# Run Stored Procedure in SQL Developer?

Ask Question

69

19

I am trying to run a stored procedure that has multiple in and out paramaters. The procedure can only be viewed in my Connections panel by navigating Other Users | | Packages | |

If I right click , the menu items are "Order Members By..." and "Create Unit Test" (greyed out). The ability to "Run" the procedure does not seem possible when it's accessed by user.

I have been trying to find an example of how to create an anonymous block so that I can run the procedure as a SQL file, but haven't found anything that works.

Does anyone know how I can execute this procedure from SQL Developer? I am using Version 2.1.1.64.

Thanks in advance!

**EDIT 1:**

The procedure I want to call has this signature:

```
user.package.procedure(
   p_1 IN  NUMBER,
   p_2 IN  NUMBER,
   p_3 OUT VARCHAR2,
   p_4 OUT VARCHAR2,
   p_5 OUT VARCHAR2,
   p_6 OUT NUMBER)
```

If I write my anonymous block like this:

```
DECLARE
   out1 VARCHAR2(100);
   out2 VARCHAR2(100);
   out3 VARCHAR2(100);
   out4 NUMBER(100);
BEGIN
   EXECUTE user.package.procedure (33,89, :out1, :out2, :out3, :out4);
END;
```

I get the error:

```
Bind Varialbe "out1" is NOT DECLCARED
anonymous block completed
```

I've tried initializing the out* variables:

```
   out1 VARCHAR2(100) := '';
```

but get the same error:

**EDIT 2:**

Based on Alex's answer, I tried removing the colons from in front of the params and get this:

```
Error starting at line 1 in command:
DECLARE
   out1 VARCHAR2(100);
   out2 VARCHAR2(100);
   out3 VARCHAR2(100);
   out4 NUMBER(100);
BEGIN
   EXECUTE user.package.procedure (33,89, out1, out2, out3, out4);
END;
Error report:
ORA-06550: line 13, column 17:
PLS-00103: Encountered the symbol "USER" when expecting one of the following:

   := . ( @ % ; immediate
The symbol ":=" was substituted for "USER" to continue.
06550. 00000 -  "line %s, column %s:\n%s"
*Cause:    Usually a PL/SQL compilation error.
*Action:
```

oracle stored-procedures oracle-sqldeveloper

edited Oct 21 '10 at 22:12

asked Oct 21 '10 at 20:39

sdoca 3,862 21 59 108

possible duplicate of Best way/tool to get the results from an oracle package procedure – OMG Ponies Oct 21 '10 at 20:52

Try putting the OUT variables inside the BEGIN, before the procedure execution statement. – OMG Ponies Oct 21 '10 at 21:56

You don't need the `execute` ; in PL/SQL that's interpreted as the start of `execute immediate` , which is different to SQL `execute` . – Alex Poole Oct 21 '10 at 22:08

2 @sdoca: you're confusing two approaches now; with your edit 2 version just remove the word `execute` . The `declare` should be before the `begin` . What I think @OMG meant was that you can declare the variables in SQL

Developer before the anonymous block with the `variable` keyword, and then use the `:out1` syntax as you had it originally, in which case you don't have a `declare` section at all. But you're mixing the two up from your last comment. – Alex Poole Oct 21 '10 at 22:24

1 Yep, I knew I was confused, but wasn't quite sure where/how. – sdoca Oct 22 '10 at 16:14

## 12 Answers

71

With simple parameter types (i.e. not refcursors etc.) you can do something like this:

```
SET serveroutput on;
DECLARE
    InParam1 number;
    InParam2 number;
    OutParam1 varchar2(100);
    OutParam2 varchar2(100);
    OutParam3 varchar2(100);
    OutParam4 number;
BEGIN
    /* Assign values to IN parameters
    InParam1 := 33;
    InParam2 := 89;

    /* Call procedure within package,
    schema.package.procedure(InParam1
        OutParam1, OutParam2, OutPara

    /* Display OUT parameters */
    dbms_output.put_line('OutParam1:
    dbms_output.put_line('OutParam2:
    dbms_output.put_line('OutParam3:
    dbms_output.put_line('OutParam4:
END;
/
```

**Edited** to use the OP's spec, and with an alternative approach to utilise `:var` bind variables:

```
var InParam1 number;
var InParam2 number;
var OutParam1 varchar2(100);
var OutParam2 varchar2(100);
var OutParam3 varchar2(100);
var OutParam4 number;

BEGIN
    /* Assign values to IN parameters
    :InParam1 := 33;
    :InParam2 := 89;

    /* Call procedure within package,
    schema.package.procedure(:InParam
        :OutParam1, :OutParam2, :OutP
END;
/
```

```
-- Display OUT parameters
print :OutParam1;
print :OutParam2;
print :OutParam3;
print :OutParam4;
```

edited Oct 21 '10 at 22:33

answered Oct 21 '10 at 21:45

Alex Poole
**133k** 6 106 179

1 +1 nice answer. Out of curiosity do you know which is preferred? – Conrad Frix Oct 21 '10 at 23:05

@Conrad: I imagine it's a preference thing, though there might be more context switching the `:var` way. I'd use the `declare` way by default if I was doing anything with PL/SQL; but I might use `:var` if, say, I was using a bit of existing code copied in from Pro*C which already had that syntax and I didn't want to touch the parms in the call. – Alex Poole Oct 22 '10 at 6:41

1 Thanks for the help and the detailed answer. I'm sure it will be a help to others as well. One thing to note is that these must be run as scripts not statements. – sdoca Oct 22 '10 at 16:15

1 When running in sql developer, I didn't have to prefix the variable names with `:` (in fact, it didn't work with it). – haridsv Mar 20 '13 at 12:48

25

Executing easy. Getting the results can be hard.

Take a look at this question I asked [Best way/tool to get the results from an oracle package procedure]

The summary of it goes like this.

Assuming you had a Package named mypackage and procedure called getQuestions. It returns a refcursor and takes in string user name.

All you have to do is create new SQL File (file new). Set the connection and paste in the following and execute.

```
var r refcursor;
exec mypackage.getquestions(:r, 'OMG P
print r;
```

edited May 23 '17 at 11:47
Community ♦
1 1

answered Oct 21 '10 at 20:48
Conrad Frix
45.8k 11 72 123

3 I had to use the full word "execute" rather than "exec" – Patrick Nov 21 '14 at 23:01

16

For those using SqlDeveloper 3+, in case you missed that:

SqlDeveloper has feature to execute stored proc/function directly, and output are displayed in a easy-to-read manner.

Just right click on the package/stored proc/ stored function, Click on `Run` and choose `target` to be the proc/func you want to execute, SqlDeveloper will generate the code snippet to execute (so that you can put your input parameters). Once executed, output parameters are displayed in lower half of the dialog box, and it even have built-in support for ref cursor: result of cursor will be displayed as a separate output tab.

answered Jun 3 '15 at 6:37
Adrian Shum
28.8k 7 63 107

1 This should be the selected answer. – EvilTeach Jun 14 '17 at 19:36

yes this should be the answer – Zcon Sep 14 '18 at 8:12

11

Open the procedure in SQL Developer and run it from there. SQL Developer displays the SQL that it runs.

```
BEGIN
  PROCEEDURE_NAME_HERE();
END;
```

answered Jan 5 '12 at 20:42

Noel
**407** 6 9

6

Use:

```
BEGIN

  PACKAGE_NAME.PROCEDURE_NAME(paramete

END;
```

Replace "PACKAGE_NAME", "PROCEDURE_NAME", and "parameter_value" with what you need. OUT parameters will need to be declared prior to.

answered Oct 21 '10 at 20:49

OMG Ponies
**259k** 62 443 469

1

None of these other answers worked for me. Here's what I had to do to run a procedure in SQL Developer 3.2.20.10:

```
SET serveroutput on;
DECLARE
  testvar varchar(100);
BEGIN
  testvar := 'dude';
  schema.MY_PROC(testvar);
  dbms_output.enable;
  dbms_output.put_line(testvar);
END;
```

And then you'd have to go check the table for whatever your proc was supposed to do with that passed-in variable -- the output will just confirm that the variable received the value (and theoretically, passed it to the proc).

**NOTE** (differences with mine vs. others):

- No `:` prior to the variable name

- No putting `.package.` or `.packages.` between the schema name and the procedure name
- No having to put an `&` in the variable's value.
- No using `print` anywhere
- No using `var` to declare the variable

All of these problems left me scratching my head for the longest and these answers that have these egregious errors out to be taken out and tarred and feathered.

edited Aug 30 '16 at 21:22

answered Aug 30 '16 at 21:12
vapcguy
3,232 27 31

0

Can't believe, this won't execute in SQL Developer:

```
var r refcursor;
exec PCK.SOME_SP(:r,
 '02619857');

print r;
```

BUT this will:

```
var r refcursor;
exec TAPI_OVLASCENJA.ARH_SELECT_NAKON_

print r;
```

Obviously everything has to be in one line..

answered Jul 2 '13 at 13:10
Amel Music
74 1 8

2 The [SQL*Plus docs for the `execute` command] state that. This is not an answer to the question that was asked though, and has been covered in more relevant answers to other questions before anyway. – Alex Poole Jul 5 '13 at 10:55

Was able to use it in SQL Developer. Thanks – Sergejs Sep 16 '15 at 9:18

0

Though this question is quite old, I keep stumbling into same result without finding an easy way to run from sql developer. After couple of tries, I found an easy way to execute the stored procedure from sql developer itself.

- Under packages, select your desired package and right click on the package name (not on the stored procedure name).
- You will find option to run. Select that and supply the required arguments. Click OK and you can see the output in output variables section below

I'm using SQL developer version 4.1.3.20

answered Mar 13 '18 at 18:46
Narasimha
**537** 7 12

0

Using SQL Developer Version 4.0.2.15 Build 15.21 the following works:

```
SET SERVEROUTPUT ON
var InParam1 varchar2(100)
var InParam2 varchar2(100)
var InParam3 varchar2(100)
var OutParam1 varchar2(100)

BEGIN
    /* Assign values to IN parameters
    :InParam1 := 'one';
    :InParam2 := 'two';
    :InParam3 := 'three';

    /* Call procedure within package,
    schema.package.procedure(:InParam1
    dbms_output.enable;
    dbms_output.put_line('OutParam1: '
END;
/
```

edited Jul 25 '14 at 20:05
JasonMArcher
**9,363** 10 48 49

answered Jul 25 '14 at 20:04
Jack
**11**

No, sorry -- `var` before the variables DOES NOT work - at least in SQL Developer 3.2.20.10, and shouldn't use colons in front of them - no need for it, again, at least in 3.2.20.10 (only reason I didn't vote this down or edit it). Also would need a semi-colon after `SET SERVEROUTPUT ON` . – vapcguy Aug 30 '16 at 21:17

-1

I wasn't able to get @Alex Poole answers working. However, by trial and error, I found the following works (using SQL Developer version 3.0.04). Posting it here in case it helps others:

```
SET serveroutput on;

DECLARE
    var InParam1 number;
    var InParam2 number;
    var OutParam1 varchar2(100);
    var OutParam2 varchar2(100);
    var OutParam3 varchar2(100);
    var OutParam4 number;

BEGIN
    /* Assign values to IN parameters
    InParam1 := 33;
    InParam2 := 89;

    /* Call procedure within package,
    schema.package.procedure(InParam1,
        OutParam1, OutParam2, OutParam

    /* Display OUT parameters */
    dbms_output.put_line('OutParam1: '
    dbms_output.put_line('OutParam2: '
    dbms_output.put_line('OutParam3: '
    dbms_output.put_line('OutParam4: '
END;
```

answered Sep 21 '13 at 19:01

ggkmath
2,148 18 61 121

2 That's the same as the first version in my answer, except you've added a `var` to each variable in the `declare` block, which is invalid. Trying to run this gives 'PLS-00103: Encountered the symbol "NUMBER" when expecting one of the following ...', and similar errors against the other five variables. – Alex Poole Nov 20 '13 at 13:24

I agree with Alex. Downvoted because the `var` , and I did not need to use `.package.` in my call, at least in 3.2.20.10, which should not have been so different, and got errors when

I did. Wasted lots of time with this answer. – vapcguy Aug 30 '16 at 21:22

-2

Creating Pl/SQL block can be painful if you have a lot of procedures which have a lot of parameters. There is an application written on python that do it for you. It parses the file with procedure declarations and creates the web app for convenient procedure invocations.

answered Dec 11 '15 at 15:25
nmax
**21** 1 5

-3

```
var out_para_name refcursor;
execute package_name.procedure_name(ir
print :out_para_name;
```

answered Sep 5 '12 at 19:29
Helper
1