DBMS PROJECT DEADLINE-5

ARYAN DHULL(2021520)
DEEPANSHU(2021524)

TRIGGERS:

Description: Adding products to a cart for a given customer decreases the stock quantity from the products table.

Final Trigger used in SQL:

```
/*Updates quantity in products table once a order containing the product is placed*/

CREATE TRIGGER update_stock_trigger
AFTER INSERT ON order_details
FOR EACH ROW
    UPDATE products
    SET Quantity = Quantity - NEW.Quantity
    WHERE Product_ID = NEW.Product_ID;
```

Description: After placing the order, the row in the cart table corresponding to the order's customer becomes empty.

Final Trigger used in SQL:

```
/*Updates the cart once an order is placed (clears the cart) */

CREATE TRIGGER clear_cart_trigger
AFTER INSERT ON orders
FOR EACH ROW
    DELETE FROM cart_items
    WHERE Customer_ID= NEW.Customer_ID;
```

Description: This trigger appoints the delivery partner whenever the order is placed.

Final Trigger used in SQL:

```
/*Appoints a delivery partner whenever an order is placed*/

CREATE TRIGGER appoint_delivery_partner
AFTER INSERT ON orders
FOR EACH ROW
    INSERT INTO order_delivery
    VALUES((Select Delivery_Partner_ID from delivery_partners order by RAND() LIMIT 1),NEW.Order_ID,NEW.Customer_ID,
    (select First_Name from customers where Customer_ID=NEW.Customer_ID),(select Street from customers where Customer_ID=NEW.Customer_ID))
```

EMBEDDED SQL QUERIES:

QUERY 1:

Description: Display names of managers , manufacturers and categories wherever the number of products manufactured are between 40,000 and 60,000

```
mycursor.execute('''select A.First_Name as Manager_Name, B.Name,B.Category_Name
                    from Managers A,Manufacturer B
                    where B.Number_Of_Products between 40000 and 60000 and
                    A.Manager_ID=B.Manager_ID''')
print("Manager Name",'\t',"     Manufacturer Name",'\t\t\t',"Category Name")
for x in mycursor:
    print(x[0]," "*(20-len(x[0])),x[1]," "*(40-len(x[1])),x[2])
```

OUTPUT-

```
PS C:\Users\Dell\Desktop\IIITD\SUBJECT_WISE_WORK\DBMS> python -u "c:\Users\Dell\Desktop\IIITD\SUBJECT_WISE_WORK\DBMS\one.py"
Manager Name        Manufacturer Name                 Category Name
Buddie              Beier Inc                         Electronics
Buddie              Effertz Group                     Daily Use
Bibbye              Volkman, Bergstrom and Hartmann   Daily Use
Bibbye              Mayer, Windler and Aufderhar      Electronics
Keely               Halvorson, Abshire and Kohler     Daily Use
Keely               Douglas-Hettinger                 Electronics
Wynnie              Schuppe LLC                       Clothes
Wynnie              Daniel LLC                        Electronics
Thoma               Fritsch, Kuhlman and Welch        Clothes
Daniel              Grimes, Schneider and Reichert    Daily Use
Norton              Sanford, Kilback and Prosacco     Electronics
Norton              Kuhlman LLC                       Cosmetics
Vilhelmina          Renner, Bradtke and Waters        Electronics
Desi                Johnson LLC                       Electronics
Johnnie             Bradtke, Jaskolski and Strosin    Daily Use
Linet               Gorczany, Kub and Welch           Electronics
Konstantin          Murray Group                      Clothes
```

QUERY 2:

Description: Displays name of customers who have placed orders with total amount above 500

```
mycursor.execute('''select First_Name from customers
                    RIGHT JOIN orders on customers.Customer_ID=orders.Customer_ID
                    where orders.Amount>500''')
print("Customers with Order Value > 500")
for x in mycursor:
    print(x[0])
```

OUTPUT-

```
Customers with Order Value > 500
Rossy
Brady
Loraine
Austin
Morlee
```

Query 3:

Description: Displays the name of products, their price and quantities wherever the total amount (i.e price* quantity) is greater than the average total amount of all products

```
mycursor.execute('''select Product_Name ,Price,Quantity, Price*Quantity as Total_Price
                    from Products where
                    Price*Quantity>(Select AVG(Price*Quantity) from Products)''')
print("Product Name",'\t'," Quantity",'\t\t'," Price",'\t\t',"Total Amount")
for x in mycursor:
    print(x[0]," "*(20-len(x[0])),x[1]," "*(20-len(str(x[1]))),x[2]," "*(20-len(str(x[2]))),x[3])
```

OUTPUT-

Product Name	Quantity	Price	Total Amount
Bread	950	249	236550
Rice	690	250	172500
Milk	660	250	165000
Tomatoes	660	250	165000
Mangoes	810	250	202500
Masale	840	250	210000
Cream	730	250	182500
Biscuits	860	250	215000
Papaya	690	250	172500
Cherries	620	250	155000
Cucumber	960	250	240000
Beetroot	980	250	245000
Flax seeds	950	220	209000
Western	950	250	237500
Indo-Western	720	250	180000
Jackets	560	250	140000
Sweatshirts	900	250	225000
Lowers	660	250	165000
Trousers	530	250	132500
Flats	720	250	180000
Microphones	930	248	230640
Tandoor	770	250	192500
Laptop	680	250	170000
Computer	680	250	170000
Refrigerator	520	250	130000
Washing Machine	600	250	150000
Heaters	660	250	165000
Boiler	900	250	225000
Headphones	720	250	180000
Stylus	900	250	225000
Cables	750	245	183750
Watches	640	250	160000
Handwash	640	250	160000
Facemask	910	250	227500
Crocin	500	250	125000
Lens	870	250	217500
Bandages	960	250	240000
Rose water	650	250	162500
Muraba	890	250	222500
Medicines	490	250	122500
Thermometer	550	250	137500
Safromycin	500	250	125000

OLAP QUERIES:

QUERY 1:

Description: **Counting the number of manufacturers of a particular category.**

```
1 •  select Category_Name ,Count(Manufacturer_ID) as Number_of_Manufacturers
2    from manufacturer
3    GROUP BY Category_Name WITH ROLLUP;
4
```

OUTPUT-

	Category_Name	Number_of_Manufacturers
▶	Clothes	25
	Cosmetics	25
	Daily Use	25
	Electronics	25
	NULL	100

QUERY 2:

Description: Displays the Customer-ID , category name, total quantity of the order placed by the customer and the total revenue generated by that order

```
1   SELECT
2       orders.Customer_ID,
3       products.Category_Name,
4       SUM(order_details.Quantity) AS TotalQuantity,
5       SUM(order_details.Quantity * order_details.Amount) AS TotalRevenue
6   FROM
7       orders
8       JOIN order_details ON orders.Order_ID = order_details.Order_ID
9       JOIN products ON order_details.Product_ID = products.Product_ID
10  GROUP BY
11      orders.Customer_ID, products.Category_Name with rollup;
12
```

OUTPUT-

Customer_ID	Category_Name	TotalQuantity	TotalRevenue
C0001	Daily Use	1	390
C0001	NULL	1	390
C0005	Daily Use	2	1340
C0005	NULL	2	1340
C0006	Daily Use	30	855000
C0006	NULL	30	855000
C0007	Electronics	7	22470
C0007	NULL	7	22470
C0008	Daily Use	2	1560
C0008	NULL	2	1560
NULL	NULL	42	880760

QUERY 3:

Description: **Displays the product name, sum of price of those of those products grouped by Category Name and the total price of all products of Category**

```
1 •  select Product_Name,sum(Price) as Amount,Category_Name from products GROUP BY Category_Name,Product_Name WITH ROLLUP;
```

OUTPUT-

	Product_Name	Amount	Category_Name
▶	Dresses	570	Clothes
	Flats	720	Clothes
	Heels	160	Clothes
	Indian	120	Clothes
	Indo-Western	720	Clothes
	Jackets	560	Clothes
	Jeans	710	Clothes
	Leggings	110	Clothes
	Loafers	400	Clothes
	Lowers	660	Clothes
	Pants	370	Clothes
	Pyjamas	380	Clothes
	Shirts	420	Clothes
	Shorts	250	Clothes
	Sneakers	500	Clothes
	Sport Shoes	490	Clothes
	Suits	720	Clothes
	Sweatshirts	900	Clothes
	T-Shirts	140	Clothes
	Ties	280	Clothes
	Tops	770	Clothes
	Trousers	530	Clothes
	Western	950	Clothes
	Work Shoes	20	Clothes
	NULL	11450	Clothes

Query 4:

Description: Displays the Manager - ID and the number of sellers under the manager

```
1 •  select Manager_ID ,Count(Seller_ID) as Number_of_Sellers
2    from sellers
3    GROUP BY Manager_ID WITH ROLLUP;
```

OUTPUT-

Manager_ID	Number_of_Sellers
M0001	4
M0002	4
M0003	4
M0004	5
M0005	4
M0006	3
M0007	4
M0008	5
M0009	5
M0010	4
M0011	4
M0012	4
M0013	4
M0014	4
M0015	4
M0016	4
M0017	4
M0018	4
M0019	4
M0020	4
M0021	4
M0022	4
M0023	4
M0024	3
M0025	3
NULL	100

Query 5:

Description: Displays the number of customers in different states and the total number of customers

```
1 •   select State, Count(Customer_ID) as
2     Number_of_Customers
3     from customers
4     GROUP BY State
5     UNION ALL
6     select NULL, Count(Customer_ID) as Number_of_Customers
7     from customers;
```

OUTPUT-

State	Number_of_Customers
LA	2
MI	2
NJ	8
AK	5
OH	7
IL	4
CA	15
SD	1
MD	4
PA	8
NY	11
TX	7
AZ	1
TN	2
WI	4
KS	3
NM	2
OR	1
FL	2
MN	1
MA	1
SC	1
RI	2
CO	1
ID	1
NC	1
IN	1
WY	2
VA	1
NULL	101