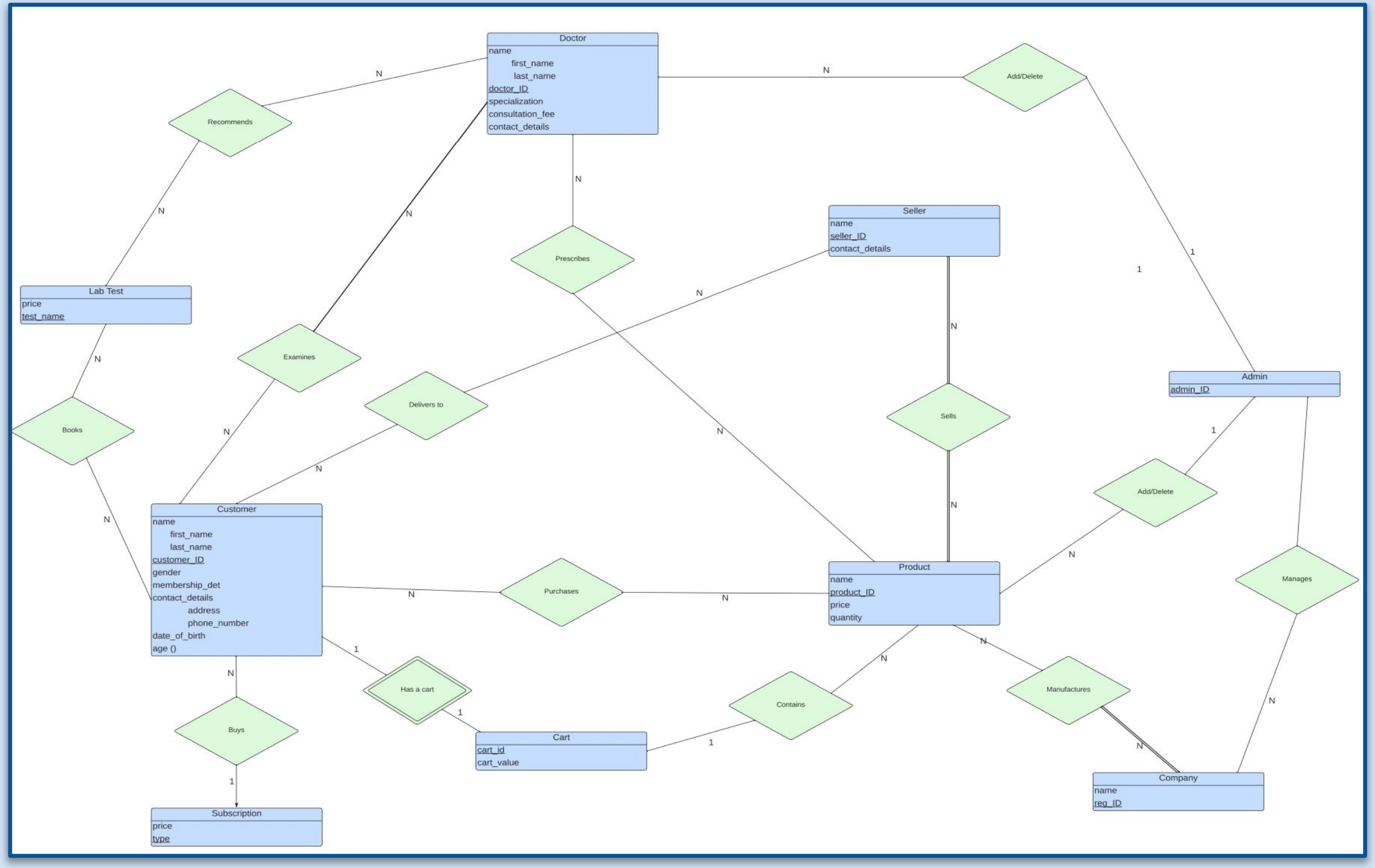
ONLINE PHARMACY STORE

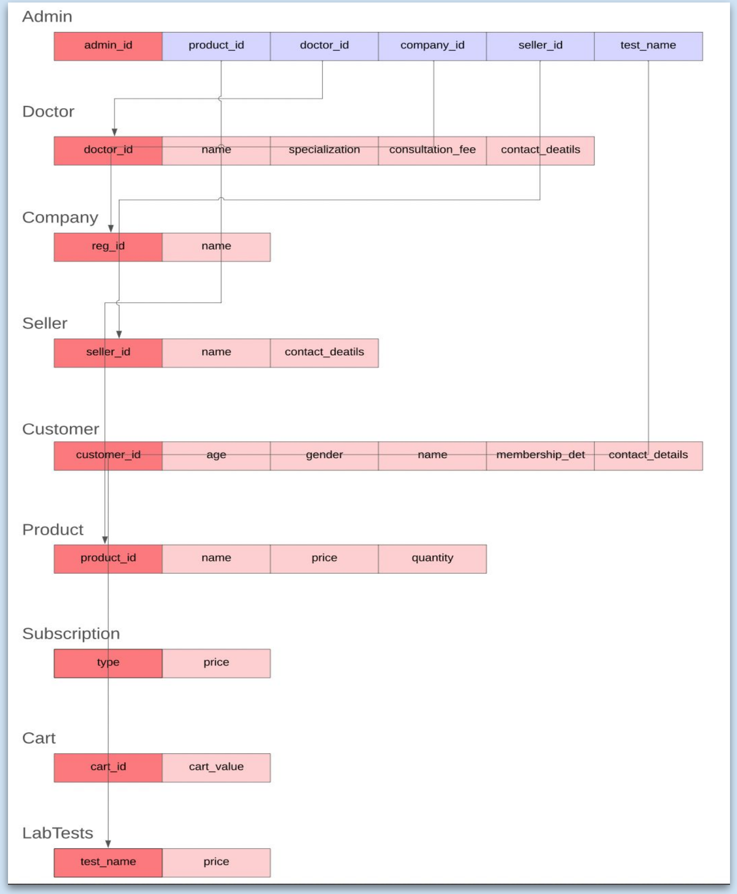
DBMS PROJECT BY-

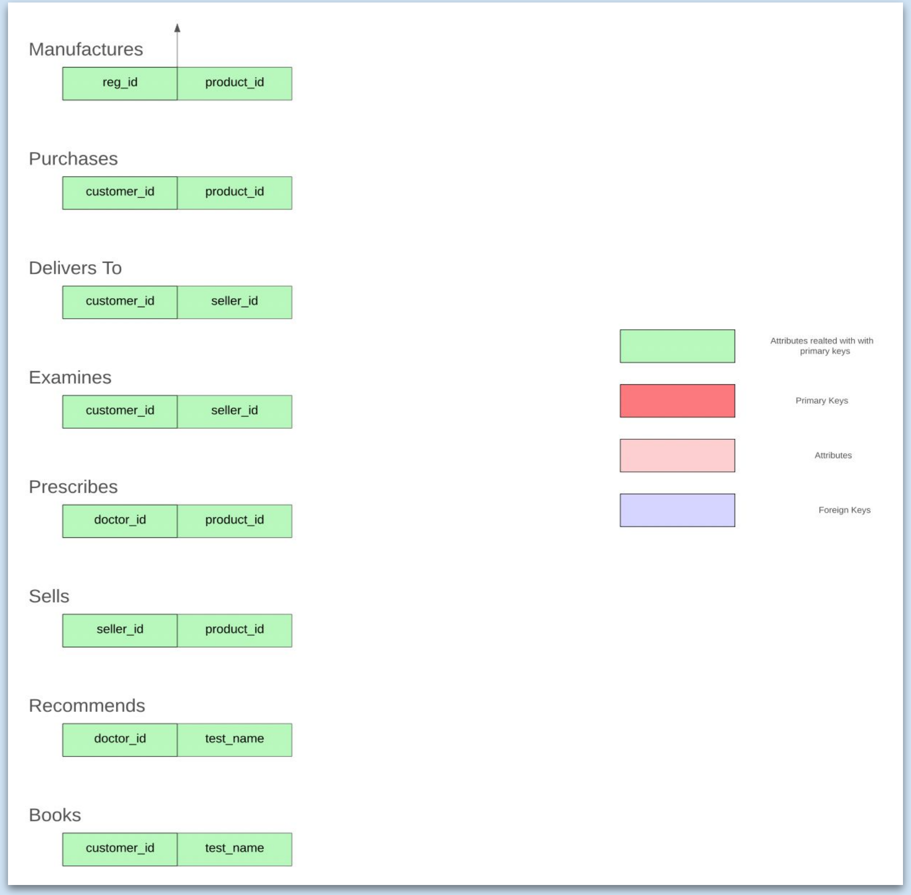
PRIYA YADAV (2021272) & PULKIT NARGOTRA (2021273)

E-R Diagram


Doctor
name
first_name
last_name
doctor_ID
specialization
consultation_fee
contact_details
Add/Delete
Recommends
Prescribes
Seller
name
seller_ID
contact_details
Lab Test
price
test_name
Examines
Delivers to
Sells
Admin
admin_ID
Books
Add/Delete
Customer
name
first_name
last_name
customer_ID
gender
membership_det
contact_details
address
phone_number
date_of_birth
age ()
Purchases
Product
name
product_ID
price
quantity
Manages
Has a cart
Contains
Manufactures
Buys
Cart
cart_id
cart_value
Company
name
reg_ID
Subscription
price
type

Relational Diagram





Tables

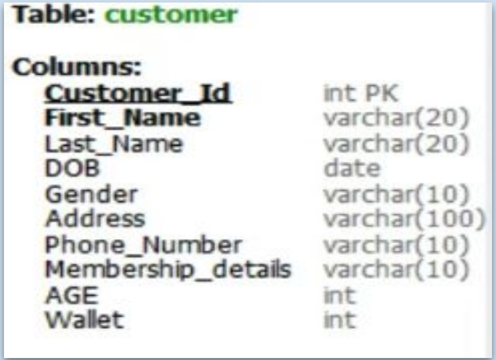


Table: customer

Columns:

Customer_Id	int PK
First_Name	varchar(20)
Last_Name	varchar(20)
DOB	date
Gender	varchar(10)
Address	varchar(100)
Phone_Number	varchar(10)
Membership_details	varchar(10)
AGE	int
Wallet	int

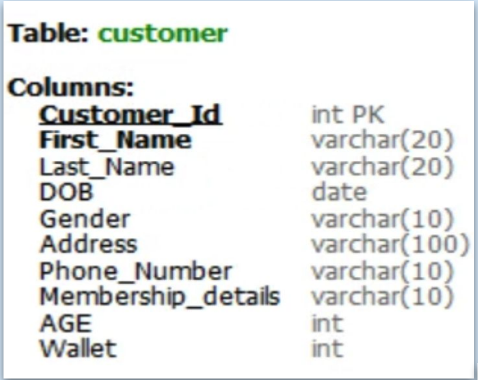


Table: customer

Columns:

Customer_Id	int PK
First_Name	varchar(20)
Last_Name	varchar(20)
DOB	date
Gender	varchar(10)
Address	varchar(100)
Phone_Number	varchar(10)
Membership_details	varchar(10)
AGE	int
Wallet	int

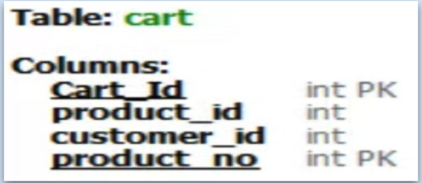


Table: cart

Columns:

Cart_Id	int PK
product_id	int
customer_id	int
product_no	int PK

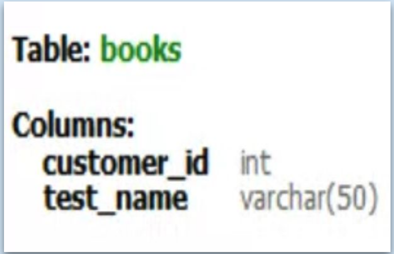


Table: books

Columns:

customer_id	int
test_name	varchar(50)

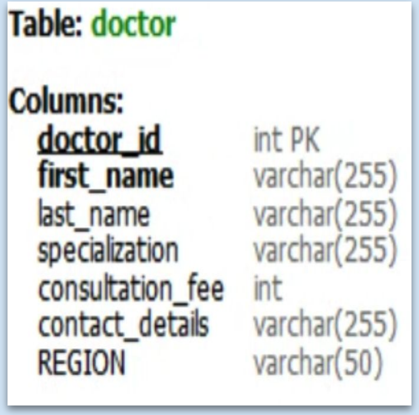


Table: doctor

Columns:

doctor_id	int PK
first_name	varchar(255)
last_name	varchar(255)
specialization	varchar(255)
consultation_fee	int
contact_details	varchar(255)
REGION	varchar(50)

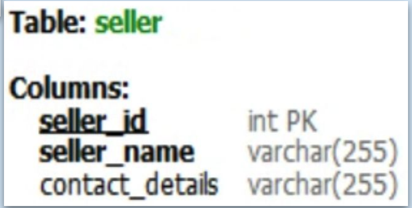


Table: seller

Columns:

seller_id	int PK
seller_name	varchar(255)
contact_details	varchar(255)

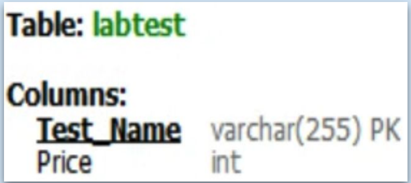


Table: labtest

Columns:

Test_Name	varchar(255) PK
Price	int

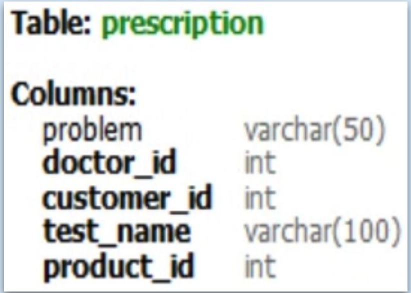


Table: prescription

Columns:

problem	varchar(50)
doctor_id	int
customer_id	int
test_name	varchar(100)
product_id	int

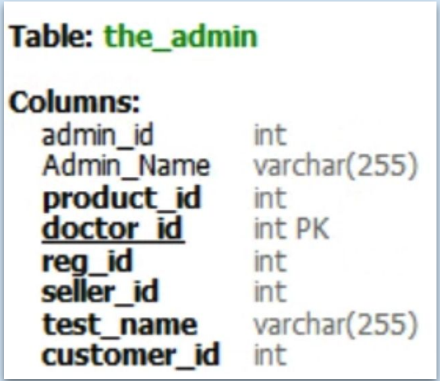


Table: the_admin

Columns:

admin_id	int
Admin_Name	varchar(255)
product_id	int
doctor_id	int PK
reg_id	int
seller_id	int
test_name	varchar(255)
customer_id	int

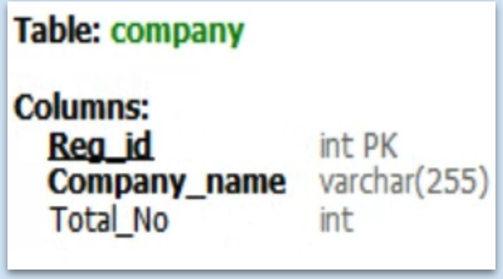


Table: company

Columns:

Reg_id	int PK
Company_name	varchar(255)
Total_No	int

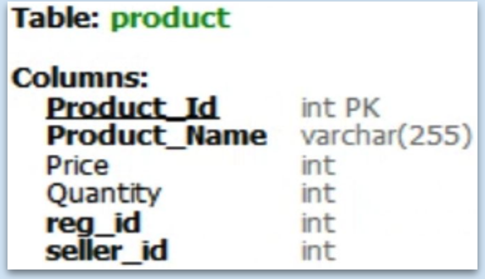


Table: product

Columns:

Product_Id	int PK
Product_Name	varchar(255)
Price	int
Quantity	int
reg_id	int
seller_id	int

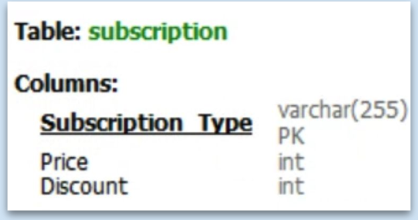


Table: subscription

Columns:

Subscription_Type	varchar(255) PK
Price	int
Discount	int

SQL Queries

```
/* QUERY1 - SELECTION
shortlisting doctor with specialization in dermatology and fee<400*/
SELECT * FROM DOCTOR
WHERE specialization="Dermatologist" AND consultation_fee <500;

/* QUERY2 - PROJECTION
Displaying the membership details of customers with their id,name and phone no*/
Select customer_id,first_name,last_name,phone_number,membership_details from customer;

/* QUERY3 - RENAME
Displaying the membership details of customers with their id,name and phone no*/
select * from customer;
select CUSTOMER_ID,first_NAME,last_name,PHONE_NUMBER
FROM CUSTOMER
WHERE CUSTOMER.MEMBERSHIP_DETAILS="Prime";

/* QUERY4- JOIN CART AND CUSTOMER */
SELECT
customer.customer_id,customer.first_name,customer.last_name,customer.phone_number,customer.
address,cart.cart_id,cart.product_id
FROM customer
JOIN cart on cart.customer_id=customer.customer_id;
```

SQL Queries

```
/* QUERY5- CART VALUE */
DROP table temp_table;
CREATE TABLE temp_table AS
SELECT cart.customer_id,cart.cart_id,cart.product_id, product.price
FROM cart
join product on cart.product_id=product.product_id;
select * from temp_table;

/* QUERY6  - DIsPLAYING CART VALUE WITH CART_ID */
CREATE TABLE finalcart_value AS
SELECT cart_id,SUM(price) AS Total_Price
FROM temp_table
GROUP BY cart_id;
select * from finalcart_value;

/* QUERY7 - UPDATING PRODUCT TABLE BY ADDING SELLER_IDS FOR SOME PRODUCTS */
UPDATE product
set seller_id=1
where product_id in (13,23,48,50);
select * from product;

/* QUERY8  - DISPLAYING DETAILS OF ALL SELLERS ASSOCIATED WITH A CUSTOMER */
CREATE TABLE sellerdetails_table AS
SELECT cart.cart_id,cart.product_id,cart.customer_id,
product.seller_id,seller.seller_name,seller.contact_details
FROM cart
join product on cart.product_id=product.product_id
join seller on product.seller_id=seller.seller_id;
select * from sellerdetails_table;
```

SQL Queries

```
/* QUERY9 - CREATION AND INSERTION IN A TABLE */
CREATE table prescription
( problem varchar(50),
doctor_id int,
customer_id int,
test_name varchar(100),
product_id int,
foreign key (doctor_id) references doctor(doctor_id),
foreign key (customer_id) references customer(customer_id),
foreign key (test_name) references labtest(test_name),
foreign key (product_id) references product(product_id));
INSERT INTO prescription
VALUES("Fever",10,1,"T1",13),
("Fever",10,1,null,48);
select * FROM PRESCRIPTION;

/* QUERY10  - DISPLAYING THE PRODUCTS RECOMMENDED BY DOCTOR THAT WERE BOUGHT BY CUSTOMER */
/*DROP TABLE inner_join;*/
CREATE TABLE inner_join as
SELECT prescription.customer_id,prescription.product_id,cart.cart_id
FROM prescription
INNER JOIN cart ON cart.customer_id=prescription.customer_id AND
cart.product_id=prescription.product_id;
/*select * from inner_join;
select * from labtest;*/
```

SQL Queries

```
/* QUERY11  - DISPLAYING THE LABTESTS RECOMMENDED BY DOCTOR THAT WERE BOOKED BY CUSTOMER */
CREATE TABLE inner_join1 as
SELECT prescription.customer_id,prescription.test_name
FROM prescription
INNER JOIN books ON books.customer_id=prescription.customer_id AND
books.test_name=prescription.test_name;
select * from inner_join1;

/* QUERY12  - SOME BASIC QUERIES */
ALTER TABLE SUBSCRIPTION
DROP DISOCUNT;
ALTER TABLE SUBSCRIPTION
ADD Discount int;
insert into subscription
values("VIP",3000,20);
UPDATE SUBSCRIPTION
SET discount=10
where subscription_type="prime";
```

SQL Queries

```
/* QUERY13 - CHECKOUT CART */
CREATE TABLE Checkout_Cart AS
SELECT
customer.customer_id,customer.membership_details,finalcart_value.cart_id,finalcart_value.to
tal_price, subscription.subscription_type,subscription.discount
FROM finalcart_value
JOIN customer on customer.customer_id=finalcart_value.cart_id
join subscription on customer.membership_details=subscription.subscription_type;
select * from checkout_cart;

/* QUERY14 - DISPLAYING CART_ID WITH FINAL PRICE/VALUE OF THE CART AFTER CONSIDERING
MEMBERSHIP */
select cart_id,ROUND((total_price-(total_price*discount/100)),2) as discounted_price
from checkout_cart;
```

TRIGGERS IN SQL

```
/* TRIGGER 1 - CREDIT FOR COMPANIES AUTOMATICALLY SET TO THEIR NO+10 */
create trigger credit
before insert
on Company
for each row
set new.total_no = new.total_no + 10;
/* insert into company
values(101,"Delmed",101); */

/* TRIGGER 2 - UPDATING DOCTOR CONSULTATION FEE */
create trigger salary_diff
before update on Doctor
for each row
set new.consultation_fee = old.consultation_fee + 100;
```

OLAP QUERIES

```
/* 1) Displays patients attended by a doctor */
select doctor_id, count(customer_id) as No_of_Patients
from prescription
group by doctor_id ;

/* 2) Displays minimum fees of a doctor grouped on the basis of region and specialization */
select specialization,region, min(consultation_fee)
from doctor1
group by specialization,region  with rollup;
```

/* 3) Products sold by each seller and their total profit */

```
SELECT SELLER_ID, COUNT(PRODUCT_ID) AS No_of_Products, SUM(PRICE) AS Total_Profit
FROM PRODUCT
GROUP BY SELLER_ID WITH ROLLUP;
```

/* 4) DISPLAYING TOP 5 COMPANIES WITH MAX NO OF PRODUCTS AND THEIR PROFITS RESPECTIVELY */

```
SELECT REG_ID , COUNT(PRODUCT_ID) AS TOTAL_PRODUCTS, sum(price) AS TOTTAL_PROFIT
FROM PRODUCT
GROUP BY REG_ID
limit 5;
```

Non-Conflicting Transactions

```
BEGIN
          INSERT INTO BOOKS VALUES (1,'A');
          update customer set wallet=100 where customer_id=1;
COMMIT
```

```
BEGIN
          UPDATE customer SET wallet = 500 WHERE customer_id =1;
         UPDATE Product JOIN cart on cart.product_id=product.product_id set
     product.quantity=product.quantity-1 where     cart_id=1;
COMMIT
```

```
BEGIN
    UPDATE customer SET wallet = wallet + 200 WHERE customer_id = 1;
COMMIT
```

```
BEGIN
    INSERT INTO COMPANY VALUES(1,'Apollo','abc');
COMMIT
```

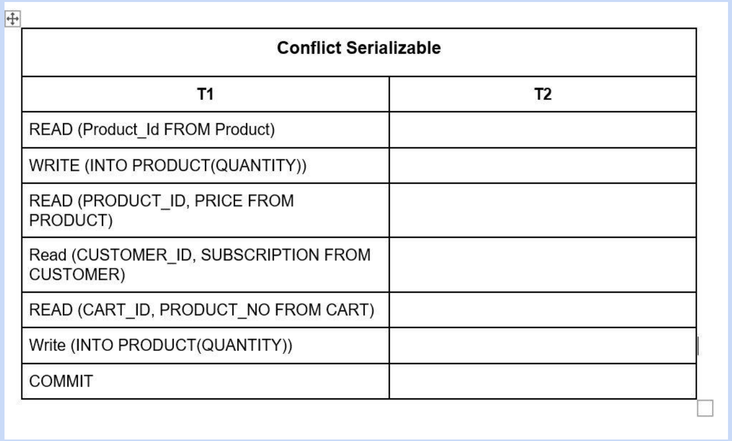
```
BEGIN
    SELECT ROUND((total_price-(total_price*discount/100)),2) FROM checkout_cart WHERE
cart_id =1;
    SELECT wallet FROM customer WHERE customer_id=1;
    UPDATE customer SET wallet = 100 WHERE customer_id =1;
COMMIT
```

Conflicting Transactions

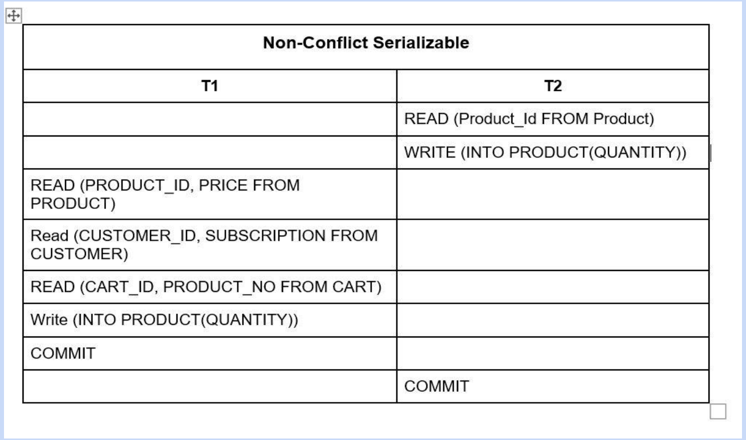
```
BEGIN
                UPDATE PRODUCT SET QUANTITY = QUANTITY+2 WHERE PRODUCT_ID = 1;
BEGIN
                   UPDATE PRODUCT SET QUANTITY = QUANTITY-2 WHERE PRODUCT_ID = 1;
COMMIT
COMMIT
```

```
BEGIN
                UPDATE customer SET wallet = 100 WHERE customer_id =1;
BEGIN
                UPDATE customer SET wallet = wallet + 50 WHERE customer_id = 1;

COMMIT
COMMIT
```

CONFLICT SERIALISABLE AND NON-CONFLICT SERIALISABLE



Conflict Serializable	
T1	T2
READ (Product_Id FROM Product)	
WRITE (INTO PRODUCT(QUANTITY))	
READ (PRODUCT_ID, PRICE FROM PRODUCT)	
Read (CUSTOMER_ID, SUBSCRIPTION FROM CUSTOMER)	
READ (CART_ID, PRODUCT_NO FROM CART)	
Write (INTO PRODUCT(QUANTITY))	
COMMIT	

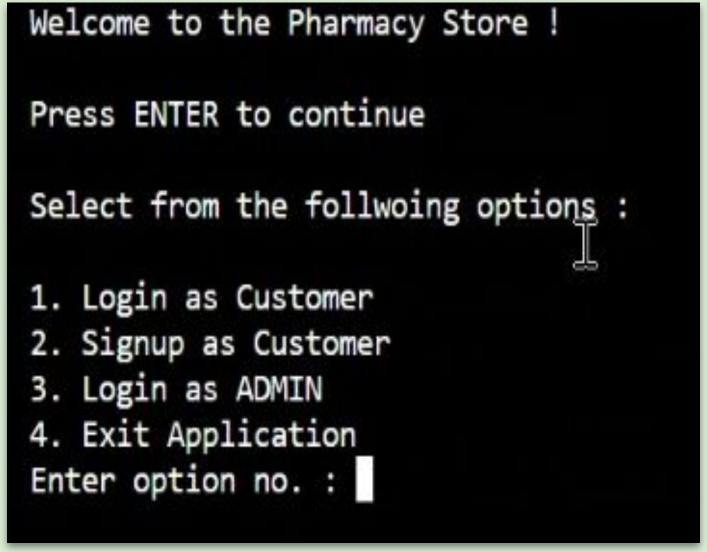


Non-Conflict Serializable	
T1	T2
	READ (Product_Id FROM Product)
	WRITE (INTO PRODUCT(QUANTITY))
READ (PRODUCT_ID, PRICE FROM PRODUCT)	
Read (CUSTOMER_ID, SUBSCRIPTION FROM CUSTOMER)	
READ (CART_ID, PRODUCT_NO FROM CART)	
Write (INTO PRODUCT(QUANTITY))	
COMMIT	
	COMMIT

In this we have considered two transactions one in which the quantity of product is being updated and another in which a customer buys the product due to which it’s quantity decreases.When it occurs in conflict serialisable we do not face any issue but when it occurs in non conflict serialisable we see that the write after write condition occurs in which the quantity is not updated before the customer checks out.

FLOW OF THE CODE

1

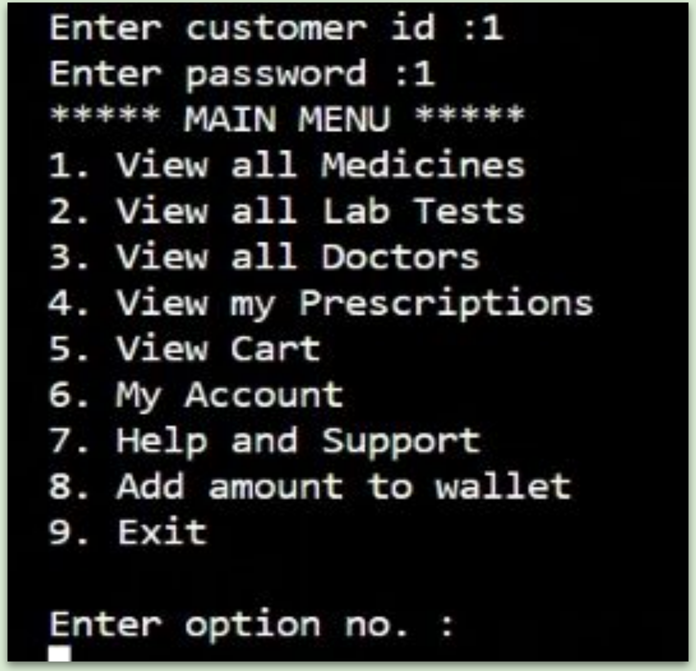


```
Welcome to the Pharmacy Store !

Press ENTER to continue

Select from the follwoing options :

1. Login as Customer
2. Signup as Customer
3. Login as ADMIN
4. Exit Application
Enter option no. :
```

This displays the starting point of the CLI.

Here the user is provided with options where they can login or sign up as a customer or login as the admin.

2

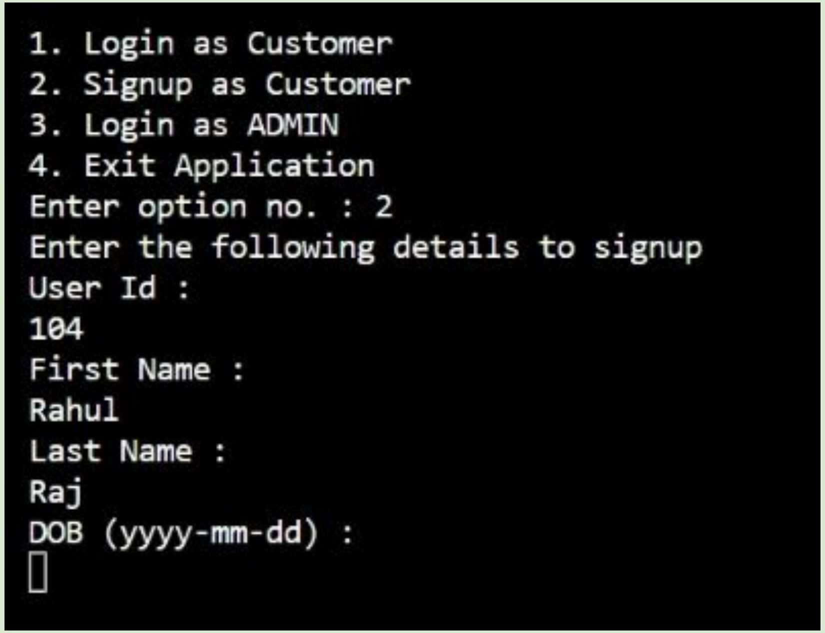


```
Enter customer id :1
Enter password :1
***** MAIN MENU *****
1. View all Medicines
2. View all Lab Tests
3. View all Doctors
4. View my Prescriptions
5. View Cart
6. My Account
7. Help and Support
8. Add amount to wallet
9. Exit

Enter option no. :
```

Login as Customer -

After entering the required credentials the customer is shown the main menu from where they can choose and perform various tasks like adding medicines to cart under view medicines option, booking appointment with doctor under view all doctors, accessing their account, etc.

3

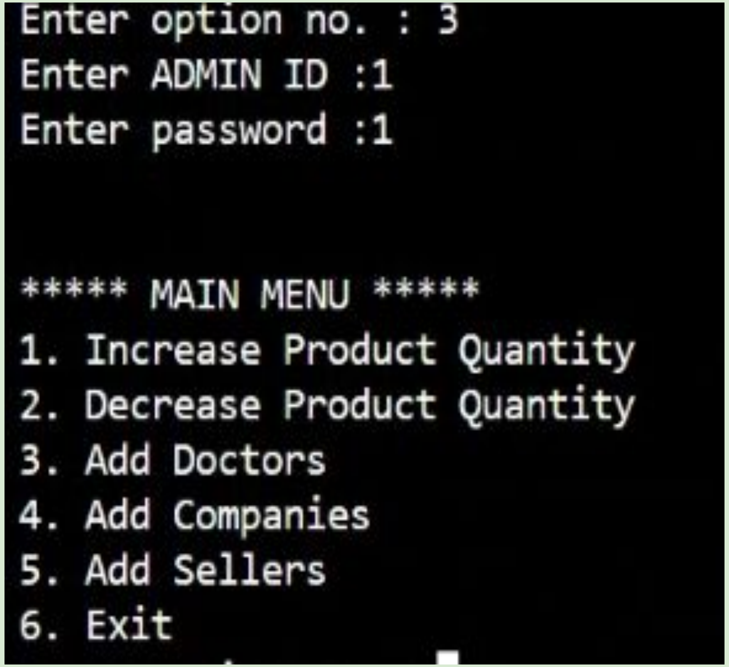


```
1. Login as Customer
2. Signup as Customer
3. Login as ADMIN
4. Exit Application
Enter option no. : 2
Enter the following details to signup
User Id :
104
First Name :
Rahul
Last Name :
Raj
DOB (yyyy-mm-dd) :
```

Sign up as Customer -

This option asks the customer to provide all necessary information and then sign up as a new customer. The customer is required to fill all necessary details like user id, name, contact details, etc.

4



```
Enter option no. : 3
Enter ADMIN ID :1
Enter password :1


***** MAIN MENU *****
1. Increase Product Quantity
2. Decrease Product Quantity
3. Add Doctors
4. Add Companies
5. Add Sellers
6. Exit
```

Login as Admin -

This option allows the administrator to login and gives access to all relevant functions like altering product quantities, adding new companies, adding new sellers, etc.