Assignment 8:

# Table creation and insertion:

```
create table Inventory(
   ProductID number(38,0) generated by default as identity primary key,
   ProductName varchar2(2000) not null,
   Category varchar2(2000) not null,
   Quantity number(38,0) not null,
   PricePerUnit number(38,10) not null
);
```

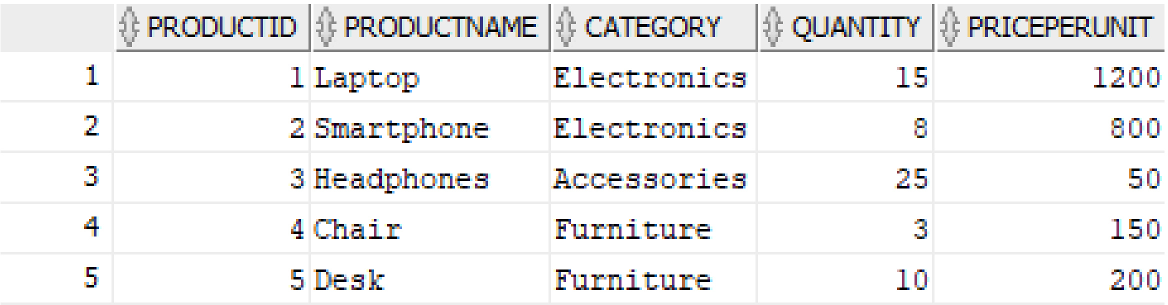
```
insert into Inventory(ProductName, Category, Quantity, PricePerUnit)
values('Laptop', 'Electronics', 15, 1200.00);
```

```
insert into Inventory(ProductName, Category, Quantity, PricePerUnit)
values('Smartphone', 'Electronics', 8, 800.00);
```

```
insert into Inventory(ProductName, Category, Quantity, PricePerUnit)
values('Headphones', 'Accessories', 25, 50.00);
```

```
insert into Inventory(ProductName, Category, Quantity, PricePerUnit)
values('Chair', 'Furniture', 3, 150.00);
```

```
insert into Inventory(ProductName, Category, Quantity, PricePerUnit)
values('Desk', 'Furniture', 10, 200.00);
```

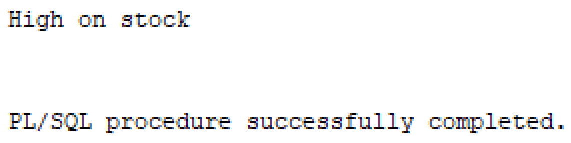


	PRODUCTID	PRODUCTNAME	CATEGORY	QUANTITY	PRICEPERUNIT
1	1	Laptop	Electronics	15	1200
2	2	Smartphone	Electronics	8	800
3	3	Headphones	Accessories	25	50
4	4	Chair	Furniture	3	150
5	5	Desk	Furniture	10	200

1- Anonymous Block with Conditions:

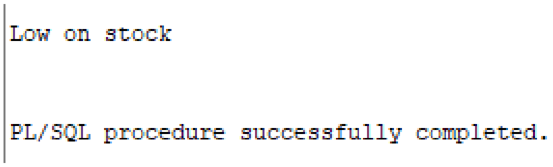
```
set SERVEROUTP on;
```

```
declare
vQuantity Inventory.Quantity%type;
vProductID Inventory.ProductID%type :=3 ;
begin
select Quantity into vQuantity from Inventory where ProductID = vProductID;
if (vQuantity < 5) then
   dbms_output.put_line('Low on stock');
elsif (vQuantity >= 5 and vQuantity <= 20) then
   dbms_output.put_line('Sufficient on Stock');
elsif (vQuantity > 20) then
   dbms_output.put_line('High on stock');
end if;
end;
```



```
High on stock


PL/SQL procedure successfully completed.
```

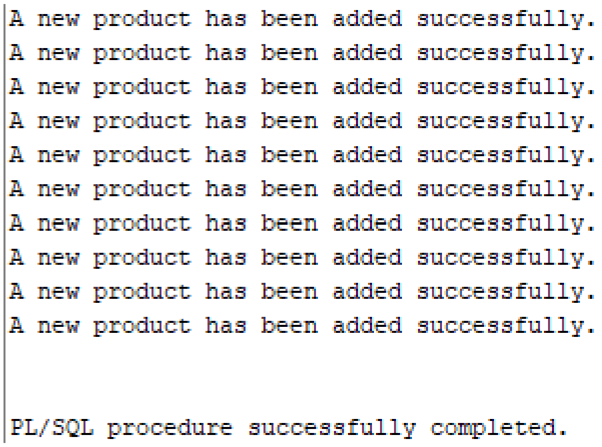
If productid = 4:



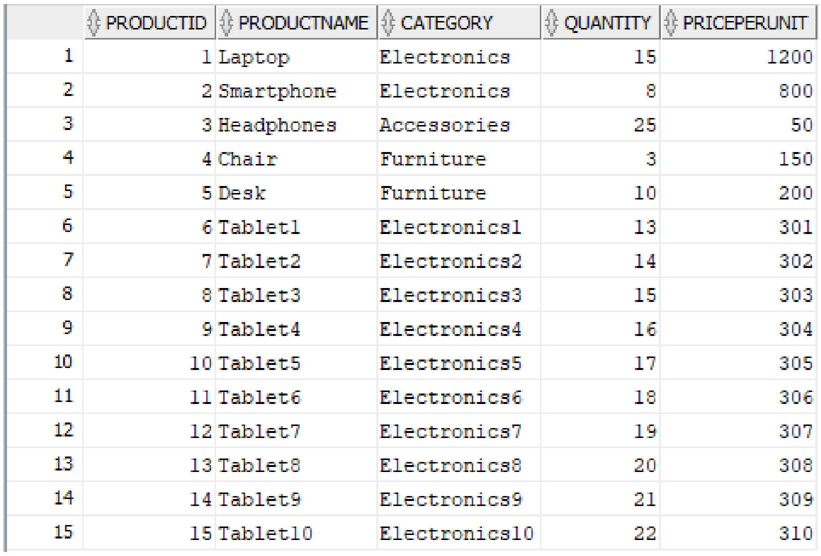
```
Low on stock


PL/SQL procedure successfully completed.
```

2- Loop with insert statements:

```
declare

vProductName Inventory.ProductName%type := 'Tablet';

vCategory Inventory.Category%type := 'Electronics';

vQuantity Inventory.Quantity%type := 12;

vPricePerUnit Inventory.PricePerUnit%type := 300.00;

counter number(10);

begin

for counter in 1 .. 10 loop

   insert into Inventory(ProductName, Category, Quantity, PricePerUnit)
values(vProductName || counter, vCategory || counter, vQuantity + counter,
vPricePerUnit + counter);

   dbms_output.put_line('A new product has been added successfully.');

end loop;

end;
```



```
A new product has been added successfully.
A new product has been added successfully.
A new product has been added successfully.
A new product has been added successfully.
A new product has been added successfully.
A new product has been added successfully.
A new product has been added successfully.
A new product has been added successfully.
A new product has been added successfully.
A new product has been added successfully.


PL/SQL procedure successfully completed.
```



	PRODUCTID	PRODUCTNAME	CATEGORY	QUANTITY	PRICEPERUNIT
1	1	Laptop	Electronics	15	1200
2	2	Smartphone	Electronics	8	800
3	3	Headphones	Accessories	25	50
4	4	Chair	Furniture	3	150
5	5	Desk	Furniture	10	200
6	6	Tablet1	Electronics1	13	301
7	7	Tablet2	Electronics2	14	302
8	8	Tablet3	Electronics3	15	303
9	9	Tablet4	Electronics4	16	304
10	10	Tablet5	Electronics5	17	305
11	11	Tablet6	Electronics6	18	306
12	12	Tablet7	Electronics7	19	307
13	13	Tablet8	Electronics8	20	308
14	14	Tablet9	Electronics9	21	309
15	15	Tablet10	Electronics10	22	310

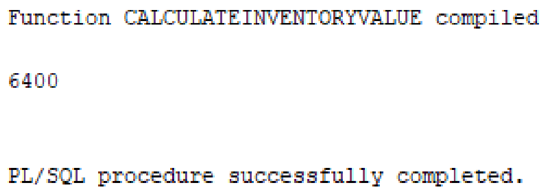
3- Function to calculate total value:

- Function Creation:

```
create or replace function CalculateInventoryValue (id number)
return number is
result number;
begin
select Quantity * PricePerUnit into result from Inventory where ProductID = id;
return result;
end;
```

- Function Call:

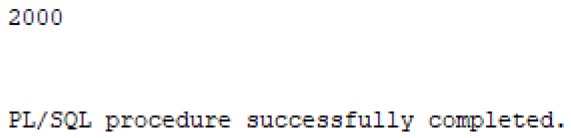
```
begin
dbms_output.put_line(CalculateInventoryValue(2));
end;
```



```
Function CALCULATEINVENTORYVALUE compiled

6400


PL/SQL procedure successfully completed.
```

- Function Call:

```
begin
dbms_output.put_line(CalculateInventoryValue(5));
end;
```



```
2000


PL/SQL procedure successfully completed.
```