(DBMS Project Report)

End-Semester Evaluation

Submitted to:

Dr. Sumit Sharma

Course Number: UCS310
Course Title: Database Management System
Computer Science and Engineering Department

Submitted by:

Amalendu Guru (102103253)
Paras Gupta (102013722)
Aarush Puri (102103723)
Aditya Raj Singh Thakur (102103724)

Group No: 2C026

BE Second Year, COE

Thapar Institute of Engineering and Technology,
Patiala.

TABLE OF CONTENT

- Introduction
- ER Diagram
- ER to Table
- Information Of Entities
- SQL/PLSQL code
- Output Screenshots

INTRODUCTION

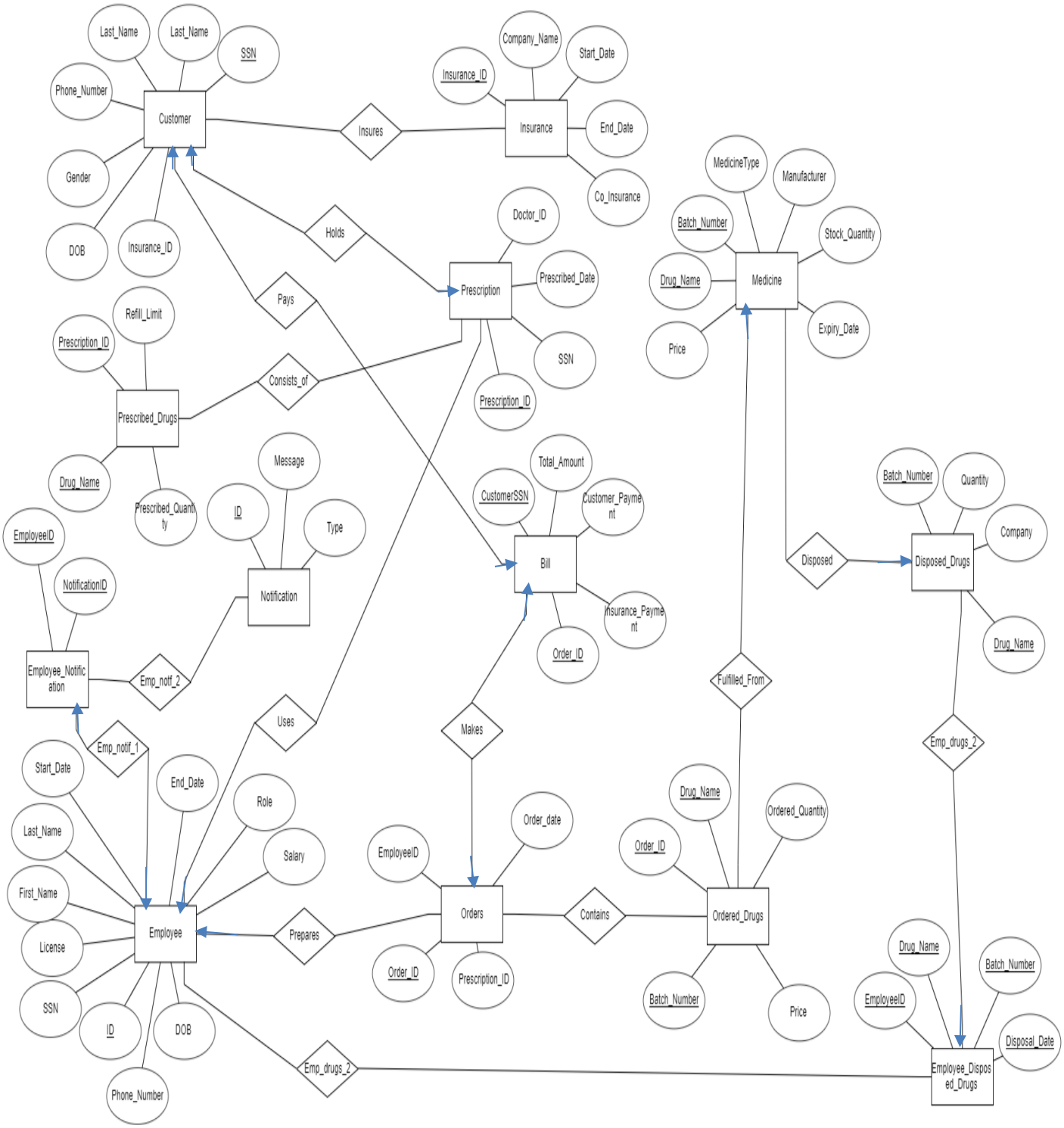
A Pharmacy Management System is a software application that helps to manage various activities in a pharmacy. It provides a centralized platform for managing medicine inventory, patient details, prescription records, sales data, and other essential information related to the pharmacy.

This project on Pharmacy Management System involves designing and developing a database to manage various activities in a pharmacy. The database will include tables for storing information related to medicines, patients, prescriptions, sales, and inventory. The project will involve designing a user-friendly interface for pharmacists to enter and retrieve information from the database. It will also involve implementing security measures to ensure the confidentiality and integrity of the data.

The Pharmacy Management System will enable pharmacists to manage inventory, track prescription history, generate reports, and perform other essential tasks. It will improve the efficiency of pharmacy operations and enhance patient safety by ensuring that the right medicine is dispensed to the right patient at the right time.

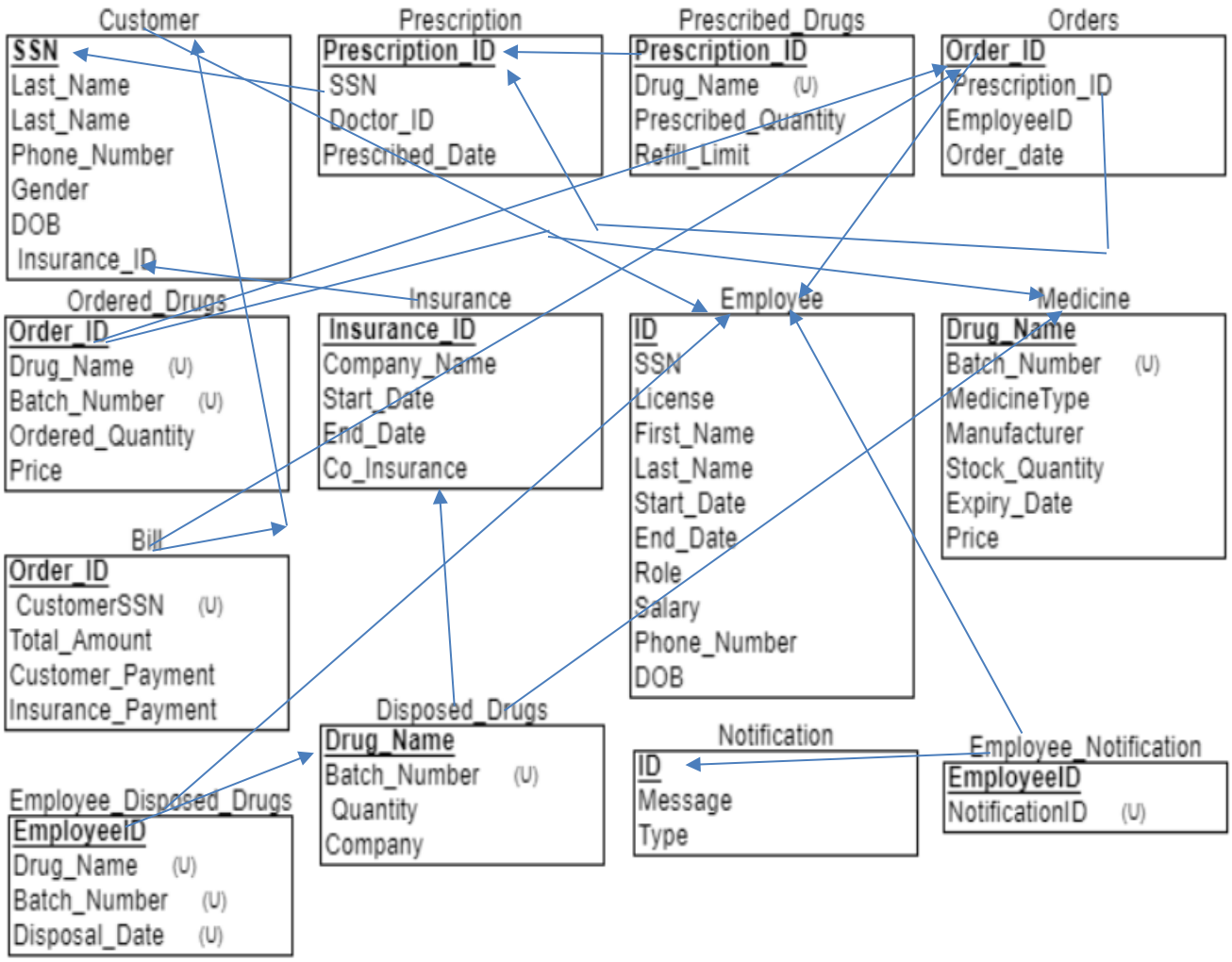
Overall, a Pharmacy Management System is a vital tool for any pharmacy, and the development of a robust DBMS project will help to ensure that pharmacies can deliver high-quality care to their patients.

ER diagram


Last_Name
Last_Name
SSN
Phone_Number
Customer
Gender
DOB
Insurance_ID
Insures
Company_Name
Start_Date
Insurance_ID
Insurance
End_Date
Co_Insurance
Holds
Pays
Doctor_ID
Prescription
Prescribed_Date
SSN
Prescription_ID
Refill_Limit
Prescription_ID
Consists_of
Prescribed_Drugs
Drug_Name
Prescribed_Quantity
MedicineType
Manufacturer
Batch_Number
Stock_Quantity
Drug_Name
Medicine
Price
Expiry_Date
Message
ID
Type
Notification
EmployeeID
NotificationID
Employee_Notification
Emp_notif_2
Emp_notif_1
Total_Amount
CustomerSSN
Customer_Payment
Bill
Insurance_Payment
Order_ID
Uses
Makes
Fulfilled_From
Disposed
Batch_Number
Quantity
Company
Disposed_Drugs
Drug_Name
Emp_drugs_2
Start_Date
Last_Name
First_Name
License
SSN
ID
DOB
Phone_Number
End_Date
Role
Salary
Employee
Prepares
EmployeeID
Order_date
Orders
Order_ID
Prescription_ID
Contains
Drug_Name
Order_ID
Ordered_Quantity
Ordered_Drugs
Batch_Number
Price
Drug_Name
Batch_Number
EmployeeID
Disposal_Date
Employee_Disposed_Drugs
Emp_drugs_2

ER TO TABLE:-

I


Customer
SSN
Last_Name
Last_Name
Phone_Number
Gender
DOB
Insurance_ID
Prescription
Prescription_ID
SSN
Doctor_ID
Prescribed_Date
Prescribed_Drugs
Prescription_ID
Drug_Name (U)
Prescribed_Quantity
Refill_Limit
Orders
Order_ID
Prescription_ID
EmployeeID
Order_date
Ordered_Drugs
Order_ID
Drug_Name (U)
Batch_Number (U)
Ordered_Quantity
Price
Insurance
Insurance_ID
Company_Name
Start_Date
End_Date
Co_Insurance
Employee
ID
SSN
License
First_Name
Last_Name
Start_Date
End_Date
Role
Salary
Phone_Number
DOB
Medicine
Drug_Name
Batch_Number (U)
MedicineType
Manufacturer
Stock_Quantity
Expiry_Date
Price
Bill
Order_ID
CustomerSSN (U)
Total_Amount
Customer_Payment
Insurance_Payment
Disposed_Drugs
Drug_Name
Batch_Number (U)
Quantity
Company
Notification
ID
Message
Type
Employee_Notification
EmployeeID
NotificationID (U)
Employee_Disposed_Drugs
EmployeeID
Drug_Name (U)
Batch_Number (U)
Disposal_Date (U)

INFORMATION OF ENTITIES

In total we have 13 Tables and information of each entity is mentioned below:-

1. **Customer: (Attributes – SSN, First_Name, Last_Name, Phone , Gender, Address , DOB, Insurance_ID)**

2. **Prescription: (Attributes –Prescription_ID, SSN , Doctor_ID , Prescribed_Date)**

3. **Prescribed_Drugs: (Attributes – Prescription_ID, Drug_Name, Prescribed_Quantity, Refill_Limit)**

4. **Orders: (Attributes – Order_ID, Prescription_ID, EmployeeID, Order_Date)**

5. **Notification: (Attributes- ID , Message , Type)**

6. **Medicine: (Attributes - Drug_Name, Batch_Number, MedicineType, Manufacturer, Stock_quantity, Expiry_Date, Price)**

7. **Employee: (Attributes – ID, SSN, License, First_Name, Last_Name, Start_Date, End_Date, Role,Salary, Phone_Number, DOB)**

8. **Disposed_Drugs : (Attributes – Drug_Name, Batch_Number, Quantity, Company)**

9. **Ordered_Drugs: (Attributes – Order_ID, Drug_Name, Batch_Number, Ordered_Quantity, Price)**

10. **Insurance: (Attributes – Insurance_ID, Company_Name, Start_Date, End_Date, Co_Insurance,)**

11. **Employee: (Attributes – ID, SSN, License, First_Name, Last_Name, Start_Date, End_Date, Role,Salary, Phone_Number, DOB)**

12. **Bill : (Attributes – Order_ID, CustomerSSN, Total_Amount, Customer_Payment, Insurance_Payment)**

13. **Employee_Notification: (Attributes- EmployeeID , NotificationID)**

14. **Employee_Disposed_Drugs (EmployeeID, Drug_Name, Batch_Number, Disposal_Date)**

SQL/PLSQL CODE

-- Table for Customer

```
DROP TABLE Customer; CREATE TABLE Customer ( SSN number(10) NOT
NULL, First_Name char(255) NOT NULL, Last_Name char(255) NOT NULL,
Phone number(10) NOT NULL UNIQUE, Gender char(1) NOT NULL, Address
char(1000) NOT NULL, DOB date NOT NULL, Insurance_ID number(10) NOT
NULL UNIQUE, PRIMARY KEY (SSN) );
```

-- Table for the prescriptions of medicines

```
DROP TABLE Prescription; CREATE TABLE Prescription ( Prescription_ID
number(10) NOT NULL, SSN number(10) NOT NULL, Doctor_ID number(10)
NOT NULL, Prescribed_Date date NOT NULL, PRIMARY KEY
(Prescription_ID) );
```

-- Table for the given prescribed drugs for patient

```
DROP table Prescribed_Drugs; CREATE TABLE Prescribed_Drugs (
Prescription_ID number(10) NOT NULL, Drug_Name char(255) NOT NULL,
Prescribed_Quantity number(10) NOT NULL, Refill_Limit number(10) NOT
NULL, PRIMARY KEY (Prescription_ID, Drug_Name) );
```

-- Table for corresponding orders as per needs

```
DROP TABLE Orders; CREATE TABLE Orders ( Order_ID number(10) NOT
NULL, Prescription_ID number(10) NOT NULL, EmployeeID number(5) NOT
NULL, Order_Date date NOT NULL, PRIMARY KEY (Order_ID) );
```

-- Table for the ordered drugs

```
DROP TABLE Ordered_Drugs; CREATE TABLE Ordered_Drugs ( Order_ID
number(10) NOT NULL, Drug_Name char(255) NOT NULL, Batch_Number
number(10) NOT NULL, Ordered_Quantity number(10) NOT NULL, Price
number(2) NOT NULL, PRIMARY KEY (Order_ID, Drug_Name, Batch_Number)
);
```

-- Table for the record of insurance of person

DROP TABLE Insurance; CREATE TABLE Insurance (Insurance_ID number(10) NOT NULL, Company_Name char(255) NOT NULL, Start_Date date NOT NULL, End_Date date NOT NULL, Co_Insurance number(4) NOT NULL, PRIMARY KEY (Insurance_ID));

-- We made index for the column Company_Name of table insurance to make searching faster

CREATE INDEX Insurance_Company_Name ON Insurance (Company_Name);

-- Table for information of staffs

DROP TABLE Employee; CREATE TABLE Employee (ID number(5) NOT NULL, SSN number(10) NOT NULL UNIQUE, License number(10) UNIQUE, First_Name char(255) NOT NULL, Last_Name char(255) NOT NULL, Start_Date date NOT NULL, End_Date date, Role char(255) NOT NULL, Salary number(4) NOT NULL, Phone_Number number(10) NOT NULL, DOB date NOT NULL, PRIMARY KEY (ID));

-- Table for all information storage of medicines

DROP TABLE Medicine; CREATE TABLE Medicine (Drug_Name char(255) NOT NULL, Batch_Number number(10) NOT NULL, MedicineType char(255) NOT NULL, Manufacturer char(255) NOT NULL, Stock_Quantity number(10) NOT NULL, Expiry_Date date NOT NULL, Price number(4) NOT NULL, PRIMARY KEY (Drug_Name, Batch_Number));

-- Table for the informations of bills

DROP TABLE Bill; CREATE TABLE Bill (Order_ID number(10) NOT NULL, CustomerSSN number(10) NOT NULL, Total_Amount number(4) NOT NULL, Customer_Payment number(4) NOT NULL, Insurance_Payment number(4) NOT NULL, PRIMARY KEY (Order_ID, CustomerSSN));

-- Table for disposed drugs

Drop table Disposed_Drugs; CREATE TABLE Disposed_Drugs (Drug_Name char(255) NOT NULL, Batch_Number number(10) NOT NULL, Quantity number(10) NOT NULL, Company char(255) NOT NULL, PRIMARY KEY (Drug_Name, Batch_Number)); -- Notification records DROP TABLE Notification; CREATE TABLE Notification (ID number(10) NOT NULL, Message char(255) NOT NULL, Type char(255) NOT NULL, PRIMARY KEY (ID));

-- Particular notification record of employees

drop table Employee_Notification; CREATE TABLE Employee_Notification (EmployeeID number(5) NOT NULL, NotificationID number(10) NOT NULL, PRIMARY KEY (EmployeeID, NotificationID));

-- Table for the employee disposed drugs

drop table Employee_Disposed_Drugs; CREATE TABLE Employee_Disposed_Drugs (EmployeeID number(5) NOT NULL, Drug_Name char(255) NOT NULL, Batch_Number number(10) NOT NULL, Disposal_Date date NOT NULL, PRIMARY KEY (EmployeeID, Drug_Name, Batch_Number, Disposal_Date));

-- Below here, there are all statements to make foreign keys, First we have made a relation between customer and insurance through insurance id

ALTER TABLE Customer ADD CONSTRAINT insures FOREIGN KEY (Insurance_ID) REFERENCES Insurance (Insurance_ID) ON DELETE Set null; -- Now we have made relation between prescription and customer through SSN ALTER TABLE Prescription ADD CONSTRAINT holds FOREIGN KEY (SSN) REFERENCES Customer (SSN);

-- Here is relation between prescribed drugs and prescription through prescription id

ALTER TABLE Prescribed_Drugs ADD CONSTRAINT consists_of FOREIGN KEY (Prescription_ID) REFERENCES Prescription (Prescription_ID) ON DELETE Cascade;

-- Now we have made two relations from order to employeeid and to prescriptionID

ALTER TABLE Orders ADD CONSTRAINT prepares FOREIGN KEY (EmployeeID) REFERENCES Employee (ID); ALTER TABLE Orders ADD CONSTRAINT uses FOREIGN KEY (Prescription_ID) REFERENCES Prescription (Prescription_ID);

-- Now we have made two relations between ordered drugs to orders and medicine

ALTER TABLE Ordered_Drugs ADD CONSTRAINT contains FOREIGN KEY (Order_ID) REFERENCES Orders (Order_ID) ON DELETE Cascade; ALTER TABLE Ordered_Drugs ADD CONSTRAINT Fulfilled_From FOREIGN KEY (Drug_Name, Batch_Number) REFERENCES Medicine (Drug_Name, Batch_Number);

-- Here we have made two relations between Bill and Orders and Customers

ALTER TABLE Bill ADD CONSTRAINT makes FOREIGN KEY (Order_ID) REFERENCES Orders (Order_ID); ALTER TABLE Bill ADD CONSTRAINT pays FOREIGN KEY (CustomerSSN) REFERENCES Customer (SSN);

-- Now we have made relation between disposed drugs to medicine through composite foreign key of drug name and batch number

ALTER TABLE Disposed_Drugs ADD CONSTRAINT disposed FOREIGN KEY (Drug_Name, Batch_Number) REFERENCES Medicine (Drug_Name, Batch_Number);

-- Below two statements are for employee notification and employee relation and Notification relation

ALTER TABLE Employee_Notification ADD CONSTRAINT emp_notif_id_1 FOREIGN KEY(EmployeeID) REFERENCES Employee (ID) ON DELETE Cascade; ALTER TABLE Employee_Notification ADD CONSTRAINT emp_notif_id_2 FOREIGN KEY(NotificationID) REFERENCES Notification (ID) ON DELETE Cascade;

-- Below two statments are for Employee disposed drugs and Employee relation and disposed drugs relation

ALTER TABLE Employee_Disposed_Drugs ADD CONSTRAINT emp_disposed_drugs_id_1 FOREIGN KEY(EmployeeID) REFERENCES Employee (ID); ALTER TABLE Employee_Disposed_Drugs ADD CONSTRAINT emp_disposed_drugs_id_2 FOREIGN KEY (Drug_Name, Batch_Number) REFERENCES Disposed_Drugs (Drug_Name, Batch_Number)

//switch case for insurance table

```
DECLARE
 choice NUMBER;
BEGIN
 DBMS_OUTPUT.PUT_LINE('Enter your choice:');
 DBMS_OUTPUT.PUT_LINE('1. Insert data into Insurance table');
 DBMS_OUTPUT.PUT_LINE('2. Delete data from Insurance table');
 DBMS_OUTPUT.PUT_LINE('3. Modify data from Insurance table');
 choice := 1;
 CASE choice
 WHEN 1 THEN
 -- DECLARE
 -- ins_id NUMBER;
 -- co_name VARCHAR2(255);
 -- start_dt DATE;
 -- end_dt DATE;
```

```
-- co_ins NUMBER(4);
 -- BEGIN
 -- DBMS_OUTPUT.PUT_LINE('Enter Insurance ID:');
 -- ins_id := &ins_id;
 -- DBMS_OUTPUT.PUT_LINE('Enter Company Name:');
 -- co_name := '&co_name';
 -- DBMS_OUTPUT.PUT_LINE('Enter Start Date (YYYY-MM-DD):');
 -- start_dt := TO_DATE('&start_dt', 'YYYY-MM-DD');
 -- DBMS_OUTPUT.PUT_LINE('Enter End Date (YYYY-MM-DD):');
 -- end_dt := TO_DATE('&end_dt', 'YYYY-MM-DD');
 -- DBMS_OUTPUT.PUT_LINE('Enter Co-Insurance:');
 -- co_ins := &co_ins;
 IF MONTHS_BETWEEN(to_date('2024-12-31', 'YYYY-MM-DD'), to_date('2022-01-01', 'YYYY-
MMDD')) < 12 THEN
 DBMS_OUTPUT.PUT_LINE('Error: Policy must be valid for at least 1 year!');
 ELSE
 -- INSERT INTO Insurance (Insurance_ID, Company_Name, Start_Date, End_Date, Co_Insurance)
 -- VALUES (ins_id, co_name, start_dt, end_dt, co_ins);
 INSERT INTO Insurance (Insurance_ID, Company_Name, Start_Date, End_Date, Co_Insurance)
 VALUES (1234567890, 'ABC Insurance Company', to_date('2022-01-01',
'YYYY-MM-DD'), to_date('2022-12-31', 'YYYY-MM-DD'), 20);
 DBMS_OUTPUT.PUT_LINE('INSERTED');
 END IF;
 -- END;
 WHEN 2 THEN
 -- DECLARE
 -- ins_id NUMBER;
 -- BEGIN
 -- DBMS_OUTPUT.PUT_LINE('Enter Insurance ID:');
 -- ins_id := &ins_id;
 DELETE FROM Insurance WHERE Insurance_ID = 1234567890;
 DBMS_OUTPUT.PUT_LINE('DELETED');
 -- END;
 WHEN 3 THEN
 -- DECLARE
 -- ins_id NUMBER;
 -- co_name VARCHAR2(255);
 -- BEGIN
 -- DBMS_OUTPUT.PUT_LINE('Enter Insurance ID:');
 -- ins_id := &ins_id;
 -- DBMS_OUTPUT.PUT_LINE('Enter New Company Name:');
 -- co_name := '&co_name';
 UPDATE Insurance SET Company_Name = 'DmmA' WHERE Insurance_ID = 1234567890;
 DBMS_OUTPUT.PUT_LINE('Updated');
 -- END;
 ELSE
 DBMS_OUTPUT.PUT_LINE('Invalid choice!');
 END CASE;
END;
```

```
//switch case for customer table
DECLARE

 choice NUMBER;
 ssn_input NUMBER(10);
 first_name_input VARCHAR2(255);
 last_name_input VARCHAR2(255);
 phone_input NUMBER(10);
 gender_input CHAR(1);
 address_input VARCHAR2(1000);
 dob_input DATE;
 insurance_id_input NUMBER(10);
 age NUMBER;
BEGIN
 DBMS_OUTPUT.PUT_LINE('Enter your choice:');
 DBMS_OUTPUT.PUT_LINE('1. Insert data into customer table');
 DBMS_OUTPUT.PUT_LINE('2. Delete data from customer table');
 DBMS_OUTPUT.PUT_LINE('3. Modify data from customer table');
 choice := 2;
 CASE choice
 WHEN 1 THEN
 -- DBMS_OUTPUT.PUT_LINE('Enter SSN:');
 -- ssn_input := &ssn_input;
 -- DBMS_OUTPUT.PUT_LINE('Enter First Name:');
 -- first_name_input := '&first_name_input';
 -- DBMS_OUTPUT.PUT_LINE('Enter Last Name:');
 -- last_name_input := '&last_name_input';
 -- DBMS_OUTPUT.PUT_LINE('Enter Phone:');
 -- phone_input := &phone_input;
 -- DBMS_OUTPUT.PUT_LINE('Enter Gender:');
 -- gender_input := '&gender_input';
 -- DBMS_OUTPUT.PUT_LINE('Enter Address:');
 -- address_input := '&address_input';
 -- DBMS_OUTPUT.PUT_LINE('Enter DOB (yyyy-mm-dd):');
 -- dob_input := TO_DATE('&dob_input', 'yyyy-mm-dd');

 -- get the age from DOB
 age := trunc(months_between(sysdate, to_date('1990-01-01', 'YYYY-MM-DD'))/12);
 -- check if age is less than 13
 IF age < 13 THEN
 DBMS_OUTPUT.PUT_LINE('Age must be at least 13 years old to create an account.');
 ELSE
 -- insert data into the customer table
 INSERT INTO Customer (SSN, First_Name, Last_Name, Phone, Gender, Address, DOB,
Insurance_ID)
 VALUES (1234567890, 'John', 'Doe', 9876543210, 'M', '123 Main St, Anytown USA',
to_date('1990-01-01', 'YYYY-MM-DD'), 9876543210);
 DBMS_OUTPUT.PUT_LINE('Data inserted successfully.');
 END IF;
 WHEN 2 THEN
 -- DBMS_OUTPUT.PUT_LINE('Enter SSN:');
 -- ssn_input := &ssn_input;
```

```
 DELETE FROM customer WHERE SSN = 1234567890;
 DBMS_OUTPUT.PUT_LINE('Data deleted successfully.');
 WHEN 3 THEN
 UPDATE customer SET First_Name = 'Jane' WHERE SSN = 1234567890;
 DBMS_OUTPUT.PUT_LINE('Data updated successfully.');
 ELSE
 DBMS_OUTPUT.PUT_LINE('Invalid choice!');
 END CASE;
END;


DECLARE
 v_company_name VARCHAR2(500) := 'ABC Insurance';
 v_first_name VARCHAR2(255);
BEGIN
 SELECT c.First_Name INTO v_first_name
 FROM Customer c
 JOIN Insurance i ON c.Insurance_ID = i.Insurance_ID
 WHERE i.Company_Name ='ABC Insurance';

 DBMS_OUTPUT.PUT_LINE('Customer(s) with insurance from ' || v_company_name || ': ' ||
v_first_name);
EXCEPTION
 WHEN NO_DATA_FOUND THEN
 DBMS_OUTPUT.PUT_LINE('No customers with insurance from ' || v_company_name || ' found');
END;
```

```
Statement processed.
Customer(s) with insurance from ABC Insurance: John
```

```
DECLARE
 v_min_salary NUMBER(4) := 4000;
BEGIN
 FOR emp IN (
 SELECT Role, COUNT(*) AS num_employees
 FROM Employee
 GROUP BY Role
 HAVING MIN(Salary) >= v_min_salary
 ) LOOP
 DBMS_OUTPUT.PUT_LINE(emp.Role || ': ' || emp.num_employees || ' employees');
 END LOOP;
END;
```

```
Statement processed.
Software Engineer
: 1 employees
Marketing Manager
: 1 employees
Product Manager
: 1 employees
HR Manager
: 1 employees
Manager
: 1 employees
```

```
DECLARE
 v_max_profit NUMBER(20,2) := 0;
 v_max_profit_drug_name VARCHAR2(255);
BEGIN
 FOR drug IN (SELECT DISTINCT Drug_Name FROM Ordered_Drugs) LOOP
 FOR order_info IN (SELECT Ordered_QuanΘty, Price FROM Ordered_Drugs WHERE Drug_Name =
drug.Drug_Name) LOOP
 DECLARE
 v_profit NUMBER(20,2);
 BEGIN
 v_profit := order_info.Ordered_QuanΘty * order_info.Price;
 IF v_profit > v_max_profit THEN
 v_max_profit := v_profit;
 v_max_profit_drug_name := drug.Drug_Name;
 END IF;
 END;
 END LOOP;
 END LOOP;

 DBMS_OUTPUT.PUT_LINE('The drug with the maximum profit is ' || v_max_profit_drug_name || '
with a profit of ' || v_max_profit);
END;
```

```
Statement processed.
The drug with the maximum profit is Ibuprofen
with a profit of 600
```