User Guide

Group 20

This user guide provides comprehensive instructions for setting up, running, and using the OPMS demo application. The application is a Java-based console program that interacts with an Oracle database to manage pharmacy operations, including user roles (customer, doctor, employee, admin) and various functionalities like viewing prescriptions, placing orders, issuing prescriptions, processing orders, and generating reports.

1. Introduction

The OPMS is a role-based system for managing an online pharmacy. It supports:

a) Customers: Register, view prescriptions, place orders.
b) Doctors: Issue prescriptions.
c) Employees: Process orders.
d) Admins: Generate reports (e.g., revenue, stock, top performers).

The system uses an Oracle database backend and a Java frontend for console interaction. Security features include role-based permissions and basic authentication (e.g., SSN/password for customers, ID/phone for doctors/employees). Test data is provided for demonstration.

2. System Requirements

a) Database: Oracle Database. Access to a schema with privileges to create tables and insert data.
b) Java: JDK 8 or higher.
c) Development Tools: Any IDE.
d) OS: Windows, macOS, or Linux (clear screen functionality adapts to OS).
e) Dependencies: Oracle JDBC JAR (e.g., ojdbc8.jar) in class path.

3. Setting Up the Database

To set up the database:

a) **Connect to Oracle:**
You should open the Command Prompt, *cd* to the directory where you store the source code file, and then enter the following command:

```
sqlplus
```

Then log in with your Oracle credentials.

b) **Create Tables:**
You should run the *create_schema.sql* script by typing:

create_schema.sql

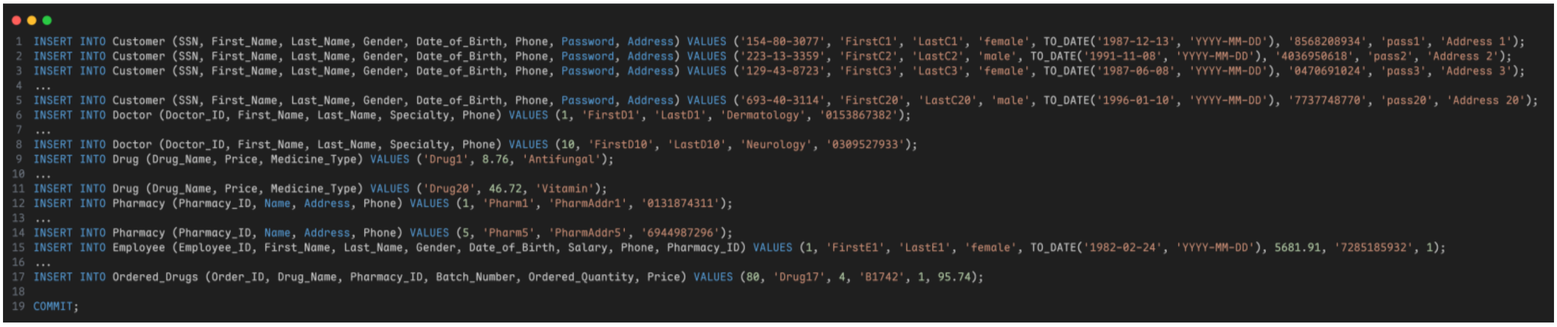
This script defines the schema using *CREATE TABLE* statements. Primary keys ensure uniqueness, foreign keys enforce referential integrity (e.g., *ON DELETE CASCADE* propagates deletions), and *CHECK* constraints validate data (e.g., *Gender IN ('male', 'female')*).

c) **Insert Test Data:**

You should run the *testing_data.sql* script by typing:

testing_data.sql

This SQL script will populate tables for testing. Inserts follow table order to satisfy FK constraints (e.g., customers before prescriptions). Data is synthetic (e.g., 'FirstC1', random prices/dates) to simulate real scenarios like expired/low stock. A total of 653 pieces of data will be inserted, and a commit will be automatically made at the end of this script.



```sql
INSERT INTO Customer (SSN, First_Name, Last_Name, Gender, Date_of_Birth, Phone, Password, Address) VALUES ('154-80-3077', 'FirstC1', 'LastC1', 'female', TO_DATE('1987-12-13', 'YYYY-MM-DD'), '8568208934', 'pass1', 'Address 1');
INSERT INTO Customer (SSN, First_Name, Last_Name, Gender, Date_of_Birth, Phone, Password, Address) VALUES ('223-13-3359', 'FirstC2', 'LastC2', 'male', TO_DATE('1991-11-08', 'YYYY-MM-DD'), '4036950618', 'pass2', 'Address 2');
INSERT INTO Customer (SSN, First_Name, Last_Name, Gender, Date_of_Birth, Phone, Password, Address) VALUES ('129-43-8723', 'FirstC3', 'LastC3', 'female', TO_DATE('1987-06-08', 'YYYY-MM-DD'), '0470691024', 'pass3', 'Address 3');
...
INSERT INTO Customer (SSN, First_Name, Last_Name, Gender, Date_of_Birth, Phone, Password, Address) VALUES ('693-40-3114', 'FirstC20', 'LastC20', 'male', TO_DATE('1996-01-10', 'YYYY-MM-DD'), '7737748770', 'pass20', 'Address 20');
INSERT INTO Doctor (Doctor_ID, First_Name, Last_Name, Specialty, Phone) VALUES (1, 'FirstD1', 'LastD1', 'Dermatology', '0153867382');
...
INSERT INTO Doctor (Doctor_ID, First_Name, Last_Name, Specialty, Phone) VALUES (10, 'FirstD10', 'LastD10', 'Neurology', '0309527933');
INSERT INTO Drug (Drug_Name, Price, Medicine_Type) VALUES ('Drug1', 8.76, 'Antifungal');
...
INSERT INTO Drug (Drug_Name, Price, Medicine_Type) VALUES ('Drug20', 46.72, 'Vitamin');
INSERT INTO Pharmacy (Pharmacy_ID, Name, Address, Phone) VALUES (1, 'Pharm1', 'PharmAddr1', '0131874311');
...
INSERT INTO Pharmacy (Pharmacy_ID, Name, Address, Phone) VALUES (5, 'Pharm5', 'PharmAddr5', '6944987296');
INSERT INTO Employee (Employee_ID, First_Name, Last_Name, Gender, Date_of_Birth, Salary, Phone, Pharmacy_ID) VALUES (1, 'FirstE1', 'LastE1', 'female', TO_DATE('1982-02-24', 'YYYY-MM-DD'), 5681.91, '7285185932', 1);
...
INSERT INTO Ordered_Drugs (Order_ID, Drug_Name, Pharmacy_ID, Batch_Number, Ordered_Quantity, Price) VALUES (80, 'Drug17', 4, 'B1742', 1, 95.74);

COMMIT;
```

Some examples of inserting statements. Symbol "…" Represents a large amount of insertion in the middle.

d) **Verify Setup:**

You may use some queries to check, like:

SELECT COUNT(*) FROM Customer;

This should return 20 if test data is loaded.

4. Running the Application

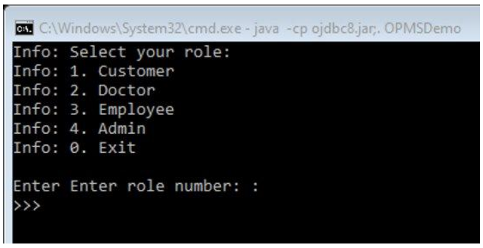
a) **Preparation:**

After setting up the Database, we need to compile the *java* file. You should enter the following command in the Command Prompt:

```
javac -cp ojdbc8.jar OPMSDemo.java
java -cp ojdbc8.jar;. OPMSDemo
```

After we have compiled the source code file, we start the program and have the following login. It is to ask the user to input the "Oracle ID" and "password".

If you see the following message, you have completed all logins. Having fun!



C:\Windows\System32\cmd.exe - java -cp ojdbc8.jar;. OPMSDemo

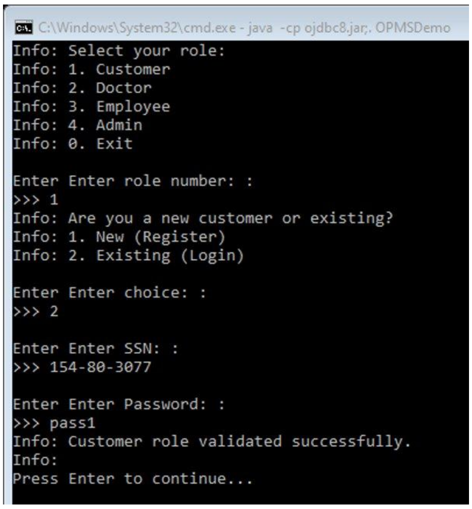
```
Info: Select your role:
Info: 1. Customer
Info: 2. Doctor
Info: 3. Employee
Info: 4. Admin
Info: 0. Exit

Enter Enter role number: :
>>>
```

b) **Using the Application:**
The app is console-based. Press Enter to continue after actions. Exit with 0.

5. Detailed Explanation of SQL Queries

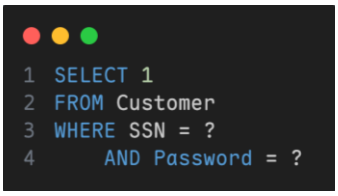
This section explains all SQL from files, including content and Java implementations (where in the *OPMSDemo.java* file).

1) Customer Login Validation (approx. line 93, in *main* method)


C:\Windows\System32\cmd.exe - java -cp ojdbc8.jar;. OPMSDemo

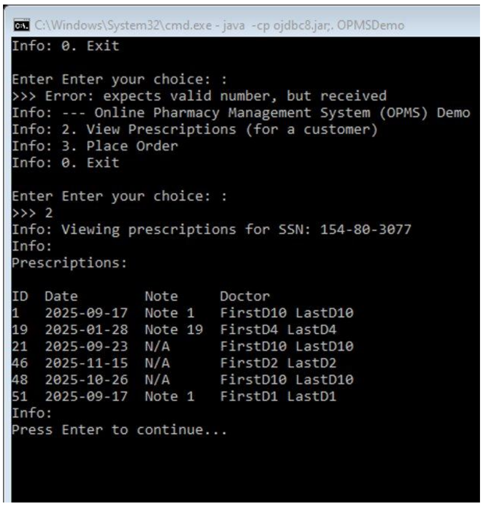
```
Info: Select your role:
Info: 1. Customer
Info: 2. Doctor
Info: 3. Employee
Info: 4. Admin
Info: 0. Exit

Enter Enter role number: :
>>> 1
Info: Are you a new customer or existing?
Info: 1. New (Register)
Info: 2. Existing (Login)

Enter Enter choice: :
>>> 2

Enter Enter SSN: :
>>> 154-80-3077

Enter Enter Password: :
>>> pass1
Info: Customer role validated successfully.
Info:
Press Enter to continue...
```

SQL used:



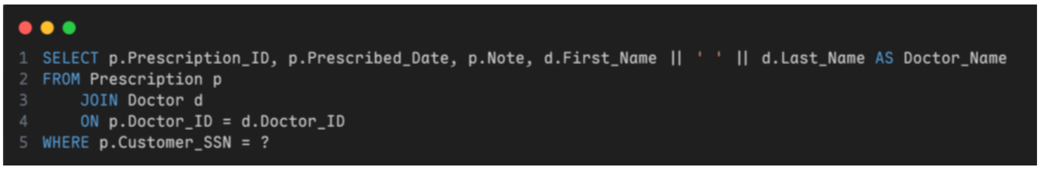
```
SELECT 1
FROM Customer
WHERE SSN = ?
      AND Password = ?
```

2) View Prescriptions (approx. line 347, in *viewPrescriptions* method)


C:\Windows\System32\cmd.exe - java -cp ojdbc8.jar;. OPMSDemo

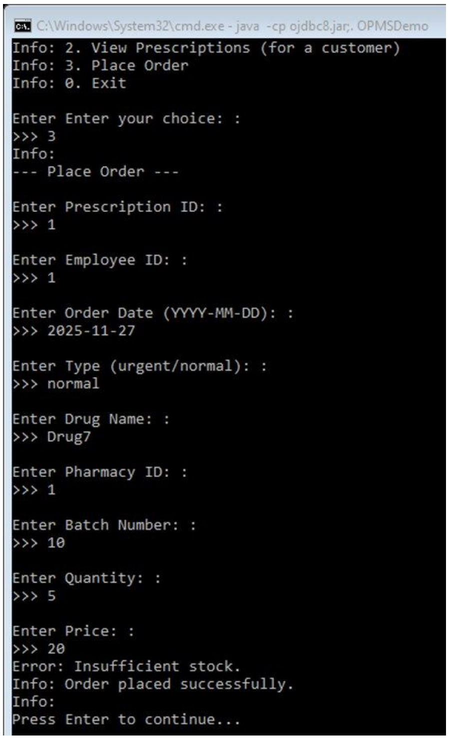
```
Info: 0. Exit

Enter Enter your choice: :
>>> Error: expects valid number, but received
Info: --- Online Pharmacy Management System (OPMS) Demo
Info: 2. View Prescriptions (for a customer)
Info: 3. Place Order
Info: 0. Exit

Enter Enter your choice: :
>>> 2
Info: Viewing prescriptions for SSN: 154-80-3077
Info:
Prescriptions:

ID  Date        Note     Doctor
1   2025-09-17  Note 1   FirstD10 LastD10
19  2025-01-28  Note 19  FirstD4 LastD4
21  2025-09-23  N/A      FirstD10 LastD10
46  2025-11-15  N/A      FirstD2 LastD2
48  2025-10-26  N/A      FirstD10 LastD10
51  2025-09-17  Note 1   FirstD1 LastD1
Info:
Press Enter to continue...
```

SQL used:



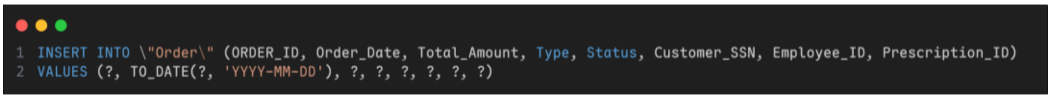
```
SELECT p.Prescription_ID, p.Prescribed_Date, p.Note, d.First_Name || ' ' || d.Last_Name AS Doctor_Name
FROM Prescription p
    JOIN Doctor d
    ON p.Doctor_ID = d.Doctor_ID
WHERE p.Customer_SSN = ?
```

3) Place Order (approx. line 401, in *placeOrder* method)


C:\Windows\System32\cmd.exe - java -cp ojdbc8.jar;. OPMSDemo

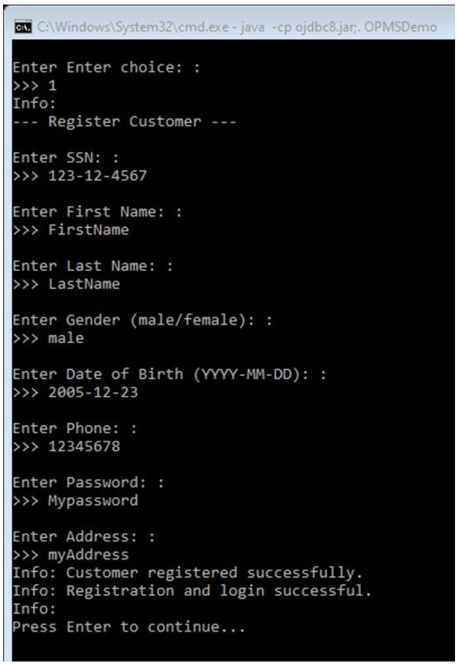
```
Info: 2. View Prescriptions (for a customer)
Info: 3. Place Order
Info: 0. Exit

Enter Enter your choice: :
>>> 3
Info:
--- Place Order ---

Enter Prescription ID: :
>>> 1

Enter Employee ID: :
>>> 1

Enter Order Date (YYYY-MM-DD): :
>>> 2025-11-27

Enter Type (urgent/normal): :
>>> normal

Enter Drug Name: :
>>> Drug7

Enter Pharmacy ID: :
>>> 1

Enter Batch Number: :
>>> 10

Enter Quantity: :
>>> 5

Enter Price: :
>>> 20
Error: Insufficient stock.
Info: Order placed successfully.
Info:
Press Enter to continue...
```

SQL used:



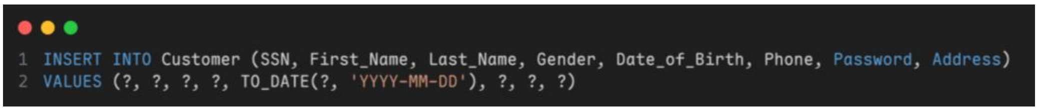
```
INSERT INTO \"Order\" (ORDER_ID, Order_Date, Total_Amount, Type, Status, Customer_SSN, Employee_ID, Prescription_ID)
VALUES (?, TO_DATE(?, 'YYYY-MM-DD'), ?, ?, ?, ?, ?, ?)
```

4) Register Customer (approx. line 317, in *registerCustomer* method)


C:\Windows\System32\cmd.exe - java -cp ojdbc8.jar;. OPMSDemo

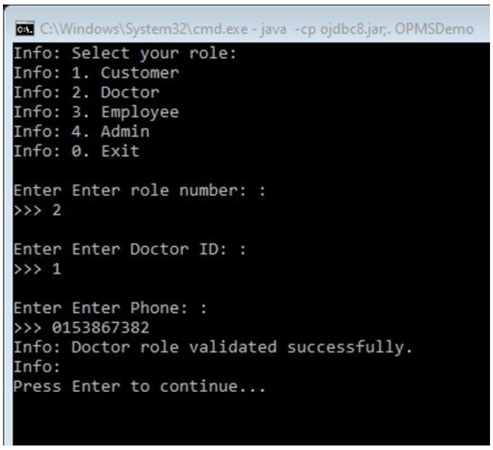
```
Enter Enter choice: :
>>> 1
Info:
--- Register Customer ---

Enter SSN: :
>>> 123-12-4567

Enter First Name: :
>>> FirstName

Enter Last Name: :
>>> LastName

Enter Gender (male/female): :
>>> male

Enter Date of Birth (YYYY-MM-DD): :
>>> 2005-12-23

Enter Phone: :
>>> 12345678

Enter Password: :
>>> Mypassword

Enter Address: :
>>> myAddress
Info: Customer registered successfully.
Info: Registration and login successful.
Info:
Press Enter to continue...
```

SQL used:



```
INSERT INTO Customer (SSN, First_Name, Last_Name, Gender, Date_of_Birth, Phone, Password, Address)
VALUES (?, ?, ?, ?, TO_DATE(?, 'YYYY-MM-DD'), ?, ?, ?)
```

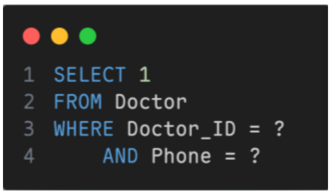
5) Doctor Role Validation (approx. line 116, in *main* method)



C:\Windows\System32\cmd.exe - java -cp ojdbc8.jar;. OPMSDemo

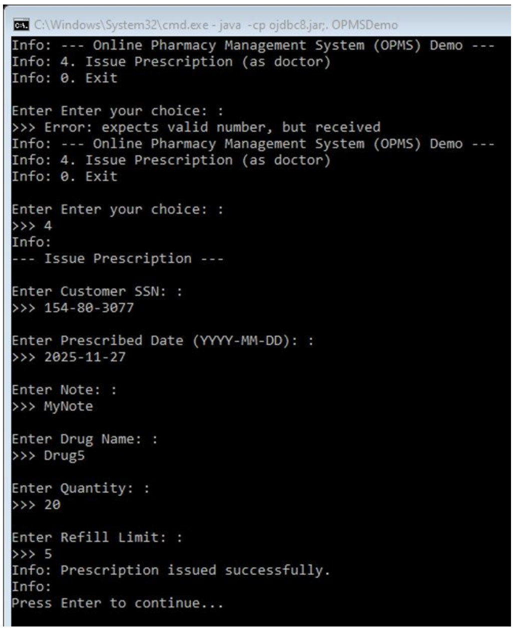
```
Info: Select your role:
Info: 1. Customer
Info: 2. Doctor
Info: 3. Employee
Info: 4. Admin
Info: 0. Exit

Enter Enter role number: :
>>> 2

Enter Enter Doctor ID: :
>>> 1

Enter Enter Phone: :
>>> 0153867382
Info: Doctor role validated successfully.
Info:
Press Enter to continue...
```

SQL used:



```sql
SELECT 1
FROM Doctor
WHERE Doctor_ID = ?
    AND Phone = ?
```

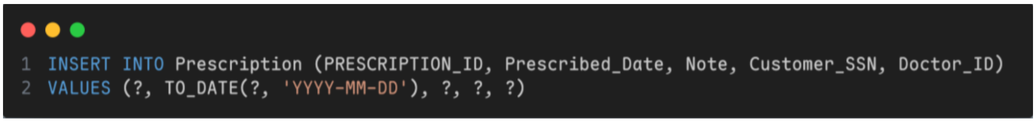
6) Issue Prescription (approx. line 487, in *issuePrescription* method):



C:\Windows\System32\cmd.exe - java -cp ojdbc8.jar;. OPMSDemo

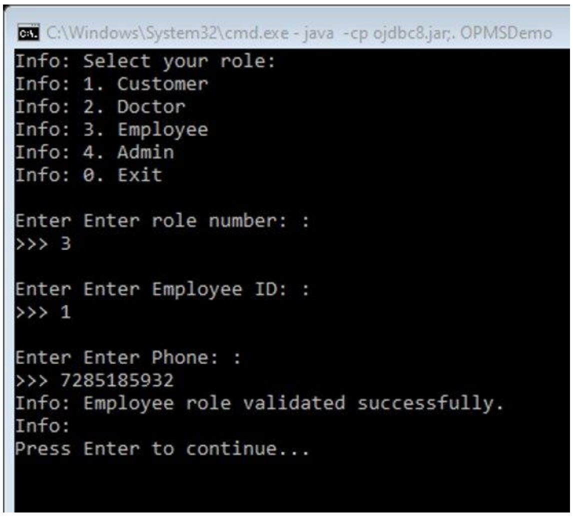
```
Info: --- Online Pharmacy Management System (OPMS) Demo ---
Info: 4. Issue Prescription (as doctor)
Info: 0. Exit

Enter Enter your choice: :
>>> Error: expects valid number, but received
Info: --- Online Pharmacy Management System (OPMS) Demo ---
Info: 4. Issue Prescription (as doctor)
Info: 0. Exit

Enter Enter your choice: :
>>> 4
Info:
--- Issue Prescription ---

Enter Customer SSN: :
>>> 154-80-3077

Enter Prescribed Date (YYYY-MM-DD): :
>>> 2025-11-27

Enter Note: :
>>> MyNote

Enter Drug Name: :
>>> Drug5

Enter Quantity: :
>>> 20

Enter Refill Limit: :
>>> 5
Info: Prescription issued successfully.
Info:
Press Enter to continue...
```

SQL used:



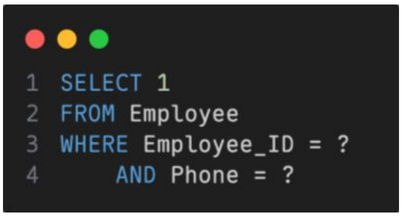
```sql
INSERT INTO Prescription (PRESCRIPTION_ID, Prescribed_Date, Note, Customer_SSN, Doctor_ID)
VALUES (?, TO_DATE(?, 'YYYY-MM-DD'), ?, ?, ?)
```

7) Employee Role Validation (approx. line 135, in *main* method)


C:\Windows\System32\cmd.exe - java -cp ojdbc8.jar;. OPMSDemo

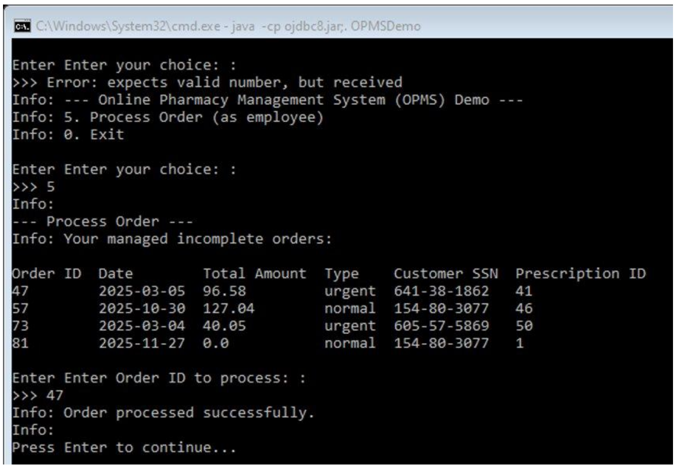
```
Info: Select your role:
Info: 1. Customer
Info: 2. Doctor
Info: 3. Employee
Info: 4. Admin
Info: 0. Exit

Enter Enter role number: :
>>> 3

Enter Enter Employee ID: :
>>> 1

Enter Enter Phone: :
>>> 7285185932
Info: Employee role validated successfully.
Info:
Press Enter to continue...
```

SQL used:



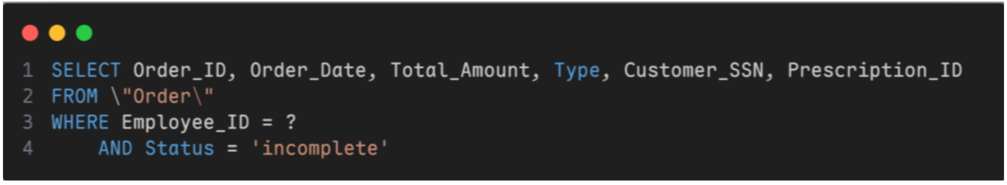
```sql
SELECT 1
FROM Employee
WHERE Employee_ID = ?
      AND Phone = ?
```

8) Process Orders (approx. line 521, in *processOrder* method)


C:\Windows\System32\cmd.exe - java -cp ojdbc8.jar;. OPMSDemo

```
Enter Enter your choice: :
>>> Error: expects valid number, but received
Info: --- Online Pharmacy Management System (OPMS) Demo ---
Info: 5. Process Order (as employee)
Info: 0. Exit

Enter Enter your choice: :
>>> 5
Info:
--- Process Order ---
Info: Your managed incomplete orders:

Order ID  Date        Total Amount  Type    Customer SSN  Prescription ID
47        2025-03-05  96.58         urgent  641-38-1862   41
57        2025-10-30  127.04        normal  154-80-3077   46
73        2025-03-04  40.05         urgent  605-57-5869   50
81        2025-11-27  0.0           normal  154-80-3077   1

Enter Enter Order ID to process: :
>>> 47
Info: Order processed successfully.
Info:
Press Enter to continue...
```

SQL used:



```sql
SELECT Order_ID, Order_Date, Total_Amount, Type, Customer_SSN, Prescription_ID
FROM \"Order\"
WHERE Employee_ID = ?
    AND Status = 'incomplete'
```

Note: The following function will be discussed in the Analysis Report; therefore, the details will not be shown here.

9) Monthly Revenue (approx. lines 400-420, in *monthlyRevenueReport* method)
10) Annual Revenue (approx. lines 430-450, in *annualRevenueReport* method)
11) Expired Stock (approx. lines 460-480, in *viewExpiredStock* method)

12) Top Sold Drugs (approx. lines 490-510, in *topSoldDrugsReport* method)
13) Top Customers (approx. lines 520-540, in *topCustomersReport* method)
14) Top Doctors (approx. lines 550-570, in *topDoctorsReport* method)
15) Low Stock (approx. lines 580-600, in *lowStockReport* method)