CIS761– Database System Concepts **Name:_______________**

SQL Assignment 2 (20 points) – due Friday, September 13th at 11:59PM

Collaboration policy*: This is an individual assignment. You are allowed to discuss the assignment with your colleagues, but your submission should reflect your own work. Sharing or copying code is not permitted.*

Assignment objectives: create a relational **social network** database, populate the database and practice advanced SQL queries, triggers and data modification statements on this database. The database and its schema are described in what follows.

Students at your hometown high school have decided to organize their social network using databases. Here's the schema they have used, together with the corresponding English description for each relation:

Highschooler (ID, name, grade) -- There is a high school student with unique *ID* and a given *first name* in a certain *grade.*

Friend (ID1, ID2) --The student with *ID1* is friends with the student with *ID2*. Friendship is mutual, so if (123, 456) is in the Friend table, so is (456, 123).

Likes (ID1, ID2) --The student with *ID1* likes the student with *ID2*. Liking someone is not necessarily mutual, so if (123, 456) is in the Likes table, there is no guarantee that (456, 123) is also present.

An SQL script for creating and populating the database is provided in the file **social-network.sql**, available on KSOL. Answer the following questions:

Advanced SQLizing

Write an SQL query for each of the following questions (run these queries on the original sample of data):

- (3p) Find the name and grade of the student(s) with the greatest number of friends. (This is a “witness” type query.)
- (3p) What is the average number of friends per student? (Your result should be just one number.)
- (3p) For each student A who likes a student B where the two are not friends, find if they have a friend C in common (who can introduce them!). For all such trios, return the name and grade of A, B, and C.

- (3p) Find the number of students who are either friends with Cassandra or are friends of friends of Cassandra. Do not count Cassandra, even though technically she is a friend of a friend.

Triggers

- (2p) Write a trigger that makes new students named 'Friendly' automatically like everyone else in their grade. That is, after the trigger runs, we should have ('Friendly', A) in the Likes table for every other Highschooler A in the same grade as 'Friendly'.

- (2p) Write a trigger that automatically deletes students when they graduate, i.e., when their grade is updated to exceed 12.

Data modification statements

- (2p) For all cases where A is friends with B, and B is friends with C, add a new friendship for the pair A and C. Do not add duplicate friendships, friendships that already exist, or friendships with oneself.

- (2p) If two students A and B are friends, and A likes B but not vice-versa, remove the Likes tuple.

What to turn it: A *.txt* file with all SQL queries (together with their results), triggers and data modification statements. Submit this file using the course Dropbox.