseamnă asta în laboratorul despre "Backup şi Recovery") e foarte important ca numărul grupurilor redolog să fie suficient de mare astfel încât procesul de arhivare să poată să ţină pasul cu ritmul de scriere în fişierele redolog.

O primă indicaţie că fişierele redolog nu sunt dimensionate corect o găsim în fişierul "alert" al bazei de date. Acesta este un log care conţine mesaje informative despre funcţionarea bazei de date. Putem găsi în el mesaje de eroare, când a fost pornită sau oprită instanţa oracle şi multe altele. Locaţia acestui fişier o putem identifica folosind interogarea de mai jos:

```
SYS@SQL> column name format a20
SYS@SQL> column value format a50
SYS@SQL> select name, value from v$diag_info where name in ('Diag Trace', 'Diag Alert');

NAME                 VALUE
-------------------- --------------------------------------------
Diag Trace           C:\APP\TALEK\diag\rdbms\testdb\testdb\trace
Diag Alert           C:\APP\TALEK\diag\rdbms\testdb\testdb\alert
```

În locația indicată de "Diag Alert" se află fișierul de alertă în format XML, iar în "Diag Trace" versiunea text a acestuia. Fisierul este denumit "alert_<nume_inst>", cu extensia "xml" sau "log".

Deschideți fișierul alert.log și identificați momentul la care instanța Oracle a fost pornită.

Mesaje ca cele de mai jos indică o dimensionare necorespunzătoare a dimensiunii fișirelor redolog sau a grupurilor redolog asociate:

```
Sat Mar 25 13:15:24 2017
Thread 1 cannot allocate new log, sequence 63
Private strand flush not complete
  Current log# 2 seq# 62 mem# 0: C:\APP\TALEK\ORADATA\TESTDB\REDO02.LOG
Sat Mar 25 13:15:25 2017
Thread 1 advanced to log sequence 63 (LGWR switch)
  Current log# 3 seq# 63 mem# 0: C:\APP\TALEK\ORADATA\TESTDB\REDO03.LOG
Sat Mar 25 13:15:40 2017
Thread 1 cannot allocate new log, sequence 64
Checkpoint not complete
  Current log# 3 seq# 63 mem# 0: C:\APP\TALEK\ORADATA\TESTDB\REDO03.LOG
```

Pentru a mai adăuga un grup redolog putem folosi următoarea succesiune de comenzi. Mai întâi, să vedem ce grupuri redolog avem disponibile:

```
SYS@SQL> select group#, members, bytes/1024/1024 size_mb from v$log order by group#;

    GROUP#    MEMBERS    SIZE_MB
---------- ---------- ----------
         1          1         50
         2          1         50
         3          1         50
```

Apoi, desigur, să vedem ce fișiere redolog sunt asociate fiecărui grup redolog:

```
SYS@SQL> column member format a40
SYS@SQL> select group#, member from v$logfile order by group#;

    GROUP# MEMBER
---------- ----------------------------------------
         1 C:\APP\TALEK\ORADATA\TESTDB\REDO01.LOG
```

```
         2 C:\APP\TALEK\ORADATA\TESTDB\REDO02.LOG
         3 C:\APP\TALEK\ORADATA\TESTDB\REDO03.LOG
```

Pentru a adăuga un al patrulea grup redolog, putem folosi următoarea comandă:

```
SYS@SQL> alter database add logfile ('C:\APP\TALEK\ORADATA\TESTDB\REDO04.LOG') size 50M;

Database altered.
```

Verificaţi câte grupuri redolog sunt disponibile după rularea comenzii de mai sus.

Raportul AWR (Automatic Workload Repository)

Baza de date Oracle colecteaza şi calculează date despre o multitudine de indicatori: procentul de utilizare a bazei de date, evenimente de aşteptare, instrucţiunile SQL cu impactul cel mai mare asupra resurselor sistemului şi aşa mai departe. Indicatorii sunt calculaţi pentru activitatea bazei de date derulată într-un anumit interval: de la momentul T0 până la momentul T1. Acestea poartă denumirea de "snapshot"-uri AWR. Implicit, baza de date ia câte un snapshot la fiecare oră, dar acestea pot fi create şi manual, folosind comanda de mai jos:

```
SYS@SQL> EXEC DBMS_WORKLOAD_REPOSITORY.create_snapshot;

PL/SQL procedure successfully completed.
```

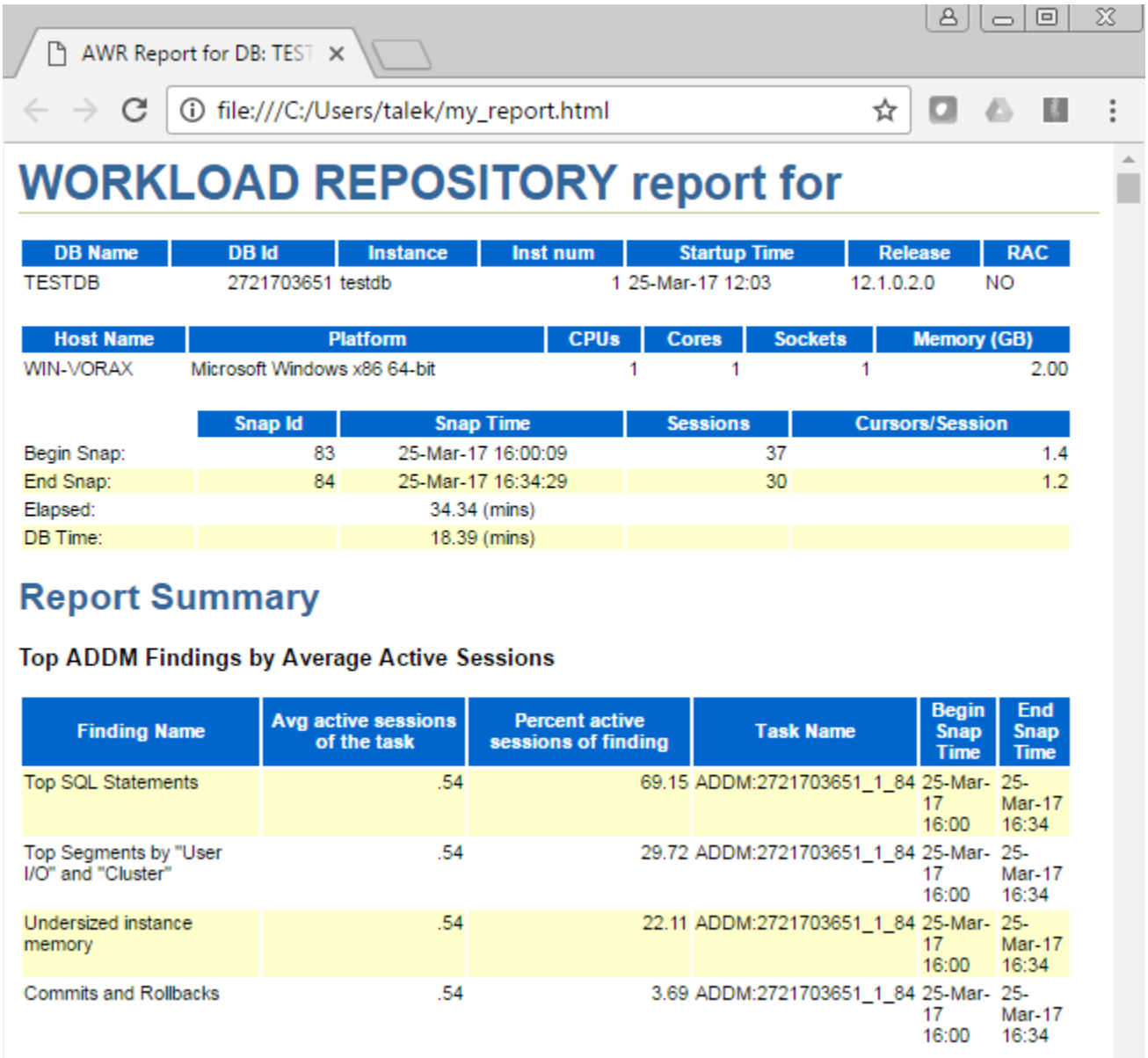
Raportul AWR conţine o sumedenie de informaţii cu privire la ce s-a întâmplat cu baza de date în intervalul dintre două snapshot-uri. Multe dintre aceste informaţii sunt greu digerabile pentru cei ce nu cunosc bine arhitectura şi bucătăria internă a server-ului Oracle, dar este un prim punct de plecare atunci când suspectăm probleme de performanţă sau alte disfuncţionalităţi ale instanţei. Pentru a genera acest raport folosim următorii paşi:

```
SYS@SQL> host echo %CD%
C:\Users\talek

SYS@SQL> @?/rdbms/admin/awrrpt

Current Instance
~~~~~~~~~~~~~~~~

   DB Id    DB Name      Inst Num Instance
----------- ------------ -------- ------------
 2721703651 TESTDB              1 testdb
```

```
Specify the Report Type
~~~~~~~~~~~~~~~~~~~~~~~
AWR reports can be generated in the following formats.  Please enter the
name of the format at the prompt.  Default value is 'html'.

'html'          HTML format (default)
'text'          Text format
'active-html'   Includes Performance Hub active report

Enter value for report_type: html

Type Specified:  html


Instances in this Workload Repository schema
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

   DB Id     Inst Num DB Name      Instance     Host
------------ -------- ------------ ------------ ------------
* 2721703651        1 TESTDB       testdb       WIN-VORAX

Using 2721703651 for database Id
Using          1 for instance number


Specify the number of days of snapshots to choose from
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
Entering the number of days (n) will result in the most recent
(n) days of snapshots being listed.  Pressing <return> without
specifying a number lists all completed snapshots.


Enter value for num_days: 1
Listing the last day's Completed Snapshots

                                                        Snap
Instance     DB Name        Snap Id    Snap Started    Level
------------ ------------ --------- ------------------ -----
testdb       TESTDB              79 25 Mar 2017 12:06      1

                                 80 25 Mar 2017 13:00      1
                                 81 25 Mar 2017 14:00      1
                                 82 25 Mar 2017 15:00      1
                                 83 25 Mar 2017 16:00      1
                                 84 25 Mar 2017 16:34      1



Specify the Begin and End Snapshot Ids
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
Enter value for begin_snap: 83
Begin Snapshot Id specified: 83

Enter value for end_snap: 84
End   Snapshot Id specified: 84
```

```
Specify the Report Name
~~~~~~~~~~~~~~~~~~~~~~~
The default report file name is awrrpt_1_83_84.html.  To use this name,
press <return> to continue, otherwise enter an alternative.

Enter value for report_name: my_report.html
```

În locaţia curentă ar trebui să găsiţi fişierul “my_report.html” pe care-l puteţi deschide cu un browser html.



WORKLOAD REPOSITORY report for

DB Name	DB Id	Instance	Inst num	Startup Time	Release	RAC
TESTDB	2721703651	testdb	1	25-Mar-17 12:03	12.1.0.2.0	NO

Host Name	Platform	CPUs	Cores	Sockets	Memory (GB)
WIN-VORAX	Microsoft Windows x86 64-bit	1	1	1	2.00

	Snap Id	Snap Time	Sessions	Cursors/Session
Begin Snap:	83	25-Mar-17 16:00:09	37	1.4
End Snap:	84	25-Mar-17 16:34:29	30	1.2
Elapsed:		34.34 (mins)		
DB Time:		18.39 (mins)		

Report Summary

Top ADDM Findings by Average Active Sessions

Finding Name	Avg active sessions of the task	Percent active sessions of finding	Task Name	Begin Snap Time	End Snap Time
Top SQL Statements	.54	69.15	ADDM:2721703651_1_84	25-Mar-17 16:00	25-Mar-17 16:34
Top Segments by "User I/O" and "Cluster"	.54	29.72	ADDM:2721703651_1_84	25-Mar-17 16:00	25-Mar-17 16:34
Undersized instance memory	.54	22.11	ADDM:2721703651_1_84	25-Mar-17 16:00	25-Mar-17 16:34
Commits and Rollbacks	.54	3.69	ADDM:2721703651_1_84	25-Mar-17 16:00	25-Mar-17 16:34

Optimizorul Oracle

Optimizarea SQL, contrar a ceea ce s-ar putea crede, nu intră doar în sarcina DBA-ului, ci mai ales a dezvoltatorilor de aplicaţii. În general, cei care implementează logică în aplicaţie, bazată pe SQL, ar trebui să aibă cunoştinţe solide legate de optimizarea acestor fraze SQL.

Pregătirea platformei de test

Schimbăm puţin registrul, de la aplicaţia de tip Facebook cu care ne-am jucat laboratorul trecut, la o aplicaţie de gestiune a vânzărilor. Pentru a crea şi popula schema VANZARI, va trebui să descărcaţi fişierul *perf_setup.sql* şi să-l rulaţi într-o fereastră de SqlPlus.

```
SQL> @C:\Users\talek\perf_setup.sql
Connecting as SYS...
Connected.

Cleanup VANZARI environment...

Create VANZARI schema.
The password is "Vanzari123"
Grant rights to VANZARI.

Connecting as VANZARI...
Connected.

Creating JUDETE table...
Creating CODURI_POSTALE table...
Creating CLIENTI table...
Creating PERSOANE table...
Creating PERSCLIENTI table...
Creating PRODUSE table...
Creating FACTURI table...
Creating LINIIFACT table...
Creating INCASARI table...
Creating INCASFACT table...
Creating WORKLOAD_PRODUCER procedure...

Populating VANZARI tables: 1000 clients and 1000 products...

Done.
```

Statistici optimizor

Pentru ca optimizorul Oracle să poată să execute optim o interogare SQL are nevoie de aşa-numitele statistici de optimizor: câte înregistrări sunt într-o tabelă, câte valori distincte sunt pentru o coloană, cam câte înregistrări sunt în medie în blocurile Oracle asociate tabelei şi multe altele.

Implicit, Oracle colectează aceste statistici automat, o dată pe zi, şi doar pentru acele tabele care au suferit modificări semnificative: multe înregistrări adăugate sau şterse. E şi cazul nostru: abia am creat şi populat schema VANZARI, prin urmare Oracle n-a apucat încă să colecteze statistici

pentru aceste tabele. Iată, dacă tragem o ochiadă în USER_TABLES să vedem câte înregistrări ne raportează că avem pentru tabela LINIIFACT, observăm o inadvertenţă:

```
SQL> select num_rows from user_tables where table_name='LINIIFACT';

  NUM_ROWS
----------

SQL> select count(*) from liniifact;

  COUNT(*)
----------
    167702
```

Putem colecta manual aceste statistici folosind pachetul DBMS_STATS:

```
SQL> exec dbms_stats.gather_table_stats(ownname => 'VANZARI', tabname => 'LINIIFACT',
estimate_percent => 80, cascade => true);

PL/SQL procedure successfully completed.

SQL> select num_rows from user_tables where table_name='LINIIFACT';

  NUM_ROWS
----------
    168854

1 row selected.
```

De ce coloana NUM_ROWS din USER_TABLES nu indică numărul exact de înregistrări din tabela LINIIFACT?

Ce face parametrul "cascade" a comenzii "dbms_stats.gather_table_stats"?

Putem colecta statistici şi pentru întreaga schemă VANZARI folosind comanda de mai jos:

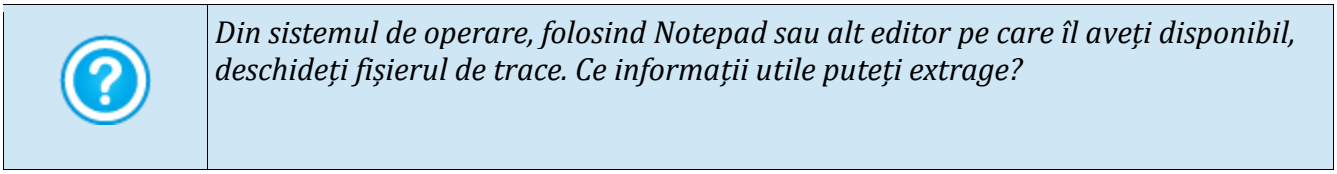
```
SQL> exec dbms_stats.gather_schema_stats(ownname => 'VANZARI', estimate_percent =>
dbms_stats.auto_sample_size, cascade => true);

PL/SQL procedure successfully completed.
```

Activarea facilității trace la nivel de sesiune

De multe ori ne dorim să urmărim toate instrucţiunile SQL care sunt executate de o sesiune pentru a identifica acele SQL-uri care pun probleme de performanţă sau pentru a diagnostica ce se întâmplă efectiv din acea sesiune. Activarea facilităţii de trace se poate face din sesiunea curentă sau o poate face DBA-ul pentru orice altă sesiune. Aici, vom exemplifica varianta activării trace-ului din propria sesiune. Va trebui să descărcaţi de pe portal fişierul "simulate_workload.sql".

```
SQL> ALTER SESSION SET EVENTS '10046 trace name context forever, level 8';

Session altered.

SQL> @C:\Users\talek\simulate_workload.sql

PL/SQL procedure successfully completed.

SQL> ALTER SESSION SET EVENTS '10046 trace name context off';

Session altered.

SQL> select value from v$diag_info where name = 'Default Trace File';

VALUE
--------------------------------------------------------------------------------
C:\APP\TALEK\diag\rdbms\testdb\testdb\trace\testdb_ora_4984.trc
```

Prima comandă activează trace-ul. Ce-a de-a doua simulează execuţia de diverse instrucţiuni SQL, după care facilitatea de trace este dezactivată. În sfârşit, cu ajutorul ultimei interogări putem afla unde au fost scrise toate informaţiile de trace.



Din sistemul de operare, folosind Notepad sau alt editor pe care îl aveţi disponibil, deschideţi fişierul de trace. Ce informaţii utile puteţi extrage?

Pentru a prelucra acest fişier de trace şi a decodifica informaţiile din el, putem folosi un utilitar Oracle denumit "tkprof".

```
C:\Users\talek>tkprof C:\APP\TALEK\diag\rdbms\testdb\testdb\trace\testdb_ora_4984.trc
report.txt waits=y sys=n

TKPROF: Release 12.1.0.2.0 - Development on Sun Mar 26 16:53:57 2017

Copyright (c) 1982, 2014, Oracle and/or its affiliates.  All rights reserved.



C:\Users\talek>notepad report.txt
```

Ar trebui să obţineţi un raport asemănător cu cel de mai jos:

```
TKPROF: Release 12.1.0.2.0 - Development on Sun Mar 26 16:53:57 2017

Copyright (c) 1982, 2014, Oracle and/or its affiliates.  All rights reserved.

Trace file: C:\APP\TALEK\diag\rdbms\testdb\testdb\trace\testdb_ora_4984.trc
Sort options: default
```

```
********************************************************************************
count    = number of times OCI procedure was executed
cpu      = cpu time in seconds executing
elapsed  = elapsed time in seconds executing
disk     = number of physical reads of buffers from disk
query    = number of buffers gotten for consistent read
current  = number of buffers gotten in current mode (usually for update)
rows     = number of rows processed by the fetch or execute call
********************************************************************************

declare
  l_codcl clienti.codcl%type;
  l_dencl clienti.dencl%type;
  l_data facturi.datafact%type;
  l_num integer;
begin
  for l_rec in (select nrfact, count(*) from liniifact group by nrfact order by 2 desc)
loop
    select count(dencl) into l_num from clienti;
    if (mod(l_rec.nrfact, 2) = 0) then
       select max(f.datafact) into l_data from facturi f;
    end if;
    execute immediate 'select f.codcl from facturi f where f.nrfact = ''' || l_rec.nrfact
|| '''' into l_codcl;
    select c.dencl into l_dencl from clienti c where c.codcl = l_codcl;
  end loop;
end;

call     count       cpu    elapsed       disk      query    current        rows
------- ------  -------- ---------- ---------- ---------- ----------  ----------
Parse        1      0.00       0.01          0         18          0           0
Execute      1     10.23      10.02          0          0          0           1
Fetch        0      0.00       0.00          0          0          0           0
------- ------  -------- ---------- ---------- ---------- ----------  ----------
total        2     10.23      10.04          0         18          0           1

Misses in library cache during parse: 1
Optimizer mode: ALL_ROWS
Parsing user id: 113

Elapsed times include waiting on following events:
  Event waited on                             Times   Max. Wait  Total Waited
  ----------------------------------------   Waited  ----------  ------------
  SQL*Net message to client                       1        0.00          0.00
  SQL*Net message from client                     1       26.71         26.71
********************************************************************************

SQL ID: 462nmqs0gfs8r Plan Hash: 4182568281

SELECT NRFACT, COUNT(*)
FROM
 LINIIFACT GROUP BY NRFACT ORDER BY 2 DESC


call     count       cpu    elapsed       disk      query    current        rows
------- ------  -------- ---------- ---------- ---------- ----------  ----------
```

```
Parse        1      0.01       0.00          0          0          0           0
Execute      1      0.00       0.00          0          0          0           0
Fetch      241      0.03       0.05          1        435          0       24000
------- ------  -------- ---------- ---------- ---------- ----------  ----------
total      243      0.04       0.05          1        435          0       24000

Misses in library cache during parse: 1
Optimizer mode: ALL_ROWS
Parsing user id: 113     (recursive depth: 1)
Number of plan statistics captured: 1

Rows (1st) Rows (avg) Rows (max)  Row Source Operation
---------- ---------- ----------  ---------------------------------------------------
     24000      24000      24000  SORT ORDER BY (cr=435 pr=1 pw=0 time=52514 us cost=569
size=120770 card=24154)
     24000      24000      24000   HASH GROUP BY (cr=435 pr=1 pw=0 time=34982 us cost=569
size=120770 card=24154)
    168520     168520     168520    INDEX FAST FULL SCAN PK_LINIIFACT (cr=435 pr=1 pw=0
time=30006 us cost=104 size=842600 card=168520)(object id 92748)


Elapsed times include waiting on following events:
  Event waited on                             Times   Max. Wait  Total Waited
  ----------------------------------------   Waited  ----------  ------------
  db file sequential read                         1        0.00          0.00
********************************************************************************

SQL ID: 9p2ywt1m4ry2k Plan Hash: 669795711

SELECT COUNT(DENCL)
FROM
 CLIENTI


call     count       cpu    elapsed       disk      query    current        rows
------- ------  -------- ---------- ---------- ---------- ----------  ----------
Parse        1      0.00       0.00          0          0          0           0
Execute  24000      0.39       0.44          0          0          0           0
Fetch    24000      1.25       1.27          0     384000          0       24000
------- ------  -------- ---------- ---------- ---------- ----------  ----------
total    48001      1.64       1.72          0     384000          0       24000

...

.. output truncated ..
```

1. *Care e semnificaţia categoriilor "Parse", "Execute" şi "Fetch"?*
2. *Planul de execuţie din fişierul trace este cel real sau cel estimat?*
3. *Identificaţi SQL-urile care au beneficiat de mecanismul "soft parse".*
4. *Ce sunt evenimentele de aşteptare afişate în trace?*