

-- 1: Proc AddNum: this proc will add two numbers X & Y, and return the result in a third variable Z

create or REPLACE PROCEDURE AddNum(x IN number, y IN number, z OUT number)

AS

BEGIN

z:= x + y;

end;

/

DECLARE

x number;

y number;

z number;

BEGIN

x := 5;

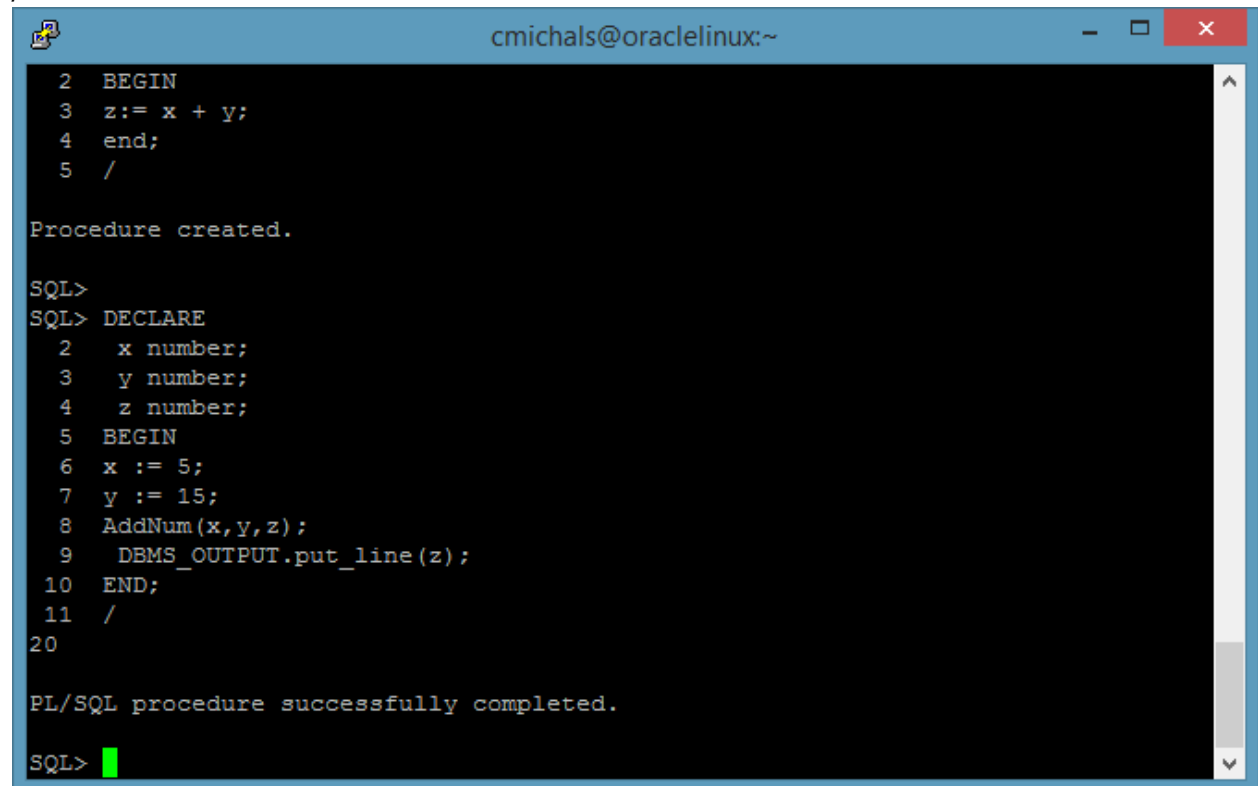
y := 15;

AddNum(x,y,z);

DBMS\_OUTPUT.put\_line(z);

END;

/



The screenshot shows a terminal window titled 'cmichals@oraclelinux:~'. The terminal contains the following text:

```
2 BEGIN
3 z:= x + y;
4 end;
5 /

Procedure created.

SQL>
SQL> DECLARE
2   x number;
3   y number;
4   z number;
5 BEGIN
6 x := 5;
7 y := 15;
8 AddNum(x,y,z);
9   DBMS_OUTPUT.put_line(z);
10 END;
11 /
20

PL/SQL procedure successfully completed.

SQL> █
```

-- 2: Proc MultiplyNum: this proc will multiply two numbers X & Y, and return the result in a third variable Z

```
create or REPLACE PROCEDURE MultiplyNum(x IN number, y IN number, z OUT  
number) AS
```

```
BEGIN
```

```
z:= x * y;
```

```
end;
```

```
/
```

```
DECLARE
```

```
x number;
```

```
y number;
```

```
z number;
```

```
BEGIN
```

```
x := 5;
```

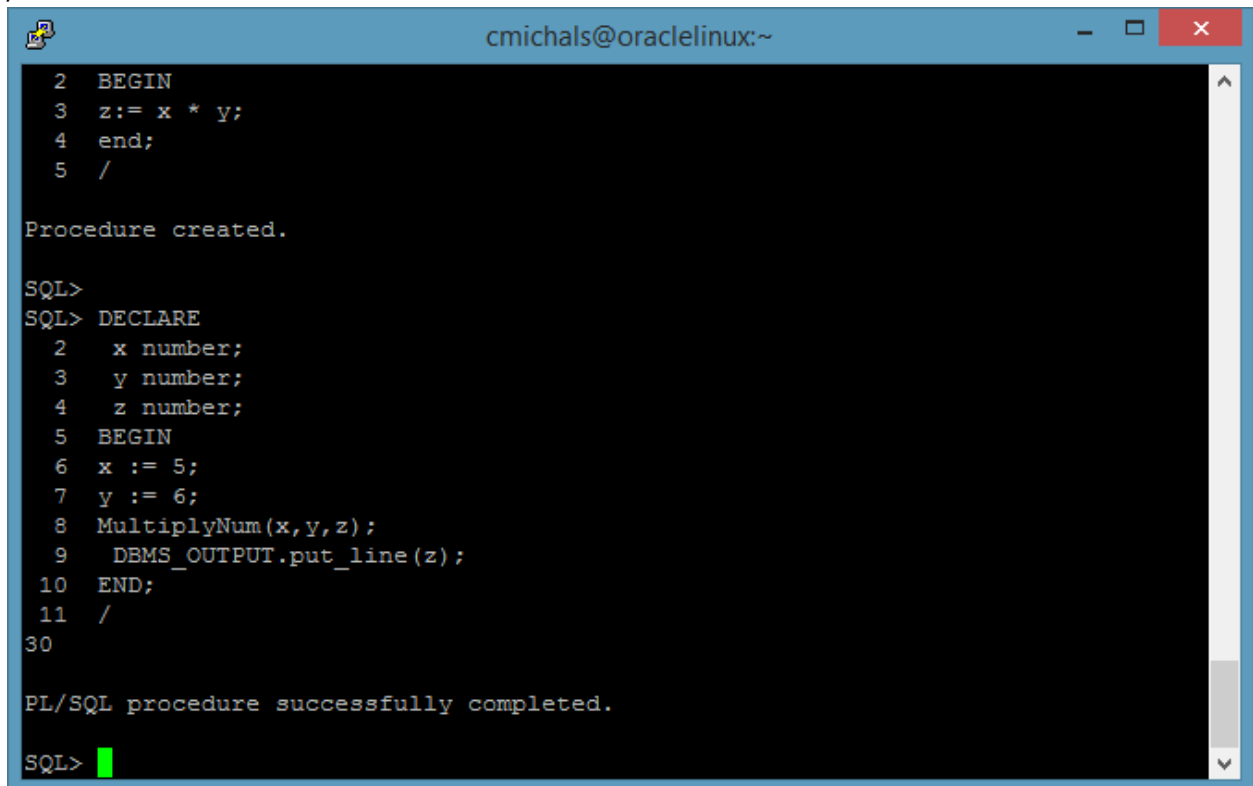
```
y := 6;
```

```
MultiplyNum(x,y,z);
```

```
DBMS_OUTPUT.put_line(z);
```

```
END;
```

```
/
```

A screenshot of a terminal window titled 'cmichals@oraclelinux:~'. The terminal shows a sequence of SQL commands and their outputs. The first command is a PL/SQL procedure definition for 'MultiplyNum', which is successfully created. The second command is a PL/SQL block that declares variables x, y, and z, assigns values to x and y, calls the 'MultiplyNum' procedure, and prints the result of z. The output shows '30' and a confirmation message 'PL/SQL procedure successfully completed.'.

```
cmichals@oraclelinux:~  
2 BEGIN  
3 z:= x * y;  
4 end;  
5 /  
  
Procedure created.  
  
SQL>  
SQL> DECLARE  
2 x number;  
3 y number;  
4 z number;  
5 BEGIN  
6 x := 5;  
7 y := 6;  
8 MultiplyNum(x,y,z);  
9 DBMS_OUTPUT.put_line(z);  
10 END;  
11 /  
30  
  
PL/SQL procedure successfully completed.  
SQL>
```

```

-- 3: Proc POWERR(n,r, R): this proc will raise the first number n, to the power r &
return the result in the variable R.
create or REPLACE PROCEDURE POWERR(n IN Integer, r IN Integer, O OUT Integer)
AS
loopCounter Integer;
BEGIN
    loopCounter := 0;
    IF r < 0 THEN
        raise_application_error(-20101, 'Can not calculate negative exponents');
    END IF;
    if r = 0 THEN
        O := 0;
        return;
    END IF;
    O := n;
    loop
        exit when loopCounter = r;
        O := n * O;
        loopCounter := loopCounter + 1;
    end loop;
end;
/

DECLARE
x Integer;
y Integer;
z Integer;
BEGIN
x := 5;
y := 6;
POWERR(x,y,z);
    DBMS_OUTPUT.put_line(z);
END;
/

```

```
cmichals@oraclelinux:~  
16         loopCounter := loopCounter + 1;  
17     end loop;  
18 end;  
19 /  
  
Procedure created.  
  
SQL>  
SQL> DECLARE  
2   x Integer;  
3   y Integer;  
4   z Integer;  
5 BEGIN  
6   x := 5;  
7   y := 6;  
8   POWERR(x,y,z);  
9   DBMS_OUTPUT.put_line(z);  
10 END;  
11 /  
78125  
  
PL/SQL procedure successfully completed.  
  
SQL> █
```

-- 4: Proc POWER3(x): this proc will take the value x and return X^3 in the same variable

create or REPLACE PROCEDURE POWER3(x in out Integer) AS

BEGIN

x := x \* x \* x;

END;

/

DECLARE

x Integer;

BEGIN

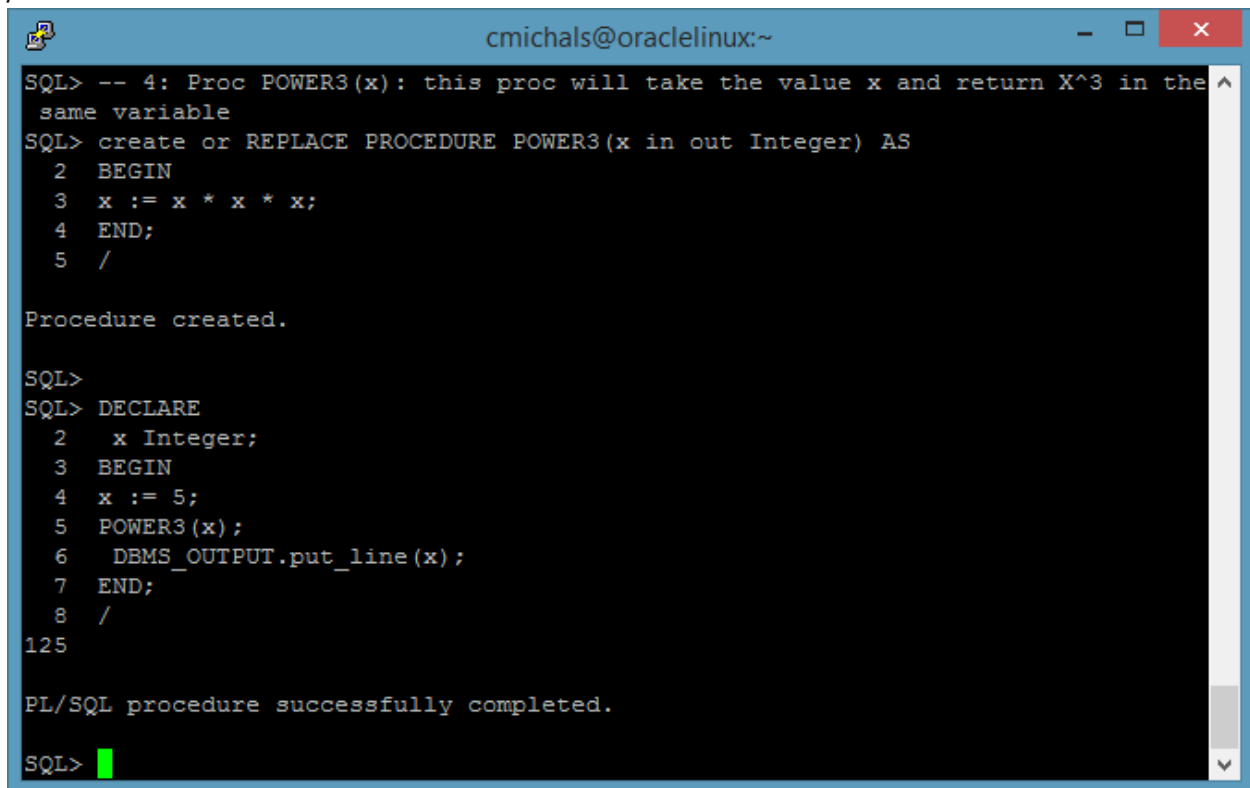
x := 5;

POWER3(x);

DBMS\_OUTPUT.put\_line(x);

END;

/



The screenshot shows a terminal window titled 'cmichals@oraclelinux:~'. The terminal contains the following text:

```
SQL> -- 4: Proc POWER3(x): this proc will take the value x and return X^3 in the
same variable
SQL> create or REPLACE PROCEDURE POWER3(x in out Integer) AS
  2 BEGIN
  3 x := x * x * x;
  4 END;
  5 /

Procedure created.

SQL>
SQL> DECLARE
  2 x Integer;
  3 BEGIN
  4 x := 5;
  5 POWER3(x);
  6 DBMS_OUTPUT.put_line(x);
  7 END;
  8 /
125

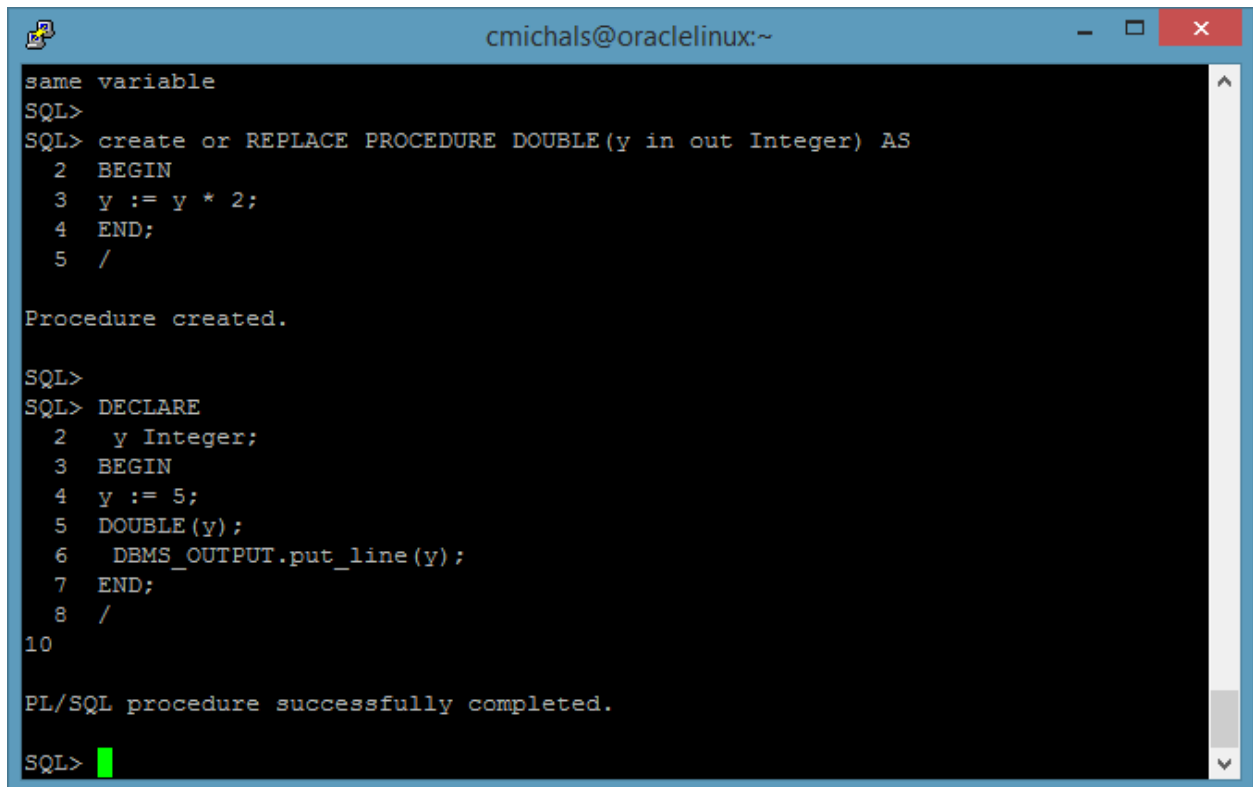
PL/SQL procedure successfully completed.

SQL> █
```

-- 5: Proc DOUBLE(y): this proc will take the value y and return 2y in the same variable

```
create or REPLACE PROCEDURE DOUBLE(y in out Integer) AS
BEGIN
y := y * 2;
END;
/
```

```
DECLARE
y Integer;
BEGIN
y := 5;
DOUBLE(y);
DBMS_OUTPUT.put_line(y);
END;
/
```

A screenshot of a terminal window titled 'cmichals@oraclelinux:~'. The terminal shows a series of SQL commands and their outputs. The first command is 'same variable', followed by 'SQL>'. Then, a PL/SQL procedure 'DOUBLE' is created with the body 'BEGIN y := y \* 2; END; /'. The output is 'Procedure created.'. Next, a DECLARE block is executed: 'SQL> DECLARE y Integer; BEGIN y := 5; DOUBLE(y); DBMS\_OUTPUT.put\_line(y); END; /'. The output is '10' followed by 'PL/SQL procedure successfully completed.'. The prompt 'SQL>' is shown at the bottom with a green cursor.

```
cmichals@oraclelinux:~
same variable
SQL>
SQL> create or REPLACE PROCEDURE DOUBLE(y in out Integer) AS
  2 BEGIN
  3 y := y * 2;
  4 END;
  5 /

Procedure created.

SQL>
SQL> DECLARE
  2 y Integer;
  3 BEGIN
  4 y := 5;
  5 DOUBLE(y);
  6 DBMS_OUTPUT.put_line(y);
  7 END;
  8 /
10

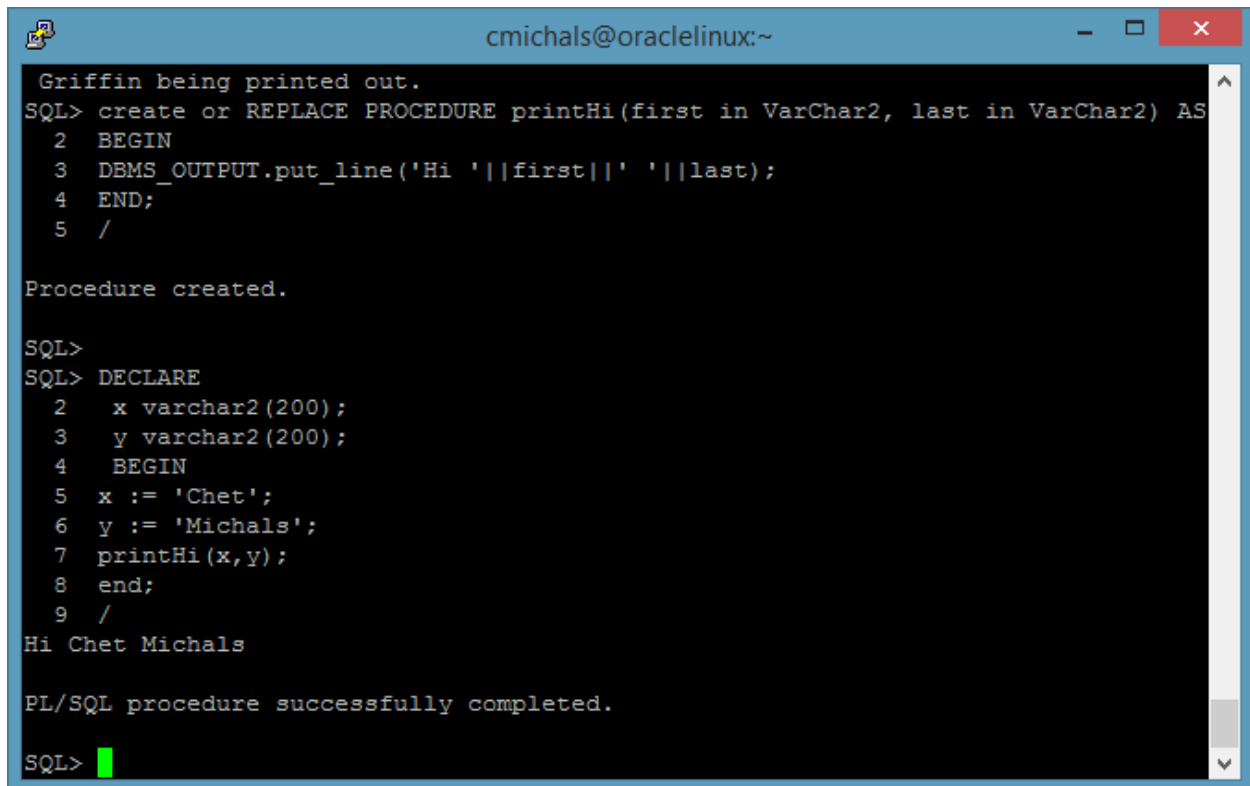
PL/SQL procedure successfully completed.

SQL> █
```

-- 6: Proc printHi(first, last): this proc takes two strings, first name & last name, and prints: "Hi <first last> to the console. E.g. calling printHi('Blake', 'Griffin') , results in: Hi Blake Griffin being printed out.

```
create or REPLACE PROCEDURE printHi(first in VarChar2, last in VarChar2) AS
BEGIN
DBMS_OUTPUT.put_line('Hi '||first||' '||last);
END;
/
```

```
DECLARE
x varchar2(200);
y varchar2(200);
BEGIN
x := 'Chet';
y := 'Michals';
printHi(x,y);
end;
/
```

A screenshot of a terminal window titled 'cmichals@oraclelinux:~'. The terminal shows the execution of SQL commands. First, a procedure 'printHi' is created. Then, variables 'x' and 'y' are declared and assigned values 'Chet' and 'Michals' respectively. Finally, the procedure 'printHi' is called with these variables, resulting in the output 'Hi Chet Michals'. The terminal also shows the message 'PL/SQL procedure successfully completed.' and a green cursor at the end of the last 'SQL>' prompt.

```
cmichals@oraclelinux:~
Griffin being printed out.
SQL> create or REPLACE PROCEDURE printHi(first in VarChar2, last in VarChar2) AS
  2 BEGIN
  3 DBMS_OUTPUT.put_line('Hi '||first||' '||last);
  4 END;
  5 /

Procedure created.

SQL>
SQL> DECLARE
  2 x varchar2(200);
  3 y varchar2(200);
  4 BEGIN
  5 x := 'Chet';
  6 y := 'Michals';
  7 printHi(x,y);
  8 end;
  9 /

Hi Chet Michals

PL/SQL procedure successfully completed.

SQL> █
```