Get resultset from oracle stored procedure

Ask Question

29

16

I'm working on converting a stored procedure from SQL server to Oracle. This stored procedure provides a direct resultset. I mean that if you call the stored procedure in eg Management Studio you directly obtain the resultset.

By converting to Oracle I walk against the problem that I in Oracle will not display the resultset

I searched on the Internet and have seen that the stored procedure should yield a REF CURSOR, but I still walk with the problem to write a little piece of code to obtain the resultset en process that.

Pseudo Code:

Call stored procedure and obtain cursor Do something with that cursor so that my resultset appears

Someone an idea?

oracle stored-procedures plsql

asked Jul 23 '09 at 9:05
jwdehaan
520 3 7 24

I wounder. This question has > 90K views and has got only 20 up-vote. It deserves up-vote per view. :D – Dr. MAF Aug 23 '17 at 11:58

@Dr.MAF The question has almost 110,000 views now. Pretty astonishing if you ask me. – Wilson Sep 10 '18 at 1:24

@Wilson Sorry, I didn't get your idea. What shall I ask you? – Dr. MAF
Sep 10 '18 at 13:50

5 Answers

58

In SQL Plus:

```
SQL> create procedure myproc (prc out sys_refcursor)
is
begin
   open prc for select * from emp;
end;
/

Procedure created.

SQL> var rc refcursor
SQL> execute myproc(:rc)

PL/SQL procedure successfully completed.

SQL> print rc

     EMPNO ENAME      JOB              MGR HIREDATE           SAL
---------- ---------- --------- ---------- ----------- ---------- --
      7839 KING       PRESIDENT            17-NOV-1981       4999
      7698 BLAKE      MANAGER         7839 01-MAY-1981       2849
      7782 CLARKE     MANAGER         7839 09-JUN-1981       2449
      7566 JONES      MANAGER         7839 02-APR-1981       2974
      7788 SCOTT      ANALYST         7566 09-DEC-1982       2999
      7902 FORD       ANALYST         7566 03-DEC-1981       2999
      7369 SMITHY     CLERK           7902 17-DEC-1980       9988
      7499 ALLEN      SALESMAN        7698 20-FEB-1981       1599
      7521 WARDS      SALESMAN        7698 22-FEB-1981       1249
      7654 MARTIN     SALESMAN        7698 28-SEP-1981       1249
      7844 TURNER     SALESMAN        7698 08-SEP-1981       1499
      7876 ADAMS      CLERK           7788 12-JAN-1983       1099
      7900 JAMES      CLERK           7698 03-DEC-1981        949
      7934 MILLER     CLERK           7782 23-JAN-1982       1299
      6668 Umberto    CLERK           7566 11-JUN-2009      19999
      9567 ALLBRIGHT  ANALYST         7788 02-JUN-2009      76999
```

answered Jul 23 '09 at 9:16

Tony Andrews
108k 17 190 236

1 Excellent! Thanks for the Answer, Tony. Can I export these results to CSV through Unix/Linux script? – Crash OR Sep 15 '14 at 16:22

8 print rc is nice in sql plus, how can I have rc displayed in a grid in SQL Developer? – Stack0verflow Oct 8 '15 at 1:07

1 I wounder. This question has > 90K views and has got only 20 up-vote. It deserves up-vote per view. And your answer deserves 10 up-votes per view. Thank you very much. – Dr. MAF Aug 23 '17 at 11:59

4

Oracle is not sql server. Try the following in SQL Developer

```
variable rc refcursor;
exec testproc(:rc2);
print rc2
```

edited Mar 3 '16 at 9:45

Praveen
6,313 3 16 33

answered Jul 23 '09 at 9:21

softveda
9,286 4 38 46

1

In SQL Plus:

```
SQL> var r refcursor
SQL> set autoprint on
SQL> exec :r := function_returning_refcursor();
```

Replace the last line with a call to your procedure / function and the contents of the refcursor will be displayed

answered Jul 23 '09 at 9:15

stjohnroe
2,778 1 21 27

1

Hi I know this was asked a while ago but I've just figured this out and it might help someone else. Not sure if this is exactly what you're looking for but this is how I call a stored proc and view the output using SQL Developer.
In SQL Developer when viewing the proc, right click and choose 'Run' or select Ctrl+F11 to bring up the Run PL/SQL window. This creates a template with the input and output params which you need to modify. My proc returns a sys_refcursor. The tricky part for me was declaring a row type that is exactly equivalent to the select stmt / sys_refcursor being returned by the proc:

```
DECLARE
  P_CAE_SEC_ID_N NUMBER;
  P_FM_SEC_CODE_C VARCHAR2(200);
  P_PAGE_INDEX NUMBER;
  P_PAGE_SIZE NUMBER;
  v_Return sys_refcursor;
  type t_row is record (CAE_SEC_ID NUMBER,FM_SEC_CODE VARCHAR2(7),row
v_total_count number);
  v_rec t_row;

BEGIN
  P_CAE_SEC_ID_N := NULL;
  P_FM_SEC_CODE_C := NULL;
  P_PAGE_INDEX := 0;
  P_PAGE_SIZE := 25;

  CAE_FOF_SECURITY_PKG.GET_LIST_FOF_SECURITY(
    P_CAE_SEC_ID_N => P_CAE_SEC_ID_N,
    P_FM_SEC_CODE_C => P_FM_SEC_CODE_C,
    P_PAGE_INDEX => P_PAGE_INDEX,
    P_PAGE_SIZE => P_PAGE_SIZE,
    P_FOF_SEC_REFCUR => v_Return
  );
  -- Modify the code to output the variable
  -- DBMS_OUTPUT.PUT_LINE('P_FOF_SEC_REFCUR = ');
  loop
    fetch v_Return into v_rec;
    exit when v_Return%notfound;
    DBMS_OUTPUT.PUT_LINE('sec_id = ' || v_rec.CAE_SEC_ID || 'sec code
||v_rec.FM_SEC_CODE);
  end loop;

END;
```

answered Jan 14 '11 at 10:26

Ciaran Bruen
3,675 13 49 64

1

My solution was to create a pipelined function. The advantages are that the query can be a single line:

- `select * from table(yourfunction(param1, param2));`
- You can join your results to other tables or filter or sort them as you please..
- the results appear as regular query results so you can easily manipulate them.

To define the function you would need to do something like the following:

```
  -- Declare the record columns
  TYPE your_record IS RECORD(
     my_col1 VARCHAR2(50),
     my_col2 varchar2(4000)
  );
  TYPE your_results IS TABLE OF your_record;

  -- Declare the function
```

```
function yourfunction(a_Param1 varchar2, a_Param2 varchar2)
return your_results pipelined is
  rt          your_results;
begin
  -- Your query to load the table type
  select s.col1,s.col2
  bulk collect into rt
  from your_table s
  where lower(s.col1) like lower('%'||a_Param1||'%');

  -- Stuff the results into the pipeline..
  if rt.count > 0 then
    for i in rt.FIRST .. rt.LAST loop
      pipe row (rt(i));
    end loop;
  end if;

  -- Add more results as you please....
  return;
end find;
```

And as mentioned above, all you would do to view your results is:

```
select * from table(yourfunction(param1, param2)) t order by t.my_col
```

answered Jun 11 '17 at 4:13

AnthonyVO
2,199 1 23 30