PHARMACY MANAGEMENT SYSTEM

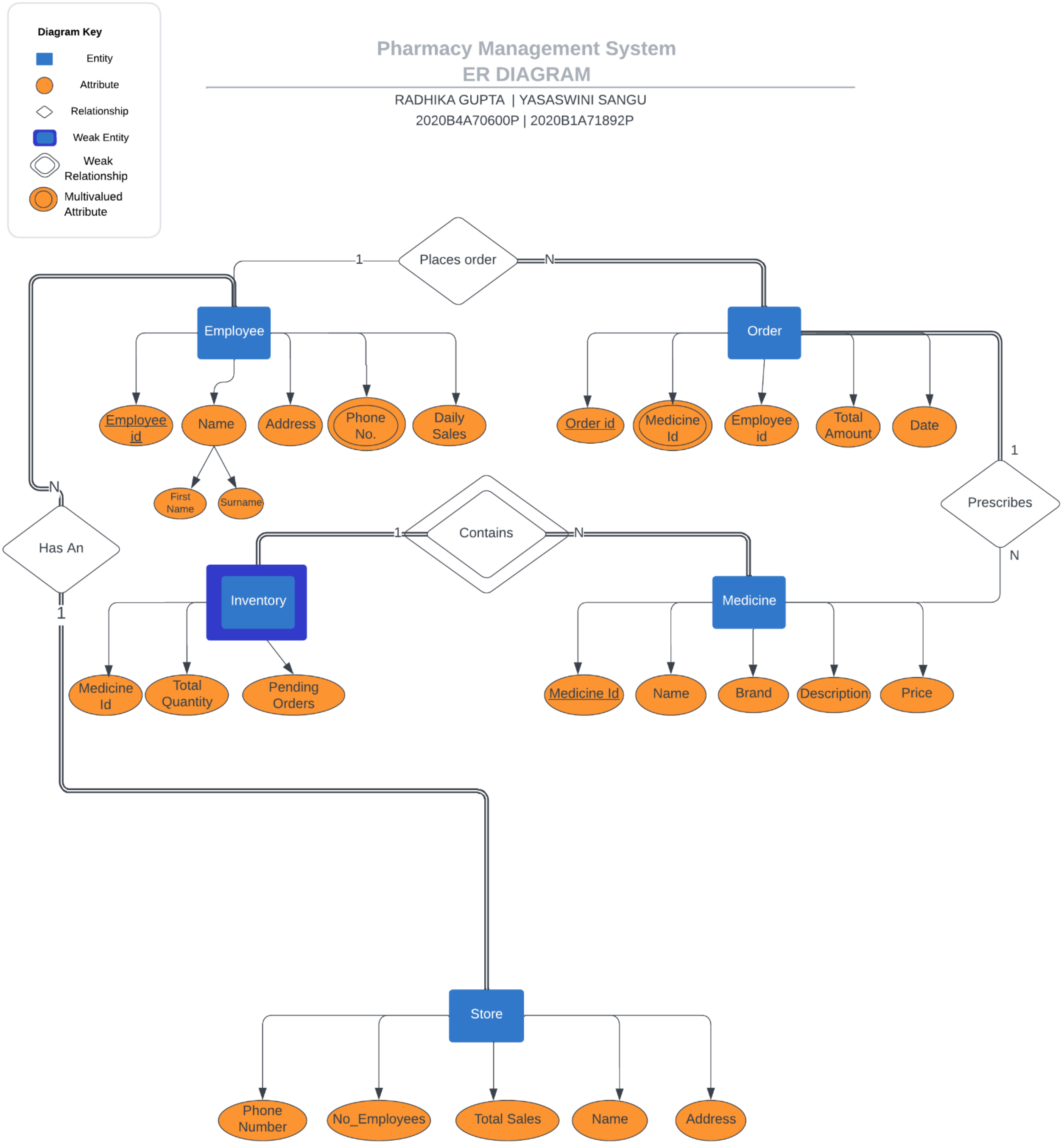
DBMS PROJECT SUBMITTED BY:

RADHIKA GUPTA (2020B4A70600P)

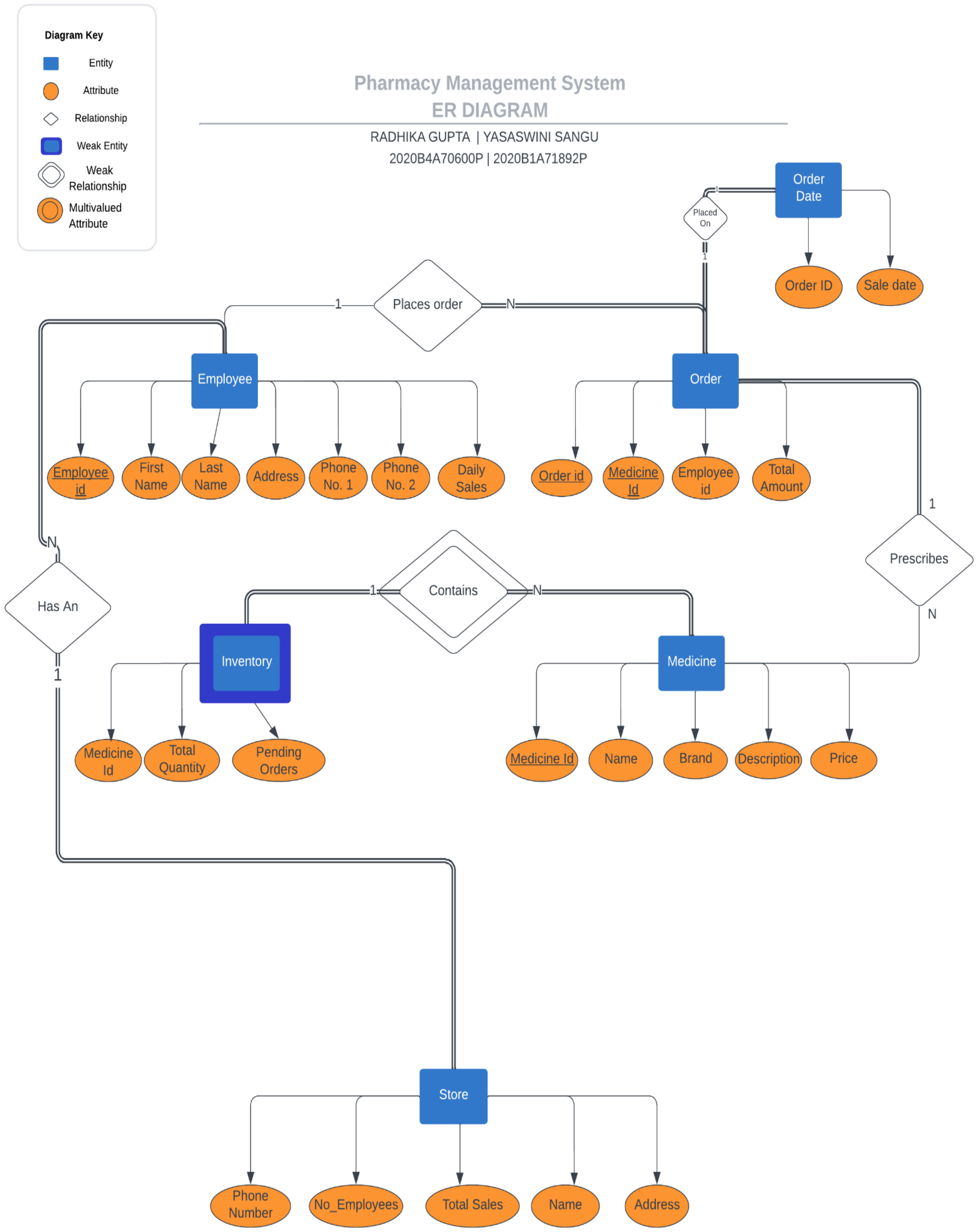
YASASWINI SANGU (2020B1A71892P)

ER DIAGRAM :

BEFORE NORMALISATION



AFTER NORMALISATION



ER DIAGRAM TO RELATIONAL MODEL:

(BEFORE NORMALISATION)

A.STORE

Store_ ID	Name Firs_Name|Last_Name	Address	Phone_no	No_Employees	Total_sales
S1					
S2					
S3					

Store_ID is the primary key that uniquely determines all other values.

B.EMPLOYEE

Employee ID	Name	Phone_No	Address	Daily_Sales
E1	ABC, XYZ	PH1		
E2		PH1 / PH2		
E3		PH3		

Employee_ID is the primary key that uniquely determines all other values.

C.MEDICINE

Medicine ID	Name	Brand	Price	Cost	Expiry_date
M1					
M2					
M3					

Medicine_ID is the primary key that uniquely determines all other values.

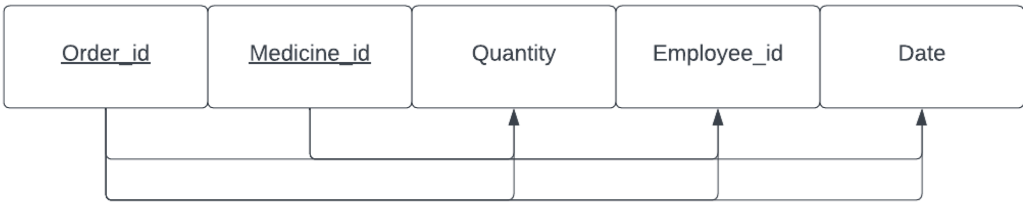
D.ORDER

	(FK)	**(FK)**		
Order ID	Medicine ID	Employee_ID	Quantity	Date

O1	M3/M2			
O2	M2/M1			

Order_ID is the primary key that uniquely determines all other values. Medicine_ID and Employee_ID are foreign keys.

And we also have the Functional dependency ,
Order_ID → Employee_id , Date
Order_ID, Medicine_Id are the candidate keys



E.INVENTORY

(FK)

Medicine ID	Total Quantity	Pending Orders
M1		
M2		

The Foreign key Medicine_ID is the primary key too.

NORMALIZATION :

A.Store:

The table is in 1NF as there are no multivalued or composite attributes.
The table is in 2NF as there are no partial dependencies.
The table is in 3NF as no transitional dependencies, i.e., no non-primary key attribute is dependent on a non-primary key attribute.

B.Employee:

Not in 1NF as it has a composite attribute Name that has First_Name and Last_Name. It also has a multivalued attribute phone number.

To convert it into 1NF, we divided Name into two columns First Name and Last Name, and added another phone number column which allows null value.

Then we make new table Emp_pho with Emp_ID and Phone_number as attributes

Employee

Employee_ID	First_Name	Last_Name	Address	No_Employees	Total_sales
S1					
S2					
S3					

Emp_phone

(FK)

Employee_ID	Phone_No
E1	P1
E2	P2a
E2	P2b
E3	P3

It is now in 2NF and 3NF also as there are no functional dependencies.

C.Medicine:

No changes

The table is in 1NF as there are no multivalued or composite attributes.

The table is in 2NF as there are no partial dependencies.

The table is in 3NF as no transitional dependencies, i.e., no non-primary key attribute is dependent on a non-primary key attribute.

D.Order:

The above table is not in 1NF as Medicine ID is a multivalued attribute, corresponding to Order ID primary key, there can be multiple Medicine ID. To convert it into 1NF, we'll make Order ID, and Medicine ID the composite primary key, as every Order ID, Medicine ID tuple will be unique.

Now,there's a partial dependency as Employee_ID and Date are functionally dependent only on Order_ID. So to remove this , we make a new table with these three values and Medicine_ID as the foreign key reference.

Order

	(FK)	(FK)
Order ID	Medicine ID	Quantity
O1	M3	

O2	M2	
O1	M1	

Order_Details

(FK)

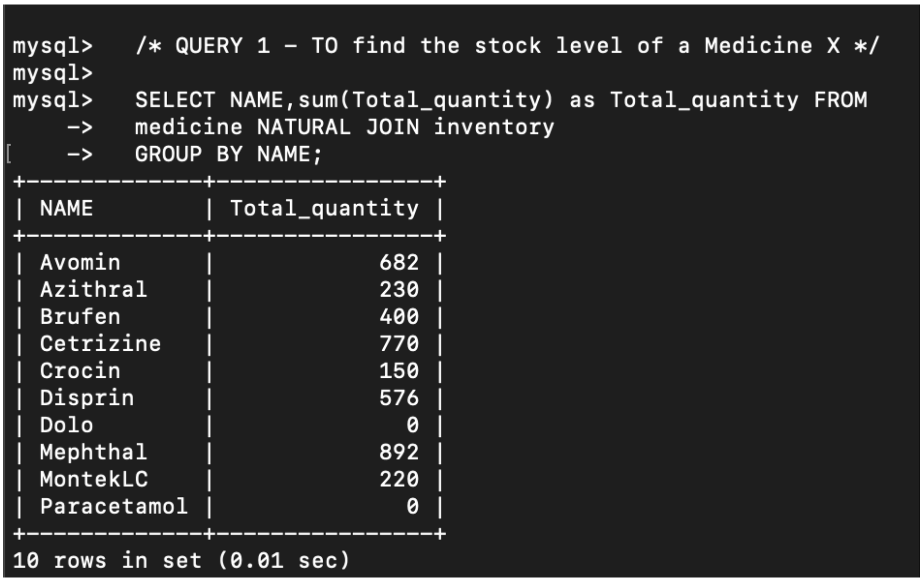
<u>Order ID</u>	Employee_ID	Date
O1	E1	D1
O2	E2	D2

E. INVENTORY:

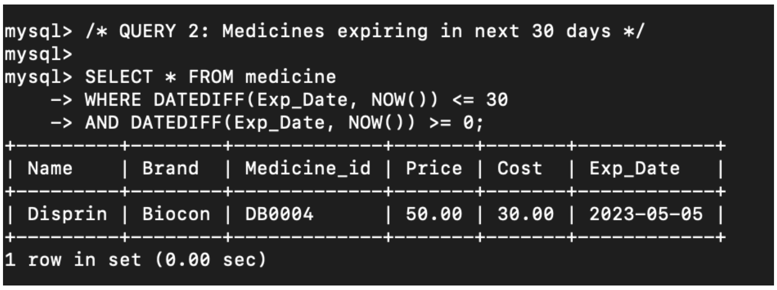
The table is in 1NF as there are no multivalued or composite attributes.
The table is in 2NF as there are no partial dependencies.
The table is in 3NF as no transitional dependencies, i.e., no non-primary key attribute is dependent on a non-primary key attribute.

<u>SQL QUERIES</u>

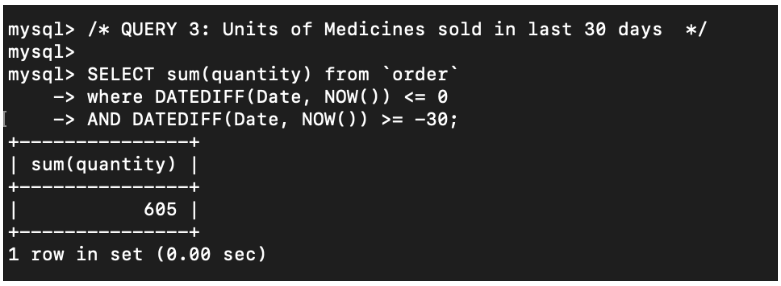
Required Queries:



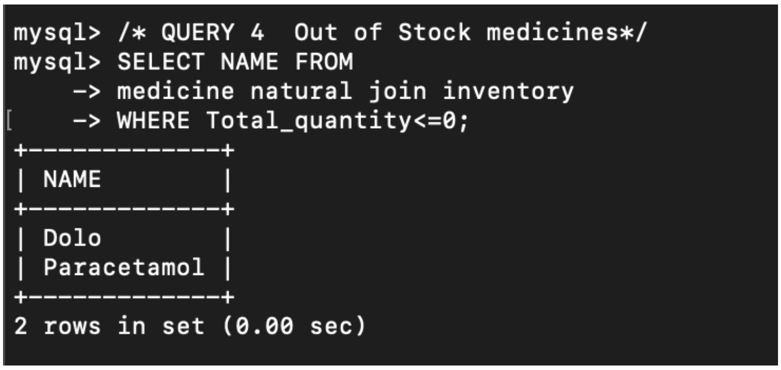
```
mysql>   /* QUERY 1 - TO find the stock level of a Medicine X */
mysql>
mysql>   SELECT NAME,sum(Total_quantity) as Total_quantity FROM
    ->   medicine NATURAL JOIN inventory
    ->   GROUP BY NAME;
+-------------+----------------+
| NAME        | Total_quantity |
+-------------+----------------+
| Avomin      |            682 |
| Azithral    |            230 |
| Brufen      |            400 |
| Cetrizine   |            770 |
| Crocin      |            150 |
| Disprin     |            576 |
| Dolo        |              0 |
| Mephthal    |            892 |
| MontekLC    |            220 |
| Paracetamol |              0 |
+-------------+----------------+
10 rows in set (0.01 sec)
```



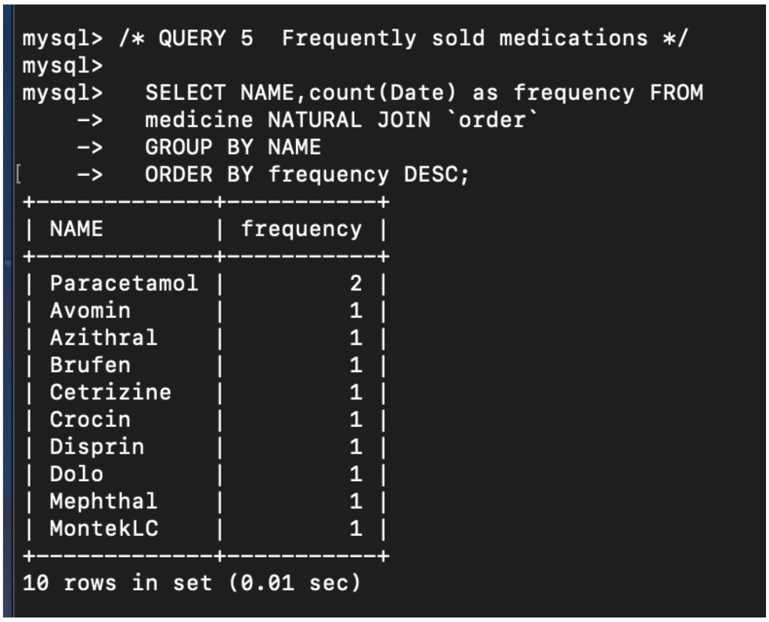
```
mysql> /* QUERY 2: Medicines expiring in next 30 days */
mysql>
mysql> SELECT * FROM medicine
    -> WHERE DATEDIFF(Exp_Date, NOW()) <= 30
    -> AND DATEDIFF(Exp_Date, NOW()) >= 0;
+---------+--------+-------------+-------+-------+------------+
| Name    | Brand  | Medicine_id | Price | Cost  | Exp_Date   |
+---------+--------+-------------+-------+-------+------------+
| Disprin | Biocon | DB0004      | 50.00 | 30.00 | 2023-05-05 |
+---------+--------+-------------+-------+-------+------------+
1 row in set (0.00 sec)
```



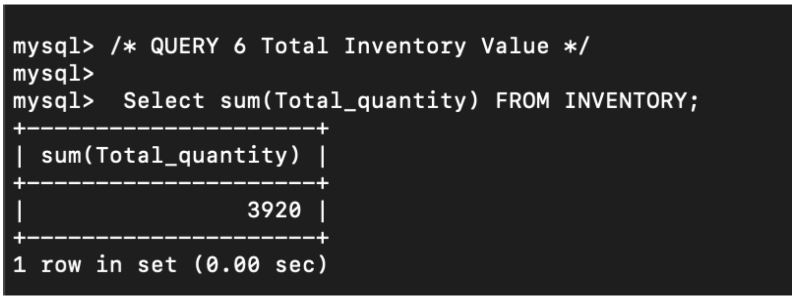
```
mysql> /* QUERY 3: Units of Medicines sold in last 30 days  */
mysql>
mysql> SELECT sum(quantity) from `order`
    -> where DATEDIFF(Date, NOW()) <= 0
    -> AND DATEDIFF(Date, NOW()) >= -30;
+---------------+
| sum(quantity) |
+---------------+
|           605 |
+---------------+
1 row in set (0.00 sec)
```



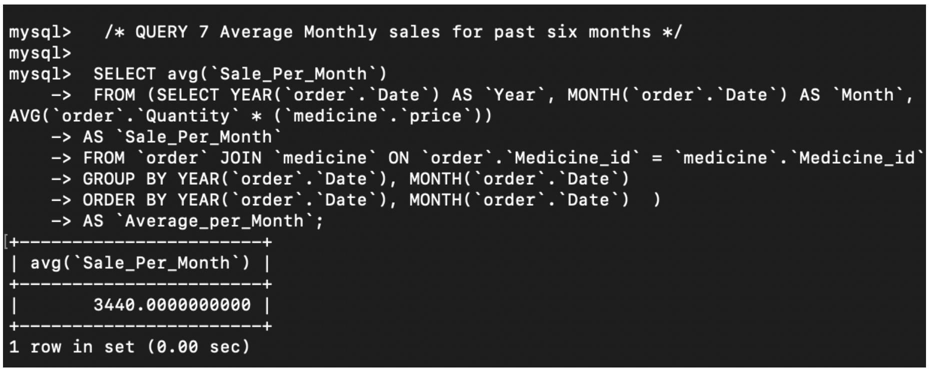
```
mysql> /* QUERY 4  Out of Stock medicines*/
mysql> SELECT NAME FROM
    -> medicine natural join inventory
    -> WHERE Total_quantity<=0;
+------------+
| NAME       |
+------------+
| Dolo       |
| Paracetamol |
+------------+
2 rows in set (0.00 sec)
```



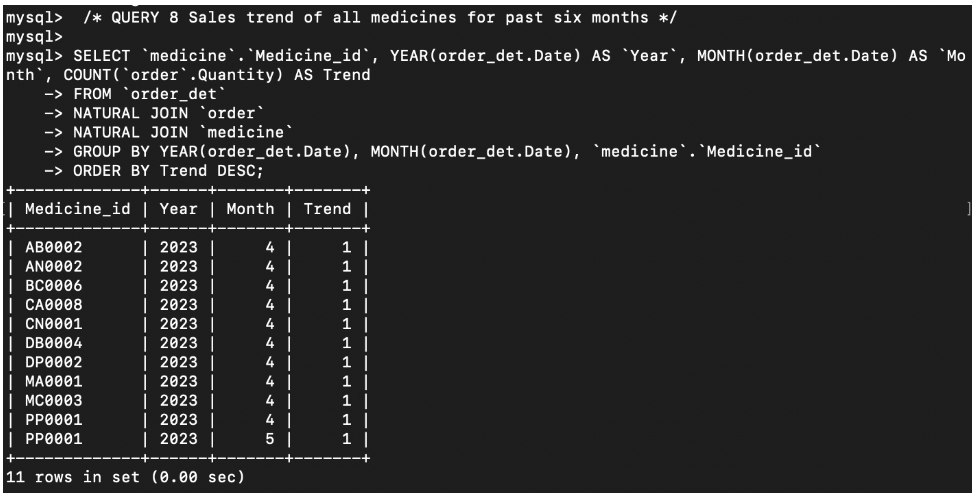
```
mysql> /* QUERY 5  Frequently sold medications */
mysql>
mysql>   SELECT NAME,count(Date) as frequency FROM
    ->   medicine NATURAL JOIN `order`
    ->   GROUP BY NAME
    ->   ORDER BY frequency DESC;
+-------------+-----------+
| NAME        | frequency |
+-------------+-----------+
| Paracetamol |         2 |
| Avomin      |         1 |
| Azithral    |         1 |
| Brufen      |         1 |
| Cetrizine   |         1 |
| Crocin      |         1 |
| Disprin     |         1 |
| Dolo        |         1 |
| Mephthal    |         1 |
| MontekLC    |         1 |
+-------------+-----------+
10 rows in set (0.01 sec)
```



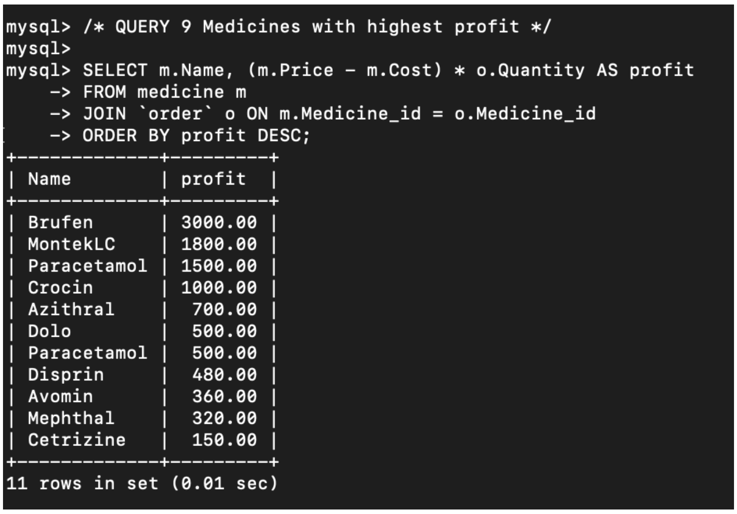
```
mysql> /* QUERY 6 Total Inventory Value */
mysql>
mysql>  Select sum(Total_quantity) FROM INVENTORY;
+---------------------+
| sum(Total_quantity) |
+---------------------+
|                3920 |
+---------------------+
1 row in set (0.00 sec)
```



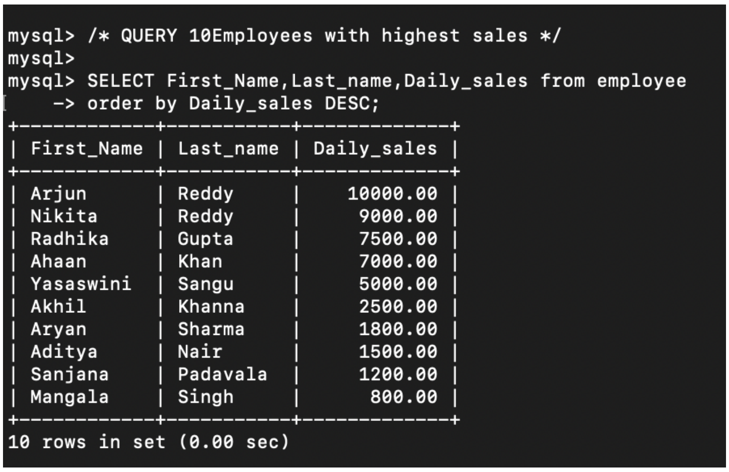
```
mysql>   /* QUERY 7 Average Monthly sales for past six months */
mysql>
mysql>  SELECT avg(`Sale_Per_Month`)
    ->  FROM (SELECT YEAR(`order`.`Date`) AS `Year`, MONTH(`order`.`Date`) AS `Month`,
AVG(`order`.`Quantity` * (`medicine`.`price`))
    -> AS `Sale_Per_Month`
    -> FROM `order` JOIN `medicine` ON `order`.`Medicine_id` = `medicine`.`Medicine_id`
    -> GROUP BY YEAR(`order`.`Date`), MONTH(`order`.`Date`)
    -> ORDER BY YEAR(`order`.`Date`), MONTH(`order`.`Date`)  )
    -> AS `Average_per_Month`;
+-----------------------+
| avg(`Sale_Per_Month`) |
+-----------------------+
|       3440.0000000000 |
+-----------------------+
1 row in set (0.00 sec)
```



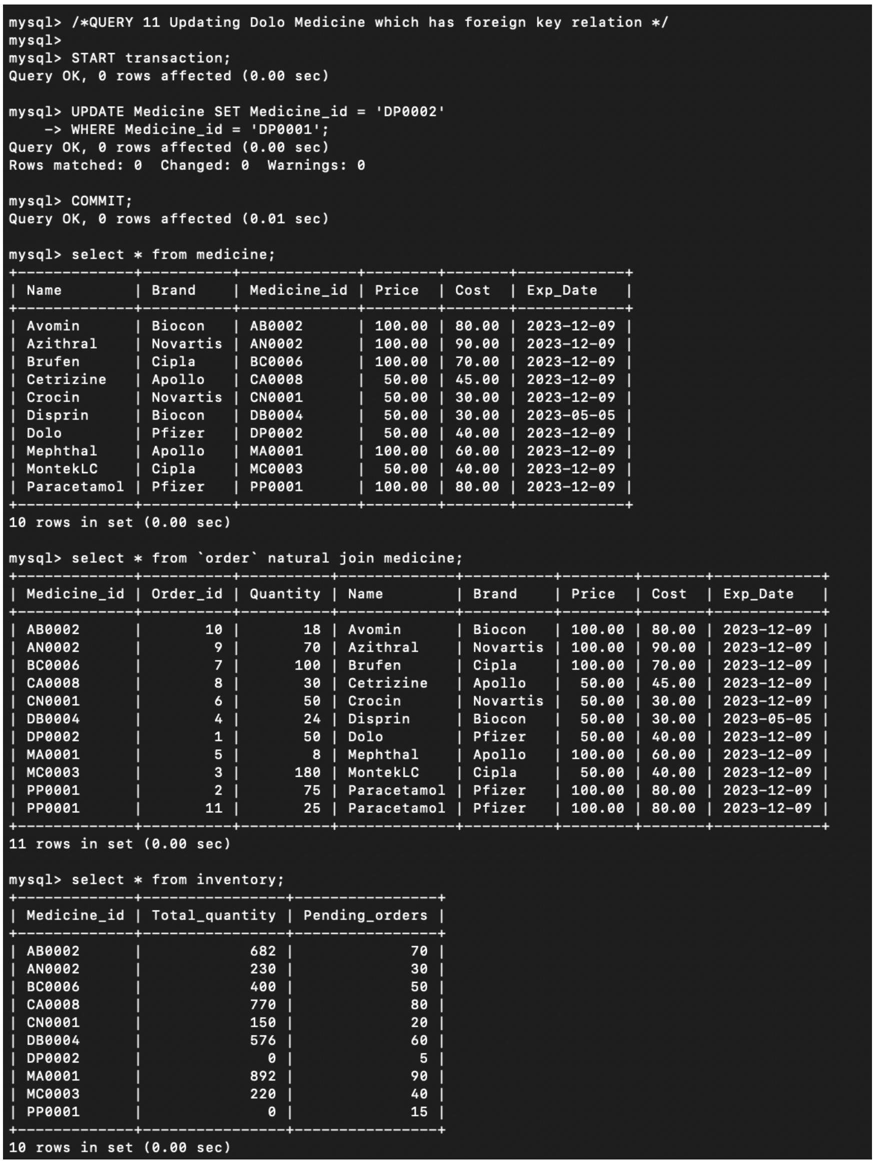
```
mysql>  /* QUERY 8 Sales trend of all medicines for past six months */
mysql>
mysql> SELECT `medicine`.`Medicine_id`, YEAR(order_det.Date) AS `Year`, MONTH(order_det.Date) AS `Mo
nth`, COUNT(`order`.Quantity) AS Trend
    -> FROM `order_det`
    -> NATURAL JOIN `order`
    -> NATURAL JOIN `medicine`
    -> GROUP BY YEAR(order_det.Date), MONTH(order_det.Date), `medicine`.`Medicine_id`
    -> ORDER BY Trend DESC;
+-------------+------+-------+-------+
| Medicine_id | Year | Month | Trend |
+-------------+------+-------+-------+
| AB0002      | 2023 |     4 |     1 |
| AN0002      | 2023 |     4 |     1 |
| BC0006      | 2023 |     4 |     1 |
| CA0008      | 2023 |     4 |     1 |
| CN0001      | 2023 |     4 |     1 |
| DB0004      | 2023 |     4 |     1 |
| DP0002      | 2023 |     4 |     1 |
| MA0001      | 2023 |     4 |     1 |
| MC0003      | 2023 |     4 |     1 |
| PP0001      | 2023 |     4 |     1 |
| PP0001      | 2023 |     5 |     1 |
+-------------+------+-------+-------+
11 rows in set (0.00 sec)
```



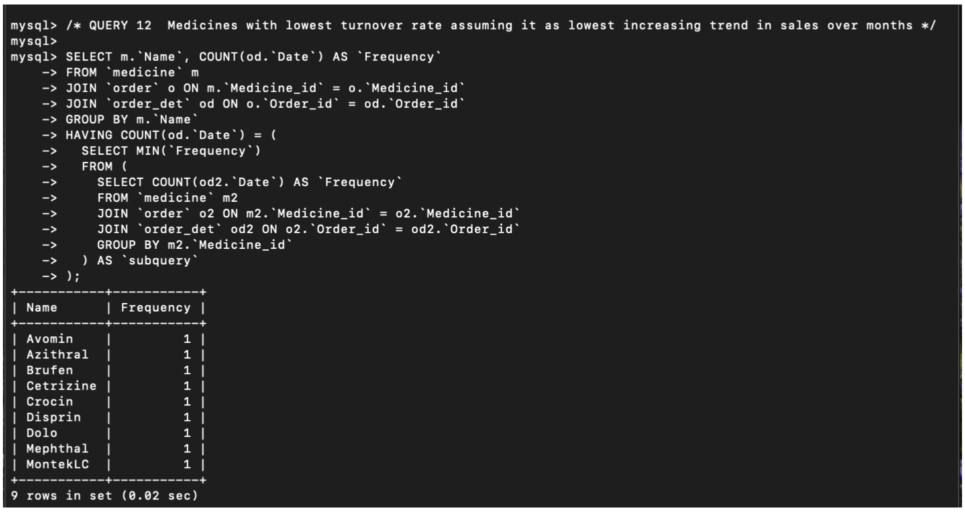
```
mysql> /* QUERY 9 Medicines with highest profit */
mysql>
mysql> SELECT m.Name, (m.Price - m.Cost) * o.Quantity AS profit
    -> FROM medicine m
    -> JOIN `order` o ON m.Medicine_id = o.Medicine_id
    -> ORDER BY profit DESC;
+-------------+---------+
| Name        | profit  |
+-------------+---------+
| Brufen      | 3000.00 |
| MontekLC    | 1800.00 |
| Paracetamol | 1500.00 |
| Crocin      | 1000.00 |
| Azithral    |  700.00 |
| Dolo        |  500.00 |
| Paracetamol |  500.00 |
| Disprin     |  480.00 |
| Avomin      |  360.00 |
| Mephthal    |  320.00 |
| Cetrizine   |  150.00 |
+-------------+---------+
11 rows in set (0.01 sec)
```



```
mysql> /* QUERY 10Employees with highest sales */
mysql>
mysql> SELECT First_Name,Last_name,Daily_sales from employee
    -> order by Daily_sales DESC;
+------------+-----------+-------------+
| First_Name | Last_name | Daily_sales |
+------------+-----------+-------------+
| Arjun      | Reddy     |    10000.00 |
| Nikita     | Reddy     |     9000.00 |
| Radhika    | Gupta     |     7500.00 |
| Ahaan      | Khan      |     7000.00 |
| Yasaswini  | Sangu     |     5000.00 |
| Akhil      | Khanna    |     2500.00 |
| Aryan      | Sharma    |     1800.00 |
| Aditya     | Nair      |     1500.00 |
| Sanjana    | Padavala  |     1200.00 |
| Mangala    | Singh     |      800.00 |
+------------+-----------+-------------+
10 rows in set (0.00 sec)
```



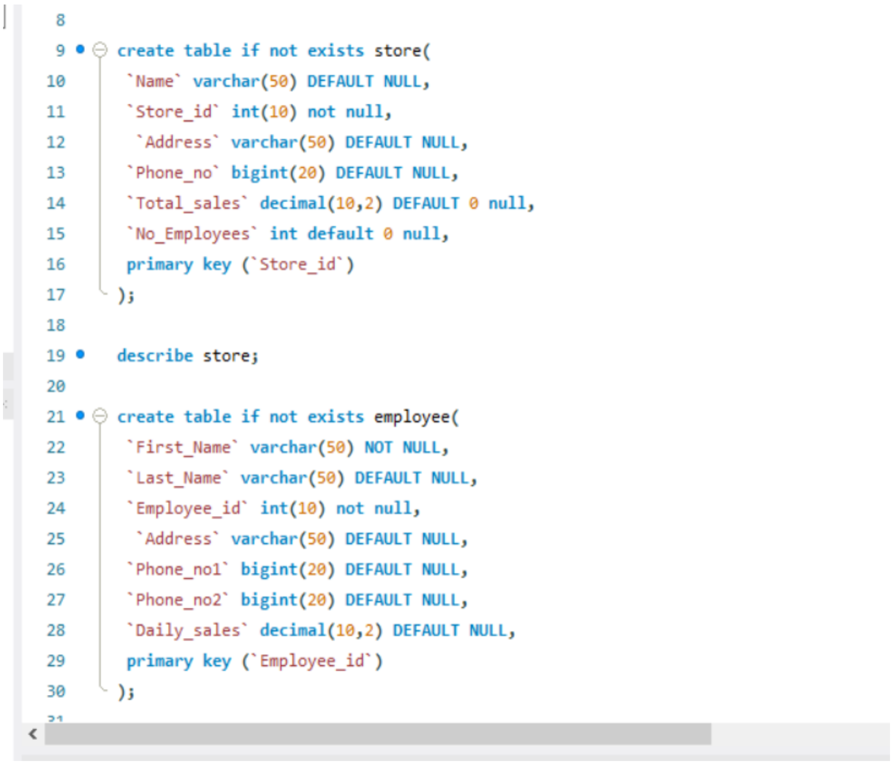
```
mysql> /*QUERY 11 Updating Dolo Medicine which has foreign key relation */
mysql>
mysql> START transaction;
Query OK, 0 rows affected (0.00 sec)

mysql> UPDATE Medicine SET Medicine_id = 'DP0002'
    -> WHERE Medicine_id = 'DP0001';
Query OK, 0 rows affected (0.00 sec)
Rows matched: 0  Changed: 0  Warnings: 0

mysql> COMMIT;
Query OK, 0 rows affected (0.01 sec)

mysql> select * from medicine;
+-------------+----------+-------------+--------+-------+------------+
| Name        | Brand    | Medicine_id | Price  | Cost  | Exp_Date   |
+-------------+----------+-------------+--------+-------+------------+
| Avomin      | Biocon   | AB0002      | 100.00 | 80.00 | 2023-12-09 |
| Azithral    | Novartis | AN0002      | 100.00 | 90.00 | 2023-12-09 |
| Brufen      | Cipla    | BC0006      | 100.00 | 70.00 | 2023-12-09 |
| Cetrizine   | Apollo   | CA0008      |  50.00 | 45.00 | 2023-12-09 |
| Crocin      | Novartis | CN0001      |  50.00 | 30.00 | 2023-12-09 |
| Disprin     | Biocon   | DB0004      |  50.00 | 30.00 | 2023-05-05 |
| Dolo        | Pfizer   | DP0002      |  50.00 | 40.00 | 2023-12-09 |
| Mephthal    | Apollo   | MA0001      | 100.00 | 60.00 | 2023-12-09 |
| MontekLC    | Cipla    | MC0003      |  50.00 | 40.00 | 2023-12-09 |
| Paracetamol | Pfizer   | PP0001      | 100.00 | 80.00 | 2023-12-09 |
+-------------+----------+-------------+--------+-------+------------+
10 rows in set (0.00 sec)

mysql> select * from `order` natural join medicine;
+-------------+----------+----------+-------------+----------+--------+-------+------------+
| Medicine_id | Order_id | Quantity | Name        | Brand    | Price  | Cost  | Exp_Date   |
+-------------+----------+----------+-------------+----------+--------+-------+------------+
| AB0002      |       10 |       18 | Avomin      | Biocon   | 100.00 | 80.00 | 2023-12-09 |
| AN0002      |        9 |       70 | Azithral    | Novartis | 100.00 | 90.00 | 2023-12-09 |
| BC0006      |        7 |      100 | Brufen      | Cipla    | 100.00 | 70.00 | 2023-12-09 |
| CA0008      |        8 |       30 | Cetrizine   | Apollo   |  50.00 | 45.00 | 2023-12-09 |
| CN0001      |        6 |       50 | Crocin      | Novartis |  50.00 | 30.00 | 2023-12-09 |
| DB0004      |        4 |       24 | Disprin     | Biocon   |  50.00 | 30.00 | 2023-05-05 |
| DP0002      |        1 |       50 | Dolo        | Pfizer   |  50.00 | 40.00 | 2023-12-09 |
| MA0001      |        5 |        8 | Mephthal    | Apollo   | 100.00 | 60.00 | 2023-12-09 |
| MC0003      |        3 |      180 | MontekLC    | Cipla    |  50.00 | 40.00 | 2023-12-09 |
| PP0001      |        2 |       75 | Paracetamol | Pfizer   | 100.00 | 80.00 | 2023-12-09 |
| PP0001      |       11 |       25 | Paracetamol | Pfizer   | 100.00 | 80.00 | 2023-12-09 |
+-------------+----------+----------+-------------+----------+--------+-------+------------+
11 rows in set (0.00 sec)

mysql> select * from inventory;
+-------------+----------------+----------------+
| Medicine_id | Total_quantity | Pending_orders |
+-------------+----------------+----------------+
| AB0002      |            682 |             70 |
| AN0002      |            230 |             30 |
| BC0006      |            400 |             50 |
| CA0008      |            770 |             80 |
| CN0001      |            150 |             20 |
| DB0004      |            576 |             60 |
| DP0002      |              0 |              5 |
| MA0001      |            892 |             90 |
| MC0003      |            220 |             40 |
| PP0001      |              0 |             15 |
+-------------+----------------+----------------+
10 rows in set (0.00 sec)
```



```
mysql> /* QUERY 12  Medicines with lowest turnover rate assuming it as lowest increasing trend in sales over months */
mysql>
mysql> SELECT m.`Name`, COUNT(od.`Date`) AS `Frequency`
    -> FROM `medicine` m
    -> JOIN `order` o ON m.`Medicine_id` = o.`Medicine_id`
    -> JOIN `order_det` od ON o.`Order_id` = od.`Order_id`
    -> GROUP BY m.`Name`
    -> HAVING COUNT(od.`Date`) = (
    ->   SELECT MIN(`Frequency`)
    ->   FROM (
    ->     SELECT COUNT(od2.`Date`) AS `Frequency`
    ->     FROM `medicine` m2
    ->     JOIN `order` o2 ON m2.`Medicine_id` = o2.`Medicine_id`
    ->     JOIN `order_det` od2 ON o2.`Order_id` = od2.`Order_id`
    ->     GROUP BY m2.`Medicine_id`
    ->   ) AS `subquery`
    -> );
+-----------+-----------+
| Name      | Frequency |
+-----------+-----------+
| Avomin    |         1 |
| Azithral  |         1 |
| Brufen    |         1 |
| Cetrizine |         1 |
| Crocin    |         1 |
| Disprin   |         1 |
| Dolo      |         1 |
| Mephthal  |         1 |
| MontekLC  |         1 |
+-----------+-----------+
9 rows in set (0.02 sec)
```

Some supplement queries and triggers that have helped ease the process:

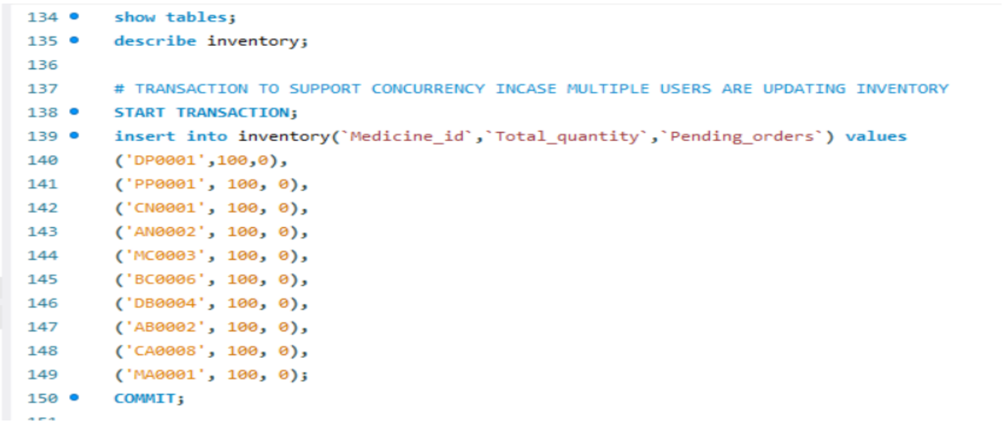
Queries to create tables for entities present in the relational schema.



```

create table if not exists store(
  `Name` varchar(50) DEFAULT NULL,
  `Store_id` int(10) not null,
   `Address` varchar(50) DEFAULT NULL,
  `Phone_no` bigint(20) DEFAULT NULL,
  `Total_sales` decimal(10,2) DEFAULT 0 null,
  `No_Employees` int default 0 null,
  primary key (`Store_id`)
);

describe store;

create table if not exists employee(
  `First_Name` varchar(50) NOT NULL,
  `Last_Name` varchar(50) DEFAULT NULL,
  `Employee_id` int(10) not null,
   `Address` varchar(50) DEFAULT NULL,
  `Phone_no1` bigint(20) DEFAULT NULL,
  `Phone_no2` bigint(20) DEFAULT NULL,
  `Daily_sales` decimal(10,2) DEFAULT NULL,
  primary key (`Employee_id`)
);
```

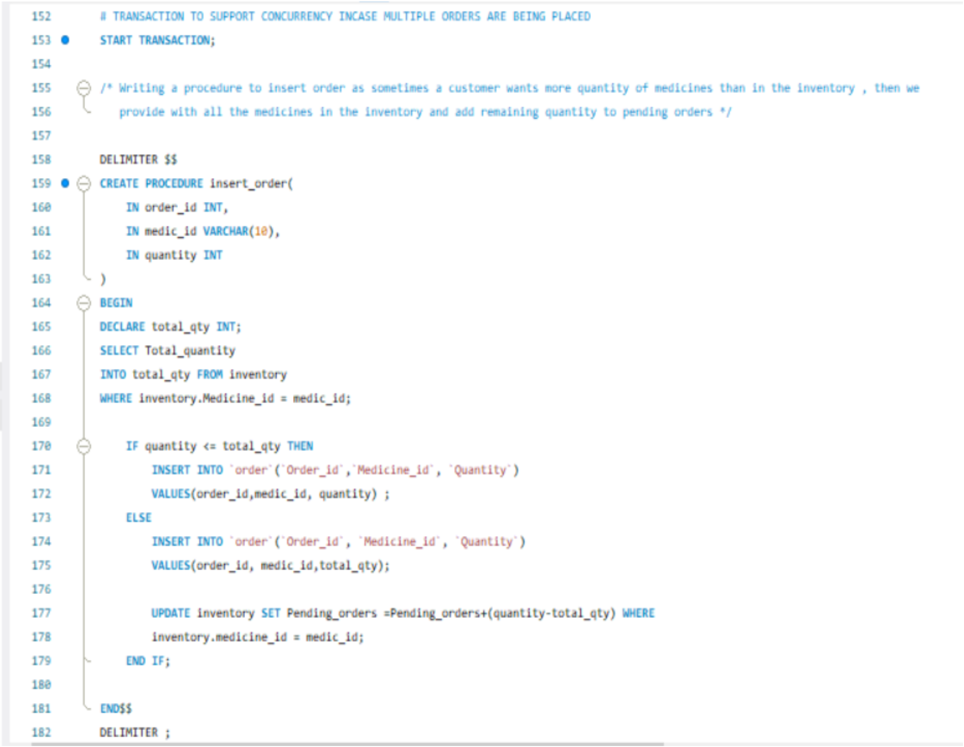
:

Queries to insert data using transactions to allow concurrency and maintain data integrity



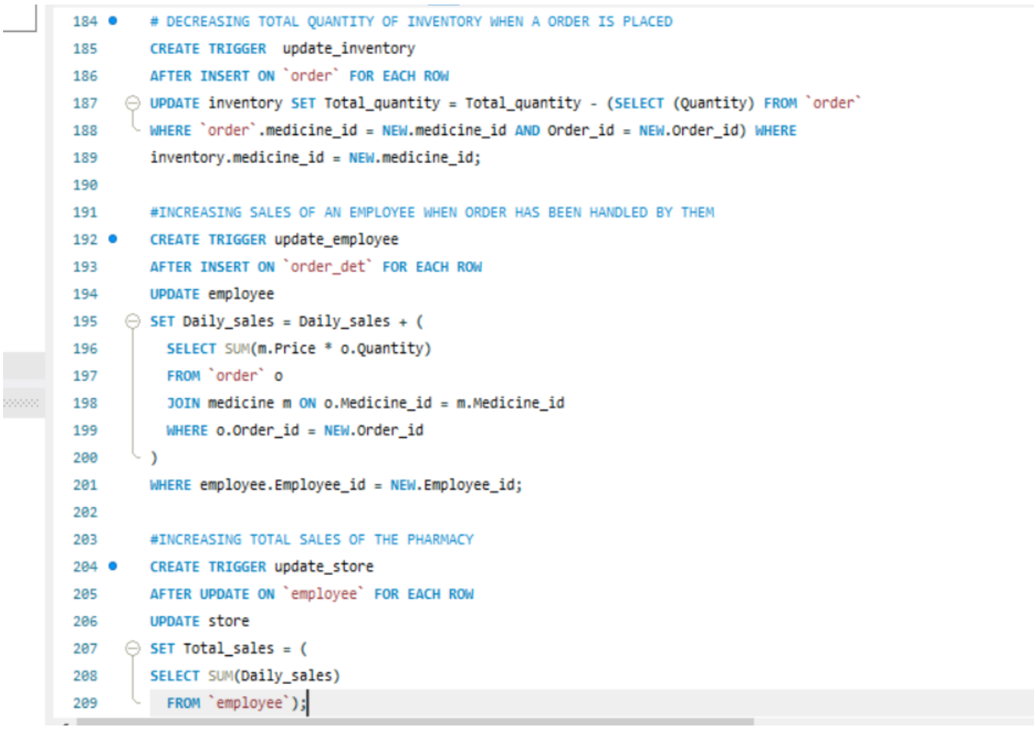
```
show tables;
describe inventory;

# TRANSACTION TO SUPPORT CONCURRENCY INCASE MULTIPLE USERS ARE UPDATING INVENTORY
START TRANSACTION;
insert into inventory(`Medicine_id`,`Total_quantity`,`Pending_orders`) values
('DP0001',100,0),
('PP0001', 100, 0),
('CN0001', 100, 0),
('AN0002', 100, 0),
('MC0003', 100, 0),
('BC0006', 100, 0),
('DB0004', 100, 0),
('AB0002', 100, 0),
('CA0008', 100, 0),
('MA0001', 100, 0);
COMMIT;
```

Procedure to insert order as sometimes we only order some quantity.



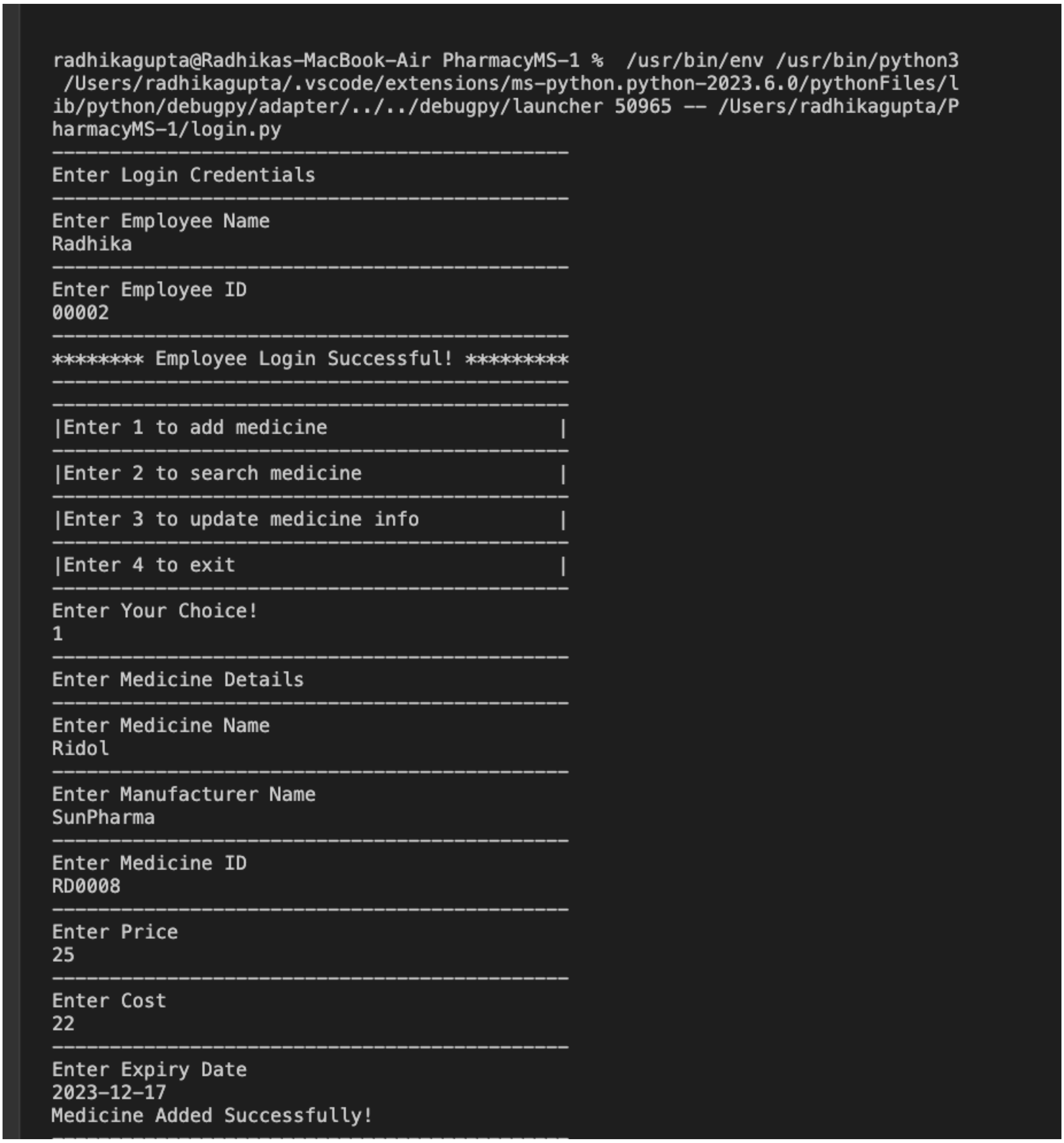
```
# TRANSACTION TO SUPPORT CONCURRENCY INCASE MULTIPLE ORDERS ARE BEING PLACED
START TRANSACTION;

/* Writing a procedure to insert order as sometimes a customer wants more quantity of medicines than in the inventory , then we
   provide with all the medicines in the inventory and add remaining quantity to pending orders */

DELIMITER $$
CREATE PROCEDURE insert_order(
    IN order_id INT,
    IN medic_id VARCHAR(10),
    IN quantity INT
)
BEGIN
DECLARE total_qty INT;
SELECT Total_quantity
INTO total_qty FROM inventory
WHERE inventory.Medicine_id = medic_id;

    IF quantity <= total_qty THEN
        INSERT INTO `order`(`Order_id`,`Medicine_id`, `Quantity`)
        VALUES(order_id,medic_id, quantity) ;
    ELSE
        INSERT INTO `order`(`Order_id`, `Medicine_id`, `Quantity`)
        VALUES(order_id, medic_id,total_qty);

        UPDATE inventory SET Pending_orders =Pending_orders+(quantity-total_qty) WHERE
        inventory.medicine_id = medic_id;
    END IF;

END$$
DELIMITER ;
```

Triggers to automatically update related tables when data is inserted in one table



```sql
# DECREASING TOTAL QUANTITY OF INVENTORY WHEN A ORDER IS PLACED
CREATE TRIGGER  update_inventory
AFTER INSERT ON `order` FOR EACH ROW
UPDATE inventory SET Total_quantity = Total_quantity - (SELECT (Quantity) FROM `order`
WHERE `order`.medicine_id = NEW.medicine_id AND Order_id = NEW.Order_id) WHERE
inventory.medicine_id = NEW.medicine_id;

#INCREASING SALES OF AN EMPLOYEE WHEN ORDER HAS BEEN HANDLED BY THEM
CREATE TRIGGER update_employee
AFTER INSERT ON `order_det` FOR EACH ROW
UPDATE employee
SET Daily_sales = Daily_sales + (
  SELECT SUM(m.Price * o.Quantity)
  FROM `order` o
  JOIN medicine m ON o.Medicine_id = m.Medicine_id
  WHERE o.Order_id = NEW.Order_id
)
WHERE employee.Employee_id = NEW.Employee_id;

#INCREASING TOTAL SALES OF THE PHARMACY
CREATE TRIGGER update_store
AFTER UPDATE ON `employee` FOR EACH ROW
UPDATE store
SET Total_sales = (
SELECT SUM(Daily_sales)
  FROM `employee`);
```

<u>FRONTEND:</u>



```
radhikagupta@Radhikas-MacBook-Air PharmacyMS-1 %  /usr/bin/env /usr/bin/python3
 /Users/radhikagupta/.vscode/extensions/ms-python.python-2023.6.0/pythonFiles/l
ib/python/debugpy/adapter/../../debugpy/launcher 50965 -- /Users/radhikagupta/P
harmacyMS-1/login.py
---------------------------------------------
Enter Login Credentials
---------------------------------------------
Enter Employee Name
Radhika
---------------------------------------------
Enter Employee ID
00002
---------------------------------------------
******** Employee Login Successful! *********
---------------------------------------------
---------------------------------------------
|Enter 1 to add medicine                    |
---------------------------------------------
|Enter 2 to search medicine                 |
---------------------------------------------
|Enter 3 to update medicine info            |
---------------------------------------------
|Enter 4 to exit                            |
---------------------------------------------
Enter Your Choice!
1
---------------------------------------------
Enter Medicine Details
---------------------------------------------
Enter Medicine Name
Ridol
---------------------------------------------
Enter Manufacturer Name
SunPharma
---------------------------------------------
Enter Medicine ID
RD0008
---------------------------------------------
Enter Price
25
---------------------------------------------
Enter Cost
22
---------------------------------------------
Enter Expiry Date
2023-12-17
Medicine Added Successfully!
---------------------------------------------
```