**Database Systems Lab - 14CS2012**

**REGISTER NO: UR14CS228**

**DATE: 31-10-16**

**EXPERIMENT-NO 8**

**Video Link :** <https://youtu.be/Szpl-eIcNG4>

**AIM:**

To Create Functions and Procedures for the following requirements.

**DESCRIPTION:**

PL/SQL subprograms are named PL/SQL blocks that can be invoked with a set of parameters. PL/SQL provides two kinds of subprograms:

* **Functions**: these subprograms return a single value, mainly used to compute and return a value.
* **Procedures**: these subprograms do not return a value directly, mainly used to perform an action.

**Program:**

1. Write a PL/SQL procedure that will accept the product id from the user, check if the product is supplied by the supplier and display the status.

create or replace procedure p1 (pid in number) is

sid supplier.S\_ID%type;

begin

select S\_ID into sid from supplier where P\_ID=pid;

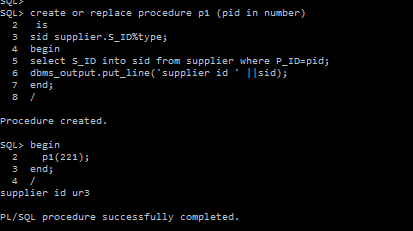
dbms\_output.put\_line('supplier id ' ||sid);

end;

begin

p1(221);

end;



2. Write a procedure to calculate total price for the product that has been supplied and Pass the supplier id as the argument.

create or replace procedure p2 (sid in varchar2)

is

pi orderquantity.price%type;

oq supplier.S\_Q%type;

begin

select orderquantity.price into pi from orderquantity,supplier where supplier.S\_ID=sid and rownum=1;

select S\_Q into oq from supplier where S\_ID=sid;

dbms\_output.put\_line('total price : '||pi\*oq);

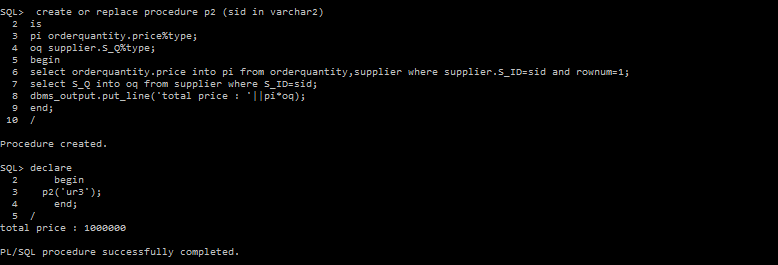
end;

declare

begin

p2('ur3');

end;



3. Write a procedure raise\_quantity which increases the quantity of the ordered product. It accepts an order id and quantity to be increased. It uses the order id to find the current quantity from the ORDER table and update the quantity.

create or replace procedure raise\_quantity(q in number,o in varchar2,q1 out number) is

begin

select distinct(order\_q) into q1 from orderquantity where order\_id=o;

dbms\_output.put\_line('present quantity : ' || q1);

update orderquantity set order\_q=(q1+q) where order\_id=o;

select distinct(order\_q) into q1 from orderquantity where order\_id=o;

dbms\_output.put\_line('updated quantity : '|| q1);

end;

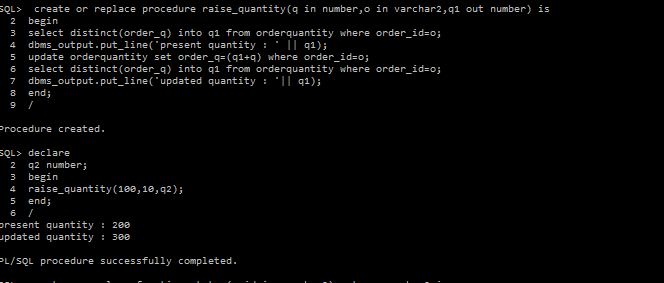
declare

q2 number;

begin

raise\_quantity(100,10,q2);

end;



4. Write a PL/SQL function STATUS to return value SUPPLIED if the product number passed to it is available in the Supplier table else will return NOT SUPPLIED.

create or replace function status(p\_id in varchar2) return varchar2 is

a varchar2(30);

b varchar2(30);

cursor br is

select distinct(p\_id) from supplier where p\_id=p\_id;

begin

open br;

fetch br into b;

if p\_id=b then

a:='supplied';

else

a:='notsupplied';

end if;

return a;

end;

declare

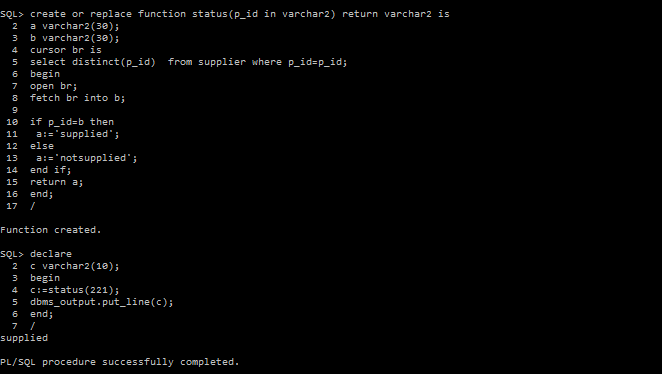
c varchar2(100);

begin

c:=status(21);

dbms\_output.put\_line(c);

end;



5. Write a PL/SQL function to return the product name when the product id is passed as an argument.

create or replace function f1(p\_id in varchar2)

return varchar2 is

pd varchar2(20);

begin

select product\_description into pd from product where product\_id=p\_id;

return pd;

end;

declare

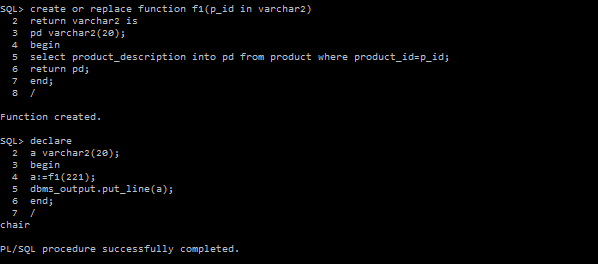
a varchar2(20);

begin

a:=f1(221);

dbms\_output.put\_line(a);

end;



6. Write a PL/SQL function to return the maximum quantity ordered by the customers.

create or replace function f2

return number is

q number;

begin

select max(order\_q) into q from orderquantity;

return q;

end;

declare

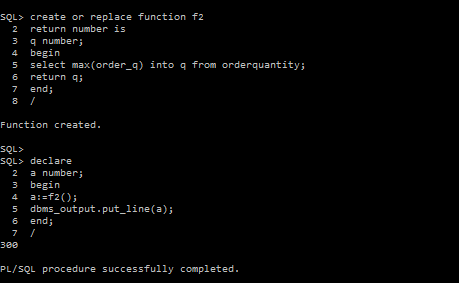
a number;

begin

a:=f2();

dbms\_output.put\_line(a);

end;



**Result:**

Functions and procedures were successfully created and tested for all the different situations