Part I Schema Understanding

1.Is friendship between two users (stored in "Friendship Table") directed (Yes/No)? Why?

Yes, the friendship between two users stored in Friendship Table is directed. The attributes Follower and Followee in our Friendship relation only reference to the UserId attribute of User relation. And we know that each tuple in the Friendship relation can be interpreted as a fact, so for instance if a tuple in “Friendship Table” has two users, user A is a follower and user B is a followee, that doesn’t mean we can reverse these two users by their attributes Followee and Follower since the reverse situation may not exist.

2. Does a group owner have to be a member of his/her group?(Yes/No)? Why?

No, since the attribute UserId of GroupMembership and Owner of GroupInfo both are foreign keys to the attribute UserId of User, which means they only reference to the User relation. So there is no any condition between “GroupInfo.Owner” and “GroupMembership.UserId” requiring the UserId value of GroupMembership in every tuple must match the Owner value of some tuple in the GroupInfo relation, they can link to different independent users. Thus, a group owner does not have to be a member of his/her group.

3. Can we have two different users with the same last name?(Yes/No) Why?

Yes, since the only key attribute in the User relation is UserId, which can be used to identify uniquely each tuple in the User relation. But the attribute LastName in the User relation is not a key, which means two distinct tuples in some state of the User relation can have the same value for the LastName attribute. Thus, we can have two different users with the same last name.

4. If we need to add a new type of object to the database which table we should use?

The ObjectType table should be used to add a new type of object to the database. By creating a new tuple in ObjectType, we can specify a new ObjectTypeID(primary key), ObjectDescription and Type. Also, the Object table need to be updated correspondingly because the attribute ObjectType of Object is a foreign key to the attribute ObjectTypeID of ObjectType.

Part II SQL Queries

1.Find the distribution of users by Gender in the social network. (8 pts)

```
SELECT Gender, Count(*) AS Distribution
FROM User
GROUP BY Gender
```

2.Find the first name and last name of users and the names of the groups they are members of, who are following user with id 2. (12 pts)

```
SELECT U.Name, U.LastName, GI.Description
FROM (((Friendship AS F JOIN User AS U ON (U.UserId=F.Follower AND F.Followee = 2))
        LEFT JOIN GroupMembership AS GM ON F.Follower=GM.UserId)
        LEFT JOIN GroupInfo AS GI ON GM.GroupId=GI.GroupID)
```

3.Find the list of users (UserId, Name, LastName) and also their group descriptions if they own any. Note that each user may own more than one group. (8 pts)

```
SELECT U.UserId, U.Name, U.LastName, GI.Description
FROM User AS U LEFT JOIN GroupInfo AS GI ON U.UserId=GI.Owner
```

4.Find the list of all users (UserId) which are influential. Influential users are those who have more than two followers. Note: Do not exclude the ended friendships from the results. (10 pts)

```
SELECT Followee AS UserId
FROM Friendship AS F
GROUP BY Followee
HAVING COUNT(*) > 2;
```

5.Find the last names of all users who have posted both images and videos. (15 pts)

```
SELECT U.LastName
FROM (Object AS O LEFT JOIN  ObjectType AS OT ON O.ObjectType=OT.ObjectTypeID)
LEFT JOIN User AS U ON O.ObjOwner=U.UserID
WHERE OT.Type="Text" OR OT.Type="Image"
GROUP BY O.ObjOwner
HAVING COUNT(DISTINCT O.ObjectType) = 2
```

6.Find the last names of all users who have liked more than one object. For each user, besides the last name show his/her number of likes. Order the results by number of likes in decreasing order. (15 pts)

```
SELECT U.LastName, COUNT(*) AS Likes
FROM ((Activity AS A LEFT JOIN User AS U ON A.UserId = U.UserId)
LEFT JOIN ActivityType as AT ON  A.ActType = AT.ActId)
WHERE AT.ActDescription="Like"
GROUP BY A.UserId
HAVING COUNT(DISTINCT A.ObjectId) > 1
ORDER BY COUNT(*) DESC
```

7.Find the number of all the different object types posted by user with id 4. (8 pts)

```
SELECT COUNT( DISTINCT O.ObjectType ) AS TOTAL
FROM Object AS O
WHERE O.ObjOwner=4
```