- **Project Name : Cosmetic Shop Management system**
- **DatabaseName : Cosmetic Shop Management system**
- **Database Software : Oracle 10g**

- **Database Description:**

Cosmetic shop management system is a complete solution for managing a shop.In other words,on enhanced tool that assists in organizing the day to day entities of a store.There is a need of an application for the sufficient management and handling customer orders.The cosmetic shop management system keeps every record and by reducing paperwork.

Product details include product id,product name,product price ,color,brand, etc. Customer table includes details such as customer id ,customer name, address and contact no,etc. Employee details includes detail such as employee id , name, address, phone no. payment details includes details such as payment no, type,totalamount.Order_info details include details such as order id, ordered date, received date, etc.

- **List of tables :**
  - **Product table** (This table include details about product. This table include attribute such as id, name, price,color, etc.)
  - **Customer table**(This table include details about customer. This table include attributes such as id, name, address,contact no, etc.)
  - **Payment table** (This table includes details about payment details. This table includes no, payment type, price,date, etc.)
  - **Employee table** (This table includes details about employees who worked in shop. This table include attributes such as name, id, salary,deptno,address,contact no etc.)
  - **Order_info table** (This table includes details of orders . This table include attributes such as order_id, ordered_date,required_date, etc.)

- **List of Constraints applied on tables:**

  1.Primary key constraint.

  2. Foreign key constraint.

  3. Unique key constraint.

  4. Not Null constraint.

  5. Check constraint.

Normalized Forms of court management system:

Database without normalization :

Product Id	**Product name**	**Product price**	**color**	**Brand**
101	Lipstick	200	Red	Lakme
102	FaceWash	300	NULL	Oriflame
103	Eyeliner	400	Black	Loreal
104	Lipbalm	250	Pink	Lakme

Customer id	**Customer name**	**Address**	**Contact no**	**Employee id**
10	Pallavi	Sinhagad Road	8989898989	111
20	Shreya	Mumbai	8767676567	112
30	Samiksha	Pune	3467890987	113
40	Shubhada	Kolhapur	9090878765	114

Employee	**Dept no**	**Employee**	**Employee**	**Contact**

name		salary	Qualification	no
Sakshi	10	90000	Engineer	5667898889
Hindavi	20	78900	Engineer	1234455666
Sanika	10	80000	Engineer	6789890987
Prerana	10	67787	Engineer	9876543201

Payment no	Payment type	Amount	Order id	Order date
1234	Cash	8000	1111	31-MAY-2018
5678	Credit Card	10000	2222	1-FEB-2018
9012	Online	12000	3333	3-MAR-2018
3456	Cash	3000	4444	24-APR-2017

1NF(First Normal Form) :

FIELD	KEY
Product id	Primary key
Customer id	Primary key
Employee id	Primary key
Payment no	Primary key
Order id	Primary Key

2)Second Normal Form(2NF):

a.Product Table

Product ID	**Product name**	**Product price**	**Product Color**	**Brand**
101	Lipstick	220	Red	Lakme
102	Eyeliner	300	Black	Lakme
103	Lipbalm	150	Pink	Lakme

b. Customer table:

Customer id	**Customer Name**	**Customer address**	**Contact no**
10	Pallavi	Pune	5678909090
11	Samiksha	Mumbai	5678989898

c. Employee table:

Employee id	Employee name	Dept no	Employee salary
111	Shreya	10	90000
112	Shubhada	20	80000
113	Sanika	10	70000

d. Payment table:

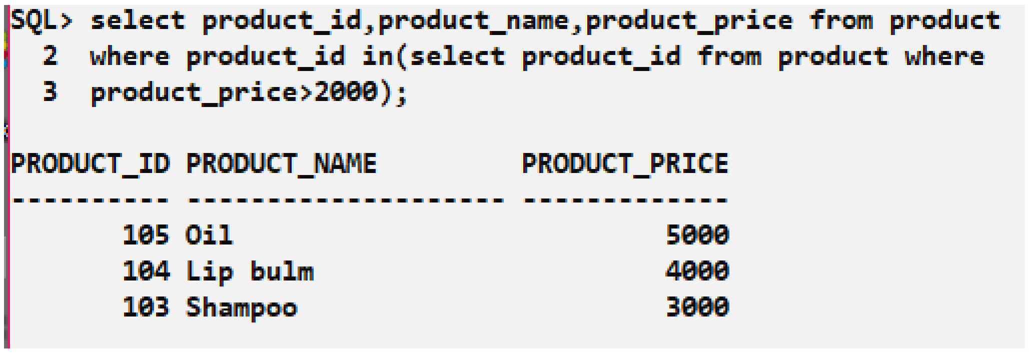
Payment no	Payment Type	amount
1234	Cash	9000
3456	Card	8000
5678	Online	10000

e. Order table:

Order id	Order date	Required date
1111	31-May-2018	1-Jun-2018
2222	21-May-2018	22-Jun-2018

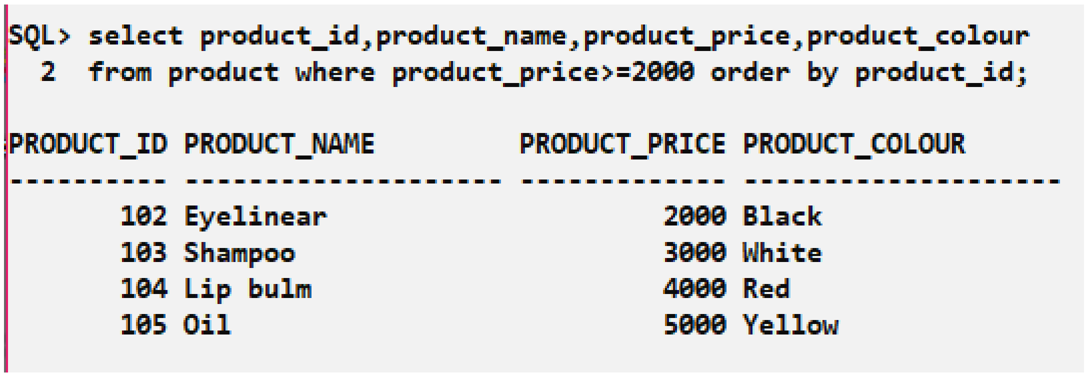
Queries on Cosmetic Shop Management System:

1.Displayproduct id,product name and product price from product table where product price is greater than 2000.



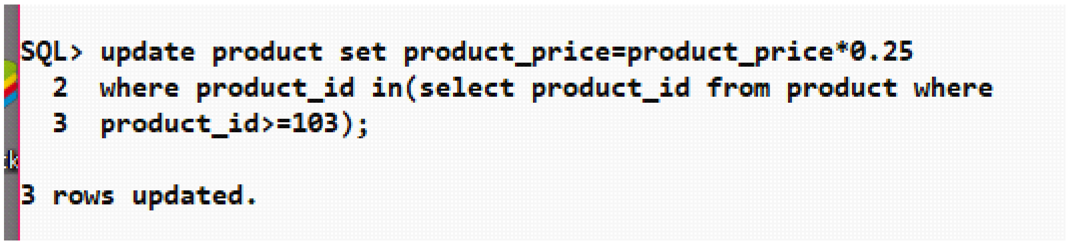
```
SQL> select product_id,product_name,product_price from product
  2  where product_id in(select product_id from product where
  3  product_price>2000);

PRODUCT_ID PRODUCT_NAME         PRODUCT_PRICE
---------- -------------------- -------------
       105 Oil                           5000
       104 Lip bulm                      4000
       103 Shampoo                       3000
```

2. Display id,name,price,color of product from product table whose price is greater than or equal to 2000 according to product id.



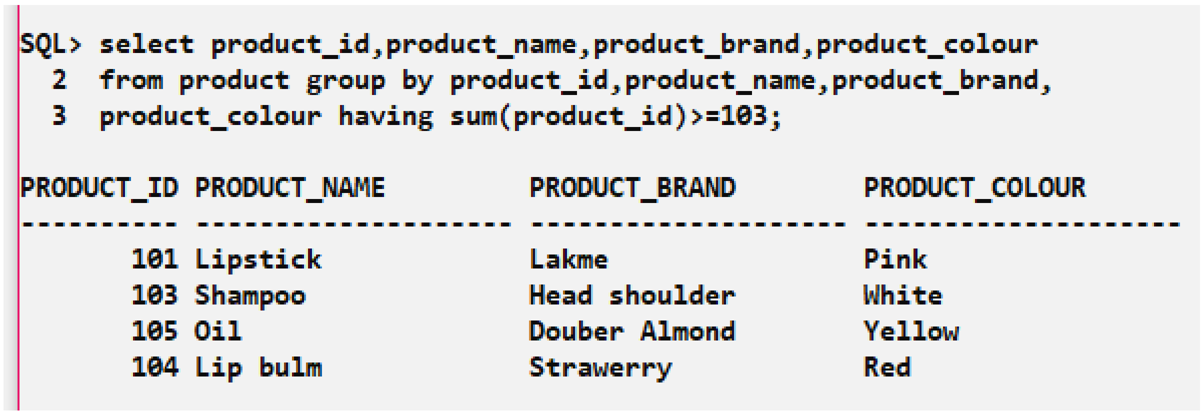
```
SQL> select product_id,product_name,product_price,product_colour
  2  from product where product_price>=2000 order by product_id;

PRODUCT_ID PRODUCT_NAME         PRODUCT_PRICE PRODUCT_COLOUR
---------- -------------------- ------------- --------------------
       102 Eyelinear                     2000 Black
       103 Shampoo                       3000 White
       104 Lip bulm                      4000 Red
       105 Oil                           5000 Yellow
```

3.Update product price from product table whose product id is greater than or equal to 103.



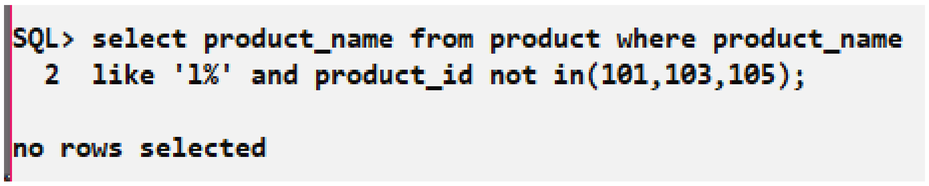
```
SQL> update product set product_price=product_price*0.25
  2  where product_id in(select product_id from product where
  3  product_id>=103);

3 rows updated.
```

4.Display id,name,colour,brand of product whose id is graeter than or equal to 103 and find sum of product id.



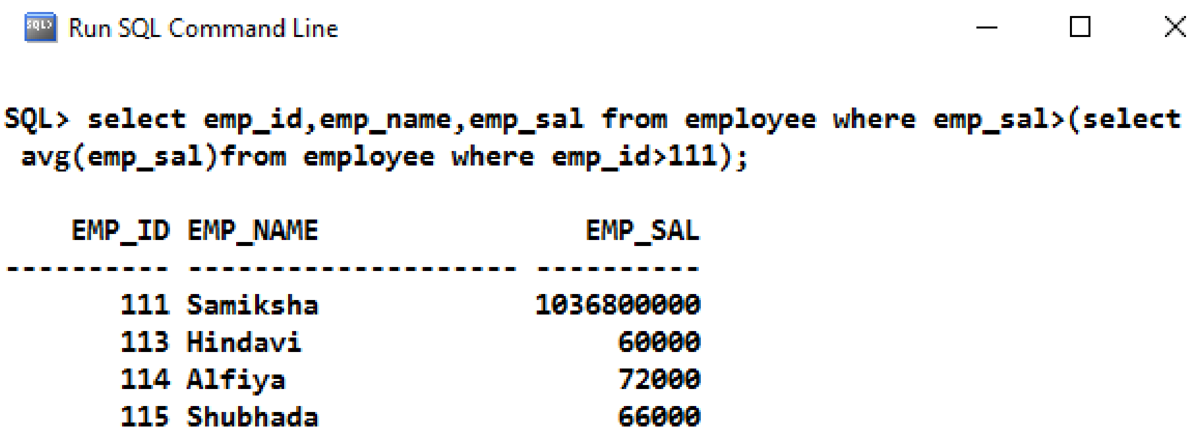
```
SQL> select product_id,product_name,product_brand,product_colour
  2  from product group by product_id,product_name,product_brand,
  3  product_colour having sum(product_id)>=103;

PRODUCT_ID PRODUCT_NAME         PRODUCT_BRAND        PRODUCT_COLOUR
---------- -------------------- -------------------- --------------------
       101 Lipstick             Lakme                Pink
       103 Shampoo              Head shoulder        White
       105 Oil                  Douber Almond        Yellow
       104 Lip bulm             Strawerry            Red
```

5)Display product name from product table whose product name start from L and product id not in 101,103 and 105.



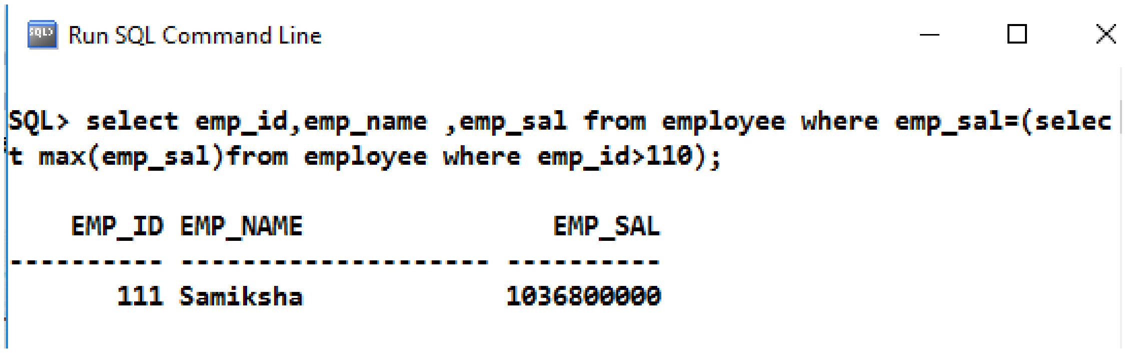
```
SQL> select product_name from product where product_name
  2  like '1%' and product_id not in(101,103,105);

no rows selected
```

5)Display id,name and salary of employee whose salary is greater than avg salary of employee.



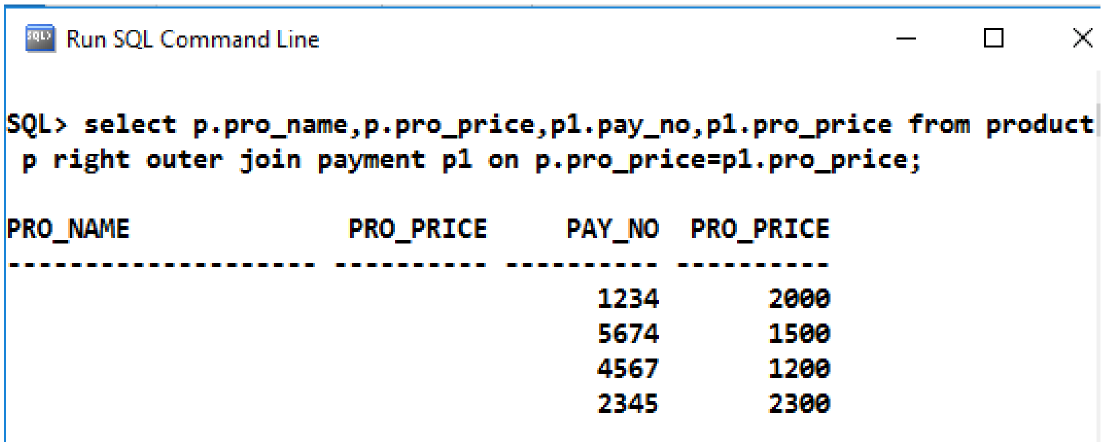
```
SQL> select emp_id,emp_name,emp_sal from employee where emp_sal>(select
 avg(emp_sal)from employee where emp_id>111);

    EMP_ID EMP_NAME                EMP_SAL
---------- -------------------- ----------
       111 Samiksha             1036800000
       113 Hindavi                   60000
       114 Alfiya                    72000
       115 Shubhada                  66000
```

6)Display emp id,emp name and emp salary of employee whose salary is maximum .



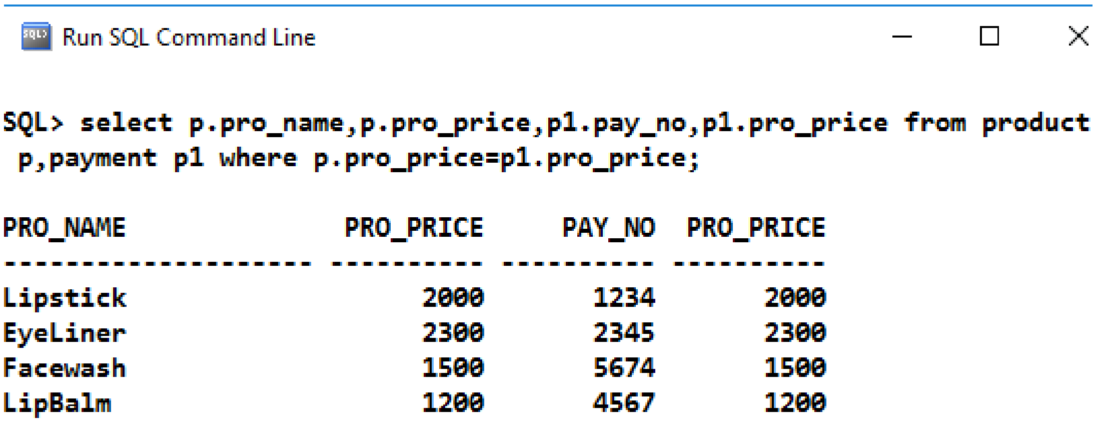
```
SQL> select emp_id,emp_name ,emp_sal from employee where emp_sal=(selec
t max(emp_sal)from employee where emp_id>110);

    EMP_ID EMP_NAME                EMP_SAL
---------- -------------------- ----------
       111 Samiksha             1036800000
```

7)Display product name,product price and payment no from two tables using left outer join.



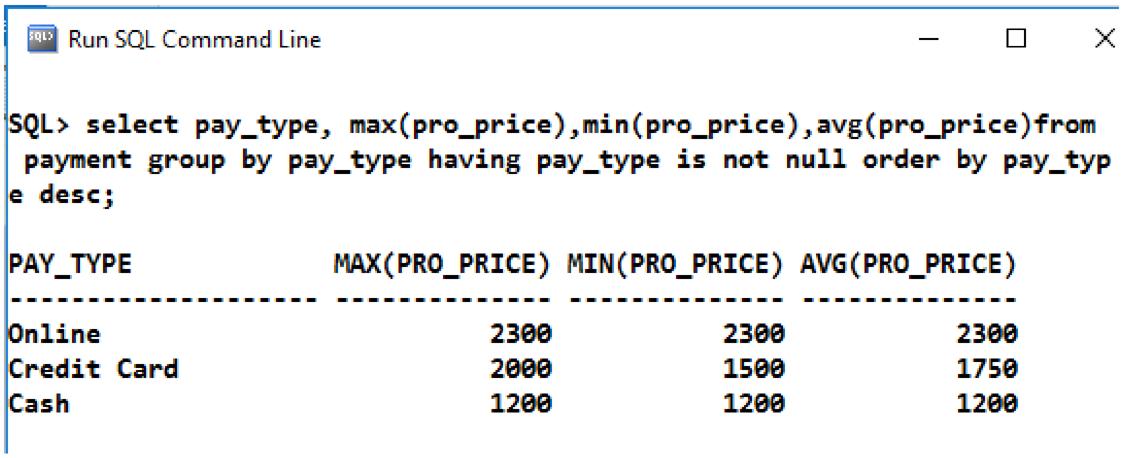
```
SQL> select p.pro_name,p.pro_price,p1.pay_no,p1.pro_price from product
 p right outer join payment p1 on p.pro_price=p1.pro_price;

PRO_NAME              PRO_PRICE     PAY_NO  PRO_PRICE
-------------------- ---------- ---------- ----------
                                      1234       2000
                                      5674       1500
                                      4567       1200
                                      2345       2300
```

8)Display product name,product name,payment no from two table if product price are same in both the table.



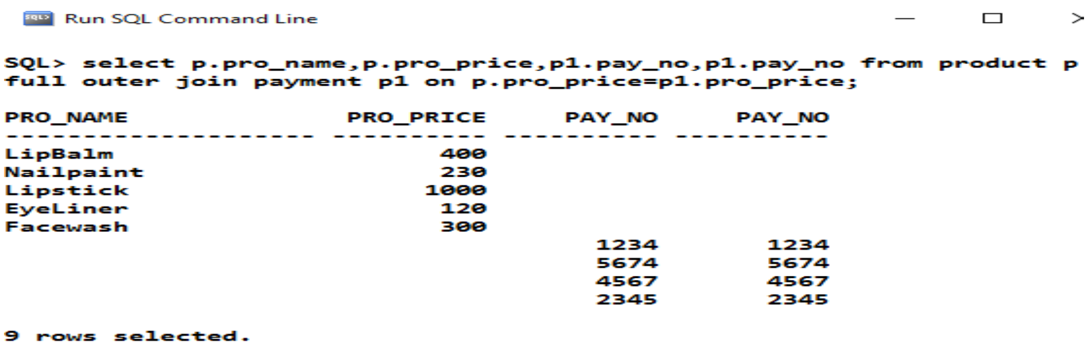
```
SQL> select p.pro_name,p.pro_price,p1.pay_no,p1.pro_price from product
 p,payment p1 where p.pro_price=p1.pro_price;

PRO_NAME              PRO_PRICE     PAY_NO  PRO_PRICE
-------------------- ---------- ---------- ----------
Lipstick                   2000       1234       2000
EyeLiner                   2300       2345       2300
Facewash                   1500       5674       1500
LipBalm                    1200       4567       1200
```

9)Display maximum,minimum and average salary from payment table where payment type is not null and display prices from descending order of payment type.



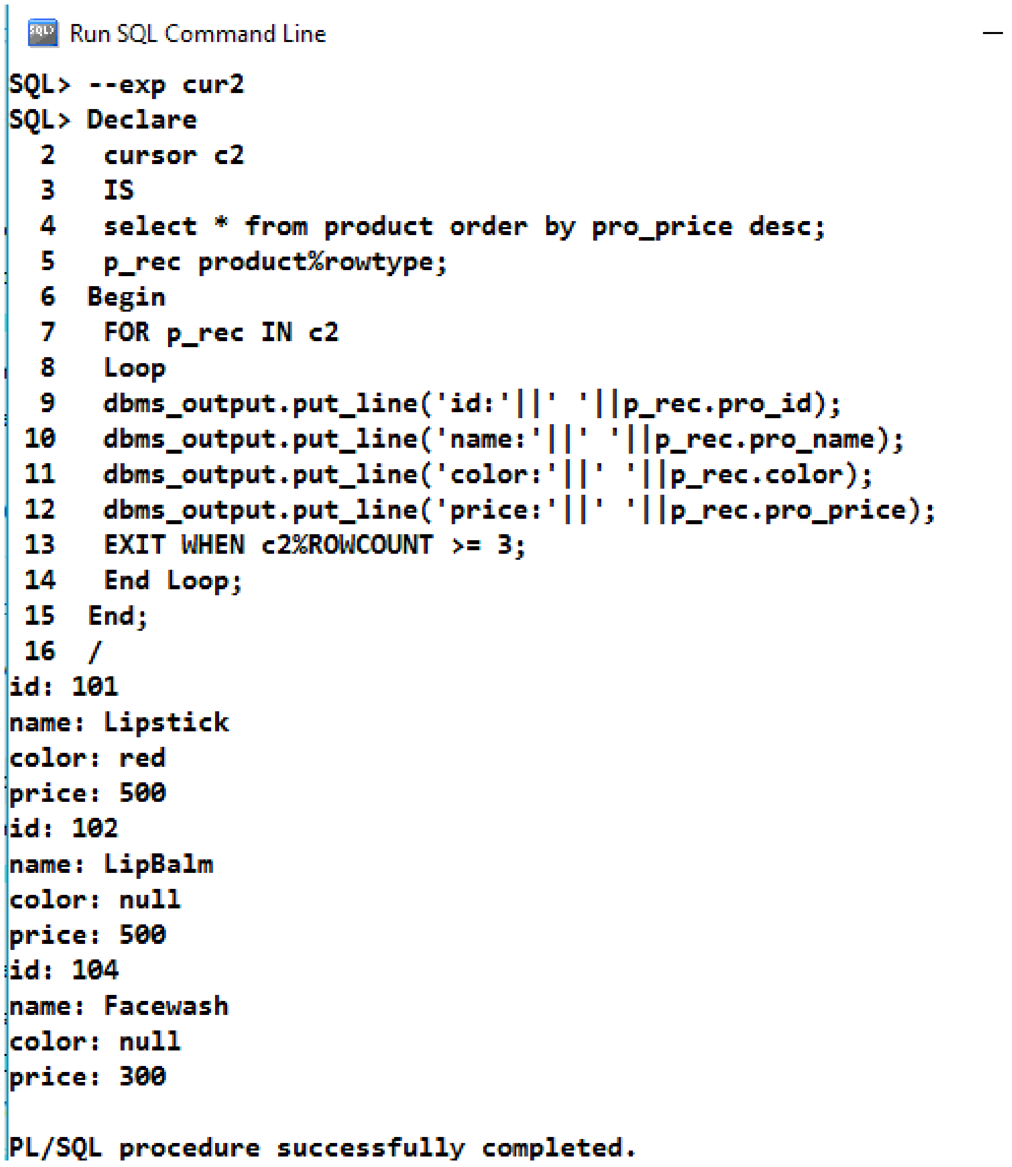
```
SQL> select pay_type, max(pro_price),min(pro_price),avg(pro_price)from
 payment group by pay_type having pay_type is not null order by pay_typ
e desc;

PAY_TYPE             MAX(PRO_PRICE) MIN(PRO_PRICE) AVG(PRO_PRICE)
-------------------- -------------- -------------- --------------
Online                         2300           2300           2300
Credit Card                    2000           1500           1750
Cash                           1200           1200           1200
```

10)Display product name ,product price and payment table from payment and product table whose prices are same.



```
SQL> select p.pro_name,p.pro_price,p1.pay_no,p1.pay_no from product p
full outer join payment p1 on p.pro_price=p1.pro_price;

PRO_NAME              PRO_PRICE     PAY_NO     PAY_NO
-------------------- ---------- ---------- ----------
LipBalm                     400
Nailpaint                   230
Lipstick                   1000
EyeLiner                    120
Facewash                    300
                                      1234       1234
                                      5674       5674
                                      4567       4567
                                      2345       2345

9 rows selected.
```

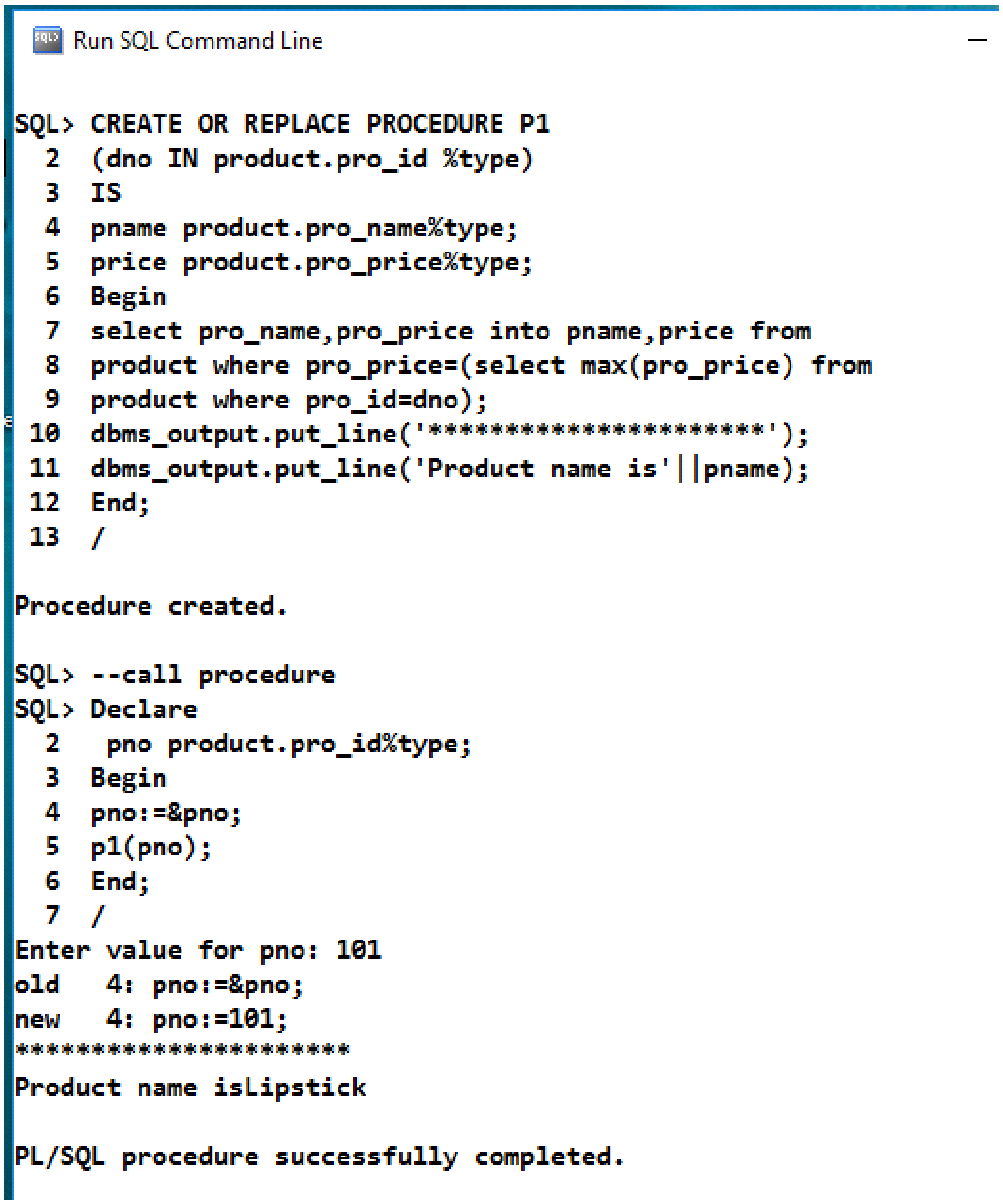
- **Cursor:**
- **Write a program to display details of highest 3 price paid products:**



```
SQL> --exp cur2
SQL> Declare
  cursor c2
  IS
  select * from product order by pro_price desc;
  p_rec product%rowtype;
 Begin
  FOR p_rec IN c2
  Loop
  dbms_output.put_line('id:'||' '||p_rec.pro_id);
  dbms_output.put_line('name:'||' '||p_rec.pro_name);
  dbms_output.put_line('color:'||' '||p_rec.color);
  dbms_output.put_line('price:'||' '||p_rec.pro_price);
  EXIT WHEN c2%ROWCOUNT >= 3;
  End Loop;
 End;
 /
id: 101
name: Lipstick
color: red
price: 500
id: 102
name: LipBalm
color: null
price: 500
id: 104
name: Facewash
color: null
price: 300

PL/SQL procedure successfully completed.
```

Procedure:

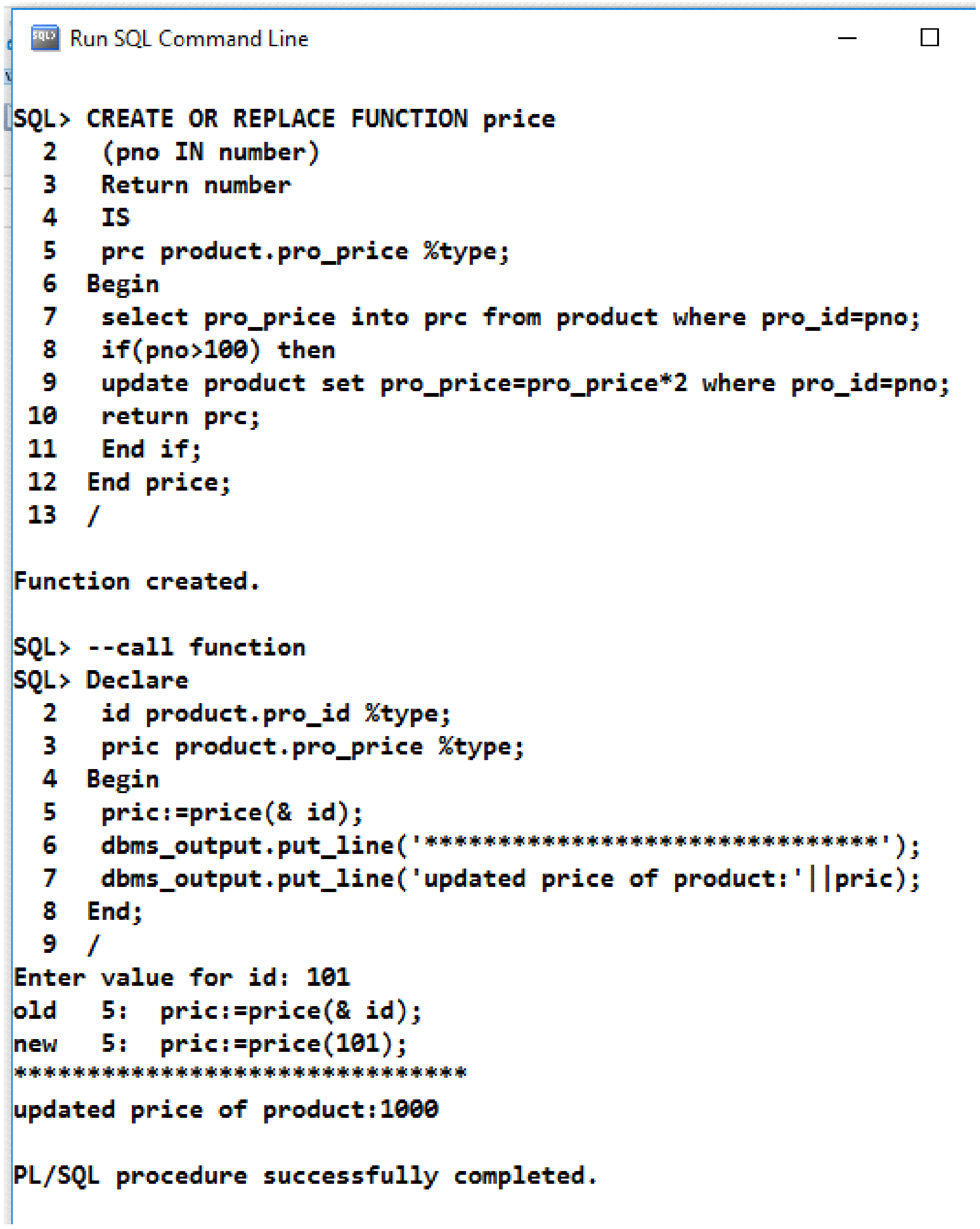
Pass product id to procedure.The procedure will return name and price of product whose price is maximum.


Run SQL Command Line

```
SQL> CREATE OR REPLACE PROCEDURE P1
  2  (dno IN product.pro_id %type)
  3  IS
  4  pname product.pro_name%type;
  5  price product.pro_price%type;
  6  Begin
  7  select pro_name,pro_price into pname,price from
  8  product where pro_price=(select max(pro_price) from
  9  product where pro_id=dno);
 10  dbms_output.put_line('**********************');
 11  dbms_output.put_line('Product name is'||pname);
 12  End;
 13  /

Procedure created.

SQL> --call procedure
SQL> Declare
  2   pno product.pro_id%type;
  3  Begin
  4  pno:=&pno;
  5  p1(pno);
  6  End;
  7  /
Enter value for pno: 101
old   4: pno:=&pno;
new   4: pno:=101;
**********************
Product name isLipstick

PL/SQL procedure successfully completed.
```

- **Function**

  Pass product id to a function and check product number and display updated price of product.

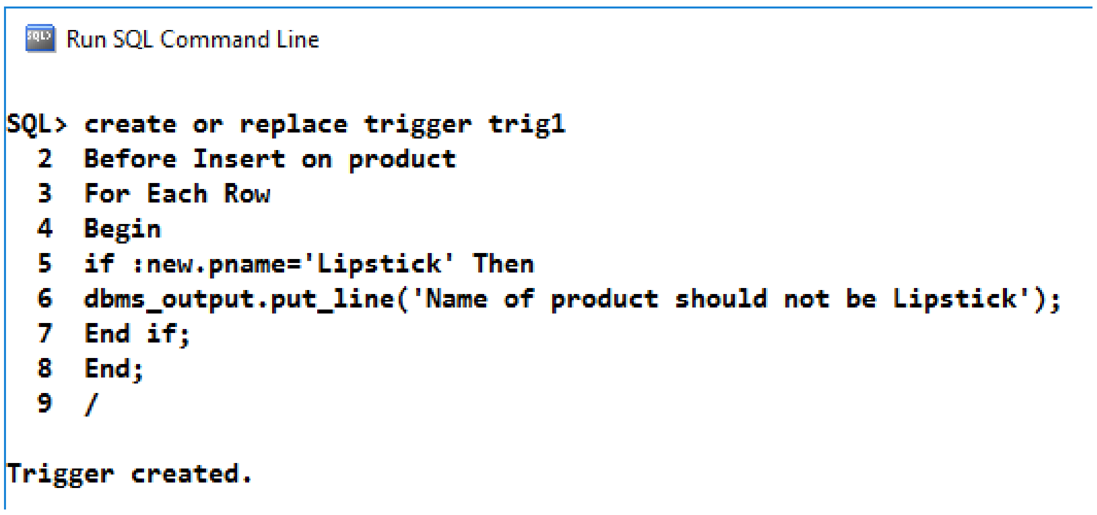


Run SQL Command Line

```
SQL> CREATE OR REPLACE FUNCTION price
  2   (pno IN number)
  3   Return number
  4   IS
  5   prc product.pro_price %type;
  6  Begin
  7   select pro_price into prc from product where pro_id=pno;
  8   if(pno>100) then
  9   update product set pro_price=pro_price*2 where pro_id=pno;
 10   return prc;
 11   End if;
 12  End price;
 13  /

Function created.

SQL> --call function
SQL> Declare
  2   id product.pro_id %type;
  3   pric product.pro_price %type;
  4  Begin
  5   pric:=price(& id);
  6   dbms_output.put_line('*******************************');
  7   dbms_output.put_line('updated price of product:'||pric);
  8  End;
  9  /
Enter value for id: 101
old   5:  pric:=price(& id);
new   5:  pric:=price(101);
*******************************
updated price of product:1000

PL/SQL procedure successfully completed.
```

- **Trigger**

- **Create trigger on product table .**



```
SQL> create or replace trigger trig1
  2  Before Insert on product
  3  For Each Row
  4  Begin
  5  if :new.pname='Lipstick' Then
  6  dbms_output.put_line('Name of product should not be Lipstick');
  7  End if;
  8  End;
  9  /

Trigger created.
```