



Tools of the Trade: All the Free stuff from Oracle Corp

Dan Hotka

Author/Speaker/Oracle Expert



www.DanHotka.com, LLC

(c) www.danhotka.com LLC.

Any reproduction or copying of this manual without the express written consent of www.danhotka.com LLC is expressly prohibited.

Limitation on Warranty. THERE ARE NO WARRANTIES, EXPRESS OR IMPLIED, WITH RESPECT THERETO, INCLUDING, WITHOUT LIMITATION, ANY WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. PURCHASER SHALL BE SOLELY RESPONSIBLE FOR THE SELECTION, USE, EFFICIENCY AND SUITABILITY OF USE OF INFORMATION CONTAINED HEREIN TO ANY PARTICULAR APPLICATION OR PROBLEM. WWW.DANHOTKA.COM LLC SHALL HAVE NO LIABILITY THEREFOR.

Limitation of Liability. IN NO EVENT SHALL WWW.DANHOTKA.COM LLC BE LIABLE TO YOU FOR ANY DAMAGES, INCLUDING, WITHOUT LIMITATION, ANY DAMAGES RELATING TO LOSS OF DATA, AND ANY INDIRECT, SPECIAL OR CONSEQUENTIAL DAMAGES OR LOST PROFITS, ARISING OUT OF OR IN ANY WAY RELATED TO YOUR USE OF THE INFORMATION CONTAINED IN THIS MANUAL. IN THE EVENT THAT THE FORGOING IS HELD UNINFORCABLE THE PARTIES AGREE THAT WWW.DANHOTKA.COM LLC'S LIABILITY TO YOU HEREUNDER, IF ANY, SHALL IN NO EVENT EXCEED THE FEE PAID BY THE INJURED PARTY FOR THE MANUAL TO WWW.DANHOTKA.COM. LLC.

Dan Hotka
Author/Instructor/Oracle Expert
CEO
DHotka@Earthlink.net
515 279 3361

Dan is a Training Consultant

❖ Dan Hotka

- Oracle Authored Expert
 - 28 Years in IT – 24 years working with Oracle
 - 10 books – hundreds of articles

❖ www.DanHotka.com

- Flat Fee Training...1 Course Fee Price for up to 15 Attendees!
 - Price includes my portable computer lab!
- Hands-on Workshops
 - Oracle Discoverer/Oracle Analytics
 - Oracle Tuning Tips and Techniques
 - TOAD and SQL Developer Courses
 - Intro to Oracle, Intro to PL/SQL, Advanced PL/SQL
 - Intro to Unix, Unix Shell Scripting, Oracle/Unix Shell Scripting

❖ Register for my quarterly Newsletter

New Book

ORACLE



ORACLE PRESS® — EXCLUSIVELY FROM MCGRAW-HILL/OSBORNE

ORACLE SQL Developer Handbook

Take Full Advantage of All the Database Development Features



ORIGINAL • AUTHENTIC

Oracle Press

ONLY FROM OSBORNE

DAN HOTKA
Oracle Expert and Trainer

ISBN: 0-07-148474-4

www.Amazon.com

www.it-ebooks.com



Agenda

- ❖ **Explain Plan Free Tools**
 - Xplan.display
 - JS Tuner
- ❖ **Oracle Trace Free Tools**
 - TKProf
 - Trace Analyzer
 - JS Analyzer
 - PL/SQL Profiler
- ❖ **Oracle Development Free Tools**
 - SQL Developer (Project Raptor)



Additional Information

❖ Ask me for:

- White Paper
- Free Tools
- Scripts
 - DHOTKA@EARTHLINK.net

Explain Plan Free Tools

❖ **DBMS_XPLAN.DISPLAY**

- Available with Oracle 8.1.7+
- Comes with Database
- Used via SQL*Plus

❖ **JS Tuner**

- Available for Oracle8+
- Download from www.DanHotka.com
- Executable Jar File

Explain Plan Free Tools

❖ **PLAN_TABLE**

- Use `<oracle home>/rdbms/admin/utlxplan.sql` for current plan_table
- Use the 'explain plan for' syntax to populate this table
- Use tools to populate this table (TOAD, SQL Developer)
- Use `SHOW_PLAN.sql` to display contents
 - Available on my website

Explain Plan Free Tools

```
Oracle SQL*Plus
File Edit Search Options Help
SQL>
SQL> EXPLAIN PLAN FOR
  2 select ename
  3 from emp
  4 where deptno in (select deptno from dept);

Explained.

SQL> start show_plan

-----
Cost      ID P_ID Access Path          Object
-----
2         0      SELECT STATEMENT
2         1      0    NESTED LOOPS
2         2      1    TABLE ACCESS FULL      EMP
         3      1    INDEX          UNIQUE SCAN  DEPT_PRIMARY_KE
                                         Y

4 rows deleted.

SQL> |
```

Explain Plan Free Tools

```
DHOTKAXP:ora9r2:USER0>1  
1* select * from table(dbms_xplan.display)  
DHOTKAXP:ora9r2:USER0>/
```

PLAN_TABLE_OUTPUT

| Id | Operation | Name | Rows | Bytes | Cost |
|-----|-----------------------------|--------------|-------|-------|------|
| 0 | SELECT STATEMENT | | 1 | 18 | 8 |
| 1 | SORT AGGREGATE | | 1 | 18 | |
| * 2 | HASH JOIN | | 10000 | 175K | 8 |
| * 3 | HASH JOIN | | 1000 | 12000 | 5 |
| 4 | TABLE ACCESS BY INDEX ROWID | B | 100 | 600 | 2 |
| * 5 | INDEX RANGE SCAN | B_STATUS_IDX | 100 | | 1 |
| * 6 | TABLE ACCESS FULL | A | 1000 | 6000 | 2 |

PLAN_TABLE_OUTPUT

| | | | | | |
|-----|-------------------|---|------|------|---|
| * 7 | TABLE ACCESS FULL | C | 1000 | 6000 | 2 |
|-----|-------------------|---|------|------|---|

Predicate Information (identified by operation id):

- 2 - access("B"."ID"="C"."B_ID")
- 3 - access("A"."B_ID"="B"."ID")
- 5 - access("B"."STATUS"='OPEN')
- 6 - filter("A"."STATUS"='OPEN')
- 7 - filter("C"."STATUS"='OPEN')



JS Tuner

- ❖ **Incorporates SQL Scripts with enhanced Trace**
- ❖ **Index Info**
 - Includes index statistics
- ❖ **Enhanced Explain Plan**
 - Works with V8+
 - Works with rule-based optimizer!
- ❖ **Available from www.DanHotka.com**



user0/*****@oraxp9i

SQL Query Results Explain Plan Statistics Enhanced Explain Plan

| ID | P_ID | Plan | Access Path | Object Name | Where Clause |
|----|------|------------------|----------------|--------------|-------------------|
| 0 | | SELECT STATEMENT | | | |
| 1 | 0 | SORT | AGGREGATE | | |
| 2 | 1 | TABLE ACCESS | BY INDEX ROWID | C | C.STATUS='OPEN' |
| 3 | 2 | NESTED LOOPS | | | |
| 4 | 3 | NESTED LOOPS | | | |
| 5 | 4 | TABLE ACCESS | BY INDEX ROWID | B | |
| 6 | 5 | INDEX | RANGE SCAN | B_STATUS_IDX | B.STATUS='OPEN' |
| 7 | 4 | TABLE ACCESS | BY INDEX ROWID | A | A.B_ID=B.ID |
| 8 | 7 | INDEX | RANGE SCAN | A_STATUS_IDX | A.STATUS=B.STATUS |
| 9 | 3 | INDEX | RANGE SCAN | C_B_ID_IDX | B.ID=C.B_ID |

Login Run Explain Desc Index Clear Open Save Help Hints History Exit

Oracle Trace Free Tools

❖ TKPROF

- Available with Oracle since V7
- Comes with Database
- Executable program

❖ Trace Analyzer

- Available for Oracle 9.2+
- Download from Metalink
- PL/SQL runs thru SQL*Plus

❖ JS Analyzer

- Runs TKPROF and Trace Analyzer
- Download from www.DanHotka.com
- Executable Jar File



SQL trace

- ❖ **SQL TRACE tool generates a raw trace file (*.trc) that contains info about SQL statements**
 - File created with adjustable default format in USER_DUMP_DEST
 - Creates a simple 10046 'Enable SQL Execution Trace'
- ❖ **Commands:**
 - TIMED_STATISTICS = TRUE
 - System level:
 - Alter system set sql_trace = true;
 - Session level:
 - Alter session set sql_trace = TRUE;
 - Alter session set timed_statistics =TRUE



SQL Trace

❖ Commands continued:

– A Different Session:

- `DBMS_SYSTEM.set_sql_trace_in_session(<sid>, <serial#>, true);`
 - `V$session`
 - Does not start the extended 10046 trace options...
- Or
- `DBMS_SUPPORT.start_trace_in_session(<sid>, <serial#>, waits=>true, binds=>false)`
- `DBMS_SUPPORT.stop_trace_in_session(<sid>, <serial#>)`
 - DOES start the 10046 extended trace options



SQL Trace

❖ Commands (continued)

- Session level:
 - Alter session set tracefile_identifier = 'xxx';

❖ Useful Parameters

- Max_dump_file_size
- Timed_statistics (v8i or before)
- User_dump_dest
- Trace_file_identifier ...up to 60 characters (or OS limit)



SQL trace

❖ SQL TRACE tool generates a raw trace file (*.trc) that contains info about SQL statements

- File created with adjustable default format in USER_DUMP_DEST
- Creates a simple 10046 'Enable SQL Execution Trace'
 - Trace Level 0 no statistics
 - Trace Level 1 includes statistics
 - Trace Level 4 includes any bind vars + statistics
 - Trace Level 8 includes wait events + statistics
 - Trace Level 12 includes all



SQL Trace

❖ Commands continued:

- A Different Session:
 - `DBMS_SYSTEM.set_sql_trace_in_session(<sid>, <serial#>, true);` ...use `FALSE` to then turn off the trace...
 - `V$session`
 - Does not start the extended 10046 trace options...
- Or
 - `DBMS_SYSTEM.set_ev(<sid>, <serial#>, 10046, 12, '')`
 - `DBMS_SYSTEM.set_ev(<sid>, <serial#>, 10046, 0, '')`
 - DOES start the 10046 extended trace options



SQL Trace

❖ Commands continued using OraDebug:

– Unix

- SQL> select spid from v\$process where addr in (select paddr from v\$session where ...);
- SQL> oradebug setospid <SPID-value>
- SQL> oradebug unlimit
- SQL> oradebug event 10046 trace name context forever, level 12

– Windows

- SQL> select pid from v\$process where addr in (select paddr from v\$session where ...);
- SQL> oradebug setorapid <PID-value>
- SQL> oradebug unlimit
- SQL> oradebug event 10046 trace name context forever, level 12

Remember to run again with 'level 0' to stop trace



SQL Trace

- ❖ **Trace file name: OracleSID_ora_<spid>.trc**
 - Spid is a column in V\$PROCESS

SPID_by_user.sql

```
select p.spid, s.program
from v$process p,
v$session s
where p.addr = s.PADDR
and s.username = '&USER';
```

SPID_by_sid_serial.sql

```
select p.spid, s.program
from v$process p,
v$session s
where p.addr = s.PADDR
and s.sid = &sid
and s.serial# = &serial
```



SQL Trace

❖ Scripts to turn on/off Trace for a different session

- Start_Trace_in_Session.sql
 - Shows current sessions
 - Prompts for input
 - Turns on an extended trace (binds, waits)
 - SQL>start start_trace_in_session
- Stop_Trace_in_session.sql
 - Ends the trace session
 - SQL> start stop_trace_in_session

SQL Trace

```
Oracle SQL*Plus
File Edit Search Options Help

Start SQL Trace in your session

Trace File Name
-----
ora9r2_ora_3596_user0.trc

SQL> select 'run your code or SQL here' from dual;

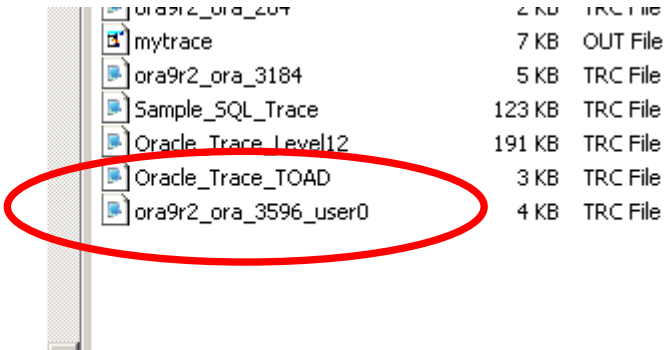
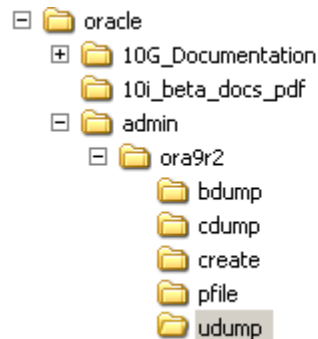
'RUNYOURCODEORSQLHERE'
-----
run your code or SQL here

1 row selected.

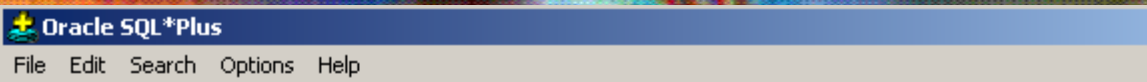
SQL> start stop_trace
SQL> SET TERMOUT OFF

Trace is turned off

SQL> |
```



SQL Trace



| USERNAME | SID | SERIAL# | PROGRAM |
|----------|-----|---------|--------------|
| USER0 | 9 | 20 | sqlplusw.exe |
| USER1 | 10 | 44 | sqlplusw.exe |
| USER0 | 12 | 28 | toad.exe |
| USER0 | 13 | 28 | toad.exe |
| USER0 | 14 | 8 | toad.exe |
| USER0 | 15 | 27 | toad.exe |
| USER0 | 16 | 12 | toad.exe |

Enter User ID from list above : user1
Enter SID from list above : 10
Enter SERIAL from list above : 44

'Write the SID and SERIAL down...will need them to stop the trace'

Started SQL Trace in session

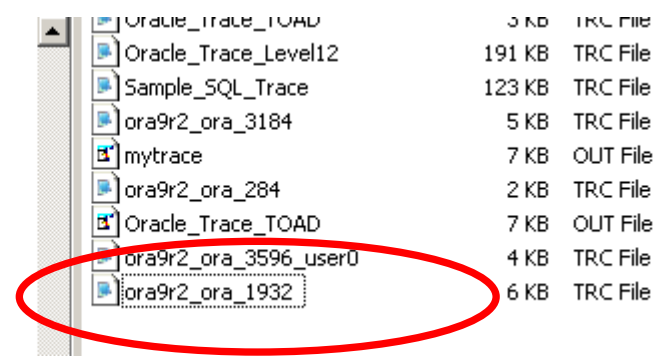
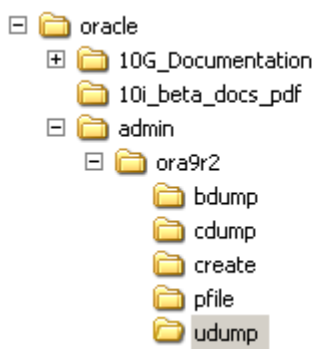
Trace File Name

ora9r2_ora_1932.trc

```
SQL>  
SQL>  
SQL>  
SQL>  
SQL>  
SQL> start stop_trace_in_session  
SQL> SET TERMOUT OFF  
Enter SID from Start Trace : 10  
Enter SERIAL from Start Trace : 44
```

Trace is turned off

SQL>





+ SQL Trace

❖ Lets see the code:

- [Start_Trace.sql](#)
 - [Stop_Trace.sql](#)
 - [Start_Trace_in_Session.sql](#)
 - [Stop_Trace_in_Session.sql](#)
- These files are on your diskette



Event Dumps

- ❖ **Alter session set max_dump_file_size=unlimited;**
- ❖ **Alter session set tracefile_identifier='HOTKA';**
- ❖ **Alter session set events '10046 trace name context forever, level 12';**
- ❖ **Select 'Hello World: Today is ' || sysdate from dual;**
- ❖ **Alter session set events '10046 trace name context off'**

```
C:\WINDOWS>tkprof
```

```
Usage: tkprof tracefile outputfile [explain= ] [table= ]
```

```
       [print= ] [insert= ] [sys= ] [sort= ]
```

```
table=schema.tablename    Use 'schema.tablename' with 'explain=' option.
```

```
explain=user/password     Connect to ORACLE and issue EXPLAIN PLAN.
```

```
print=integer             List only the first 'integer' SQL statements.
```

```
aggregate=yes|no
```

```
insert=filename          List SQL statements and data inside INSERT statements.
```

```
sys=no                   TKPROF does not list SQL statements run as user SYS.
```

```
record=filename         Record non-recursive statements found in the trace file.
```

```
waits=yes|no            Record summary for any wait events found in the trace file.
```

```
sort=option             Set of zero or more of the following sort options:
```

```
  prscnt  number of times parse was called
```

```
  prscpu  cpu time parsing
```

```
  prsela  elapsed time parsing
```

```
  prsdsk  number of disk reads during parse
```

```
  prsqry  number of buffers for consistent read during parse
```

```
  prscu   number of buffers for current read during parse
```

```
  prsmis  number of misses in library cache during parse
```

```
  execnt  number of execute was called
```

```
  execpu  cpu time spent executing
```

```
  exeela  elapsed time executing
```

```
  exedsk  number of disk reads during execute
```

```
  exeqry  number of buffers for consistent read during execute
```

```
  execu   number of buffers for current read during execute
```

```
  exerow  number of rows processed during execute
```

```
  exemis  number of library cache misses during execute
```

```
  fchcnt  number of times fetch was called
```

```
  fchcpu  cpu time spent fetching
```

```
  fchela  elapsed time fetching
```

```
  fchdsk  number of disk reads during fetch
```

```
  fchqry  number of buffers for consistent read during fetch
```

```
  fchcu   number of buffers for current read during fetch
```

```
  fchrow  number of rows fetched
```

```
  userid  userid of user that parsed the cursor
```



SQL Trace

❖ TKPROF

- Explain plans are included – don't need explain= option
- Combine sort & print
 - Sort=(exeela, fchela) print=10
 - Sort=(execpu, fchcpu) print=10
- Sys=no – does not display dictionary SQL



SQL Trace

- ❖ **Use TKPROF utility to produce a formatted output**
- ❖ **Formatted file contains:**
 - Parse, execute, and fetch counts
 - Parse: SQL Syntax & Execution Plan
 - Execute:
 - Modifies data on INSERT, UPDATE, DELETE
 - Identifies rows for SELECT
 - Fetch: Retrieves rows for SELECT
 - CPU time to process
 - Elapsed time to execute
 - Read info:
 - Disk: Physical block reads
 - Query: Consistent Logical Block Reads
 - Current: Logical block Reads

Copyright (c) 1982, 2002, Oracle Corporation. All rights reserved.

Trace file: ora9r2_ora_Level12.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

alter session set events '10046 trace name context forever, level 12'

| call | count | cpu | elapsed | disk | query | current | rows |
|---------|-------|------|---------|------|-------|---------|------|
| Parse | 0 | 0.00 | 0.00 | 0 | 0 | 0 | 0 |
| Execute | 1 | 0.00 | 0.03 | 0 | 0 | 0 | 0 |
| Fetch | 0 | 0.00 | 0.00 | 0 | 0 | 0 | 0 |
| total | 1 | 0.00 | 0.03 | 0 | 0 | 0 | 0 |

Misses in library cache during parse: 0

Misses in library cache during execute: 1

Optimizer goal: CHOOSE

Parsing user id: 88

Elapsed times include waiting on following events:

| Event waited on | Times Waited | Max. Wait | Total Waited |
|-----------------------------|-----------------|-----------|--------------|
| SQL*Net message to client | 1 | 0.00 | 0.00 |
| SQL*Net message from client | 1 | 15.93 | 15.93 |



Trace Analyzer

- ❖ **Oracle Corp's new trace file analyzer**
- ❖ **Available via MetaLink**
 - Search on Trace Analyzer
 - Download trca.zip
 - Read the Instructions.txt
 - Easy to Install
 - Easy to use



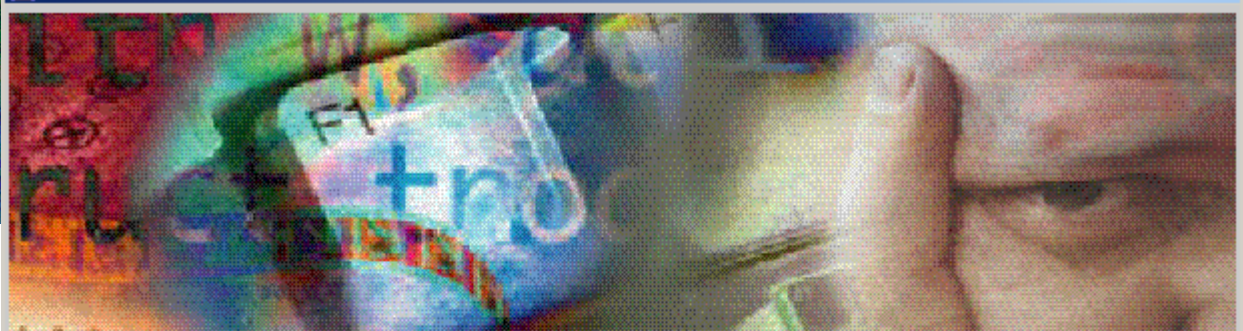
Trace Analyzer

- ❖ **Sqlplus userid/pwd@host @trcanlzl <trace file>**
- ❖ **SQL> start trcanlzl <trace file>**
 - <trace file> is the name of the trace file
 - Trace Analyzer will find it in the UDUMP folder
 - Creates a html file in the UDUMP and the folder where SQL*Plus was started from
- ❖ [Trace Analyzer Output](#)



JS Analyzer

- ❖ **Runs command-line TKProf**
 - Starts output in Notepad (or editor of choice)
- ❖ **Runs Trace Analyzer**
- ❖ **Simply point at a trace file, click and go.**
- ❖ **Available from www.DanHotka.com**



Logged-In User

user0/*****@oraxp9i

Open File

C:\oracle\admin\ora9r2\udump\Oracle_Trace_TOAD.trc

TKProf Trace Analyzer

| Parse | Execute | Fetch | Print |
|---------------------------------------|---|--|-------|
| <input type="checkbox"/> Count | <input type="checkbox"/> Count | <input type="checkbox"/> Count | 10 |
| <input type="checkbox"/> CPU Time | <input type="checkbox"/> CPU Time | <input type="checkbox"/> CPU Time | |
| <input type="checkbox"/> Disk Reads | <input type="checkbox"/> Time | <input type="checkbox"/> Time | |
| <input type="checkbox"/> Time | <input type="checkbox"/> Disk Reads | <input type="checkbox"/> Disk Reads | |
| <input type="checkbox"/> Consist Read | <input type="checkbox"/> Consist Read | <input type="checkbox"/> Consist Reads | |
| <input type="checkbox"/> Current Read | <input type="checkbox"/> Current Read | <input type="checkbox"/> Current Reads | |
| <input type="checkbox"/> Lib misses | <input type="checkbox"/> Rows Processed | <input type="checkbox"/> Rows Fetched | |
| | <input type="checkbox"/> Lib misses | | |

OK Clear Help

Login Open Exit



PL/SQL Profiler

- ❖ Shows how long each line of PL/SQL Code took to execute!
- ❖ Easy to Setup
- ❖ Easy to use
- ❖ Use:
 - Supplied Scripts



PL/SQL Profiler

❖ Installation

- Run once as SYSDBA, logged in as SYS
- `<ORACLE_HOME>/rdbms/admin/profload.sql`
 - Connect `sys/pwd@<host>` as `sysdba`
- Run once per User Account:
- `<ORACLE_HOME>/rdbms/admin/proftab.sql`
 - Connect as the schema account owner
- These scripts are on your diskette



PL/SQL Profile

❖ To Use:

- Tool, SQL Developer, SQL*Plus
 - `Dbms_profiler.start_profiler('some comment');`
 - `Exec <your PL/SQL package/function/procedure>`
 - `Dbms_profiler.stop_profiler`
- Watch for JS Profiler!!!

PL/SQL Profile

❖ Examine Output:

```
SQL>
SQL> create or replace function seconds (nsec1 IN NUMBER)
  2  RETURN NUMBER AS
  3  nsec2 number(12,3);
  4  begin
  5  nsec2 := nsec1 / 1000000000000;
  6  return (nsec2);
  7  END;
  8  /
```

Function created.

```
SQL> select runid, run_owner, run_date, run_comment, seconds(run_total_time) run_seconds
  2  from plsqli_profiler_runs;
```

| RUNID | RUN_OWNER | RUN_DATE | RUN_COMMENT | RUN_SECONDS |
|-------|-----------|-----------|-----------------------|-------------|
| 1 | USER0 | 21-SEP-07 | User0 Looping_Example | 1.856 |

PL/SQL Profile

```
SQL> 1
      2 select runid, unit_number, unit_type, unit_owner, unit_name, unit_timestamp
      3 from plsqli_profiler_units
      4 where unit_owner <> '<anonymous>'
      5* and runid = 1
SQL> /
```

| RUNID | UNIT_NUMBER | UNIT_TYPE | UNIT_OWNER | UNIT_NAME | UNIT_TIME |
|-------|-------------|-----------|------------|-----------------|-----------|
| 1 | 3 | PROCEDURE | USER0 | LOOPING_EXAMPLE | 20-SEP-07 |

```
SQL> |
```

PL/SQL Profile

```
select pu.unit_name, pd.line#, pd.total occur passes,
round(pd.total time / 1000000000,5) total time, us.text text
from plsql_profiler data pd, plsql_profiler_units pu, user_source us
where pd.runid = &rpt runid
and pd.unit number = &rpt unitid
and pd.runid = pu.runid
and pd.unit number = pu.unit number
and us.name = pu.unit_name
and us.line = pd.line#
and us.type in ('PACKAGE BODY','PROCEDURE','FUNCTION');
```

| UNIT_NAME | LINE# | PASSES | TOTAL_TIME | TEXT |
|-----------------|-------|--------|------------|---|
| LOOPING_EXAMPLE | 3 | 1 | .00159 | loop_counter NUMBER := 0; |
| LOOPING_EXAMPLE | 5 | 15 | 4.07546 | FOR rec IN (SELECT * |
| LOOPING_EXAMPLE | 8 | 14 | .01396 | loop_counter := loop_counter + 1; |
| LOOPING_EXAMPLE | 9 | 29 | .06673 | DBMS_OUTPUT.put_line (|
| LOOPING_EXAMPLE | 14 | 2 | .008 | DBMS_OUTPUT.put_line ('Procedure Looping E xample is done'); |



PL/SQL Profile

❖ Use supplied PROFILER_RPT.sql

- Start c:\temp\profiler_rpt.sql
 - Ask me for this script!!!
- Prompts for required information
- Displays and prints above information
- [Review output](#)

❖ Cleanup

- Delete from:
 - delete from plsql_profiler_data;
 - delete from plsql_profiler_units;
 - delete from plsql_profiler_runs;

Oracle Development Free Tools

❖ SQL Developer

- Available since December
- Formerly known as Project Raptor
- Download from:
www.oracle.com/technology/software/products/sql/index.html
- Java Application



SQL Developer

- ❖ **New IDE tool for Oracle**
- ❖ **Very easy to add/modify any object**
- ❖ **Very easy to see related objects, data, relationships, and more**
- ❖ **Very easy to develop and debug PL/SQL**
- ❖ **Very easy to work with Procedures/Packages/Functions**

New Book

ORACLE



ORACLE PRESS® — EXCLUSIVELY FROM MCGRAW-HILL/OSBORNE

ORACLE SQL Developer Handbook

Take Full Advantage of All the Database Development Features



ORIGINAL • AUTHENTIC

Oracle Press

ONLY FROM OSBORNE

DAN HOTKA
Oracle Expert and Trainer

ISBN: 0-07-148474-4

Order today!

www.com

Oracle SQL Developer

File Edit View Navigate Run Debug Source Tools Help

Connections Reports

user0_oraxp9i EMP

Columns Data Indexes Constraints Grants Statistics Column Statistics

Actions...

| Column Name | Data Type | Nullable | Da |
|-------------|--------------------|----------|----|
| EMPNO | NUMBER(4,0) | No | |
| ENAME | VARCHAR2(10 Bytes) | Yes | |
| JOB | VARCHAR2(9 Bytes) | Yes | |
| MGR | NUMBER(4,0) | Yes | |
| HIREDATE | DATE | Yes | |
| SAL | NUMBER(7,2) | Yes | |
| COMM | NUMBER(7,2) | Yes | |
| DEPTNO | NUMBER(2,0) | Yes | |

Connections

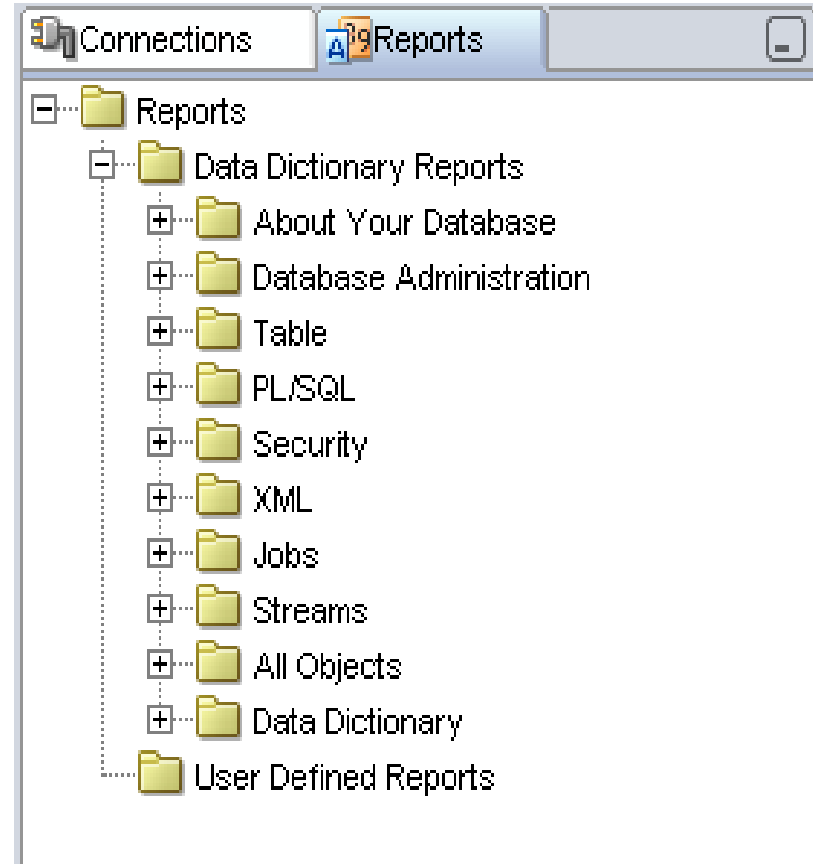
- sales_tracking_oraxp9i
- system_oraxp9i
- user0_oraxp9i
 - DEPT_CHAR
 - DUMMY
 - EMP
 - MANAGERS
 - MASTER
 - MULTI_KEY_TBL1
 - MULTI KEY TBL2

Context Menu:

- Create TABLE
- Apply Filter...
- Refresh
- Clear Filter
- Help

SQL Developer

- ❖ There are many more features than can be discussed in this limited time...
- ❖ Dan provides a 1-day training course on this topic that will allow you to maximize your productivity with this new tool!



Additional Reading

❖ Oracle Professional

- March 2006 SQL Developer
- May 2006 Trace Analyzer

❖ www.DBAZine.com

- Trace Analyzer
- Tuning Topics

Oracle Professional
Solutions for High-End Oracle® DBAs and Developers

Project Raptor/SQL Developer Review

Dan Hotka

The Oracle development world hasn't had a quality and free development test area. Over software purchased 7000, Mail, Usat 5 and Project Raptor came along in the article, Dan Hotka introduces us to this new Oracle graphical interface tool.

Project Raptor is Oracle's newly released free graphical database query and object development/debugging tool. This tool is in its infancy and already has quality features found in many available development tools. Project Raptor is Oracle's internal name for this project. When it's released for production, it will be named SQL Developer. In the meantime, you might hear both names used for this tool.

This article will start with Project Raptor's system requirements and illustrate the ease of installation. It then demonstrates the tool's main features, including connecting to databases, database object browsing, and running SQL and scripts.

Raptor capabilities
Project Raptor gives the user a nice graphical interface in the Oracle BEEM environment (see [Figure 1](#)). Navigation is easy via the tree browser on the left. Raptor can run SQL statements or SQL scripts. If the scripts contain variables, it will prompt the user for each variable. Raptor gives a single as well as plan, Tables, indexes, and database objects can easily be created, data viewed, relationships illustrated, dependencies listed, and Raptor can even create a script for the object selected.

The SQL area has a history. There are code "snippets" to aid in the fine details of syntax. These snippets include various data format options, number formats, character formats, hints, and even PL/SQL code examples. The code can be dragged and dropped from the snippets window.

All kinds of PL/SQL can be created and maintained. Raptor even includes

March 2006
Issue 11 Number 1

1. Project Raptor/SQL Developer Review Dan Hotka
2. Credential Complexity in PL/SQL, Ben J. Evans-Fourie
3. March 2006 Developer's

DOWNLOAD
Write a comment for this item!
www.oracle.com



More Free Stuff

❖ More Free Stuff not covered here:

- UTLBStat/UTLEstat
 - <oracle home>/rdbms/admin/utlbstat.sql & utlestat.sql
- Stats Pack
 - Read the <oracle home>/rdbms/admin/spdoc.txt
 - [SP Report](#)
- Oracle10g Express
- Rman
- Flashback Query
- Log Miner
- Data Guard/Standby Database
- Let me know if you know of others!!!