CS143: Homework #4 (Normalization and Integrity)

1. Suppose that we decompose the schema $R(A, B, C, D, E, F)$ into (A, B, C, F) and (A, D, E). When the following set of functional dependencies hold, is the decomposition lossless?
   $A \rightarrow BC, CD \rightarrow E, B \rightarrow D, E \rightarrow A$
   Explain your answer.

2. List non-trivial functional dependencies satisfied by the following relation. You do not need to find all functional dependencies. It is enough to identify a set of functional dependencies that imply all functional dependencies that is satisfied by the relation.

A	B	C
a_1	b_1	c_2
a_1	b_1	c_2
a_2	b_1	c_1
a_2	b_1	c_3

3. Assume *Student* and *Class* entity sets that we have used in the class. The *Student* and *Class* sets are connected by Take relationship set. We now convert the Take relationship set into a table **Take(sid, dept, cnum)** using our standard translation algorithm, where sid is the key for a student and (dept, cnum) is the key for a class.
   Explain how functional dependencies can be used to indicate the following:

   (a) A one-to-one relationship exists between entity sets *Student* and *Class*.

   (b) A many-to-one relationship exists between entity sets *Student* and *Class*.

4. Assume the following set of functional dependencies hold for the relation R(A, B, C, D, E):
   $A \rightarrow BC$, $CD \rightarrow E$, $B \rightarrow D$, $E \rightarrow A$

   (a) Is A a key for R? Explain your answer.

   (b) Is BC a key for R? Explain your answer.

5. Assume the following set of functional dependencies hold for the relation *R(A, B, C, D, E, F)*:
   $A \rightarrow BC$, $C \rightarrow E$, $B \rightarrow D$
   Is it in **BCNF**? Explain your answer. If it is not, normalize it into a set of relations in **BCNF**.

6. Assume the following tables for this problem:

```
ComputerProduct(manufacturer, model, price)
Desktop(model, speed, ram, hdd)
Laptop(model, speed, ram, hdd, weight)
```

   A computer product is either a desktop or a laptop.

   (a) Using a `CHECK` constraint on the `Laptop` table, express the constraint that a laptop cannot have weight larger than 5kg and the weight must be greater than 0kg. You do not need to show the entire `CREATE TABLE` statement. Show only the `CHECK` constraint part in the `CREATE TABLE` statement.

(b) Write a trigger to replace the `CHECK` constraint in (a), so that when trying to add a laptop with weight larger than 5kg or smaller and equal to 0kg, the tuple is still inserted, but the value of the "weight" attribute is set to `NULL`.

7. All employees at the ABC corporation have to swipe their identification cards before they leave their office, so that the company can keep track of when each employee leaves work every day.

```
Employee(eid, name, salary)
LeavingTime(eid, date, time)
```

The above database is maintained to record this information. Whenever an employee swipes her card, her leaving time is recorded in the `LeavingTime` table. The underlined attributes represent the primary key of each table. LeavingTime.eid is a foreign key to Employee.eid.

(a) Write down the SQL `CREATE TABLE` statements to create the above two tables with `PRIMARY KEY` and `FOREIGN KEY` constraints

(b) Write down the SQL statement issued to the database when the employee 143 swipes her card at 4PM on 4/1/2015.

(c) If an employee swipes her card at the exit, stays for another hour, swipes her card again and leave, what would happen?

(d) Assume that our database programmer forgot to specify the primary key constraint when he created the `LeavingTime` table. Now that the database has been in operation for a while and the programmer has to "fix" the database by deleting all tuples that would not have been there if the primary key constraint had been specified. Assume that tuples in the `LeavingTime` table do not change their values once they are inserted. Also, swiping an employee's card is the only way to insert a new tuple to the table. Write a SQL statement that performs this "clean-up" operation.

8. Consider the table `R(A, B)`, which currently has only one tuple (1,0). Assume that the following trigger has already been created for the database.

```
CREATE TRIGGER Times3
AFTER UPDATE ON R
   REFERENCING NEW ROW AS n
   FOR EACH ROW
WHEN (n.B < 10)
BEGIN
   UPDATE R SET B=B*3 WHERE A=n.A;
   INSERT INTO R VALUES(100, 0);
END
```

List all tuples in the table `R` after the following update statement is executed:

```
UPDATE R SET B=3 WHERE A=1
```