TP-3: *Social_network*

1. Find the names of all students who are friends with someone named Gabriel.
2. 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.
3. 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.
4. 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.

join, natural join , left join

1. (*Movie_rating*) Find the movies that are reviewed by no one (using left join).
2. (*Social_network*) Find names of Highschoolers that do not like anybody....
3. (*Social_network*) Find names of Highschoolers that is not liked by anybody....

max, min , avg, count

1. (*Movie_rating*) find the title(s) of the oldest movie(s) available.
2. Find the movie titles that have the best rating.

Homework

1. (*Social_network*) 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.
2. (*Social_network*) Find those students for whom all of their friends are in different grades from themselves. Return the students' names and grades.
3. (*Social_network*) 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.
4. (*Social_network*) 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.