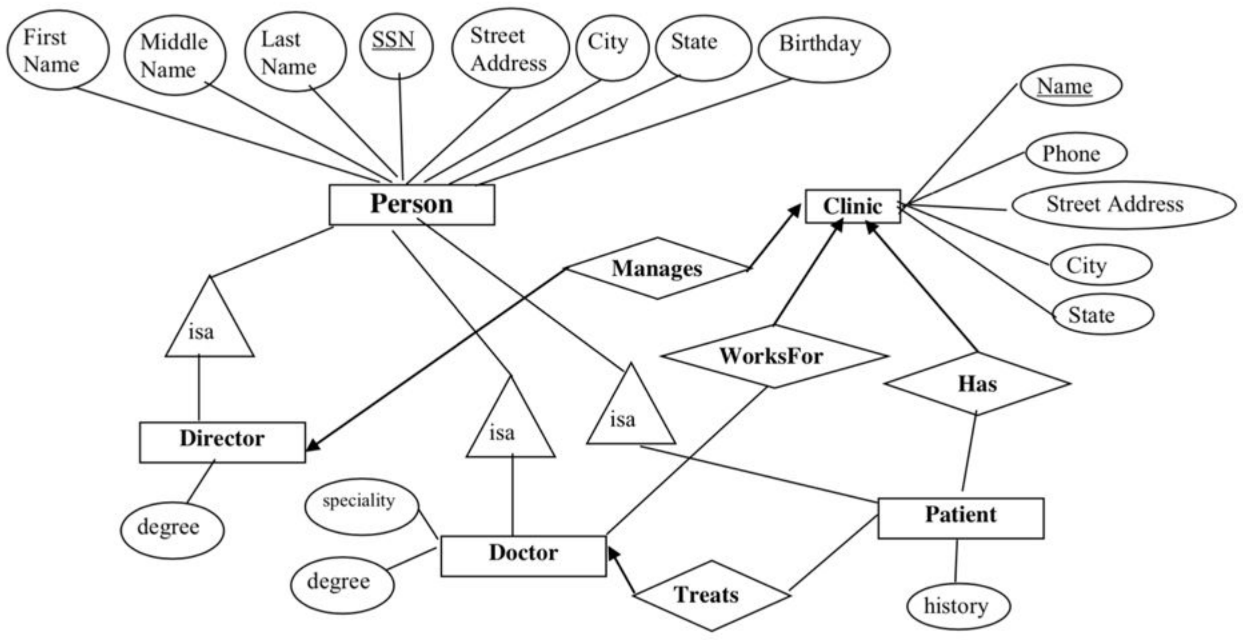
Chrystal Mingo

Database -- CSC 33600

Project 3

Overview:

Convert the following ER diagram into MySQL tables. Develop proper constraints and triggers to enforce database integrity.



Part One:

1) **All primary and foreign key constraints can be found in the tables below:**

Table for *Person:*

```
CREATE TABLE Person (
   First_Name VARCHAR(255),
   Middle_Name VARCHAR(255),
   Last_Name VARCHAR(255),
   SSN INTEGER,
   Street_Address VARCHAR(255),
   City VARCHAR(255),
   State VARCHAR(255),
   Birthday DATE,
   PRIMARY KEY(SSN),
   CONSTRAINT LNAME CHECK (State <>'California' OR State <> 'Texas' OR((State =
"California" OR State = "Texas") AND Last_Name = 'Smith'))
);
```

Table for *Doctor:*

```
CREATE TABLE Doctor (
    SSN INTEGER,
    Degree VARCHAR(255),
    Speciality VARCHAR(255) CHECK(Speciality IN('Family Practice', 'Internal Medicine',
'Pediatrics', 'Obstetrics', 'Gynecology')),
    PRIMARY KEY (SSN),
    FOREIGN KEY (SSN) REFERENCES Person(SSN)
);
```

Table for *Director:*

```
CREATE TABLE Director (
    SSN INTEGER,
    Degree  VARCHAR(255) CHECK(Degree IN (SELECT Degree FROM Doctor)),
    PRIMARY KEY(SSN),
    FOREIGN KEY (SSN) REFERENCES Person(SSN)
);
```

Table for *Patient:*

```
CREATE TABLE Patient (
    SSN INTEGER,
    History VARCHAR(255),
    PRIMARY KEY(SSN),
    FOREIGN KEY (SSN) REFERENCES Person(SSN)
);
```

Table for *Clinic:*

```
CREATE TABLE Clinic (
    Name VARCHAR(255),
    Phone INTEGER,
    Street_Address VARCHAR(255),
    City VARCHAR(255),
    State VARCHAR(255),
    PRIMARY KEY(Name)
);
```

Table for *Manages:*

```
CREATE TABLE Manages (
    Clinic_Name VARCHAR(255),
    Director_SSN INTEGER,
    PRIMARY KEY (Clinic_Name, Director_SSN),
    FOREIGN KEY (Clinic_Name) REFERENCES Clinic(Name),
    FOREIGN KEY (Director_SSN) REFERENCES Director(SSN)
);
```

Table for *Works_For:*

```
CREATE TABLE Works_For (
    Clinic_Name VARCHAR(255),
    Doctor_SSN INTEGER,
    PRIMARY KEY (Clinic_Name, Doctor_SSN),
    FOREIGN KEY (Clinic_Name) REFERENCES Clinic(Name),
    FOREIGN KEY (Doctor_SSN) REFERENCES Doctor(SSN)
);
```

Table for *Treats:*

```
CREATE TABLE Treats (
    Doctor_SSN INTEGER,
    Patient_SSN INTEGER,
    PRIMARY KEY (Doctor_SSN, Patient_SSN),
    FOREIGN KEY (Doctor_SSN) REFERENCES Doctor(SSN),
    FOREIGN KEY (Patient_SSN) REFERENCES Patient(SSN)
);
```

Table for *Clinic_Patient:*

```
CREATE TABLE Clinic_Patient (
    Clinic_Name VARCHAR(255),
    Patient_SSN INTEGER,
    PRIMARY KEY (Clinic_Name, Patient_SSN),
    FOREIGN KEY (Clinic_Name) REFERENCES Clinic(Name),
    FOREIGN KEY (Patient_SSN) REFERENCES Patient(SSN));
```

2) **The constraint is located in the Doctor table:** *CHECK(Speciality IN('Family Practice', 'Internal Medicine', 'Pediatrics', 'Obstetrics', 'Gynecology')),*
3) **The constraint is located in the Director table:** CHECK(Degree IN (SELECT Degree FROM Doctor))
4) **The constraint is located in the Person table:** *LNAME_CHECK CHECK (State <>'California' OR State <> 'Texas' OR((State = "California" OR State = "Texas") AND Last_Name = 'Smith'))*
5) **Audit all the additions of doctors using a trigger (the trigger is below):**

In order for the Trigger to work I needed to create another table and then wrote the trigger:

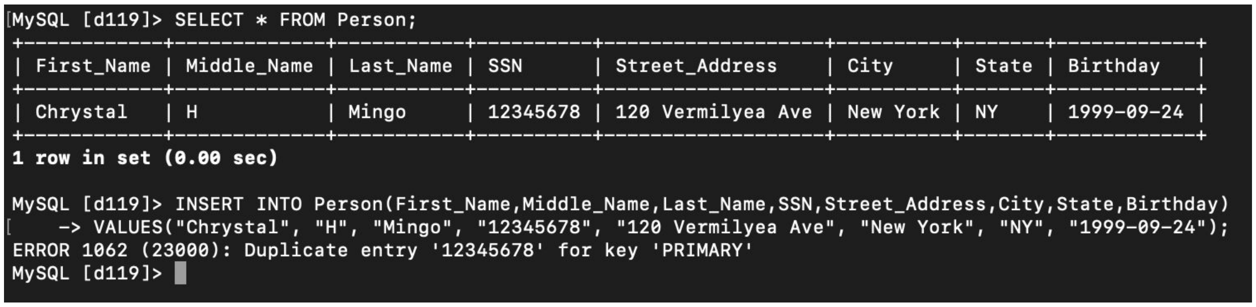
```
CREATE TABLE Doc_Date(
   SSN INTEGER,
   DateAdd DATE
);

CREATE TRIGGER doc_date ON Doc_Date
AFTER INSERT ON
FOR EACH ROW
BEGIN
INSERT INTO Doc_Date(SSN, DateAdd)
VALUES(Doctor.SSN,CURDATE())
END;
```

Part Two:

1) **Insert a person record to demonstrate the utilization of the primary key constraint**

To show that the Primary Key constraint for Person table works, the database should reject inserts of the same SSN. As seen in figure1.

Figure1:

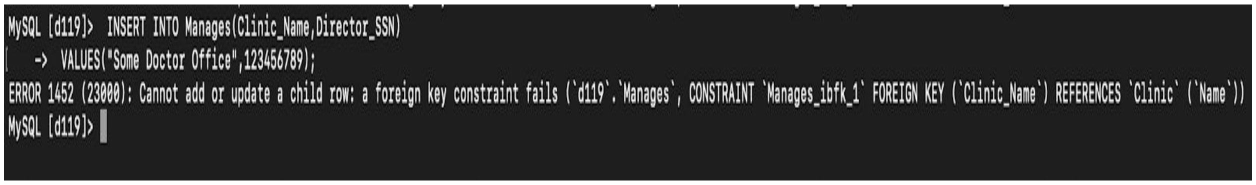


```
[MySQL [d119]> SELECT * FROM Person;
+------------+-------------+-----------+----------+-------------------+----------+-------+------------+
| First_Name | Middle_Name | Last_Name | SSN      | Street_Address    | City     | State | Birthday   |
+------------+-------------+-----------+----------+-------------------+----------+-------+------------+
| Chrystal   | H           | Mingo     | 12345678 | 120 Vermilyea Ave | New York | NY    | 1999-09-24 |
+------------+-------------+-----------+----------+-------------------+----------+-------+------------+
1 row in set (0.00 sec)

MySQL [d119]> INSERT INTO Person(First_Name,Middle_Name,Last_Name,SSN,Street_Address,City,State,Birthday)
[    -> VALUES("Chrystal", "H", "Mingo", "12345678", "120 Vermilyea Ave", "New York", "NY", "1999-09-24");
ERROR 1062 (23000): Duplicate entry '12345678' for key 'PRIMARY'
MySQL [d119]>
```

2) **Insert a manages record to demonstrate the utilization of the foreign key constraint**

To show that the Foreign Key for the Manage table works, the database should reject any inserts that involves a director or clinic that does not exist. As seen in figure2.

Figure2:



```
MySQL [d119]>  INSERT INTO Manages(Clinic_Name,Director_SSN)
[    -> VALUES("Some Doctor Office",123456789);
ERROR 1452 (23000): Cannot add or update a child row: a foreign key constraint fails (`d119`.`Manages`, CONSTRAINT `Manages_ibfk_1` FOREIGN KEY (`Clinic_Name`) REFERENCES `Clinic` (`Name`))
MySQL [d119]>
```

Part Three:

1) **Assertion:**

```
CREATE ASSERTION one NOT EXISTS (
   SELECT Patient.Last_Name, Patient.Birthday
   FROM Patient A, Patient B
   WHERE A.Last_Name = B.Last_Name AND A.Birthday = B.Birthday
   AND (
      NOT EXISTS (
         SELECT Doctor_SSN, Patient_SSN
         FROM Treats
         WHERE (Treats.Patient_SSN = A.SSN AND Treats.Patient_SSN = B.SSN))));
```

2) **Trigger:**

```
CREATE TABLE Doctor_Has_Too_Many_Patients(
   SSN INTEGER
);
```

```
CREATE TRIGGER Too_Many_Patients
AFTER INSERT ON Patient
REFRENCEING NEW ROW AS new
BEGIN
IF ((SELECT Count(History) FROM Patients WHERE SSN = new.SNN) >= 5)
INSERT INTO Doctor_Has_Too_Many_Patients(SSN) SELECT Doctor_SSN FROM Treats
WHERE Patient_SSN = new.SSN;
END;
```