Autotrace in SQLPLUS

Here is what I like to do to get autotrace working:

- cd $oracle_home/rdbms/admin
- log into sqlplus as system
- run SQL> @utlxplan
- run SQL> create public synonym plan_table for plan_table
- run SQL> grant all on plan_table to public
- exit sqlplus and cd $oracle_home/sqlplus/admin
- log into sqlplus as SYS
- run SQL> @plustrce
- run SQL> grant plustrace to public

You can replace **public** with some **user** if you want. by making it public, you let anyone trace using sqlplus (not a bad thing in my opinion).

About Autotrace

You can automatically get a report on the execution path used by the SQL optimizer and the statement execution statistics. The report is generated after successful SQL DML (that is, SELECT, DELETE, UPDATE and INSERT) statements. It is useful for monitoring and tuning the performance of these statements.

Controlling the Report

You can control the report by setting the AUTOTRACE system variable.

```
SET AUTOTRACE OFF           - No AUTOTRACE report is generated. This is the
                              default.
SET AUTOTRACE ON EXPLAIN    - The AUTOTRACE report shows only the optimizer
                              execution path.
SET AUTOTRACE ON STATISTICS - The AUTOTRACE report shows only the SQL
                              statement execution statistics.
SET AUTOTRACE ON            - The AUTOTRACE report includes both the
                              optimizer execution path and the SQL
                              statement execution statistics.
SET AUTOTRACE TRACEONLY     - Like SET AUTOTRACE ON, but suppresses the
                              printing of the user's query output, if any.
```

To use this feature, you must have the PLUSTRACE role granted to you and a PLAN_TABLE table created in your schema. For more information on the PLUSTRACE role and PLAN_TABLE table, see the AUTOTRACE variable of the SET command in Chapter 6 of the SQL*Plus Guide.

Execution Plan

The Execution Plan shows the SQL optimizer's query execution path.

Each line of the Execution Plan has a sequential line number. SQL*Plus also displays the line number of the parent operation.

The Execution Plan consists of four columns displayed in the following order:

Column Name	Description
ID_PLUS_EXP	Shows the line number of each execution step.
PARENT_ID_PLUS_EXP	Shows the relationship between each step and its parent. This column is useful for large reports.
PLAN_PLUS_EXP	Shows each step of the report.
OBJECT_NODE_PLUS_EXP	Shows the database links or parallel query servers used.

The format of the columns may be altered with the COLUMN command. For example, to stop the PARENT_ID_PLUS_EXP column being displayed, enter:

```
SQL> COLUMN PARENT_ID_PLUS_EXP NOPRINT
```

The default formats can be found in the site profile (for example, glogin.sql).

The Execution Plan output is generated using the EXPLAIN PLAN command. For information about interpreting the output of EXPLAIN PLAN, see the Oracle7 Server Tuning guide.

The following is an example of tracing statements for performance statistics and query execution path.

If the SQL buffer contains the following statement:

```
SQL> SELECT D.DNAME, E.ENAME, E.SAL, E.JOB
2 FROM EMP E, DEPT D
3 WHERE E.DEPTNO = D.DEPTNO
```

The statement can be automatically traced when it is run:

```
SQL> SET AUTOTRACE ON
SQL> /

DNAME          ENAME             SAL JOB
-------------- ---------- ---------- ---------
ACCOUNTING     CLARK            2450 MANAGER
ACCOUNTING     KING             5000 PRESIDENT
ACCOUNTING     MILLER           1300 CLERK
RESEARCH       SMITH             800 CLERK
RESEARCH       ADAMS            1100 CLERK
RESEARCH       FORD             3000 ANALYST
RESEARCH       SCOTT            3000 ANALYST
RESEARCH       JONES            2975 MANAGER
SALES          ALLEN            1600 SALESMAN
SALES          BLAKE            2850 MANAGER
SALES          MARTIN           1250 SALESMAN
SALES          JAMES             950 CLERK
SALES          TURNER           1500 SALESMAN
SALES          WARD             1250 SALESMAN

14 rows selected.

Execution Plan
-----------------------------------------------------------
0       SELECT STATEMENT Optimizer=CHOOSE
1    0   MERGE JOIN
2    1     SORT (JOIN)
3    2       TABLE ACCESS (FULL) OF 'DEPT'
4    1     SORT (JOIN)
5    4       TABLE ACCESS (FULL) OF 'EMP'

Statistics
----------------------------------------------------------
148  recursive calls
  4  db block gets
 24  consistent gets
  6  physical reads
 43  redo size
591  bytes sent via SQL*Net to client
256  bytes received via SQL*Net from client
 33  SQL*Net roundtrips to/from client
  2  sorts (memory)
  0  sorts (disk)
 14  rows processed
```

Note: The output may vary depending on the version of the server to which you are connected and the configuration of the server.