DBMS ASSIGNMENT – 7

***Roll Number: U19CS012***

***Name: BHAGYA VINOD RANA***

1.) Create the Following Tables:

**Master tables**: Seller, Category, Product and Customer

A.) **Seller**

(Seller Id, Seller Name, Rating)

CREATE TABLE Seller(

    Seller\_Id CHAR(3) PRIMARY KEY,

    Seller\_Name CHAR(10),

    Rating FLOAT,

    CHECK(RATING BETWEEN 0 AND 5)

);

B.) **Category**

(Category Id, Category)

CREATE TABLE Category (

    Category\_Id CHAR(3) PRIMARY KEY,

    Category CHAR(10)

);

C.) **Product**

(Product Id, Product, amount, Quantity remaining, Category Id, seller id, Rating)

CREATE TABLE Product (

    Product\_Id CHAR(4) PRIMARY KEY,

    Product VARCHAR(25),

    amount INT,

    Quantity\_remaining INT,

    Category\_Id CHAR(3),

    seller\_id CHAR(3),

    Rating INT DEFAULT 0,

    FOREIGN KEY(CATEGORY\_ID) REFERENCES CATEGORY(CATEGORY\_ID),

    FOREIGN KEY(SELLER\_ID) REFERENCES SELLER(SELLER\_ID)

);

D.) **Customer**

(Customer Id, name, password)

CREATE TABLE Customer (

    Customer\_Id CHAR(5) PRIMARY KEY,

    name CHAR(10),

    password VARCHAR(15)

);

**Transaction tables**: Order, Order\_Products

E.) **Orders**

(Order id, customer id, amount, date, time)

CREATE TABLE Orders (

    Order\_id CHAR(5) PRIMARY KEY,

    customer\_id CHAR(10),

    amount INT,

    TIME\_STAMP\_OF\_ORDER DATE,

    FOREIGN KEY(CUSTOMER\_ID) REFERENCES CUSTOMER(CUSTOMER\_ID)

);

F.) **Order\_Products**

(Order id, Product id, quantity, seller id, Original amount, discount, product Rating)

CREATE TABLE Order\_Products(

    Order\_id CHAR(5) PRIMARY KEY,

    Product\_id CHAR(4),

    quantity INT ,

    seller\_id CHAR(3),

    Original\_amount INT,

    discount INT,

    product\_Rating INT ,

    FOREIGN KEY(ORDER\_ID) REFERENCES ORDERS(ORDER\_ID),

    FOREIGN KEY(PRODUCT\_ID,PRODUCT\_RATING) REFERENCES PRODUCT(PRODUCT\_ID,RATING),

    FOREIGN KEY(SELLER\_ID) REFERENCES SELLER(SELLER\_ID)

);

Insert the Given Data into the Database.

INSERT INTO SELLER (Seller\_Id, Seller\_Name, Rating) VALUES ('1S', 'Abhay', 3.3);

INSERT INTO SELLER (Seller\_Id, Seller\_Name, Rating) VALUES ('2S', 'Priya', 1);

INSERT INTO SELLER (Seller\_Id, Seller\_Name, Rating) VALUES ('3S', 'Kishan', 4.8);

INSERT INTO SELLER (Seller\_Id, Seller\_Name, Rating) VALUES ('4S', 'Vicky', 4.3);

INSERT INTO SELLER (Seller\_Id, Seller\_Name, Rating) VALUES ('5S', 'Sneha', 3.6);

INSERT INTO SELLER (Seller\_Id, Seller\_Name, Rating) VALUES ('6S', 'Pushpa', 2.8);

INSERT INTO CATEGORY (Category\_Id, Category) VALUES ('1C', 'Books');

INSERT INTO CATEGORY (Category\_Id, Category) VALUES ('2C', 'Footwear');

INSERT INTO CATEGORY (Category\_Id, Category) VALUES ('3C', 'Home Decor');

INSERT INTO CATEGORY (Category\_Id, Category) VALUES ('4C', 'Accessories');

INSERT INTO PRODUCT (Product\_Id, Product, Amount, Quantity\_Remaining, Category\_Id, Seller\_Id) VALUES

('1P', 'The Programming language of ORACLE', 350, 4, '1C', '1S'),

('2P', 'Nike White shoes', 7000, 2, '2C', '3S'),

('3P', 'White Lamp', 800, 3, '3C', '5S'),

('4P', 'Antique Silver Earrings', 400, 7, '4C', '2S'),

('5P', 'Antique Silver Bracelet', 700, 5, '4C', '6S'),

('6P', 'Catwalk leather flats', 1599, 3, '2C', '4S'),

('7P', 'Introduction to Java', 650, 8, '1C', '5S'),

('8P', 'Portico King size bedsheet', 1999, 1, '3C', '1S'),

('9P', 'Book rack', 999, 7, '3C', '4S'),

('10P', 'Artificial Intelligence 3rd Edition', 570, 9, '1C', '2S'),

('11P', 'Introduction to python', 630, 10, '1C', '5S');

INSERT INTO CUSTOMER (CUSTOMER\_ID, NAME, PASSWORD) VALUES

('CST01','ABRAHM LINCON','AB@LI'),

('CST02','GRAHAM BELL','#BELL'),

('CST03','NICHOLA TESLA','@TESLA'),

('CST04','SWAMI VIVEKANAND','@SWAMI'),

('CST05','VIRAT KOHLI','@RUN MACHINE'),

('CST06','LIONELL MESSI','FOOTBALL'),

('CST07','DUCKWARD LEWIS','drs'),

('CST08','PIED PIPPER','SILICONVALLEY'),

('CST09','STUART LITTLE','@MOUSE'),

('CST10','AXAR PATEL','MOTERA');

INSERT INTO ORDERS(Order\_id, customer\_id, amount, TIME\_STAMP\_OF\_ORDER) VALUES

('O0001', 'CST01', 6500, '2021-03-09 10:00'),

('O0002', 'CST03', 760, '2021-03-07 12:00'),

('O0003', 'CST04', 350, '2021-03-08 18:00'),

('O0004', 'CST06', 6900, '2021-03-09 12:00'),

('O0005', 'CST08', 400, '2021-03-08 16:00'),

('O0006', 'CST09', 700, '2021-03-09 17:00'),

('O0007', 'CST01', 350, '2021-03-05 09:00'),

('O0008', 'CST03', 1900, '2021-03-09 11:00'),

('O0009', 'CST08', 350, '2021-03-06 10:00'),

('O0010', 'CST01', 970, '2021-03-08 14:00');

INSERT INTO Order\_Products

( Order\_id, Product\_id, quantity, seller\_id, Original\_amount, discount, product\_Rating) VALUES

('O0001', '2P', 1, '3S', 7000, 500, 4),

('O0002', '3P', 1, '5S', 800, 40, 3.5),

('O0003', '1P', 1, '1S', 350, 0, 3.8),

('O0004', '2P', 1, '3S', 7000, 100, 4.5),

('O0005', '4P', 1, '2S', 400, 0, 2.9),

('O0006', '5P', 1, '6S', 700, 0, 4),

('O0007', '1P', 1, '1S', 350, 0, 3.6),

('O0008', '8P', 1, '1S', 1999, 99, 4.8),

('O0009', '1P', 1, '1S', 350, 0, 4.7),

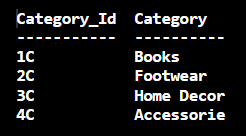
('O0010', '9P', 1, '4S', 999, 29, 4.3);

**Initial Table:**

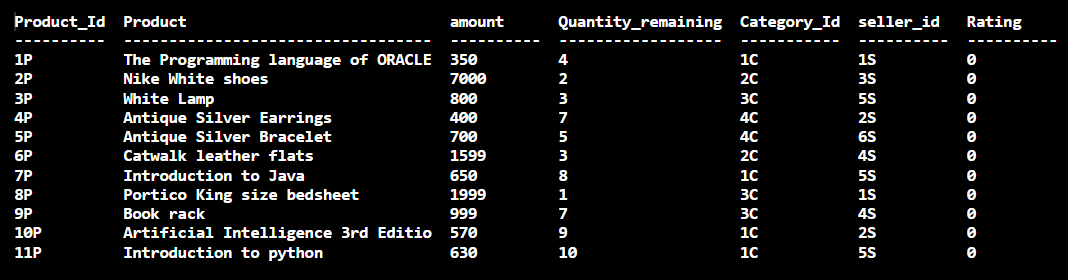
**SELLER TABLE**



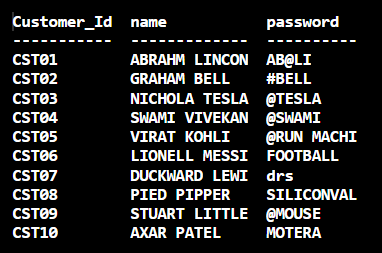
**CATEGORY TABLE**



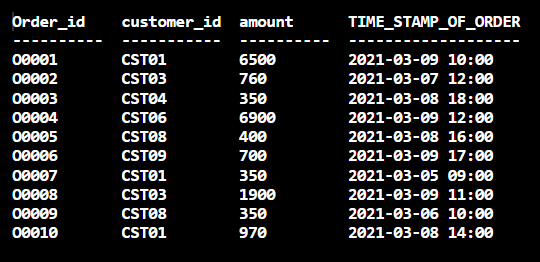
**PRODUCT TABLE**



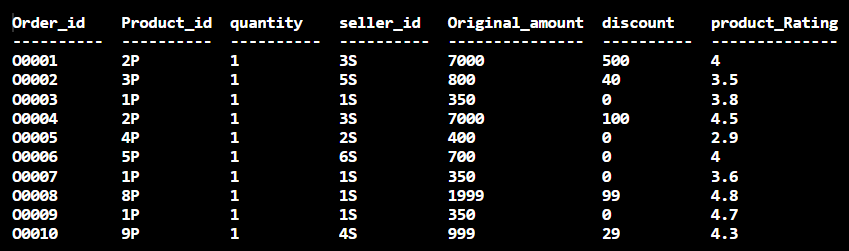
**CUSTOMERS TABLE**



**ORDER TABLE**



**ORDER\_PRODUCTS TABLE**



**Q. Write queries for the following:**

1. Display the highest sold product details.

**Query:**

SELECT \*

FROM PRODUCT, ( SELECT P,MAX(COUNTS)

                FROM (  SELECT PRODUCT\_ID P, COUNT(PRODUCT\_ID) AS COUNTS

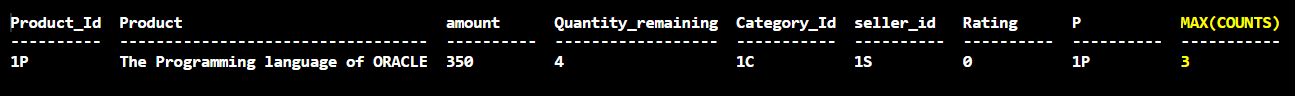
                        FROM ORDER\_PRODUCTS O

                        GROUP BY PRODUCT\_ID

                     ))

                WHERE PRODUCT.PRODUCT\_ID = P;

**Output:**



2. Update product rating column in product table as per the entries in order\_product table (calculate average).

**Query:**

UPDATE PRODUCT

SET RATING=(    SELECT AVG(PRODUCT\_RATING)

                FROM ORDER\_PRODUCTS OLD

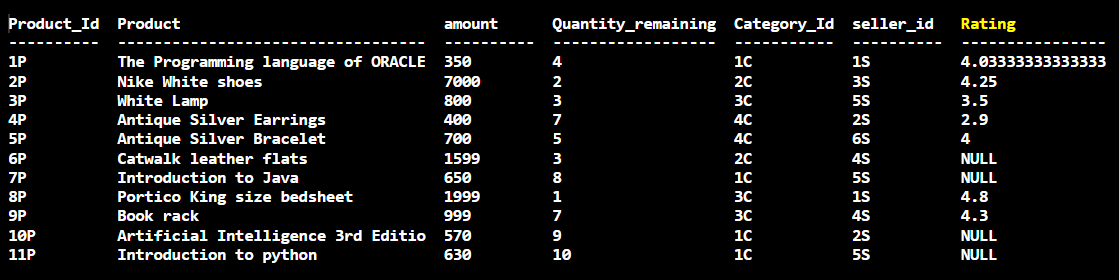
                GROUP BY PRODUCT\_ID

                HAVING OLD.PRODUCT\_ID=PRODUCT.PRODUCT\_ID

);

select \* from PRODUCT;

**Output:**



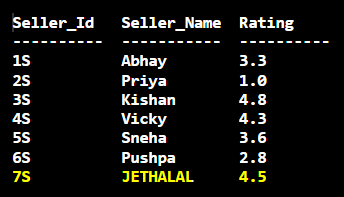
3. Add a new seller with all details.

**Query:**

INSERT INTO SELLER VALUES('7S','JETHALAL','4.5');

SELECT \* FROM SELLER;

**Output:**



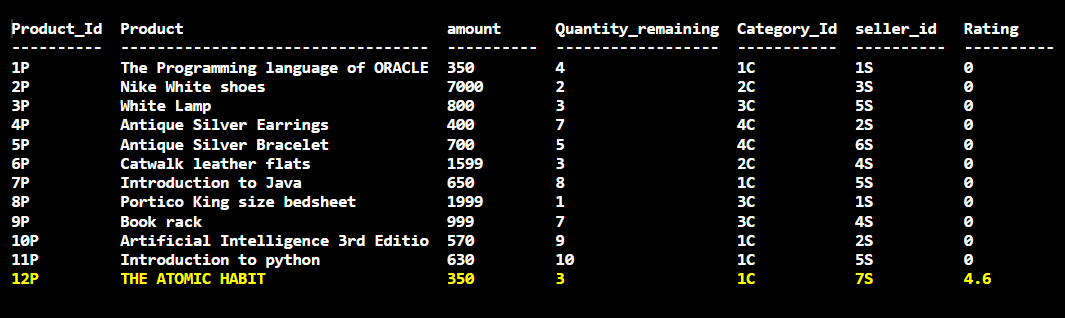
4. Add a new product with all details.

**Query:**

INSERT INTO PRODUCT VALUES('12P','THE ATOMIC HABIT',350,3,'1C','7S',4.6);

SELECT \* FROM PRODUCT;

**Output:**



5. Display the details of the products which have never sold.

**Query:**

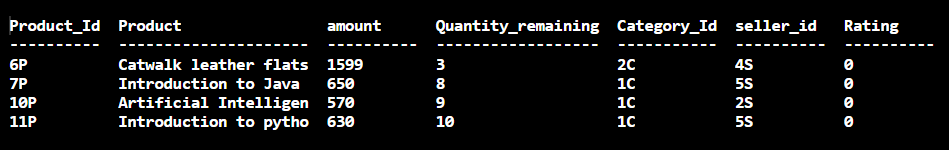
SELECT \*

FROM PRODUCT P

WHERE P.PRODUCT\_ID NOT IN ( SELECT PRODUCT\_ID

                            FROM ORDER\_PRODUCTS);

**Output:**



6. Display the details of the seller who has not sold any product today.

**Query:**

SELECT \*

FROM SELLER

WHERE SELLER\_ID NOT IN ( SELECT SELLER\_ID

                         FROM ORDER\_PRODUCTS

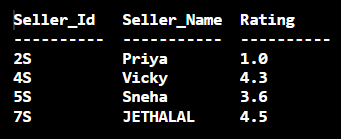
                         WHERE ORDER\_ID IN (SELECT ORDER\_ID

                                            FROM ORDERS

                                            WHERE STRFTIME("%Y-%m-%d",ORDERS.TIME\_STAMP\_OF\_ORDER) IS '2021-03-09')

);

**Output:**



NOTE: Observe that the New Seller “JETHALAL” who was added recently, has not sold any Product and therefore is also there in the List.

7. Display the details of the seller who has sold the highest amount of products today.

**Query:**

SELECT \*

FROM SELLER S, (SELECT SI ,MAX(SUMATION)

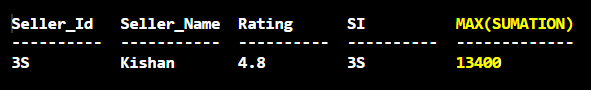
                FROM (SELECT SELLER\_ID AS SI, SUM(ORIGINAL\_AMOUNT- DISCOUNT) AS SUMATION

                FROM ORDER\_PRODUCTS

                GROUP BY SELLER\_ID))

                WHERE S.SELLER\_ID=SI;

**Output:**



8. Display the product details with the highest rating.

**Query:**

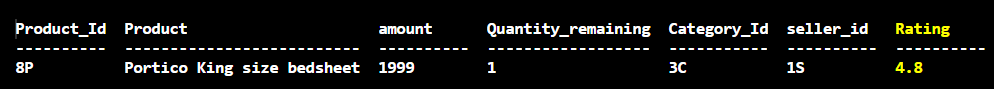
SELECT \*

FROM PRODUCT

WHERE RATING IS ( SELECT MAX(RATING)

                  FROM PRODUCT);

**Output:**



9. Display the customer details who has repeated the same product purchase in the last three months.

**Query:**

INSERT INTO ORDERS( Order\_id, customer\_id, amount, TIME\_STAMP\_OF\_ORDER ) VALUES

('O0011','CST01',350,'2021-03-09 16:00');

INSERT INTO Order\_Products

( Order\_id , Product\_id , quantity , seller\_id ,Original\_amount , discount ,product\_Rating )

VALUES ('O0011', '1P',1,'1S',350,0,4);

SELECT \*

FROM CUSTOMER

WHERE CUSTOMER\_ID IN (  SELECT O1.CUSTOMER\_ID

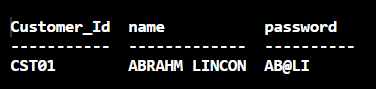
                        FROM (ORDERS NATURAL JOIN ORDER\_PRODUCTS) O1 ,(ORDERS NATURAL JOIN ORDER\_PRODUCTS) O2

                        WHERE O1.CUSTOMER\_ID=O2.CUSTOMER\_ID AND O1.ORDER\_ID IS NOT O2.ORDER\_ID AND O1.PRODUCT\_ID=O2.PRODUCT\_ID

                        AND JULIANDAY('now')-JULIANDAY(O1.TIME\_STAMP\_OF\_ORDER)<=90

);

**Output:**



10. Display the seller details who is second highest in selling products in the last three months.

**Query:**

*-- Note : HERE IN THIS QUESTION IT IS CONSIDERED THAT "HIGHEST IN SELLING PRODUCTS" = "HIGHEST QUANTITY SOLD"*

*-- IF ABOVE ONE IS NOT TRUE RATHER IF IT IS "HIGHEST IN SELLING PRODUCTS" = "HIGHEST AMOUNT"*

*-- THEN REPLACING QUANTITY TO AMOUNT WILL WORK AS PER REQUIREMENTS*

CREATE TABLE TEMPORARY(

    SELLER\_ID CHAR(3),

    TOTAL\_QUANTITY INT

);

INSERT INTO TEMPORARY

SELECT SELLER\_ID ,SUM(QUANTITY)

FROM (  SELECT \*

        FROM ORDER\_PRODUCTS NATURAL JOIN ORDERS

        WHERE JULIANDAY('now')-JULIANDAY(TIME\_STAMP\_OF\_ORDER)<=90)

GROUP BY SELLER\_ID;

SELECT \*

FROM SELLER

WHERE SELLER\_ID IN (    SELECT SELLER\_ID

                        FROM (  SELECT SELLER\_ID , MAX(TOTAL\_QUANTITY) AS salary

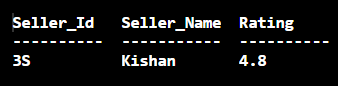
                                FROM TEMPORARY

                                WHERE TOTAL\_QUANTITY < (SELECT MAX(TOTAL\_QUANTITY) FROM TEMPORARY))

);

DROP TABLE TEMPORARY;

**Output:**



11. Display products in the descending order of product amount sold by the seller who is having the highest rating.

**Query:**

SELECT DISTINCT PRODUCT\_ID, PRODUCT , AMOUNT , QUANTITY\_REMAINING , CATEGORY\_ID , SELLER\_ID

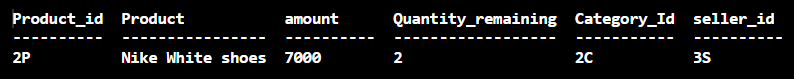
FROM (ORDER\_PRODUCTS NATURAL JOIN PRODUCT)

WHERE SELLER\_ID IN (SELECT SID

                    FROM (SELECT SELLER\_ID SID, MAX(RATING) FROM SELLER))

                    ORDER BY ORIGINAL\_AMOUNT-DISCOUNT DESC;

**Output:**



12. Update the seller ratings as per the new entries in Order\_Products table.

**Query:**

SELECT \* FROM SELLER;

UPDATE SELLER

SET RATING=(

    SELECT AVG(PRODUCT\_RATING)

    FROM ORDER\_PRODUCTS OP

    GROUP BY SELLER\_ID

    HAVING OP.SELLER\_ID=SELLER.SELLER\_ID

);

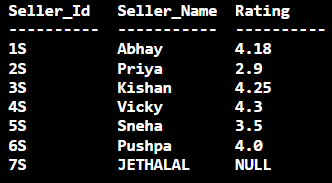
SELECT \* FROM SELLER;

**Output:**

**Before Updating:**



**After Updating:**



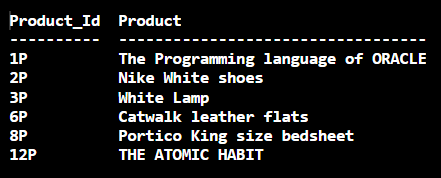
13. Display the list of products having quantity remaining <= 4.

**Query:**

select product\_id , product from product

where quantity\_remaining <= 4;

**Output:**



**Submitted By:**

**BHAGYA VINOD RANA**

**U19CS012**