DBMS Assignment 8

1. Create a Function which returns the seller's name with the highest rating.

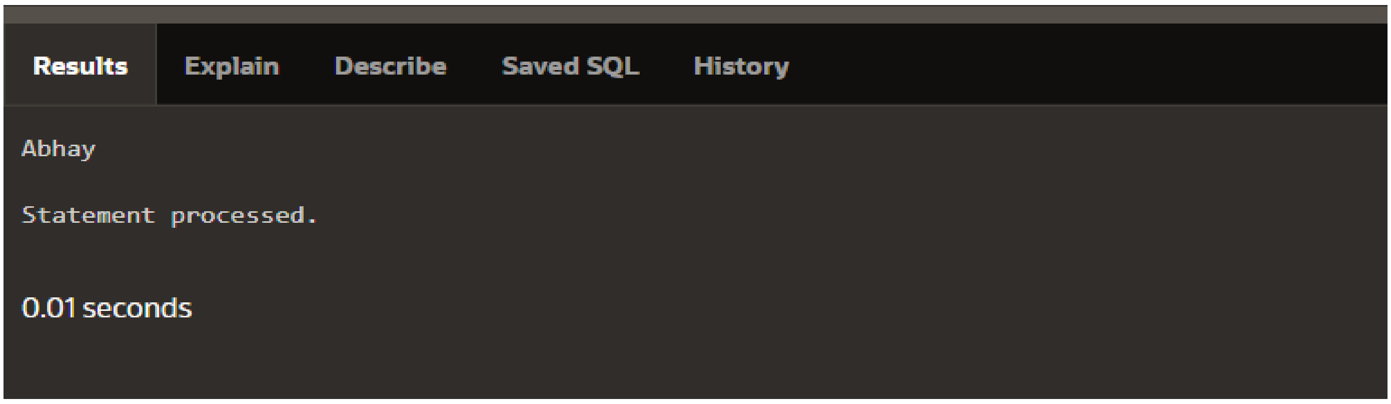
Script:

```
CREATE OR REPLACE FUNCTION SELLER_MAX_RATING
RETURN seller.Sellername%TYPE
IS
maxrating seller.Sellername%TYPE;
BEGIN
    SELECT Sellername INTO maxrating FROM Seller WHERE Rating=(SELECT MAX(Rating) FROM Seller);
RETURN maxrating;
END;
```

Command:

```
DECLARE
 ans  Seller.Sellername%TYPE;
BEGIN
 ans:=seller_max_rating;
dbms_output.put_line(ans);
END;
```

Output:



```
Abhay

Statement processed.

0.01 seconds
```

2. Create Stored procedure which takes as an input 'category' and outputs all the products of that category.

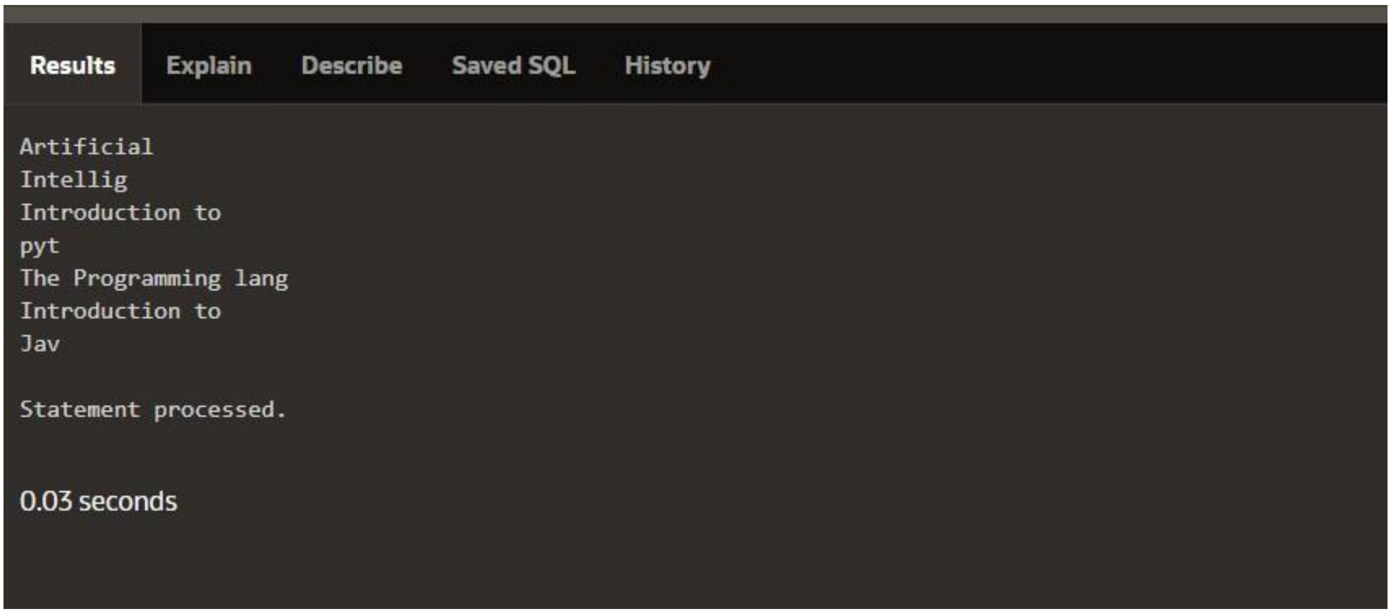
Script:

```
CREATE OR REPLACE PROCEDURE prod_of_cat(cat IN category.category%type)
is
 pro product.product%type;
 CURSOR c_product IS
 SELECT product FROM product WHERE Categoryid=(SELECT Categoryid FROM Category Where Category
=cat);
BEGIN
 Open c_product;
 LOOP
 FETCH c_product INTO pro;
 EXIT WHEN c_product%NOTFOUND;
 dbms_output.put_line(pro); END
 LOOP;
 CLOSE c_product;
 end;
```

Command:

```
BEGIN
 prod_of_cat('Books');
END;
```

Output:



```
Artificial
Intellig
Introduction to
pyt
The Programming lang
Introduction to
Jav

Statement processed.


0.03 seconds
```

3. Create Stored procedure to take a range of prices as input and output all the products in the provided range.

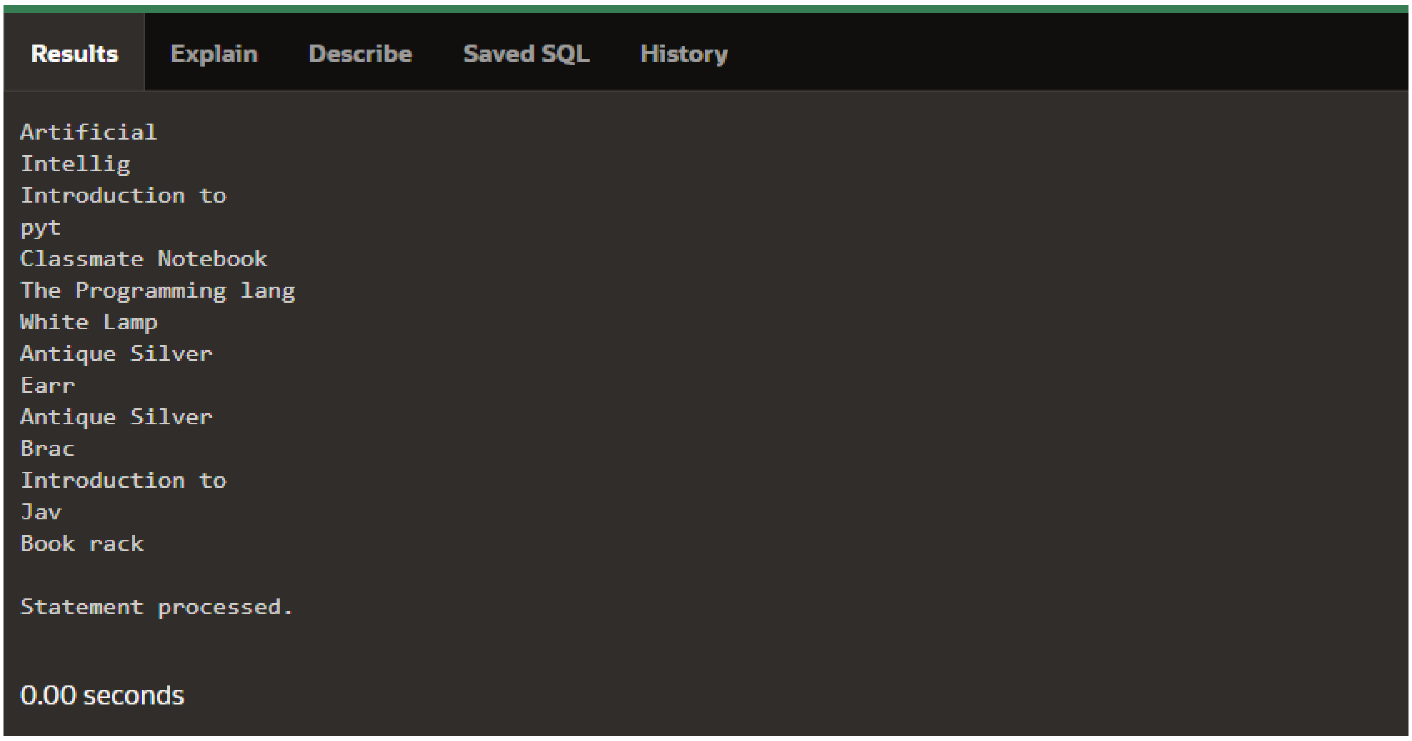
Script:

```
CREATE OR REPLACE PROCEDURE range(ll in product.amount%type, ul in product.amount%type)
is
 prod product.product%type;
 cursor c_product is
 SELECT PRODUCT FROM PRODUCT WHERE AMOUNT BETWEEN ll AND ul;
BEGIN
 open c_product;
 LOOP
 FETCH c_product INTO prod;
 EXIT WHEN c_product%NOTFOUND;
 dbms_output.put_line(prod); END
 LOOP;
 close c_product;
end;
```

Command:

```
BEGIN
 range(100,1000);
END;
```

Output:



```
Artificial
Intellig
Introduction to
pyt
Classmate Notebook
The Programming lang
White Lamp
Antique Silver
Earr
Antique Silver
Brac
Introduction to
Jav
Book rack

Statement processed.


0.00 seconds
```

4. Create function to display all the seller details with rating more than 3.

Script:

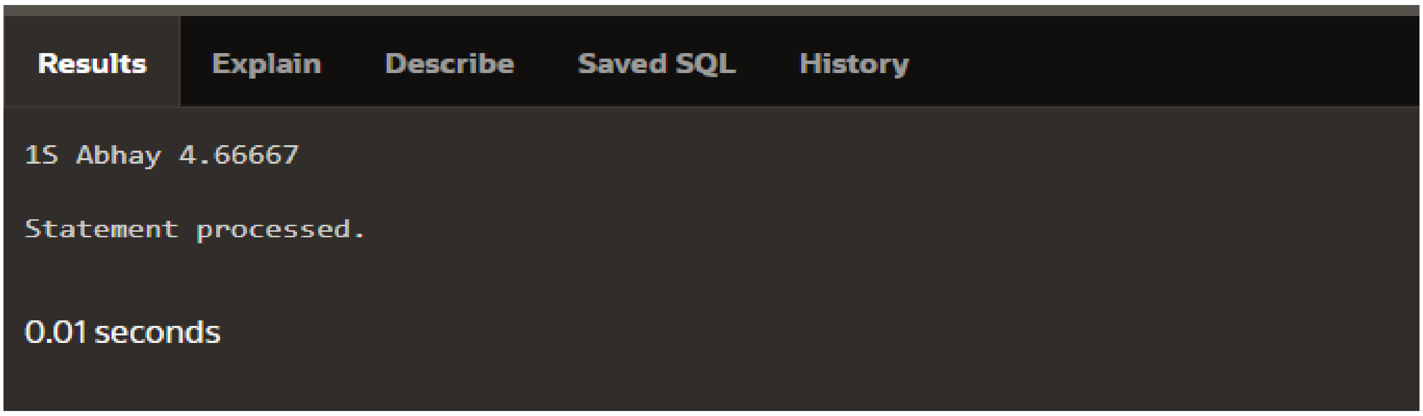
```
CREATE OR REPLACE FUNCTION getsellerdeatils
RETURN SYS_REFCURSOR
IS
```

```
 s_details SYS_REFCURSOR;
BEGIN
 OPEN s_details FOR
 SELECT DISTINCT Sellerid, Sellername, Rating FROM Seller WHERE Rating>3;
 RETURN s_details;
END;
```

Command:

```
DECLARE
 s_details SYS_REFCURSOR;
 s_id SELLER.SELLERID%type;
 s_name SELLER.SELLERNAME%type;
s_rating SELLER.RATING%type;
BEGIN
 s_details:=getsellerdeatils;
 LOOP
 FETCH s_details INTO s_id, s_name, s_rating;
 EXIT WHEN s_details%NOTFOUND;
 dbms_output.put_line(s_id || ' ' || s_name || ' ' || s_rating);
 END LOOP;
END;
```

Output:



```
1S Abhay 4.66667

Statement processed.

0.01 seconds
```

5. Create a function to display all the products, seller wise.

Script:

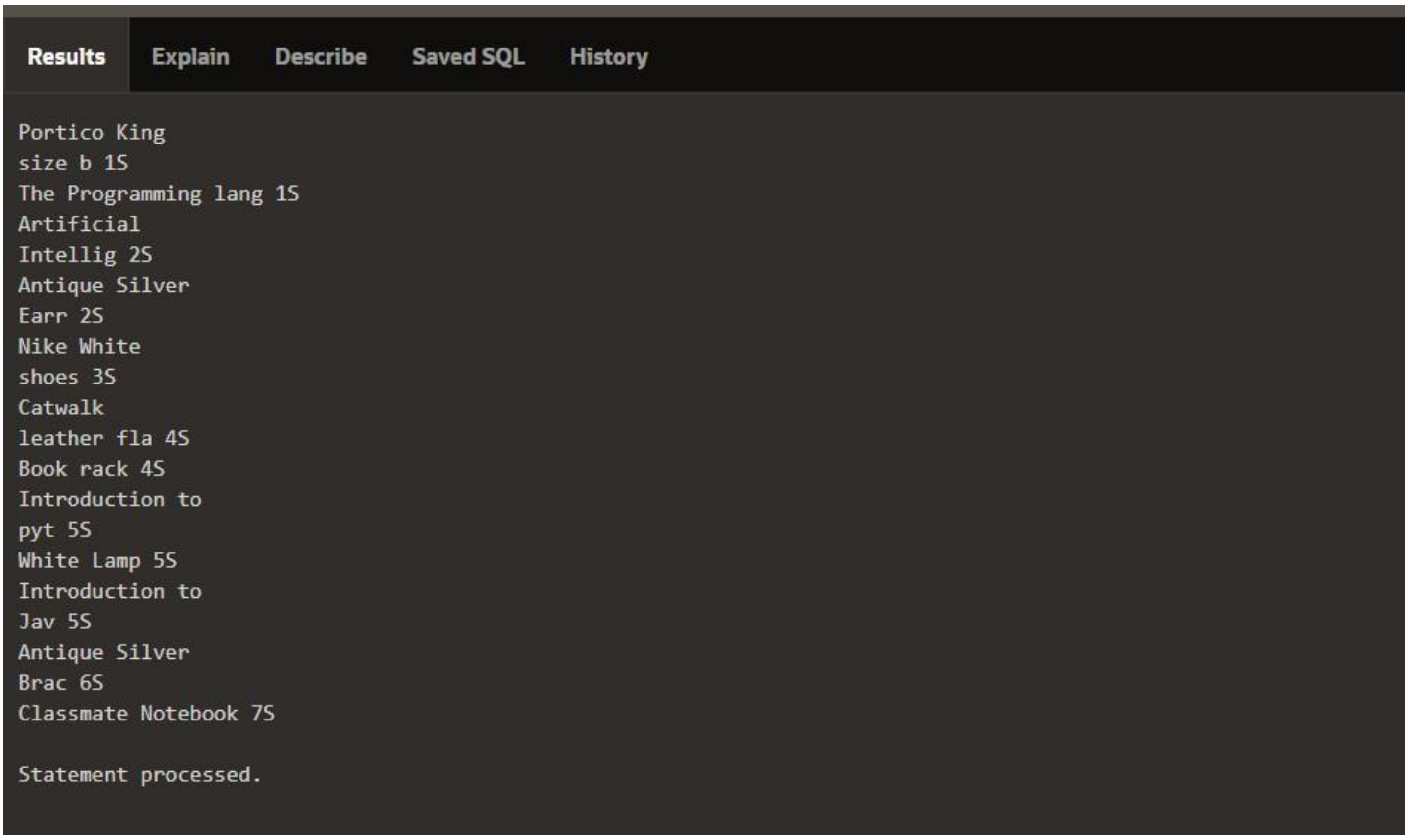
```
CREATE OR REPLACE FUNCTION disp_product_seller
RETURN SYS_REFCURSOR
IS
prods SYS_REFCURSOR;
BEGIN
OPEN prods FOR SELECT PRODUCT,SELLERID FROM PRODUCT SELLER ORDER BY SELLERID;
RETURN prods;
END;
```

Command:

```
DECLARE
```

```
details SYS_REFCURSOR;
s_id seller.sellerid%type;
product seller.Sellername%type;
BEGIN
details:=disp_product_seller;
LOOP
FETCH details INTO s_id,product;
EXIT WHEN details%notfound;
dbms_output.put_line(s_id || ' ' ||product);
END LOOP;
END;
```

Output:



```
Portico King
size b 1S
The Programming lang 1S
Artificial
Intellig 2S
Antique Silver
Earr 2S
Nike White
shoes 3S
Catwalk
leather fla 4S
Book rack 4S
Introduction to
pyt 5S
White Lamp 5S
Introduction to
Jav 5S
Antique Silver
Brac 6S
Classmate Notebook 7S

Statement processed.
```

6. Create a Stored procedure which checks all the entries in Order_Products table and update seller and product table accordingly.

Script:

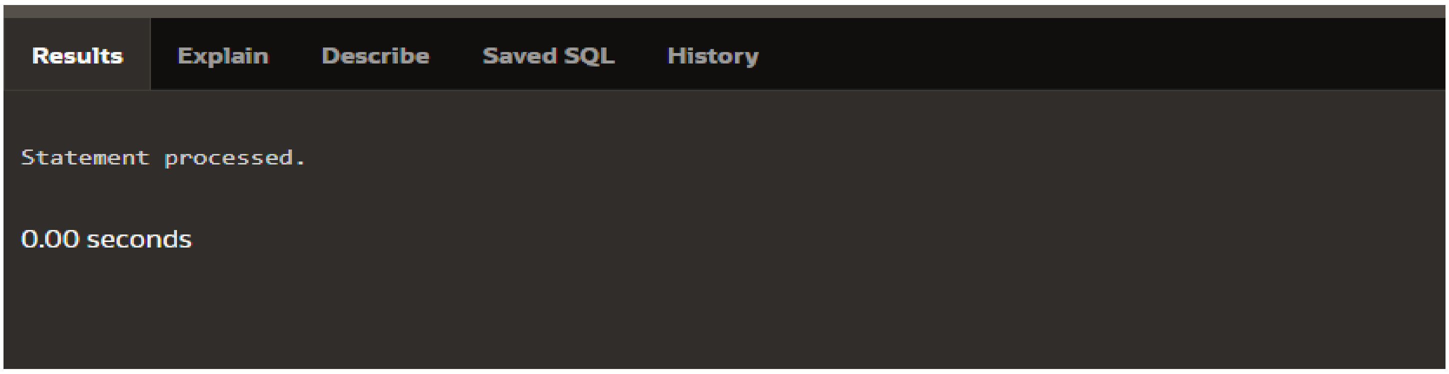
```
CREATE OR REPLACE PROCEDURE update_product_seller
AS
BEGIN
 UPDATE product p SET p.rating = (SELECT AVG(rating) FROM order_products GROUP BY productid H
AVING productid = p.productid);
```

```
 UPDATE seller s SET s.rating = (SELECT AVG(rating) FROM order_products GROUP BY sellerid HAV
ING sellerid = s.sellerid);
END;
```

Command:

```
BEGIN
 update_product_seller;
END;
```

Output:



```
Statement processed.

0.00 seconds
```

7. Create Stored procedure which takes as input different filters such as price range, category, product rating, seller rating, out of stock and displays the list of products with all the details after applying filters.

Script:

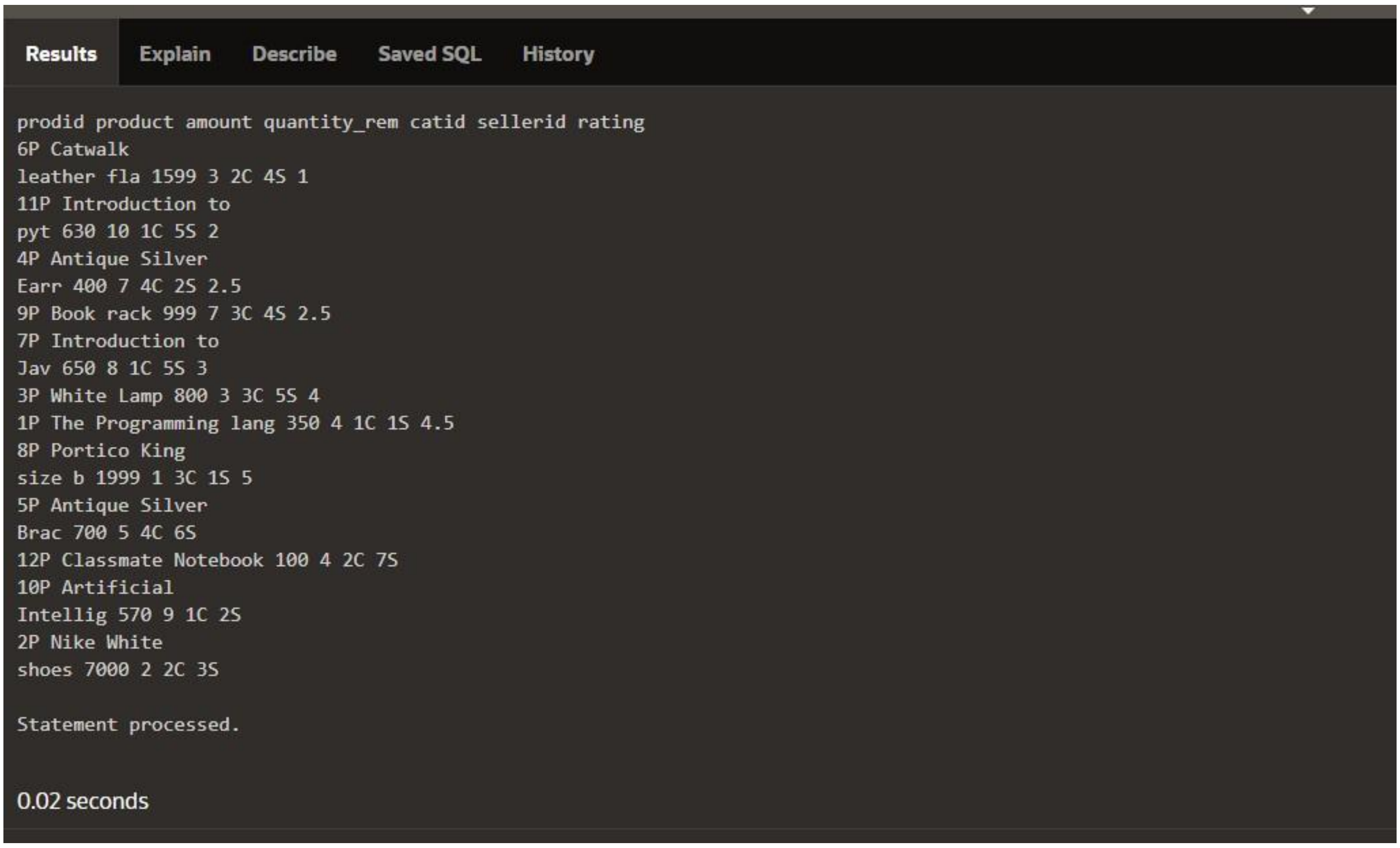
```
CREATE OR REPLACE PROCEDURE filter_criteria(opt IN NUMBER)
IS
 prod_details SYS_REFCURSOR;
 prod_prodid PRODUCT.PRODUCTID%type;
 prod_name PRODUCT.PRODUCT%type;
 prod_amt PRODUCT.AMOUNT%type;
 prod_quant PRODUCT.QUANTITYREMAINING%type;
 prod_catid PRODUCT.CATEGORYID%type;
 prod_sellerid PRODUCT.SELLERID%type;
 prod_rating PRODUCT.RATING%type;
BEGIN
 CASE opt
 WHEN 1 THEN OPEN prod_details FOR SELECT PRODUCTID, PRODUCT, AMOUNT,QUANTITYREMAINING, CATEG
ORYID, SELLERID, RATING FROM PRODUCT ORDER BY AMOUNT;
 WHEN 2 THEN OPEN prod_details FOR SELECT PRODUCTID, PRODUCT, AMOUNT,QUANTITYREMAINING, CATEG
ORYID, SELLERID, RATING FROM PRODUCT ORDER BY CATEGORYID;
 WHEN 3 THEN OPEN prod_details FOR SELECT PRODUCTID, PRODUCT, AMOUNT,QUANTITYREMAINING, CATEG
ORYID, SELLERID, RATING FROM PRODUCT ORDER BY RATING;
```

```
 WHEN 4 THEN OPEN prod_details FOR SELECT P.PRODUCTID, P.PRODUCT, P.AMOUNT, P.QUANTITYREMAINI
NG, P.CATEGORYID, P.SELLERID, P.RATING FROM PRODUCT P,
 SELLER S WHERE P.SELLERID=S.SELLERID ORDER BY S.RATING;
 WHEN 5 THEN OPEN prod_details FOR SELECT PRODUCTID, PRODUCT, AMOUNT,QUANTITYREMAINING, CATEG
ORYID, SELLERID, RATING FROM PRODUCT ORDER BY QUANTITYREMAINING;
 END CASE;
 LOOP
 FETCH prod_details INTO prod_prodid, prod_name, prod_amt, prod_quant,prod_catid, prod_seller
id, prod_rating;
 EXIT WHEN prod_details%NOTFOUND;
 dbms_output.put_line(prod_prodid || ' ' || prod_name || ' ' || prod_amt || ' ' || prod_quan
t || ' ' || prod_catid || ' ' || prod_sellerid || ' ' || prod_rating);
 END LOOP;
END;
```

Command:

```
BEGIN
 dbms_output.put_line( 'prodid' || ' ' || 'product' || ' ' || 'amount' || ' ' || 'quantity_re
m' || ' ' || 'catid' || ' ' || 'sellerid' || ' ' || 'rating');
 --sorting: 1 for amount wise, 2 for category wise, 3 for productrating wise, 4 for seller-
rating wise, 5 for quantity-wise
 filter_criteria(3);
END;
```

Output:



```
prodid product amount quantity_rem catid sellerid rating
6P Catwalk
leather fla 1599 3 2C 4S 1
11P Introduction to
pyt 630 10 1C 5S 2
4P Antique Silver
Earr 400 7 4C 2S 2.5
9P Book rack 999 7 3C 4S 2.5
7P Introduction to
Jav 650 8 1C 5S 3
3P White Lamp 800 3 3C 5S 4
1P The Programming lang 350 4 1C 1S 4.5
8P Portico King
size b 1999 1 3C 1S 5
5P Antique Silver
Brac 700 5 4C 6S
12P Classmate Notebook 100 4 2C 7S
10P Artificial
Intellig 570 9 1C 2S
2P Nike White
shoes 7000 2 2C 3S

Statement processed.

0.02 seconds
```

8. Create a function which takes as input sorting criteria like popularity or lowest price or highest price and display the product list accordingly.

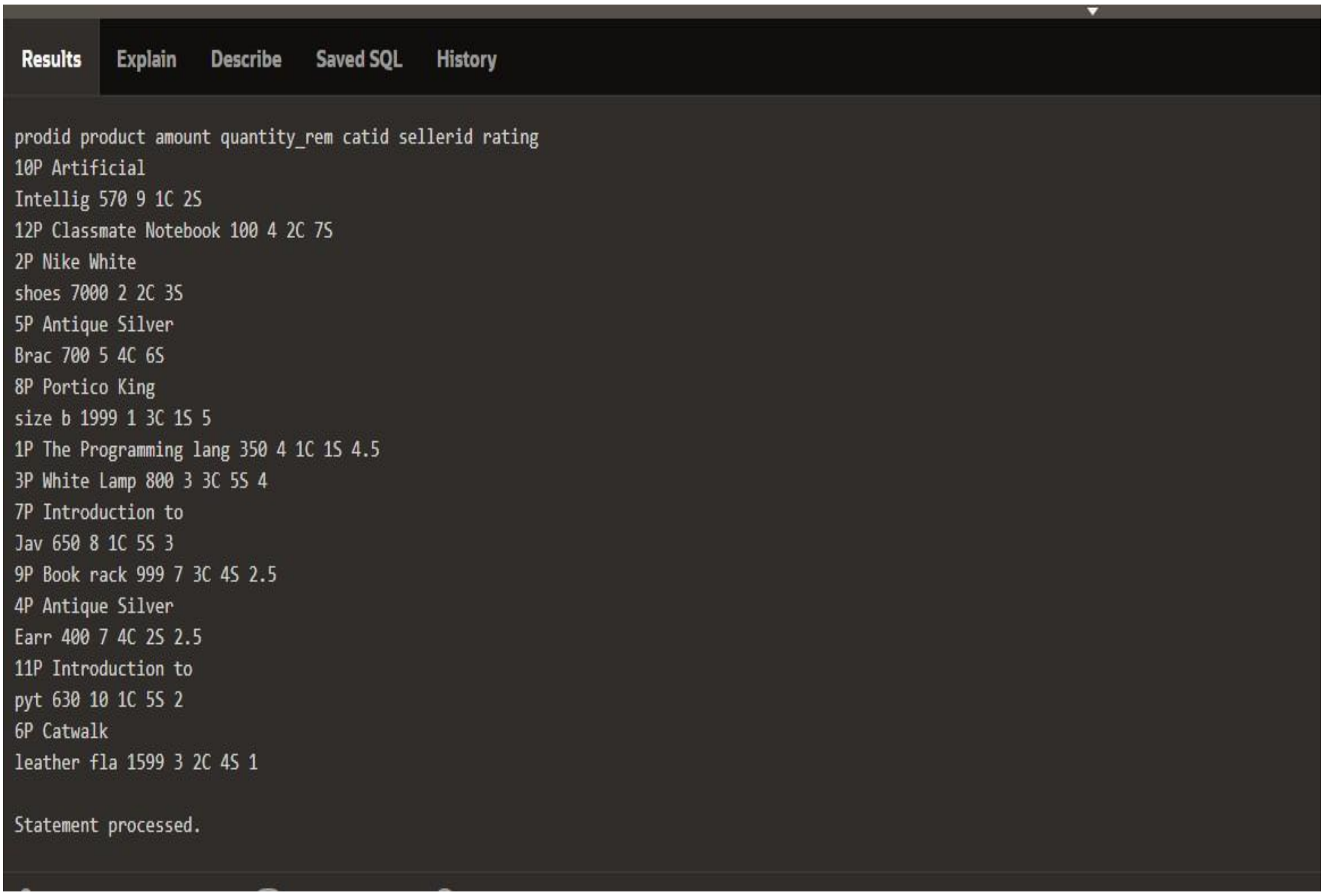
Script:

```
CREATE OR REPLACE FUNCTION sort_criteria(opt IN number)
RETURN SYS_REFCURSOR
IS
 prod_details SYS_REFCURSOR;
BEGIN
 CASE opt
 WHEN 1 THEN OPEN prod_details FOR SELECT PRODUCTID, PRODUCT, AMOUNT, QUANTITYREMAINING, CATE
GORYID, SELLERID, RATING FROM PRODUCT ORDER BY AMOUNT;
 WHEN 2 THEN OPEN prod_details FOR SELECT PRODUCTID, PRODUCT, AMOUNT, QUANTITYREMAINING, CATE
GORYID, SELLERID, RATING FROM PRODUCT ORDER BY AMOUNT DESC;
 WHEN 3 THEN OPEN prod_details FOR SELECT PRODUCTID, PRODUCT, AMOUNT, QUANTITYREMAINING, CATE
GORYID, SELLERID, RATING FROM PRODUCT ORDER BY RATING DESC;
 END CASE;
 RETURN prod_details;
END;
```

Command:

```
DECLARE
 prod_details SYS_REFCURSOR;
 prod_prodid PRODUCT.PRODUCTID%type;
 prod_name PRODUCT.PRODUCT%type;
 prod_amt PRODUCT.AMOUNT%type;
 prod_quant PRODUCT.QUANTITYREMAINING%type;
 prod_catid PRODUCT.CATEGORYID%type;
 prod_sellerid PRODUCT.SELLERID%type;
 prod_rating PRODUCT.RATING%type;
BEGIN
 dbms_output.put_line( 'prodid' || ' ' || 'product' || ' ' || 'amount' || ' ' || 'quantity_re
m' || ' ' || 'catid' || ' ' || 'sellerid' || ' ' || 'rating');
 --criteria for sorting: 1 for amount ascending, 2 for amount descnding, 3 for rating wise
 prod_details:=sort_criteria(3);
 LOOP
 FETCH prod_details INTO prod_prodid, prod_name, prod_amt, prod_quant,prod_catid, prod_seller
id, prod_rating;
 EXIT WHEN prod_details%NOTFOUND;
 dbms_output.put_line( prod_prodid || ' ' || prod_name || ' ' || prod_amt || ' ' || prod_quan
t || ' ' || prod_catid || ' ' || prod_sellerid || ' ' || prod_rating);
 END LOOP;
END;
```

Output:



```
prodid product amount quantity_rem catid sellerid rating
10P Artificial
Intellig 570 9 1C 2S
12P Classmate Notebook 100 4 2C 7S
2P Nike White
shoes 7000 2 2C 3S
5P Antique Silver
Brac 700 5 4C 6S
8P Portico King
size b 1999 1 3C 1S 5
1P The Programming lang 350 4 1C 1S 4.5
3P White Lamp 800 3 3C 5S 4
7P Introduction to
Jav 650 8 1C 5S 3
9P Book rack 999 7 3C 4S 2.5
4P Antique Silver
Earr 400 7 4C 2S 2.5
11P Introduction to
pyt 630 10 1C 5S 2
6P Catwalk
leather fla 1599 3 2C 4S 1

Statement processed.
```