CSCE 5350 004

FUNDAMENTALS OF DATABASE SYSTEMS

PROJECT - PART 5

GROUP-8

Name	ID
Nitin Chakravarthy Chilukuri	11601099
Surya Vamsi Chintapalli	11645442
Prathyusharani Dumpala	11656342
Yamini Gollamudi	11642723
Lohitha Sai Bonthu	11611601

ATTRIBUTES:

Stores: Store Id, Address, Manager, Assigned Pharmacist, Assigned Doctor, Region Code

Warehouses: Warehouse Id, Address, Warehouse Manager, Current Stock, Capacity, Region Code

Region: Region Name, Region Code, Region Manager

Employees: EID, Name, SSN, Age, Gender, Address, Ph NO, Wage, Type, Location, Bank Account Number

Payroll: EID, Date, Hours Worked

Patients: PID, Name, Age, Gender, Ph NO, Address, SSN

Insurance: INM, Name of Insurance, PID, Amount, Date Claimed, Status

Prescription: PrescriptionID, DoctorID, PatientID, Date_Prescribed

Drug: Drug ID, Name, Price, Drug Type, Dosage, Manf By, Manf Date, Batch NO, Expiry Date

Inventory: Drug ID, Building ID, Current Stock

Logistics: Good, Date, Warehouse ID, Store ID, Quantity, Status

Sales: No of sales per month, Month, Year, Drug ID, Store ID

IDENTIFIED FUNCTIONAL DEPENDENCIES:

Stores:

Store Id->Address, Manager, Assigned Pharmacist, Assigned Doctor, Region Code

Warehouses:

Warehouse Id-> Address, Warehouse Manager, Current Stock, Capacity, Region Code

Region:

Region Code-> Region Name, Region Manager

Employees:

EID-> Name, SSN, Age, Gender, Address, Ph NO, Wage, Type, Location, Bank Account Number

SSN-> Name, Age, Gender

Type-> Wage

Payroll:

EID, Date->Hours Worked

Patients:

PID-> Name, Age, Gender, Ph NO, Address, SSN

SSN-> Name, Age, Gender

Insurance:

INM->Name of Insurance, PID, Amount, Date Claimed, Status

PID->INM

Prescription:

Prescription ID->Doctor ID, Patient ID, Date_Prescribed

Drug:

Drug ID->Name, Price, Drug Type, Dosage, Manf By, Manf Date, Batch NO, Expiry Date

Inventory:

Drug ID, Building ID->Current Stock

Logistics:

Good, Date, Warehouse ID, Store ID->Quantity, Status

Sales:

Month, Year, Drug ID, Store ID->No of sales per month

NORMALIZATION:

1. Stores:

Stores: Store Id, Address, Manager, Assigned Pharmacist, Assigned Doctor, Region Code

Store Id->Address, Manager, Assigned Pharmacist, Assigned Doctor, Region Code

NORMAL FORM:

Store Id+ = {Store Id, Address, Manager, Assigned Pharmacist, Assigned Doctor, Region Code}

Therefore, Store Id is the candidate key.

Since there is only one FD in which the LHS contains the candidate key which is also a super key this table is in BCNF.

2. Warehouses:

Warehouses: Warehouse Id, Address, Warehouse Manager, Current Stock, Capacity, Region

Code

Warehouse Id-> Address, Warehouse Manager, Current Stock, Capacity, Region Code

NORMAL FORM:

Warehouse Id+ = {Warehouse Id, Address, Warehouse Manager, Current Stock, Capacity, Region Code}

Therefore, Warehouse Id is the candidate key.

Since there is only one FD in which the LHS contains the candidate key which is also a super key this table is in BCNF.

3. Region:

Region: Region Name, Region Code, Region Manager

Region Code-> Region Name, Region Manager

NORMAL FORM:

Region Code+ = {Region Code, Region Name, Region Manager}

Region Name+ = {Region Name}

Region Manager+ = {Region Manager}

We have Region Code as the candidate key.

Since there is only one FD in which the LHS contains the candidate key which is also a super key this table is in BCNF.

4. Employees:

Employees: EID, Name, SSN, Age, Gender, Address, Ph NO, Wage, Type, Location, Bank Account Number

EID-> Name, SSN, Age, Gender, Address, Ph NO, Wage, Type, Location, Bank Account Number

SSN-> Name, Age, Gender

Type-> Wage

EID+ = {EID, Name, SSN, Age, Gender, Address, Ph NO, Wage, Type, Location, Bank Account Number}

SSN+ = {SSN, Name, Age, Gender}

Type+ = {Type, Wage}

NORMAL FORM:

We have EID as the candidate key.

Prime Attributes = {EID}

Non-Prime Attributes = {Name, SSN, Age, Gender, Address, Ph NO, Wage, Type, Location, Bank Account Number}

-We know that the table is in 1NF as multiple values are not allowed.

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

-The table is not in 3NF as we have (SSN-> Name, Age, Gender) and (Type-> Wage) where a non-prime attribute is dependent on another non-prime attribute.

Therefore, the highest normal form of Employees table is 2NF.

NORMALIZING TO BCNF:

Divide Employees table into R1, R2 such that:

R1(EID, Name, SSN, Age, Gender, Address, Ph NO, Type, Location, Bank Account Number)

R2(Type, Wage).

FD corresponding to R2: Type-> Wage

In R2 ‘Type’ is the candidate key from (Type-> Wage) and all FD’s corresponding to R2 have a super key in the LHS. So R2 is in BCNF.

Now Decompose R1 into R3, R4 such that:

R3(SSN, Name, Age, Gender)

R4(EID, SSN, Address, Ph NO, Type, Location, Bank Account Number)

FD corresponding to R3: SSN-> Name, Age, Gender

In R3 ‘SSN’ is the candidate key from (SSN-> Name, Age, Gender) and all FD’s corresponding to R3 have a super key in the LHS. So R3 is in BCNF.

FD corresponding to R4: EID-> SSN, Address, Ph NO, Type, Location, Bank Account Number.

This FD is decomposed from (EID-> Name, SSN, Age, Gender, Address, Ph NO, Wage, Type, Location, Bank Account Number) using Armstrong’s Axioms Decomposition Rule.

In R4 ‘EID’ is the candidate key and all FD’s corresponding to R4 have a super key in the LHS. So R4 is in BCNF.

Dependency Preserving Check:

We have (Type-> Wage) preserved in R2 and (SSN-> Name, Age, Gender) is preserved in R3.

We need to check for EID-> Name, SSN, Age, Gender, Address, Ph NO, Wage, Type, Location, Bank Account Number

This can done by computing w = w U ((w ∩ Ri)+ ∩ Ri) and check whether w gives the FD we are checking for or not.

Initially w = EID

Iteration 1:

w = w U ((w ∩ Ri)+ ∩ Ri) for Ri = R2:

w = EID U ((EID ∩ (Type, Wage))+ ∩ (Type, Wage)) = EID

w = w U ((w ∩ Ri)+ ∩ Ri) for Ri = R3:

w = EID U ((EID ∩ (SSN, Name, Age, Gender))+ ∩ (SSN, Name, Age, Gender))

w = EID

w = w U ((w ∩ Ri)+ ∩ Ri) for Ri = R4:

w = EID U ((EID ∩ (EID, SSN, Address, Ph NO, Type, Location, Bank Account Number))+ ∩ (EID, SSN, Address, Ph NO, Type, Location, Bank Account Number))

w = (EID, SSN, Address, Ph NO, Type, Location, Bank Account Number)

Iteration 2:

w = w U ((w ∩ Ri)+ ∩ Ri) for Ri = R2:

w = (EID, SSN, Address, Ph NO, Type, Location, Bank Account Number) U (((EID, SSN, Address, Ph NO, Type, Location, Bank Account Number) ∩ (Type, Wage))+ ∩ (Type, Wage))

w = (EID, SSN, Address, Ph NO, Type, Location, Bank Account Number, Wage)

w = w U ((w ∩ Ri)+ ∩ Ri) for Ri = R3:

w = (EID, SSN, Address, Ph NO, Type, Location, Bank Account Number, Wage) U (((EID, SSN, Address, Ph NO, Type, Location, Bank Account Number, Wage) ∩ (SSN, Name, Age, Gender))+ ∩ (SSN, Name, Age, Gender))

w = (EID, SSN, Address, Ph NO, Type, Location, Bank Account Number, Wage, Name, Age, Gender)

We can obtain the FD EID-> Name, SSN, Age, Gender, Address, Ph NO, Wage, Type, Location, Bank Account Number

Hence Dependency is Preserved and converted into BCNF.

5. Payroll

Payroll: EID, Date, Hours Worked

EID, Date->Hours Worked

NORMAL FORM:

EID+ = {EID}

Date+ = {Date}

Hours Worked+ = {Hours Worked}

(EID, Date)+ = { EID, Date, Hours Worked}

We have (EID, Date) as the candidate key.

Since there is only one FD and it contains candidate key in its LHS which is also a super key Payroll is in BCNF.

6. Patients

Patients: PID, Name, Age, Gender, Ph NO, Address, SSN

PID-> Name, Age, Gender, Ph NO, Address, SSN

SSN-> Name, Age, Gender

NORMAL FORM:

PID+ = {Name, Age, Gender, Ph NO, Address, SSN}

SSN+ = {SSN, Name, Age, Gender}

Therefore, PID is the candidate keys.

Prime Attributes = {PID}

Non-Prime Attributes = {Name, Age, Gender, Ph NO, Address, SSN}

-We know that the table is in 1NF as multiple values are not allowed.

- It is in 2NF as there are no partial dependencies.

- It is not in 3NF as there are transitive dependencies.

NORMALIZING TO BCNF:

Divide the Patients Table into R1, R2 such that:

R1(SSN, Name, Age, Gender)

R2(PID, SSN, Ph NO, Address)

FD corresponding to R1: SSN-> Name, Age, Gender

In R1 ‘SSN’ is the candidate key from (SSN-> Name, Age, Gender) and all FD’s corresponding to R1 have a super key in the LHS. So R2 is in BCNF.

FD corresponding to R2: PID-> SSN, Ph NO, Address

This FD is decomposed from (PID-> Name, Age, Gender, Ph NO, Address, SSN) using Armstrong's Axioms Decomposition Rule.

In R2 'PID' is the candidate key and all FD's corresponding to R2 have a super key in the LHS. So R2 is in BCNF.

Dependency Preserving Check: We have SSN-> Name, Age, Gender preserved in R1. We need to check for PID-> Name, Age, Gender, Ph NO, Address, SSN

This can done by computing w = w U ((w ∩ Ri)+ ∩ Ri) and check whether w gives the FD we are checking for or not.

Initially w = PID

Iteration 1:

w = w U ((w ∩ Ri)+ ∩ Ri) for Ri = R1:

w = PID U ((PID ∩ (SSN,Name,Age,Gender))+ ∩ (SSN,Name,Age,Gender)) = PID

w = w U ((w ∩ Ri)+ ∩ Ri) for Ri = R2:

w = PID U ((PID ∩ (PID, SSN, Ph NO, Address))+ ∩ (PID, SSN, Ph NO, Address))

w = (PID, SSN, Ph NO, Address)

Iteration 2:

w = w U ((w ∩ Ri)+ ∩ Ri) for Ri = R1:

w = (PID, SSN, Ph NO, Address) U (((PID, SSN, Ph NO, Address) ∩ (SSN,Name,Age,Gender))+ ∩ (SSN,Name,Age,Gender))

w = (PID, Name, Age, Gender, Ph NO, Address, SSN)

Since we can obtain PID-> Name, Age, Gender, Ph NO, Address, SSN, the decomposition of R into R1, R2 is in BCNF and Dependency Preserving.

7. Insurance

Insurance: INM, Name of Insurance, PID, Amount, Date Claimed, Status

INM->Name of Insurance, PID, Amount, Date Claimed, Status

PID->INM

NORMAL FORM:

INM+ = {INM, Name of Insurance, PID, Amount, Date Claimed, Status}

PID+ = {PID, INM, Name of Insurance, Amount, Date Claimed, Status}

Therefore, INM and PID are the candidate keys.

Prime Attributes = {INM, PID}

Non-Prime Attributes = {Name of Insurance, Amount, Date Claimed, Status}

-We know that the table is in 1NF as multiple values are not allowed.

-The table is not in 2NF as we have PID -> INM which is a partial dependency.

Therefore, the highest normal form of Employees table is 1NF.

NORMALIZING TO BCNF:

Divide the Insurance Table into R1, R2 such that:

R1(INM, Name of Insurance, Amount, Date Claimed, Status)

R2(PID, INM)

FD corresponding to R2: PID-> INM

In R2 'PID' is the candidate key from (PID-> INM) and all FD's corresponding to R2 have a super key in the LHS. So R2 is in BCNF.

FD corresponding to R1: INM-> Name of Insurance, Amount, Date Claimed, Status

This FD is decomposed from (INM->Name of Insurance, PID, Amount, Date Claimed, Status) using Armstrong's Axioms Decomposition Rule.

In R1 'INM' is the candidate key and all FD's corresponding to R1 have a super key in the LHS. So R1 is in BCNF.

Dependency Preserving Check: We have (PID-> INM) preserved in R2.

We need to check for INM->Name of Insurance, PID, Amount, Date Claimed, Status

This can done by computing w = w U ((w ∩ Ri)+ ∩ Ri) and check whether w gives the FD we are checking for or not.

Initially w = INM

Iteration 1:

w = w U ((w ∩ Ri)+ ∩ Ri) for Ri = R2:

w = INM U ((INM ∩ (INM,PID))+ ∩ (INM,PID)) = (INM, PID)

w = w U ((w ∩ Ri)+ ∩ Ri) for Ri = R1:

w = (INM, PID) U (((INM, PID) ∩ (INM, Name of Insurance, Amount, Date Claimed, Status))+ ∩ (INM, Name of Insurance, Amount, Date Claimed, Status))

w = (INM, PID, Name of Insurance, Amount, Date Claimed, Status)

Hence Dependency is Preserved and converted into BCNF.

8. Prescription

Precsription: PrescriptionID, DoctorID, PatientID, Date_Prescribed

Prescription ID->Doctor ID, Patient ID, Date_Prescribed

NORMAL FORM:

Prescription ID+ = {PrescriptionID, DoctorID, PatientID, Date_Prescribed}

We have (Prescription ID) as the candidate key.

Since there is only one FD and it contains candidate key in its LHS which is also a super key Payroll is in BCNF.

9. Drug

Drug: Drug ID, Name, Price, Drug Type, Dosage, Manf By, Manf Date, Batch NO, Expiry Date

Drug ID->Name, Price, Drug Type, Dosage, Manf By, Manf Date, Batch NO, Expiry Date

NORMAL FORM:

Drug ID+ = {Drug ID, Name, Price, Drug Type, Dosage, Manf By, Manf Date, Batch NO, Expiry Date}

We have (Drug ID) as the candidate key.

Since there is only one FD and it contains candidate key in its LHS which is also a super key Payroll is in BCNF.

10. Inventory

Inventory: Drug ID, Building ID, Current Stock

Drug ID, Building ID->Current Stock

NORMAL FORM:

Drug ID+ = {Drug ID}

Building ID+ = {Building ID}

Current Stock+ = {Current Stock}

(Drug ID, Building ID)+ = {Drug ID, Building ID, Current Stock}

We have (Drug ID, Building ID) as the candidate key.

Since there is only one FD and it contains candidate key in its LHS which is also a super key Payroll is in BCNF.

11. Logistics

Logistics: Good, Date, Warehouse ID, Store ID, Quantity, Status

Good, Date, Warehouse ID, Store ID->Quantity, Status

NORMAL FORM:

We have (Good, Date, Warehouse ID, Store ID) as the candidate key.

Since there is only one FD and it contains candidate key in its LHS which is also a super key Payroll is in BCNF.

12. Sales

Sales: No of sales per month, Month, Year, Drug ID, Store ID

Month, Year, Drug ID, Store ID->No of sales per month

NORMAL FORM:

We have (Month, Year, Drug ID, Store ID) as the candidate key.

Since there is only one FD and it contains candidate key in its LHS which is also a super key Payroll is in BCNF.

UPDATED TABLES & ENTITIES:

Stores: Store Id, Address, Manager, Assigned Pharmacist, Assigned Doctor, Region Code

Warehouses: Warehouse Id, Address, Warehouse Manager, Current Stock, Capacity, Region Code

Region: Region Name, Region Code, Region Manager

Employees: EID, SSN, Address, Ph NO, Type, Location, Bank Account Number

Employee SSN: SSN, Name, Age, Gender

Employee Type: Type, Wage

Patients: PID, Ph NO, Address, SSN

Patients SSN: SSN**,** Name, Age, Gender

Patients Insurance: PID, INM

Insurance Details: INM, Name of Insurance, Amount, Date Claimed, Status

Prescription: PrescriptionID, DoctorID, PatientID, Date_Prescribed

Drug: Drug ID, Name, Price, Drug Type, Dosage, Manf By, Manf Date, Batch NO, Expiry Date

Inventory: Drug ID, Building ID, Current Stock

Logistics: Good, Date, Warehouse ID, Store ID, Quantity, Status

Sales: No of sales per month, Month, Year, Drug ID, Store ID

The New tables are those related to Employees, Patients, and Insurance.

- The Employee Personal Information like Name, Age, Gender is now stored in a new table Employee SSN as we can get the info using SSN.

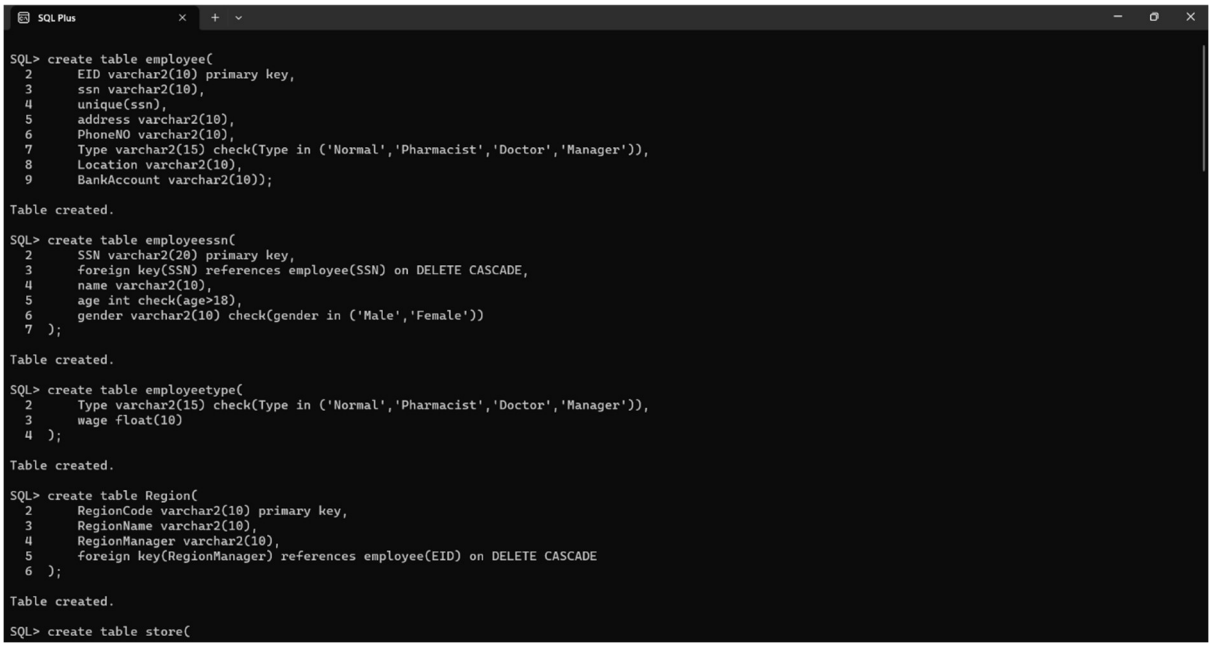
- The Wage of employee which depends on his type is also now stored in a separate table named Employee Type.

- Similar to employee table the personal information of patients is now stored in a separate table named Patients SSN.

- The Insurance table is split into two tables. One contains the Insurance Number and the corresponding Patient ID, whereas the other contains the details of the Insurance.

CREATION OF NEW TABLES:

Creation of Employee, Employees SSN, Employee Type and Region Tables:



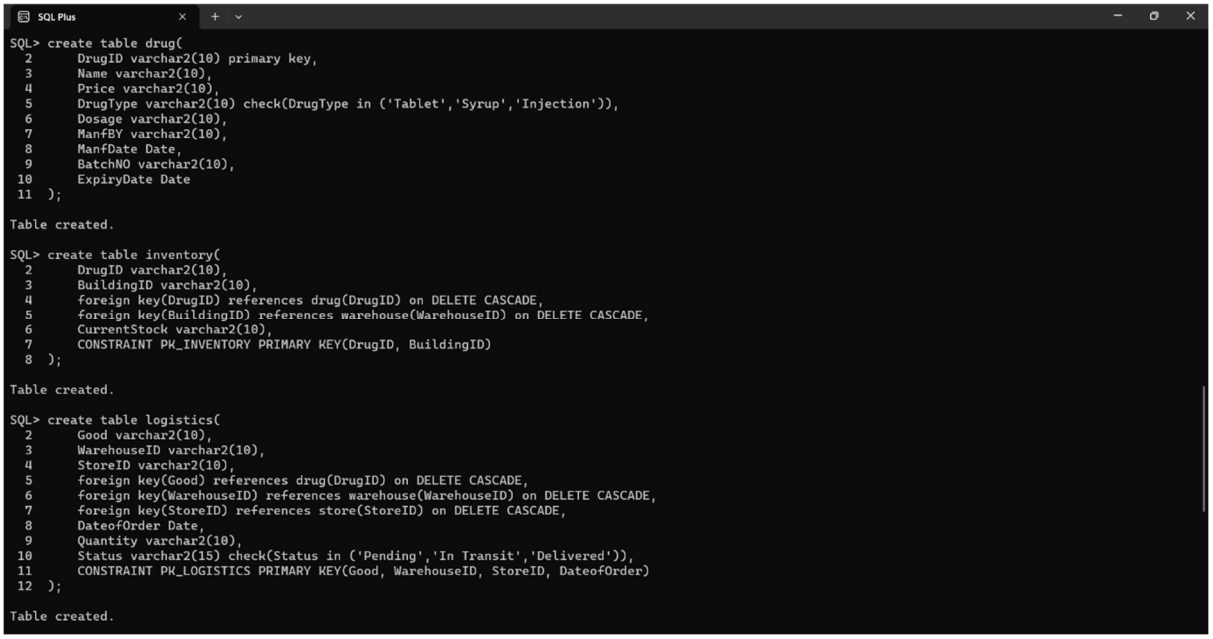
Creation of Stores, Warehouse and Patients Table:

```
SQL> create table store(
  2      StoreID varchar2(10) primary key,
  3      address varchar2(10),
  4      ManagerID varchar2(10),
  5      PharmacistID varchar2(10),
  6      foreign key(PharmacistID) references employee(EID) on DELETE CASCADE,
  7      DoctorID varchar2(10),
  8      foreign key(DoctorID) references employee(EID) on DELETE CASCADE,
  9      Region varchar2(10),
 10      foreign key(Region) references region(RegionCode) on DELETE CASCADE
 11  );

Table created.

SQL> create table warehouse(
  2      WarehouseID varchar2(10) primary key,
  3      address varchar2(10),
  4      ManagerID varchar2(10),
  5      foreign key(ManagerID) references employee(EID) on DELETE CASCADE,
  6      CurrentStock varchar2(10),
  7      Capacity varchar2(10),
  8      Region varchar2(10),
  9      foreign key(Region) references region(RegionCode) on DELETE CASCADE
 10  );

Table created.

SQL> create table Patients(
  2      PID varchar2(10) primary key,
  3      SSN varchar2(10),
  4      unique(SSN),
  5      PhoneNO varchar2(10),
  6      Address varchar2(10)
  7  );

Table created.
```

Creation of Patients SSN, Patients INS and Insurance Details Tables:

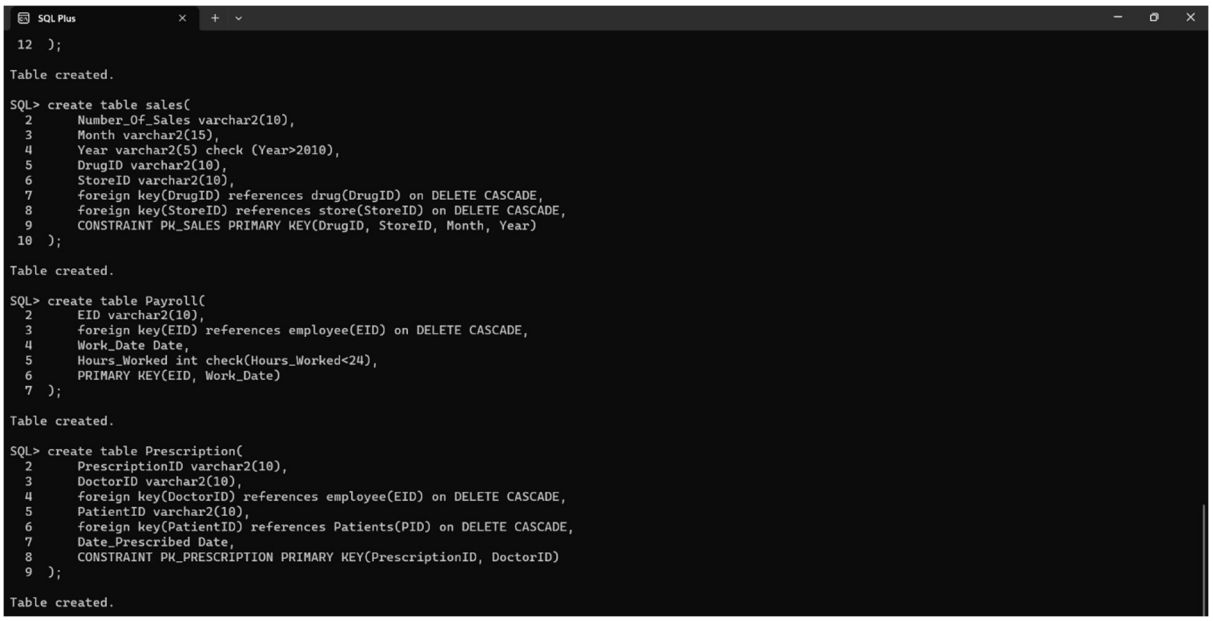
```
SQL> create table PatientSSN(
  2      SSN varchar2(10) primary key,
  3      foreign key(SSN) references Patients(SSN) on DELETE CASCADE,
  4      Age int,
  5      Gender varchar2(10) check(Gender in ('Male','Female'))
  6  );

Table created.

SQL> create table PatientsINS(
  2      InsuranceNUM varchar2(10),
  3      PID varchar2(10),
  4      CONSTRAINT PK_INSURANCE PRIMARY KEY(InsuranceNUM, PID)
  5  );

Table created.

SQL>
SQL> create table INSDetails(
  2      InsuranceNUM varchar2(10) primary key,
  3      Name varchar2(10),
  4      Amount varchar2(10),
  5      DateClaimed Date,
  6      Status varchar2(10) check(Status in ('Pending','Approved','Failed'))
  7  );

Table created.
```

Creation of Drug, Inventory and Logistics Tables:



```
SQL Plus

SQL> create table drug(
  2      DrugID varchar2(10) primary key,
  3      Name varchar2(10),
  4      Price varchar2(10),
  5      DrugType varchar2(10) check(DrugType in ('Tablet','Syrup','Injection')),
  6      Dosage varchar2(10),
  7      ManfBY varchar2(10),
  8      ManfDate Date,
  9      BatchNO varchar2(10),
 10      ExpiryDate Date
 11  );

Table created.

SQL> create table inventory(
  2      DrugID varchar2(10),
  3      BuildingID varchar2(10),
  4      foreign key(DrugID) references drug(DrugID) on DELETE CASCADE,
  5      foreign key(BuildingID) references warehouse(WarehouseID) on DELETE CASCADE,
  6      CurrentStock varchar2(10),
  7      CONSTRAINT PK_INVENTORY PRIMARY KEY(DrugID, BuildingID)
  8  );

Table created.

SQL> create table logistics(
  2      Good varchar2(10),
  3      WarehouseID varchar2(10),
  4      StoreID varchar2(10),
  5      foreign key(Good) references drug(DrugID) on DELETE CASCADE,
  6      foreign key(WarehouseID) references warehouse(WarehouseID) on DELETE CASCADE,
  7      foreign key(StoreID) references store(StoreID) on DELETE CASCADE,
  8      DateofOrder Date,
  9      Quantity varchar2(10),
 10      Status varchar2(15) check(Status in ('Pending','In Transit','Delivered')),
 11      CONSTRAINT PK_LOGISTICS PRIMARY KEY(Good, WarehouseID, StoreID, DateofOrder)
 12  );

Table created.
```

Creation of Sales, Payroll and Prescription Tables:



```
SQL Plus

 12  );

Table created.

SQL> create table sales(
  2      Number_Of_Sales varchar2(10),
  3      Month varchar2(15),
  4      Year varchar2(5) check (Year>2010),
  5      DrugID varchar2(10),
  6      StoreID varchar2(10),
  7      foreign key(DrugID) references drug(DrugID) on DELETE CASCADE,
  8      foreign key(StoreID) references store(StoreID) on DELETE CASCADE,
  9      CONSTRAINT PK_SALES PRIMARY KEY(DrugID, StoreID, Month, Year)
 10  );

Table created.

SQL> create table Payroll(
  2      EID varchar2(10),
  3      foreign key(EID) references employee(EID) on DELETE CASCADE,
  4      Work_Date Date,
  5      Hours_Worked int check(Hours_Worked<24),
  6      PRIMARY KEY(EID, Work_Date)
  7  );

Table created.

SQL> create table Prescription(
  2      PrescriptionID varchar2(10),
  3      DoctorID varchar2(10),
  4      foreign key(DoctorID) references employee(EID) on DELETE CASCADE,
  5      PatientID varchar2(10),
  6      foreign key(PatientID) references Patients(PID) on DELETE CASCADE,
  7      Date_Prescribed Date,
  8      CONSTRAINT PK_PRESCRIPTION PRIMARY KEY(PrescriptionID, DoctorID)
  9  );

Table created.
```