Assignment 1: ER Diagram and Relational Model
Due: Oct 7, Sat, 11:59pm

Question 1 (18 marks, 3 marks each). For each of the following statements, represent it using CREATE TABLE statement:

1. **Patients attend doctors. You want to store the attending information and the date of attending. Patients identified by pid. Doctors is identified by did.**

```
CREATE_TABLE Doctors (
        did CHAR(10),
        name CHAR(15),
        dateOfBirth DATE,
        specialty CHAR(15),
        PRIMARY KEY(did)
)

CREATE_TABLE Patients (
        pid CHAR(10),
        name CHAR(15),
        dateOfBirth DATE,
        PRIMARY KEY (pid)
)

CREATE_TABLE Attending (
        attendingDate DATE,
        pid CHAR(10),
        did CHAR(10),
        PRIMARY KEY (attendingDate, pid, did),
        FOREIGN KEY (pid) REFERENCES Patients,
        FOREIGN KEY (did) REFERENCES Doctors
)
```

2. **Continue 1, each patient attends doctors at most once.**

```
CREATE_TABLE Doctors (
        did CHAR(10),
        name CHAR(15),
        dateOfBirth DATE,
        specialty CHAR(15),
        PRIMARY KEY(did)
)

CREATE_TABLE Patients (
        pid CHAR(10),
        name CHAR(15),
        dateOfBirth DATE,
        PRIMARY KEY (pid)
)
```

```
CREATE_TABLE Attending (
        attendingDate DATE,
        pid CHAR(10),
        did CHAR(10),
        PRIMARY KEY (pid, did),
        FOREIGN KEY (pid) REFERENCES Patients,
        FOREIGN KEY (did) REFERENCES Doctors
)
```

3. **Continue 2, each patient attends doctors at least once.**

```
CREATE_TABLE Doctors (
        did CHAR(10),
        name CHAR(15),
        dateOfBirth DATE,
        specialty CHAR(15),
        PRIMARY KEY(did)
)

CREATE_TABLE Patients (
        pid CHAR(10),
        name CHAR(15),
        dateOfBirth DATE,
        FOREIGN KEY (attendingDate) REFERENCES Attending,
        PRIMARY KEY (pid, attendingDate)
)

CREATE_TABLE Attending (
        attendingDate DATE,
        pid CHAR(10),
        did CHAR(10),
        PRIMARY KEY (pid, did),
        FOREIGN KEY (pid) REFERENCES Patients,
        FOREIGN KEY (did) REFERENCES Doctors NOT NULL
)
```

4. **Continue 1, only existing doctors can be attended by a patient.**

```
CREATE_TABLE Doctors (
        did CHAR(10),
        name CHAR(15),
        dateOfBirth DATE,
        specialty CHAR(15),
        PRIMARY KEY(did)
)

CREATE_TABLE Patients (
        pid CHAR(10),
        name CHAR(15),
        dateOfBirth DATE,
        attendingID CHAR(10) NOT NULL,
```

```
        FOREIGN KEY (attendingID) REFERENCES Attending,
        PRIMARY KEY (pid)
)

CREATE_TABLE Attending (
        attendingID INT,
        attendingDate DATE,
        pid CHAR(10),
        did CHAR(10),
        PRIMARY KEY (pid, did),
        FOREIGN KEY (pid) REFERENCES Patients,
        FOREIGN KEY (did) REFERENCES Doctors NOT NULL
)
```

5. **Continue 1, every doctor must be attended by a patient.**

```
CREATE TABLE Attending (
        attendingDate DATE NOT NULL,
        pid CHAR(10) NOT NULL,
        did CHAR(10) NOT NULL,
        PRIMARY KEY (AttendingDate),
        FOREIGN KEY (pid) REFERENCES Patients(pid),
        FOREIGN KEY (did) REFERENCES Doctors(did)
    )
```

6. **Continue 1, each of (Name, Address) and (Name,Age) uniquely identifies a patient.**

Question 2 (18 marks, 3 marks each) A university database contains information about professors (identified by ssn) and courses (identified by courseid). Professors teach courses; each of the following situations concerns the Teaches relationship set. For each situation, draw an ER diagram that describes it (assuming that no further constraints hold) and using CREATE TABLE to model the information in the ER diagram.

1. **Professors can teach the same course in several semesters, and each offering must be recorded.**

```
CREATE_TABLE Professors (
        ssn CHAR(10),
        name CHAR(20),
        PRIMARY KEY (ssn)
)

CREATE_TABLE Courses (
        courseid INTEGER,
        name CHAR(10),
        PRIMARY KEY (courseid)
)

CREATE_TABLE Teaches (
        ssn CHAR(10),
        courseid INTEGER,
        semester DATE,
        PRIMARY KEY (ssn, courseid, semester),
        FOREIGN KEY (ssn) REFERENCES Professors,
```

```
        FOREIGN KEY(courseid) REFERENCES Courses
)
```

2. **Professors can teach the same course in several semesters, and only the most recent such offering needs to be recorded. (Assume this condition applies in all subsequent questions.)**

```
CREATE_TABLE Professors (
        ssn CHAR(10),
        name CHAR(20),
        PRIMARY KEY (ssn)
)

CREATE_TABLE Courses (
        courseid INTEGER,
        name CHAR(10),
        PRIMARY KEY (courseid)
)

CREATE_TABLE Teaches (
        ssn CHAR(10),
        courseid INTEGER,
        semester DATE,
        PRIMARY KEY (ssn, courseid),
        FOREIGN KEY (ssn) REFERENCES Professors,
        FOREIGN KEY(courseid) REFERENCES Courses
)
```

3. **Every professor must teach some course.**

```
CREATE_TABLE Professors (
        ssn CHAR(10),
        name CHAR(20),
        courseid INTEGER NOT NULL,
        PRIMARY KEY (ssn),
        FOREIGN KEY(courseid) REFERENCES Courses
)

CREATE_TABLE Courses (
        courseid INTEGER,
        name CHAR(10),
        PRIMARY KEY (courseid)
)

CREATE_TABLE Teaches (
        ssn CHAR(10),
        courseid INTEGER NOT NULL,
        semester DATE,
        PRIMARY KEY (ssn, courseid),
        FOREIGN KEY (ssn) REFERENCES Professors,
        FOREIGN KEY(courseid) REFERENCES Courses
)
```

4. **Every professor teaches exactly one course (no more, no less).**

```
CREATE_TABLE Professors (
        ssn CHAR(10),
        name CHAR(20),
        PRIMARY KEY (ssn)
)

CREATE_TABLE Courses (
        courseid INTEGER,
        name CHAR(10),
        PRIMARY KEY (courseid)
)

CREATE_TABLE Teaches (
        ssn CHAR(10),
        courseid INTEGER,
        semester DATE,
        PRIMARY KEY (ssn),
        FOREIGN KEY (ssn) REFERENCES Professors,
        FOREIGN KEY(courseid) REFERENCES Courses
)
```

5. **Every professor teaches exactly one course (no more, no less), and every course must be taught by some professor.**

```
CREATE_TABLE Professors (
        ssn CHAR(10),
        name CHAR(20),
        courseid INTE   GER NOT NULL UNIQUE,
        FOREIGN KEY(courseid) REFERENCES Courses,
        PRIMARY KEY (ssn)
)

CREATE_TABLE Courses (
        courseid INTEGER,
        name CHAR(10),
        ssn CHAR(10),
        FOREIGN KEY (ssn) REFERENCES Professors,
        PRIMARY KEY (courseid)
)
```

6. **Now suppose that certain courses can be taught by a team of professors jointly, but it is possible that no one professor in a team can teach the course. Model this situation, introducing additional entity sets and relationship sets if necessary.**

```
CREATE_TABLE Professors (
        ssn CHAR(10),
```

```
        teamid INT,
        name CHAR(20),
        PRIMARY KEY (ssn)
)

CREATE_TABLE Courses (
        courseid INTEGER,
        name CHAR(10),
        PRIMARY KEY (courseid)
)

CREATE_TABLE Teaches (
        ssn CHAR(10),
        courseid INTEGER,
        semester DATE,
        PRIMARY KEY (teamID),
        FOREIGN KEY (ssn) REFERENCES Professors,
        FOREIGN KEY(courseid) REFERENCES Courses
)

CREATE_TABLE BelongsTo (
        ssn CHAR(10),
        teamid CHAR(10),
        PRIMARY KEY(ssn, teamid),
        FOREIGN KEY(ssn) REFERENCES Professors,
        FOREIGN KEY(teamid) REFERENCES Team
)
```

Question 3 (25 marks). You are asked to set up a database, ArtBase, for art galleries. This database will capture all the information that galleries need to
Maintain:

- Galleries keep information about artists, their names (which are unique), birthplaces, age, and style of art.
- For each piece of artwork, the artist, the year it was made, its unique title, its type of art (e.g., painting, lithograph, sculpture, photograph), and its price must be stored.
- Pieces of artwork are also classified into groups of various kinds, for example, portraits, still lifes, works by Picasso, or works of the 19th century; a given piece may belong to more than one group.
- Each group is identified by a name (like those above) that describes the group.
- Finally, galleries keep information about customers. For each customer, galleries keep their unique name, address, total amount of dollars they have spent in the gallery (very important!), and the artists and groups of art that each customer tends to like.

(1) Draw the ER diagram for the database (10 marks)

(2) Represent the data in the ER diagram using CREATE TABLE statements (10 marks)

```
CREATE_TABLE Artists (
        Birthplace CHAR(255),
        Age INTEGER,
        Style CHAR(255),
```

```
        ArtistName CHAR(255),
        PRIMARY KEY (ArtistName)
)

CREATE_TABLE Artworks (
        Title CHAR(255),

        YearMade DATE,
        ArtType CHAR(255),
        Price INT,
        PRIMARY KEY (Title),
        FOREIGN KEY Artist REFERENCES Artists
)

CREATE_TABLE Customers (
        CustomerName CHAR(255),
        Address CHAR(255),
        TotalSpent DECIMAL,
        PRIMARY KEY(CustomerName)
)

CREATE_TABLE ArtworkGroups (
        Title CHAR(255),
        GroupName CHAR(255),
        PRIMARY KEY (Title, GroupName),
        FOREIGN KEY(Title) REFERENCES Artworks,
        FOREIGN KEY (GroupName) REFERENCES Groups
)




CREATE_TABLE CustomerArtistPreferences (
        CustomerName CHAR(255),
        ArtistName CHAR(255),
        PRIMARY KEY (CustomerName, Artist Name),
        FOREIGN KEY (CustomerName) REFERENCES Customers
        FOREIGN KEY (ArtistName) REFERENCES Artists
)

CREATE TABLE CustomerGroupsPreferences (
        CustomerName CHAR(255),
        GroupName CHAR(255),
        PRIMARY KEY (CustomerName, GroupName),
        FOREIGN KEY (CustomerName) REFERENCES Customers,
        FOREIGN KEY (GroupName) REFERENCES Groups
)
```

(3) If Artwork as a weak entity set with the partial key title and the owner entity set Artist, describe the changes needed in (1) and (2) (5 marks)

In (1), we Artwork becomes a week entity and so the relationship between artists and artworks is identifying. It does not need a primary key.

In (2), we would make the following change to ensure uniqueness and imply that an artwork is identified by its title and the name of its artist.

```
CREATE TABLE Artworks (
    Title CHAR(255),
    Artist CHAR(255),
    YearMade DATE,
    ArtType CHAR(255),
    Price INT,
    PRIMARY KEY (Title, Artist),
    FOREIGN KEY (Artist) REFERENCES Artists
)
```