This is an individual assignment. Please review the academic integrity policy and contact the instructor or TA if you have any questions.

The university registrar database has the following schema:

student: sid (string)
course: cid (string)
prerequisite: cid (string), pre_cid (string)
record: sid (string), cid (string), qtr (string), year (integer), grade (integer)

Relation student provides the ids of all students enrolled in the university, and course provides all offered courses. Relation prerequisite provides the prerequisites pre_cid for each course cid. Relation record indicates which courses were taken by each student in a given year and quarter (F, W, S), and the grade obtained (for simplicity, grades are provided as integers 4, 3, 2, 1, 0, corresponding to A, B, C, D, F). Assume that a class can only be taken once. Note that students may have gaps in enrollment, i.e. they need not take courses every quarter.

In addition, the following hold:

- sid is the primary key of student
- cid is the primary key of course
- \{cid, pre_cid\} is the primary key of prerequisite; cid and pre_cid are foreign keys referencing course.
- \{sid, cid, qtr, year\} is the primary key of record (so grades are unique for each course taken by a student in a given year and quarter), sid is a foreign key referencing student and cid is a foreign key referencing course.

A sample instance will be posted separately, together with outputs to the queries in the assignment.
Write the following queries in SQL (of course, the queries must work on all data, not just the sample one):

1. Compute the GPA for the academic year\(^1\) 2015/16 for all students who have taken at least one class during this academic year. The answer should have attributes \{sid, gpa\}.

2. List all pairs of students who have taken at least one course together in Fall 2016. The answer should have attributes \{sid1, sid2\}. Avoid listing pairs of the form \(<a, a>\), or listing both \(<a, b>\) and \(<b, a>\). Specifically, list only pairs \(<a, b>\) where \(a\) is strictly less than \(b\) in alphabetical order.

3. List, for each quarter, the number of courses with fewer than 5 students (but at least one student) enrolled that quarter. The answer should have attributes \{qtr, year, num\}.

4. Find the number of courses each student has taken in Fall 2016. If a student has taken no class that quarter, the number of classes should be zero. The answer should have attributes \{sid, num\}.

5. List the students who have taken all prerequisites for CSE132X with a grade of 2 or higher (this refers to direct prerequisites only, not to prerequisites of prerequisites). The answer should have one attribute \(sid\). Note that, if CSE132X has no prerequisites, then all students should be in the answer. Provide two SQL queries, using nested sub-queries in different ways:

   (a) with NOT IN tests only;
   (b) with NOT EXISTS tests only;

6. Find the students whose quarterly GPA has gone up every quarter they have been enrolled (their GPA in each quarter is strictly higher than their GPA in previous quarters). Recall that students may have gaps in enrollment. **Hint:** It may be helpful to observe that the order of occurrence of quarters in a given calendar year is in reverse alphabetical order ("W" > "S" > "F"). For example, the quarters occurring in 2016 are, in chronological order, W, S and F.

\(^1\)The academic year 2015/16 comprises the following quarters: Fall 2015, Winter 2016, Spring 2016.
7. Update relation `record` by swapping enrollments in CSE132A and CSE132B in Fall 2016, without explicitly naming the students involved. That is, all students enrolled in CSE132A should be dropped from that class and enrolled in CSE132B, and conversely. You may use several update commands if needed. Do not create new relations.

What to submit:

- Please clone this github repository (https://github.com/nhatminhbeo/cse132a-sql-assignment), fill in and submit to it the files listed below.

- Each file corresponds to a specific problem or subproblem and is named accordingly (i.e. Q1.sql, ..., Q5a.sql, ..., Q7.sql).

- Each file should be formatted such that calling the command ".read Q*.sql" on a populated database in sqlite3 would give you the desired result (you can test this during submission).

- For each file, please only fill in the SQL statements needed for that problem. You don’t need to include the statements to create the database schema or the sample database.

Submission instructions:

- Zip the required files and upload to gradescope. Make sure to zip all the files together rather than just the parent directory.

- Once you submit your files on gradescope, the autograder will run your submission against some test cases. A working submission (i.e. correctly formatted, compressed) should pass the public test cases.

Grading:

- The assignment is graded solely on the correctness of the queries against some public and hidden test cases.

- The public test cases are derived from the posted sample database, and are visible for your testing purposes.

- The hidden test cases are derived from a different instance of the database, and won’t be visible to you. Therefore, it’s important to make sure the SQL statements work for all databases.