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

);

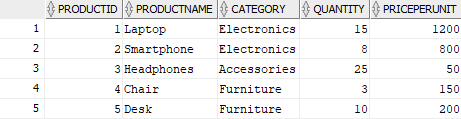
insert into Inventory(ProductName, Category, Quantity, PricePerUnit) values('Laptop', 'Electronics', 15, 1200.00);

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);



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

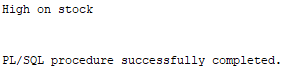
dbms\_output.put\_line('Sufficient on Stock');

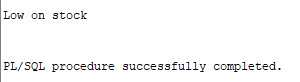
elsif (vQuantity > 20) then

dbms\_output.put\_line('High on stock');

end if;

end;



If productid = 4:

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

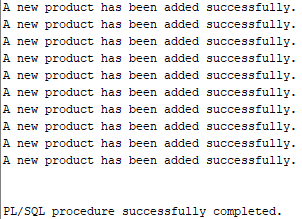
for counter in 1 .. 10 loop

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 table with numbers and words

Description automatically generated

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;

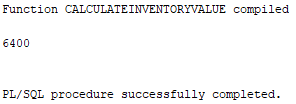
end;

* Function Call:

begin

dbms\_output.put\_line(CalculateInventoryValue(2));

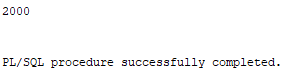
end;



* Function Call:

begin

dbms\_output.put\_line(CalculateInventoryValue(5));

end;