ORDS Tracing Tool

The purpose of the tool is to allow tracing Oracle API for MongoDB sessions in an Oracle Database in case, when there is need to trace statements hardcoded in an application, as in such case it is not possible to use SQL Monitor.

Disclaimer

The use of the scripts is at your sole risk. All information is provided "as -is", without any warranty of its accuracy, completeness, fitness for a particular purpose.

Requirements

- Oracle Database

  All editions and versions of Oracle Database, which support Oracle API for MongoDB are supported.

- Oracle ORDS component, which uses this database with Mongo API enabled.

- Schema owning the tool with the following roles and privileges:

  - DB_DEVELOPER_ROLE
  - CREATE JOB
  - SELECT ON SYS.GV_$SESSION
  - SELECT ON SYS.GV_$SQL
  - SELECT ON SYS.GV_$SQL_PLAN
  - SELECT ON SYS.V_$SQL_PLAN
  - SELECT ON SYS.V_$SQLCOMMAND
  - EXECUTE ON DBMS_SCHEDULER

Installation and deinstallation

- To install the tool there is need to execute install.sql script located in sql directory
- To remove the tool from the database there is need to execute remove.sql located in sql directory

Components

- ORDS_TRC_TAB table
    - stores snapshots of sessions connected through ORDS
- ORDS_TRC_PKG PL/SQL package
    - this package automatizes tracing ORDS sessions (see "usage" section)

ORDS_TRC_PKG components

`session_array_type`	Nested table type used to store ORDS session data
`procedure snapshot`	This procedure creates a snapshot of ORDS sessions. If p_username is provided, then the snapshot is limited to sessions of this user

`procedure snapshot`
`(p_username varchar2 := null)`
	This procedure creates a snapshot of ORDS sessions. If value of p_username parameter is provided, then the snapshot is limited only to sessions of this username

`procedure create_job`
`( p_username varchar2 := null,`
`  p_frequency integer := 5)`
	This procedure creates a DBMS_SCHEDULER job, which will collect snapshots of ORDS sessions. If value of p_username parameter is provided, then snapshots will contain data coming only from such user sessions. Parameter p_frequency is used to provide information about frequency of the job executions, expressed in seconds

`procedure drop_job`
`(p_username varchar2 := null,`
` p_purge_logs Boolean := false);`
	This procedure drops a DBMS_SCHEDULER job created by create_job procedure, described above. If p_username is provided then this procedure drops a job tracing only sessions created by this user. If p_purge_logs is set to true then all corresponding log entries are also deleted.

`procedure drop_all_jobs`
`(p_purge_logs Boolean := false)`
	This procedure drops all tracer jobs. If p_purge_logs is set to true, then this procedure deletes also all log entries

`function get_jobs return job_array_type`
`pipelined`
`parallel_enable`
	This function returns nested array containing data of all tracer jobs

procedure print_jobs	This procedure prints (by executing DBMS_OUTPUT.PUT_LINE procedure) information about all tracer jobs
function get_sessions (p_username varchar2 := null) return session_array_type pipelined parallel enable	This function returns nested table of ORDS sessions connected to the database. If p_username is not null, then this function returns information about only session of this user, otherwise it returns information about all ORDS sessions
procedure print_sessions (p_username varchar2 := null)	This procedure prints (by executing DBMS_OUTPUT.PUT_LINE procedure) information about ORDS sessions connected to the database. If p_username is not null, then this procedure prints information about only session of this user, otherwise it prints information about all ORDS sessions
function get_report (p_username varchar2 := null) return clob	This function generates a tracing report. If value of p_username parameter is provided, then the report will be limited only to sessions of this user
procedure print_report (p_username varchar2 := null)	This procedure will print out (by executing DBMS_OUTPUT.PUT_LINE procedure) report. It requires setting serveroutput to on. If value of p_username is provided, then the report will be limited only to sessions of this user
procedure purge_logs (p_username varchar2 := null)	This procedure deletes tracing data from ORDS_TRC_TAB table. If p_username is provided then it will delete only data coming from database sessions of this user.

Usage

The tool allows for gathering snapshots of sessions connected through ORDS. To start to trace these sessions there is need to run the following procedure:

ORDS_TRC_PKG.CREATE_ JOB(FREQ INTEGER)

Example:

```
SQL> exec ords_trc_pkg.create_job('oradev')

PL/SQL procedure successfully completed.
```

Above example creates a snapshot gathering job, which will be executed every 5 seconds. This job will gather data limited only to ORADEV database (and Oracle API for MongoDB) user.

Job created in this way is already enabled, so in the next step we can run application/workload/scripts, which we want to trace.
After some time we can generate report.

It is possible to generate a report and store it in a CLOB value:

```
SQL> declare
  2     v_report clob;
  3  begin
  4    v_report := ords_trc_pkg.get_report('oradev');
  5    …
```

It is also possible to print out the report in a SQL*Plus session:

```
SQL> set serveroutput on
SQL> exec ords_trc_pkg.print_report('oradev')
```

Sample report is presented in the appendix A of this document

To stop gathering data there is need to execute DROP_JOB procedure from ORDS_TRC_PKG package

```
SQL> select job_name
  2* from user_scheduler_jobs;

JOB_NAME
______________________
ORDS_TRC_JOB_ORADEV

SQL> exec ords_trc_pkg.drop_job('oradev')

PL/SQL procedure successfully completed.

SQL> select job_name
  2* from user_scheduler_jobs;

no rows selected
```

Finally, we can purge log by executing purge_logs procedure:

```
SQL> begin
  2 -- this deletes only data of ORADEV user sessions
  3    ords_trc_pkg.purge_logs('oradev');
  4 -- this deletes all data
  5    ords_trc_pkg.purge_logs;
  6  end;
  7* /
PL/SQL procedure successfully completed.
```

Appendix A: Sample report

```
SQL> exec ords_trc_pkg.print_report

 USERNAME : ORADEV
======================================================

inst_id:sid:serial# : 8:23076:7179
-----------------------------------------------------

SQL_ID  6y51fmj1wtv2c, child number 0

begin   DBMS_SODA_ADMIN.DESCRIBE_COLLECTION(
P_URI_NAME    => :1 ,                    P_DESCRIPTOR => :2 ,
        P_23C_DRIVER => true); end;
```

```
NOTE: cannot fetch plan for SQL_ID: 6y51fmj1wtv2c, CHILD_NUMBER: 0
      Please verify value of SQL_ID and CHILD_NUMBER;
      It could also be that the plan is no longer in cursor cache (check v$sql_plan)



inst_id:sid:serial# : 8:37135:37950
-----------------------------------------------------

command type : 0 ()
plan         :
SQL_ID  b3748jjrxmm8g, child number 0
-------------------------------------
select "DATA",rawtohex("RESID"),"ETAG" from "ORADEV"."test01"

Plan hash value: 3481372005

------------------------------------------------------------------------------------
| Id  | Operation                 | Name   | Rows  | Bytes | Cost (%CPU)| Time     |
------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT          |        |       |       |     3 (100)|          |
|   1 |  TABLE ACCESS STORAGE FULL| test01 |     2 |   246 |     3   (0)| 00:00:01 |
------------------------------------------------------------------------------------


SQL_ID  6y51fmj1wtv2c, child number 0

begin   DBMS_SODA_ADMIN.DESCRIBE_COLLECTION(
P_URI_NAME   => :1 ,                    P_DESCRIPTOR => :2 ,
        P_23C_DRIVER => true); end;

NOTE: cannot fetch plan for SQL_ID: 6y51fmj1wtv2c, CHILD_NUMBER: 0
      Please verify value of SQL_ID and CHILD_NUMBER;
      It could also be that the plan is no longer in cursor cache (check v$sql_plan)


command type : 3 (SELECT)
plan         :
SQL_ID  6k24nmf72sfn6, child number 0
-------------------------------------
select "DATA",rawtohex("RESID"),"ETAG" from "ORADEV"."DEPT_JSON_VIEW"

Plan hash value: 2907360136

--------------------------------------------------------------------------------------------
| Id  | Operation                 | Name           | Rows  | Bytes | Cost (%CPU)| Time     |
--------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT          |                |       |       |     3 (100)|          |
|   1 |  TABLE ACCESS STORAGE FULL| DEPT_XML_TABLE |    27 |  1107 |     3   (0)| 00:00:01 |
--------------------------------------------------------------------------------------------
```