Exercise I. Consider the following database schema for a social network application.

'User(id, name)' the user whose name is 'name' has id 'id'.

'Friend(id1, id2)' the user with id 'id1' has user with id 'id2' as a friend.

'id' is the primary key of the user relation.
'{id1, id2}' is the primary key of the friend relation.
'id1' in the friend relation references the user relation.
'id2' in the friend relation references the user relation.
There are no other constraints.

Note that the friendship relationship is not necessarily symmetric, reflexive or transitive.

Question 1. Which of the following operations on the database may violate a constraint on the friend table?

a) Add a friend to user with id 123X.
b) Remove a friend of user 123X.
c) Add a new user in the user table.
d) All of the above.
e) None of the above.

Question 2. Which of the following operations on the database may violate a constraint on the user table?

a) Add a user.
b) Remove user 123X.
c) Change the name of user 123X.
d) All of the above.
e) None of the above.

Question 3. Which of the following operations may violate a constraint on the user or the friend table?

a) Delete all friends of user 123X.
b) Delete a user.
c) Change the name of user 123X.
d) All of the above.
e) None of the above.

Question 4. Which of the following operations may violate a constraint on both tables: user and friend?

a) Change the name of a user.
b) Change the id to null for user 123X in table user.
c) Change id2 to a null value for all friends of 123X in table friend.
d) All of the above.
e) None of the above.

Question 5. Find the names of the different users who are their own friend (notice that different users may have the same name).

Question 6. Find the names of the different users who have some symmetrical friendships (notice that different sers may have the same name).

Question 7. Find for each user the total number of different friends. Print the name of the user and the total number. (notice that different users may have the same name).

Question 8. Find the names of different users who have more than two friends. Use aggregates. (notice that different users may have the same name).

Exercise II.

Nang-Si-Da Resort has several villages on the island of Koh Lek. For each village we record its name, the name of the village manager and the telephone number of the reception office of the village. Each village has a different name, a different manager and a different telephone. For instance, Mr. Sak is the manager of Deedee Village. The telephone number of Deedee village is 651652.

Each village consists of several bungalows (every village has at least one bungalow). Each bungalow in a given village has a unique number (unique within the village) and a type. For instance, bungalow number 5 in Dokmai village is a family suite.

We record the name, age, passport number and nationality of each customer. For instance, Mr. Schmidt is 58 years hold. His passport number is G1234D. He is German.

We record for each customer booking a bungalow the check-in and check-out dates. For instance, Mr. Franscesci booked a bungalow from 12 December 2013 to 25 December 2013.

A paying customer may book several bungalows for the same or for overlapping periods. For instance, Mr. Franscesci also booked a bungalow from 23 December 2013 to 25 December 2013 and another one from 2 January 2014 to 5 January 2014. For the sake of simplicity we do not check for overbooking. For instance, Mr. Franscesci and Mr. Tan booked the same bungalow on 24 December 2013.

Your company Araikodai Pte Ltd has been commissioned to design and implement the database application for the resort.

Question 9. Draw the ER diagram.

Question 10. Give the SQL DDL statements.

Some Answers

Exercise I

Answer key for questions 1 to 4: (a,a,b,b)

Question 5.

```
SELECT u.name
FROM user u, friend f
WHERE u.id=f.id1 AND f.id1=f.id2;
% there is no need for DISTINCT or GROUP BY
```

Question 6.

```
SELECT u.name
FROM user u, friend f1, friend f2
WHERE u.id=f1.id1 AND f1.id1=f2.id2 AND f1.id2=f2.id1;
GROUP BY u.id, u.name; % Only GROUP BY is possible here.
```

Question 7.

```
SELECT u.name, COUNT(*)
FROM user u, friend f
WHERE u.id=f.id1
GROUP BY u.id, u.name;
```

Question 8.

```
SELECT u.name,
FROM user u, friend f
WHERE u.id=f.id1
GROUP BY u.id, u.name;
HAVING COUNT(*) > 1;
```

Exercise II

Answers not provided. Feel free to share your answers on the forum.