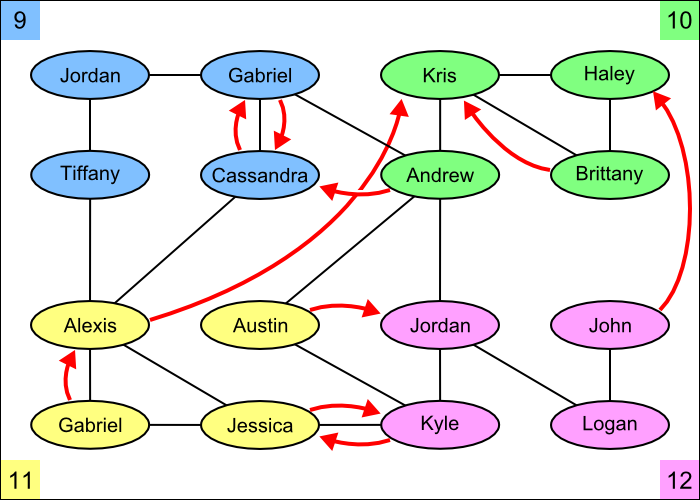
Students at your hometown high school have decided to organize their social network using databases. So far, they have collected information about sixteen students in four grades, 9-12. Here's the schema:   
  
Highschooler ( ID, name, grade )   
English: There is a high school student with unique *ID* and a given *first name* in a certain *grade*.   
  
Friend ( ID1, ID2 )   
English: 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 )   
English: 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.   
  
  
For your convenience, here is a graph showing the various connections between the students in our database. 9th graders are blue, 10th graders are green, 11th graders are yellow, and 12th graders are purple. Undirected black edges indicate friendships, and directed red edges indicate that one student likes another student.

  
**Instructions:** Each problem asks you to write a query in SQL.

Q1

Find the names of all students who are friends with someone named Gabriel. 

Q2

For every student who likes someone 2 or more grades younger than themselves, return that student's name and grade, and the name and grade of the student they like.

Q3

For every pair of students who both like each other, return the name and grade of both students. Include each pair only once, with the two names in alphabetical order.

Q4

Find names and grades of students who only have friends in the same grade. Return the result sorted by grade, then by name within each grade.

Q5

Find the name and grade of all students who are liked by more than one other student. 

Q6

Find all students who do not appear in the Likes table (as a student who likes or is liked) and return their names and grades. Sort by grade, then by name within each grade.

Q7

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.

Q8

Find the difference between the number of students in the school and the number of different first names.

Q9

What is the average number of friends per student? (Your result should be just one number.)

Q10

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.

Q11

Find the name and grade of the student(s) with the greatest number of friends.

Q12

For every situation where student A likes student B, but we have no information about whom B likes (that is, B does not appear as an ID1 in the Likes table), return A and B's names and grades.

Q13

For every situation where student A likes student B, but student B likes a different student C, return the names and grades of A, B, and C.

Q14

Find those students for whom all of their friends are in different grades from themselves. Return the students' names and grades.