DBMS ASSIGNMENT - 9

Cursors and Triggers

Name: BHAGYA VINOD RANA **Roll Number: U19CS012**

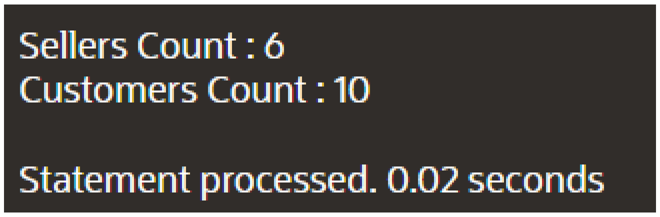
(A) Cursors:

1. Create a cursor to fetch the count of customers and sellers.

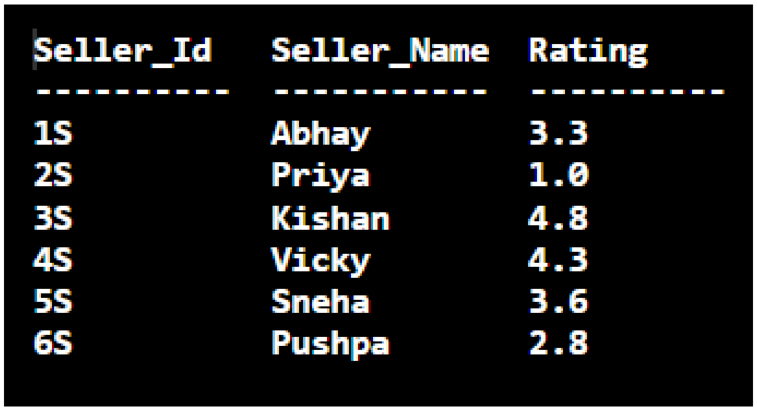
Cursor:

```
DECLARE

-- Variables to Hold Data
s_id SELLER.SELLER_ID%TYPE;
cus_id CUSTOMER.CUSTOMER_ID%TYPE;

-- CURSOR to Count the Number of Sellers
CURSOR seller_cnt IS
SELECT
    DISTINCT SELLER_ID
FROM
    SELLER;

-- CURSOR to Count the Number of Customers
CURSOR customer_cnt IS
SELECT
    DISTINCT CUSTOMER_ID
FROM
    CUSTOMER;
BEGIN

OPEN seller_cnt;
    LOOP FETCH seller_cnt INTO s_id;
    EXIT WHEN seller_cnt%notfound;
    END LOOP;
    -- Print the Seller Count after Iterating Whole SELLER Table
    dbms_output.put_line('Sellers Count : ' || seller_cnt%rowcount);
CLOSE seller_cnt;

OPEN customer_cnt;
    LOOP FETCH customer_cnt INTO cus_id;
    EXIT WHEN customer_cnt%notfound;
    END LOOP;
     -- Print the Customer Count after Iterating Whole SELLER Table
    dbms_output.put_line('Customers Count : ' || customer_cnt%rowcount);
CLOSE customer_cnt;
END;/
```

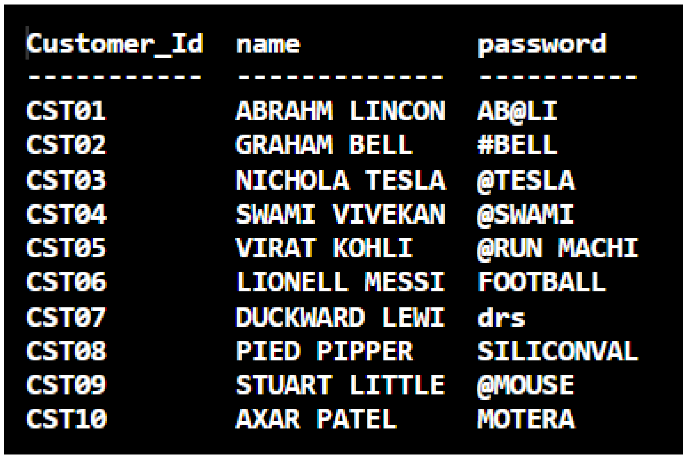
Output:



```
Sellers Count : 6
Customers Count : 10

Statement processed. 0.02 seconds
```



```
Seller_Id    Seller_Name  Rating
----------   -----------  ----------
1S           Abhay        3.3
2S           Priya        1.0
3S           Kishan       4.8
4S           Vicky        4.3
5S           Sneha        3.6
6S           Pushpa       2.8
```



```
Customer_Id  name          password
-----------  ------------  ----------
CST01        ABRAHM LINCON AB@LI
CST02        GRAHAM BELL   #BELL
CST03        NICHOLA TESLA @TESLA
CST04        SWAMI VIVEKAN @SWAMI
CST05        VIRAT KOHLI   @RUN MACHI
CST06        LIONELL MESSI FOOTBALL
CST07        DUCKWARD LEWI drs
CST08        PIED PIPPER   SILICONVAL
CST09        STUART LITTLE @MOUSE
CST10        AXAR PATEL    MOTERA
```

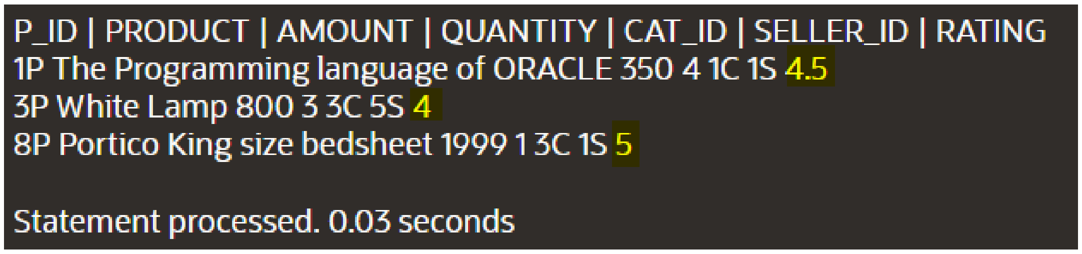
2. Create a cursor to display all the product details with rating more than 3.5.

Cursor:

```
DECLARE

-- Variables to Hold Data
prod_prodid PRODUCT.PRODUCT_ID%TYPE;
prod_name PRODUCT.PRODUCT%TYPE;
prod_amt PRODUCT.AMOUNT%TYPE;
prod_quant PRODUCT.QUANTITY_REM%TYPE;
prod_catid PRODUCT.CATEGORY_ID%TYPE;
prod_sellerid PRODUCT.SELLER_ID%TYPE;
prod_rating PRODUCT.RATING%TYPE;

-- CURSOR
CURSOR prod_details IS
SELECT PRODUCT_ID,PRODUCT,AMOUNT,QUANTITY_REM,CATEGORY_ID,SELLER_ID,RATING
FROM
    PRODUCT
WHERE
    RATING IS NOT NULL AND RATING>3.5;

BEGIN
    dbms_output.put_line( 'P_ID' || ' | ' || 'PRODUCT' || ' | ' || 'AMOUNT' || ' | ' || 'QUAN
TITY' || ' | ' || 'CAT_ID' || ' | ' || 'SELLER_ID' || ' | ' || 'RATING');
```

```
OPEN prod_details;
    -- Loop to Print the Output
    LOOP FETCH prod_details INTO prod_prodid, prod_name, prod_amt, prod_quant, prod_catid, prod_sellerid, prod_rating;
    EXIT WHEN prod_details%NOTFOUND;
    dbms_output.put_line( prod_prodid || ' ' || prod_name || ' ' || prod_amt || ' ' || prod_quant || ' ' || prod_catid || ' ' || prod_sellerid || ' ' || prod_rating);
    END LOOP;
CLOSE prod_details;

END;/
```

<u>Output:</u>



```
P_ID | PRODUCT | AMOUNT | QUANTITY | CAT_ID | SELLER_ID | RATING
1P The Programming language of ORACLE 350 4 1C 1S 4.5
3P White Lamp 800 3 3C 5S 4
8P Portico King size bedsheet 1999 1 3C 1S 5

Statement processed. 0.03 seconds
```

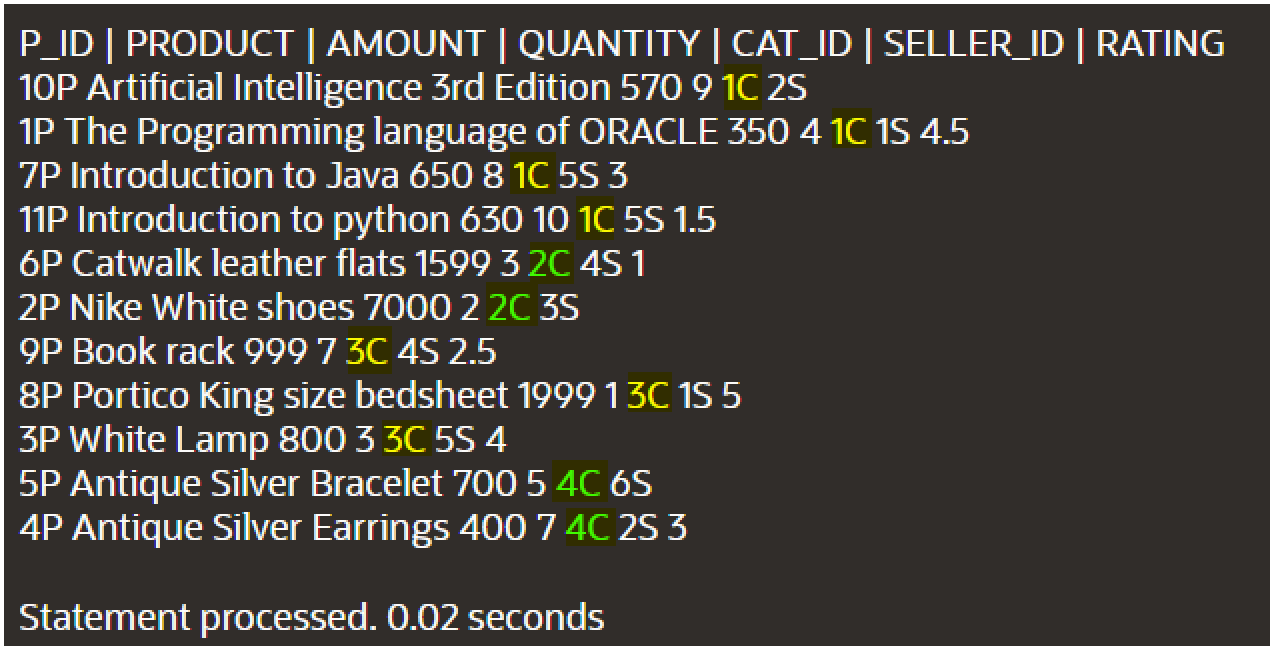
3. Create a cursor to display all the products category wise.

<u>Cursor:</u>

```
DECLARE

-- Variables to Hold Data
prod_prodid PRODUCT.PRODUCT_ID%TYPE;
prod_name PRODUCT.PRODUCT%TYPE;
prod_amt PRODUCT.AMOUNT%TYPE;
prod_quant PRODUCT.QUANTITY_REM%TYPE;
prod_catid PRODUCT.CATEGORY_ID%TYPE;
prod_sellerid PRODUCT.SELLER_ID%TYPE;
prod_rating PRODUCT.RATING%TYPE;

-- CURSOR
CURSOR prod_details IS
SELECT PRODUCT_ID,PRODUCT,AMOUNT,QUANTITY_REM,CATEGORY_ID,SELLER_ID,RATING
FROM
    PRODUCT
ORDER BY
    CATEGORY_ID;

BEGIN
```

```
    dbms_output.put_line( 'P_ID' || ' | ' || 'PRODUCT' || ' | ' || 'AMOUNT' || ' | ' || 'QUAN
TITY' || ' | ' || 'CAT_ID' || ' | ' || 'SELLER_ID' || ' | ' || 'RATING');

OPEN prod_details;
    -- Loop to Print the Output
    LOOP FETCH prod_details INTO prod_prodid, prod_name, prod_amt, prod_quant, prod_catid, pr
od_sellerid, prod_rating;
    EXIT WHEN prod_details%NOTFOUND;
    dbms_output.put_line( prod_prodid || ' ' || prod_name || ' ' || prod_amt || ' ' || prod_q
uant || ' ' || prod_catid || ' ' || prod_sellerid || ' ' || prod_rating);
    END LOOP;
CLOSE prod_details;

END;/
```

<u>Output:</u>



```
P_ID | PRODUCT | AMOUNT | QUANTITY | CAT_ID | SELLER_ID | RATING
10P Artificial Intelligence 3rd Edition 570 9 1C 2S
1P The Programming language of ORACLE 350 4 1C 1S 4.5
7P Introduction to Java 650 8 1C 5S 3
11P Introduction to python 630 10 1C 5S 1.5
6P Catwalk leather flats 1599 3 2C 4S 1
2P Nike White shoes 7000 2 2C 3S
9P Book rack 999 7 3C 4S 2.5
8P Portico King size bedsheet 1999 1 3C 1S 5
3P White Lamp 800 3 3C 5S 4
5P Antique Silver Bracelet 700 5 4C 6S
4P Antique Silver Earrings 400 7 4C 2S 3

Statement processed. 0.02 seconds
```

Triggers:

1. Create a trigger to update the remaining quantity of product in the product table, when a new entry in order_products table is inserted.

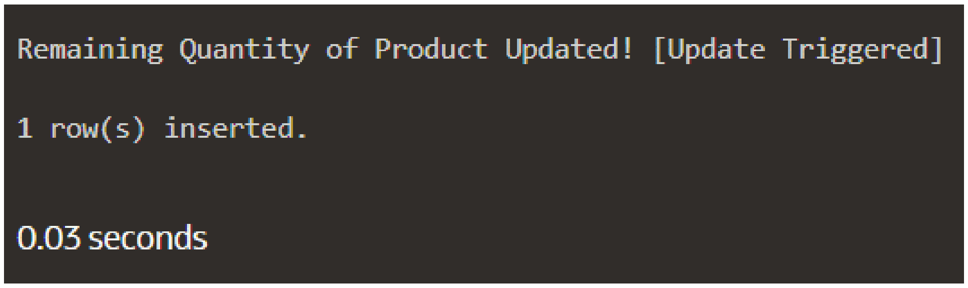
<u>Trigger:</u>

```
CREATE OR REPLACE TRIGGER qty_trigger
AFTER INSERT ON
    ORDER_PRODUCT
FOR EACH ROW
```

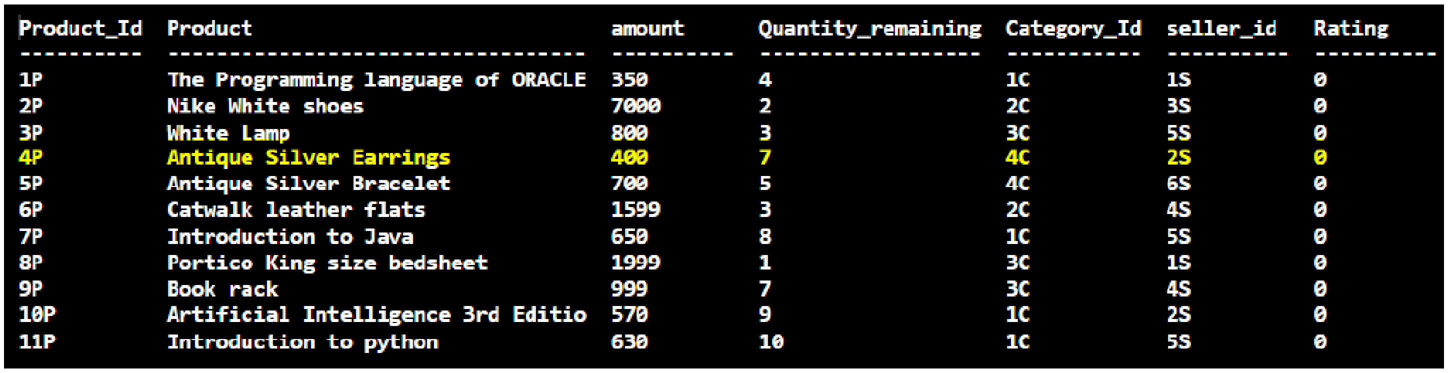
```
BEGIN
    UPDATE
        PRODUCT
    SET
        QUANTITY_REM = QUANTITY_REM - :NEW.QUANTITY
    WHERE
        QUANTITY_REM > 0 AND PRODUCT_ID = :NEW.PRODUCT_ID;

    IF SQL%ROWCOUNT = 0 THEN
        dbms_output.put_line('No Row Updated! [Update was Triggered]');
    ELSE
        dbms_output.put_line('Remaining Quantity of Product Updated! [Update Triggered]');
    END IF;
END qty_trigger;
```

<u>Test:</u>

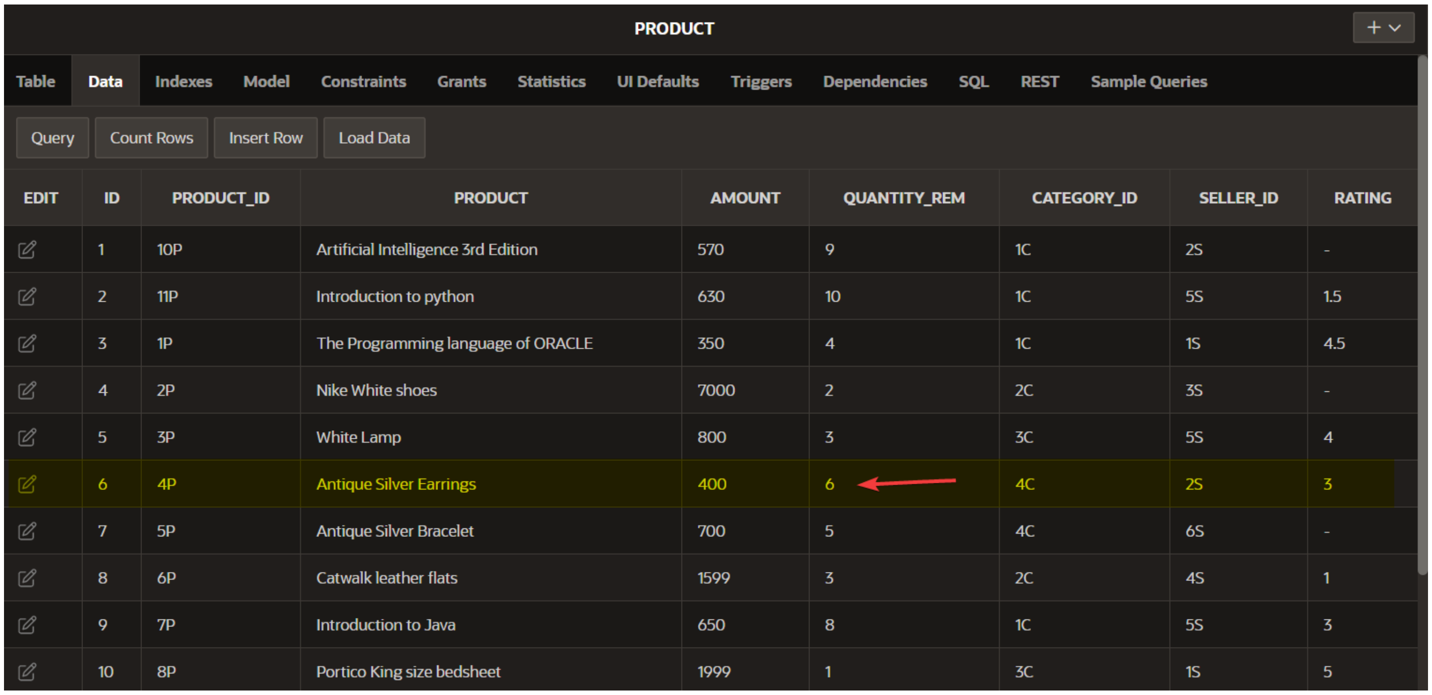
```
--
Lets Order on Product with Product ID 4P and after this Order the Quantity in Product Table S
hould Decrease!

INSERT INTO ORDER_PRODUCT(ORDER_ID, PRODUCT_ID, QUANTITY, SELLER_ID, ORIGINAL_AMT, DISCOUNT,
PROD_RATING) VALUES('6O' , '4P' , 1 , '2S' , 400 , 0 , 4);
```



```
Remaining Quantity of Product Updated! [Update Triggered]

1 row(s) inserted.


0.03 seconds
```



```
Product_Id  Product                             amount      Quantity_remaining  Category_Id  seller_id   Rating
----------  ----------------------------------  ----------  ------------------  -----------  ----------  ----------
1P          The Programming language of ORACLE  350         4                   1C           1S          0
2P          Nike White shoes                    7000        2                   2C           3S          0
3P          White Lamp                          800         3                   3C           5S          0
4P          Antique Silver Earrings             400         7                   4C           2S          0
5P          Antique Silver Bracelet             700         5                   4C           6S          0
6P          Catwalk leather flats               1599        3                   2C           4S          0
7P          Introduction to Java                650         8                   1C           5S          0
8P          Portico King size bedsheet          1999        1                   3C           1S          0
9P          Book rack                           999         7                   3C           4S          0
10P         Artificial Intelligence 3rd Editio  570         9                   1C           2S          0
11P         Introduction to python              630         10                  1C           5S          0
```



EDIT	ID	PRODUCT_ID	PRODUCT	AMOUNT	QUANTITY_REM	CATEGORY_ID	SELLER_ID	RATING
	1	10P	Artificial Intelligence 3rd Edition	570	9	1C	2S	-
	2	11P	Introduction to python	630	10	1C	5S	1.5
	3	1P	The Programming language of ORACLE	350	4	1C	1S	4.5
	4	2P	Nike White shoes	7000	2	2C	3S	-
	5	3P	White Lamp	800	3	3C	5S	4
	6	4P	Antique Silver Earrings	400	6	4C	2S	3
	7	5P	Antique Silver Bracelet	700	5	4C	6S	-
	8	6P	Catwalk leather flats	1599	3	2C	4S	1
	9	7P	Introduction to Java	650	8	1C	5S	3
	10	8P	Portico King size bedsheet	1999	1	3C	1S	5

2. Create a trigger to update product rating and seller rating when a new entry in the order_products table is inserted.

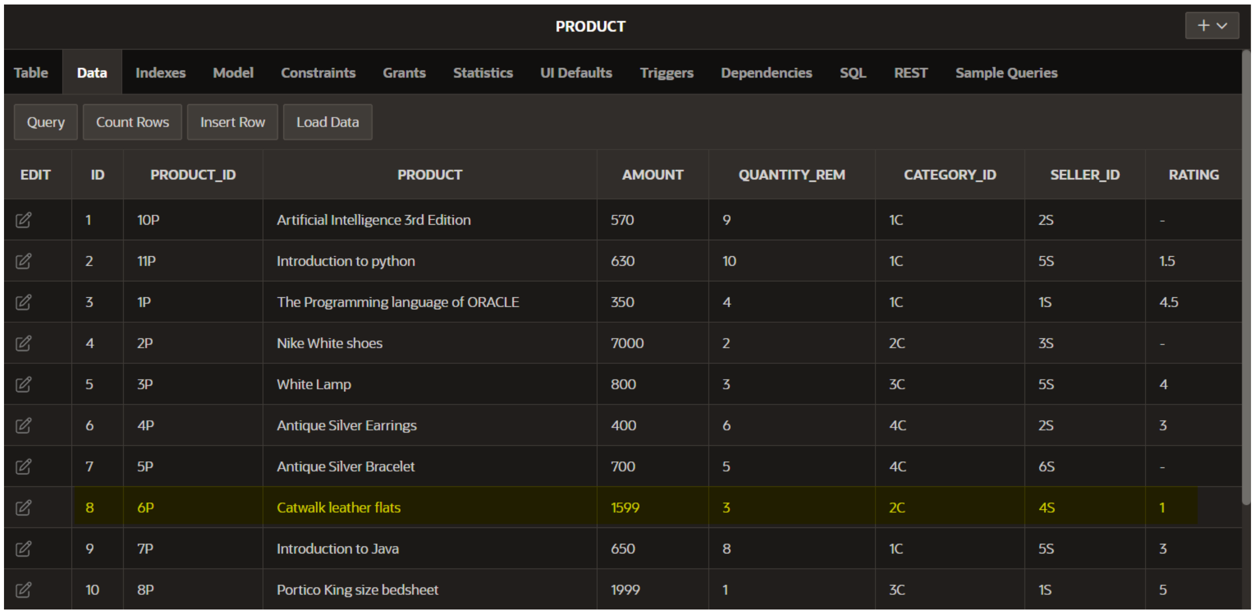
Trigger:

```
CREATE OR REPLACE TRIGGER rating_trigger
AFTER INSERT ON ORDER_PRODUCT

BEGIN
-- Update Product Rating
UPDATE
   product p
SET
   p.rating =
   (
      SELECT
         AVG(prod_rating)
      FROM
         order_product
      GROUP BY
         product_id
      HAVING
         product_id = p.product_id
   );

-- Update Seller Rating
UPDATE
   seller s
SET
```

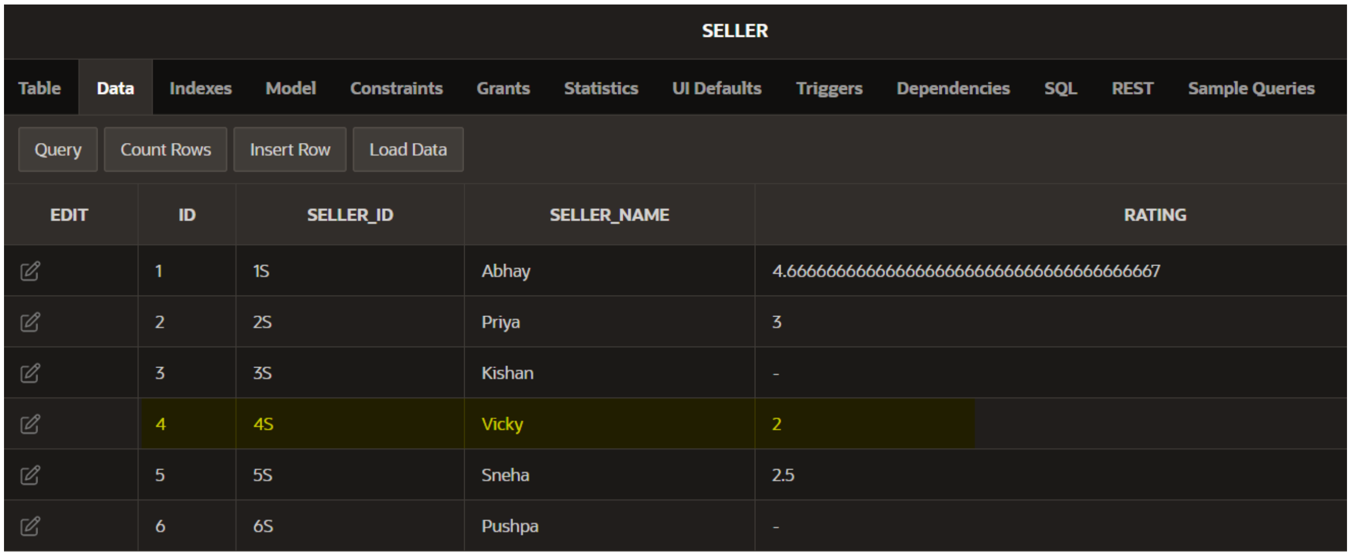
```
    s.rating = (
    SELECT
        AVG(prod_rating)
    FROM
        order_product
    GROUP BY
        seller_id
    HAVING
        seller_id = s.seller_id
   );

-- Output Update Info
    IF SQL%ROWCOUNT = 0 THEN
        dbms_output.put_line('No Updates Made in Database!');
    ELSE
        dbms_output.put_line('Product Rating and Seller Rating Updated Successfully! [Update
Triggered]');
    END IF;

END rating_trigger;
```

Test:

Before Query:


PRODUCT
Table Data Indexes Model Constraints Grants Statistics UI Defaults Triggers Dependencies SQL REST Sample Queries
Query Count Rows Insert Row Load Data

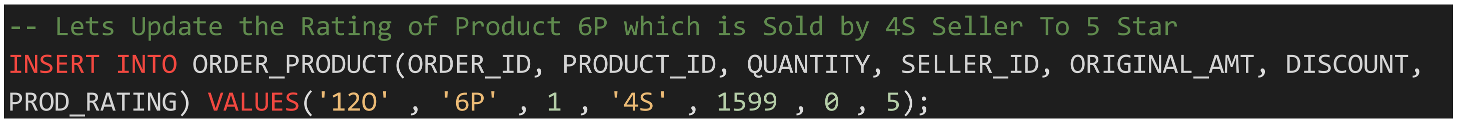
EDIT	ID	PRODUCT_ID	PRODUCT	AMOUNT	QUANTITY_REM	CATEGORY_ID	SELLER_ID	RATING
	1	10P	Artificial Intelligence 3rd Edition	570	9	1C	2S	-
	2	11P	Introduction to python	630	10	1C	5S	1.5
	3	1P	The Programming language of ORACLE	350	4	1C	1S	4.5
	4	2P	Nike White shoes	7000	2	2C	3S	-
	5	3P	White Lamp	800	3	3C	5S	4
	6	4P	Antique Silver Earrings	400	6	4C	2S	3
	7	5P	Antique Silver Bracelet	700	5	4C	6S	-
	8	6P	Catwalk leather flats	1599	3	2C	4S	1
	9	7P	Introduction to Java	650	8	1C	5S	3
	10	8P	Portico King size bedsheet	1999	1	3C	1S	5



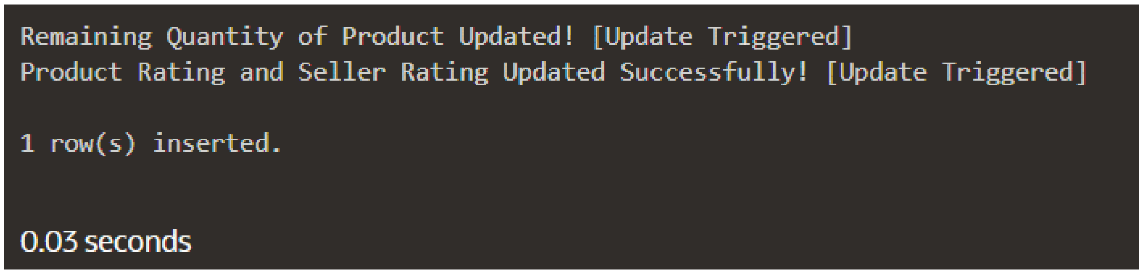
SELLER

EDIT	ID	SELLER_ID	SELLER_NAME	RATING
	1	1S	Abhay	4.666666666666666666666666666666666666667
	2	2S	Priya	3
	3	3S	Kishan	-
	4	4S	Vicky	2
	5	5S	Sneha	2.5
	6	6S	Pushpa	-

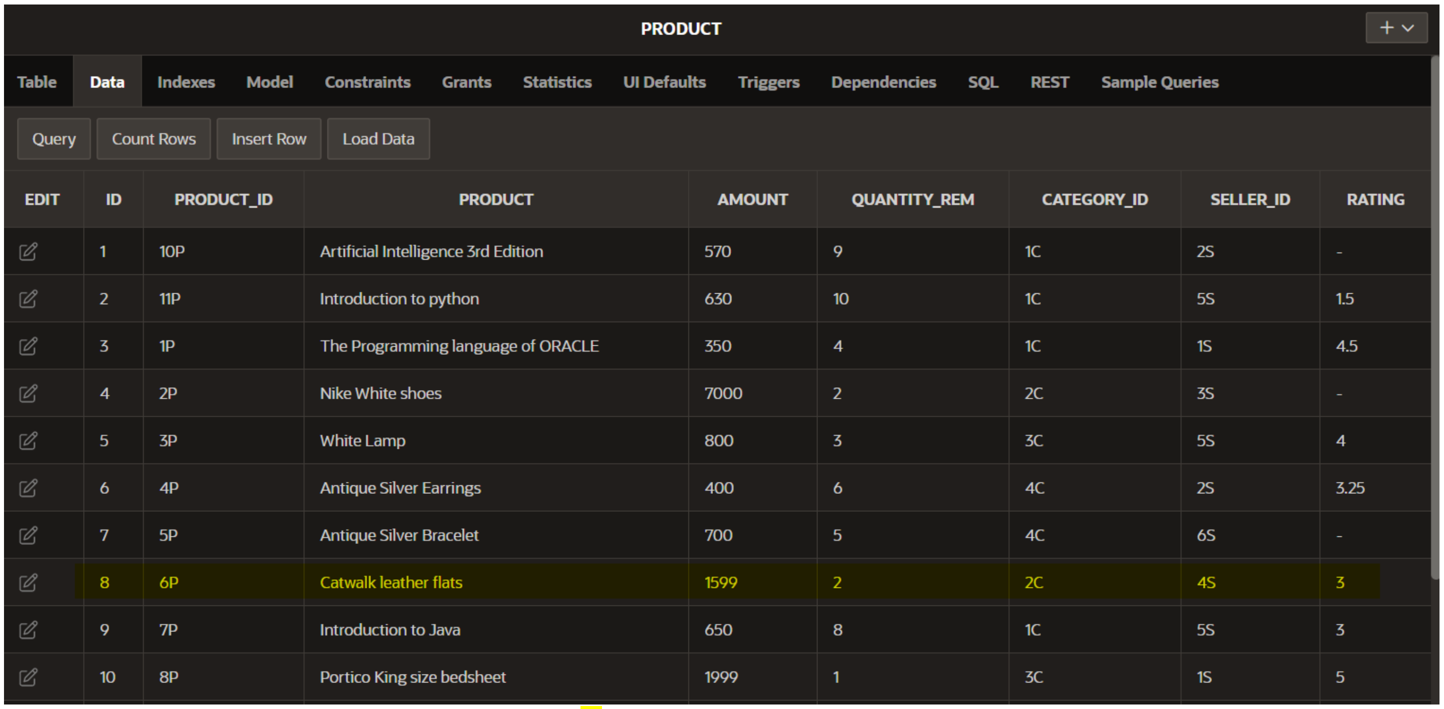
Test:



```sql
-- Lets Update the Rating of Product 6P which is Sold by 4S Seller To 5 Star
INSERT INTO ORDER_PRODUCT(ORDER_ID, PRODUCT_ID, QUANTITY, SELLER_ID, ORIGINAL_AMT, DISCOUNT,
PROD_RATING) VALUES('120' , '6P' , 1 , '4S' , 1599 , 0 , 5);
```



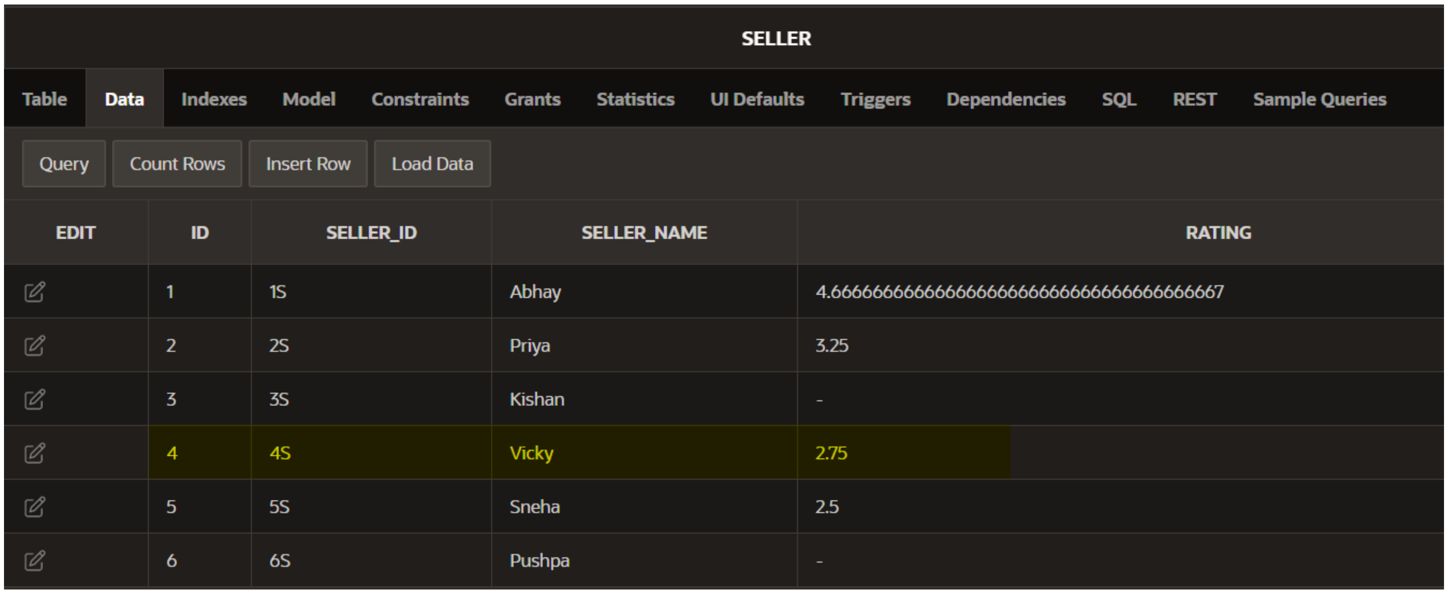
```
Remaining Quantity of Product Updated! [Update Triggered]
Product Rating and Seller Rating Updated Successfully! [Update Triggered]

1 row(s) inserted.


0.03 seconds
```



PRODUCT

EDIT	ID	PRODUCT_ID	PRODUCT	AMOUNT	QUANTITY_REM	CATEGORY_ID	SELLER_ID	RATING
	1	10P	Artificial Intelligence 3rd Edition	570	9	1C	2S	-
	2	11P	Introduction to python	630	10	1C	5S	1.5
	3	1P	The Programming language of ORACLE	350	4	1C	1S	4.5
	4	2P	Nike White shoes	7000	2	2C	3S	-
	5	3P	White Lamp	800	3	3C	5S	4
	6	4P	Antique Silver Earrings	400	6	4C	2S	3.25
	7	5P	Antique Silver Bracelet	700	5	4C	6S	-
	8	6P	Catwalk leather flats	1599	2	2C	4S	3
	9	7P	Introduction to Java	650	8	1C	5S	3
	10	8P	Portico King size bedsheet	1999	1	3C	1S	5

(1+5)/2 = 3 <= New Rating of Product



SELLER

Table | Data | Indexes | Model | Constraints | Grants | Statistics | UI Defaults | Triggers | Dependencies | SQL | REST | Sample Queries

Query | Count Rows | Insert Row | Load Data

EDIT	ID	SELLER_ID	SELLER_NAME	RATING
	1	1S	Abhay	4.66666666666666666666666666666666666667
	2	2S	Priya	3.25
	3	3S	Kishan	-
	4	4S	Vicky	2.75
	5	5S	Sneha	2.5
	6	6S	Pushpa	-

3. Create a trigger to check when a new entry is to be inserted in the order_products table the quantity column satisfies the remaining quantity column from the product table.

<u>Trigger:</u>

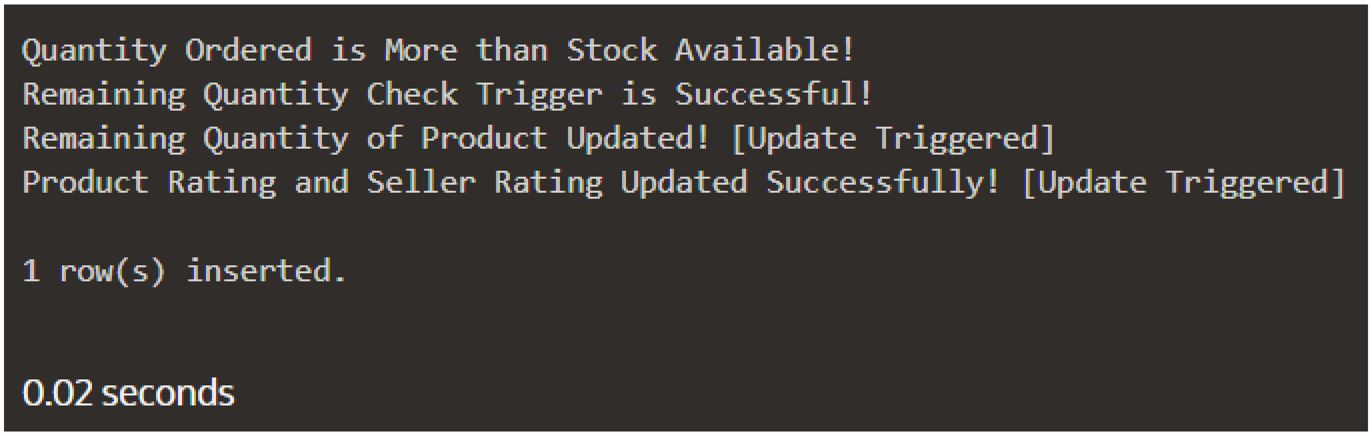
```
CREATE OR REPLACE TRIGGER qty_check
AFTER INSERT
ON ORDER_PRODUCT
FOR EACH ROW

-- To Store Current Quantity & Compare with Given Input One
DECLARE
    tmp PRODUCT.QUANTITY_REM%TYPE;
BEGIN

    SELECT
        QUANTITY_REM INTO tmp
    FROM
        PRODUCT
    WHERE
        product_id = :NEW.product_id;

    IF (tmp > :NEW.QUANTITY) THEN
        UPDATE PRODUCT SET QUANTITY_REM = QUANTITY_REM - :NEW.QUANTITY;
        dbms_output.put_line('Remaining Quantity is Satisfied by New Order!');
    ELSE
        dbms_output.put_line('Quantity Ordered is More than Stock Available!');
    END IF;

    IF SQL%ROWCOUNT = 0 THEN
```

```
        dbms_output.put_line('No row affected');
    ELSE
        dbms_output.put_line('Remaining Quantity Check Trigger is Successful!');
    END IF;

END qty_check;
```

Test:

```
-- Lets Order More than Quantity Available for Product 7P which is Sold by Seller 5S
INSERT INTO ORDER_PRODUCT(ORDER_ID, PRODUCT_ID, QUANTITY, SELLER_ID, ORIGINAL_AMT, DISCOUNT,
PROD_RATING) VALUES('15O' , '7P' , 15 , '5S' , 650 , 0 , 3);
```

Output:



```
Quantity Ordered is More than Stock Available!
Remaining Quantity Check Trigger is Successful!
Remaining Quantity of Product Updated! [Update Triggered]
Product Rating and Seller Rating Updated Successfully! [Update Triggered]

1 row(s) inserted.


0.02 seconds
```

Submitted By:

BHAGYA VINOD RANA

U19CS012