1. **(3 pts.)** In this problem, we are going to design a simple movie database. The database will include information about movies, critical reviews, and people in Hollywood (actors, directors, and producers).

   - People in Hollywood all have a name, birth date, and social security number (unique by Federal Law). We care about three types of people: actors, directors, and producers. We assume that these categories are mutually exclusive (e.g., an actor cannot be a director or a producer). Actors have a salary, directors are paid by a percentage of the movies’ gross income (each director specifies his/her own percentage, which is the same across movies), and producers have capital.

   - Information about movies includes the title, rating, release date, and gross income. We assume that no two films with the same title will be released on the same date. Actors star in movies, directors direct movies, and producers produce movies. Actors star in many movies, and movies must have at least one actor. Movies must have exactly one director. Each movie also has at least one producer; each producer may produce multiple films.

   - Information about reviews includes the name of the movie critic, a score, and the text of the review. Each review is associated with a specific movie. We assume that movie critics all have unique names and that a critic will write at most one review per movie. We also assume that when we delete a movie, we do not need to retain its reviews.

What you need to do:

   a. **(2.5 pts.)** Draw an ER diagram for the movies database using the notation shown in class. Include the constraints and requirements shown above. Please state any assumptions you make, but do not introduce constraints not listed in the problem definition above.

   b. **(0.5 pts.)** Change the diagram so that each actor must star in at least one movie.
2. (4 pts.) Translate the following ER diagram into the relational model by writing the SQL table creation commands necessary to define the relations. Be sure to include primary keys and foreign keys where applicable.
3. (3 pts.) In this question we ask you to apply constraint checking to a database. The database manages customer orders, a common commercial application of database systems, and consists of three relations, defined as follows:

CREATE TABLE Customers(
    id INTEGER,
    name CHAR(40),
    email CHAR(64),
    PRIMARY KEY (id),
    UNIQUE (email))

CREATE TABLE Orders(
    id INTEGER,
    custID INTEGER,
    saledate DATE,
    PRIMARY KEY (id),
    FOREIGN KEY (custID) REFERENCES Customers
    ON DELETE NO ACTION
    ON UPDATE CASCADE)

CREATE TABLE LineItems(
    id INTEGER,
    orderID INTEGER,
    item CHAR(64),
    quantity INTEGER,
    cost REAL,
    PRIMARY KEY (id, orderID),
    FOREIGN KEY (orderID) REFERENCES Orders
    ON DELETE CASCADE
    ON UPDATE CASCADE)

Customers

<table>
<thead>
<tr>
<th>id</th>
<th>name</th>
<th>email</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Thomas Jefferson</td>
<td><a href="mailto:tj@monticello.com">tj@monticello.com</a></td>
</tr>
<tr>
<td>2</td>
<td>George Washington</td>
<td><a href="mailto:gw@vernon.net">gw@vernon.net</a></td>
</tr>
<tr>
<td>3</td>
<td>John Kennedy</td>
<td><a href="mailto:jfk@hyannis.org">jfk@hyannis.org</a></td>
</tr>
<tr>
<td>4</td>
<td>George H. W. Bush</td>
<td><a href="mailto:bush41@kennebunkport.us">bush41@kennebunkport.us</a></td>
</tr>
</tbody>
</table>

Orders

<table>
<thead>
<tr>
<th>id</th>
<th>custID</th>
<th>saledate</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>03/05/1801</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>07/04/1961</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>04/06/2005</td>
</tr>
</tbody>
</table>

LineItems

<table>
<thead>
<tr>
<th>id</th>
<th>orderID</th>
<th>item</th>
<th>quantity</th>
<th>cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>Quills</td>
<td>12</td>
<td>3.50</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>Parchment</td>
<td>5</td>
<td>3.88</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>Pens</td>
<td>7</td>
<td>23.00</td>
</tr>
<tr>
<td>1</td>
<td>3</td>
<td>Broccoli</td>
<td>3</td>
<td>4.20</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>Sushi</td>
<td>4</td>
<td>3.90</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>Cuban Cigars</td>
<td>12</td>
<td>65.00</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>Tapered Candles</td>
<td>12</td>
<td>6.99</td>
</tr>
</tbody>
</table>
What you need to do: Apply each of the following commands to the database in the order they are listed. If a command fails because of a constraint violation, identify which constraint is violated and explain how. Show the state of the three relations after each legal command (i.e., each command that does not violate any constraints) has been applied.

```
INSERT INTO Customers (id, name, email)
VALUES (6, 'Bill Clinton', 'wjc@whitewater.net')

UPDATE Customers C
SET C.id = 6
WHERE C.name = 'George Washington'

UPDATE Customers C
SET C.email = 'gw@vernon.net'
WHERE C.id = 6

DELETE FROM Orders O
WHERE O.id = 3

DELETE FROM Customers C
WHERE C.id = 3

UPDATE Orders O
SET O.id = 10
WHERE O.id = 1

INSERT INTO LineItems (id, orderID, item, quantity, cost)
VALUES (1, 5, 'Socks', 5, 7.99)

DELETE FROM Orders O
WHERE O.saledate > 01/01/1900

DELETE FROM Customers C
WHERE C.id = 3
```