NYU Polytechnic School of Engineering Computer
Science and Engineering Department CS 3083
Prof. Phyllis Frankl

HOMEWORK #4

Hand in your solutions via NYU Classes in a plain text file named `hw4.txt` or `hw4.sql` so we can easily copy/paste your queries in order to test them.
You only need to hand in the queries, but I strongly recommend that you execute them and check that the results are what you expect them to be. You might find it helpful to test the subqueries separately before testing the whole queries in which they are nested. Modify the data in the university database in order to test your queries thoroughly.
You may work with classmates. If you work with others, you may hand in one file for the whole group. Make sure all of your names are in your file (near the top.)
If you're using a DBMS other than MySQL, note which one near the top of the file. NOTE: MySQL doesn't have INTERSECT or EXCEPT. Use IN or NOT IN (subquery), instead.
For example:

```
SELECT a1, a2 FROM T1 WHERE predicate1
AND (a1, a2) NOT IN
(SELECT a1, a2 FROM T2 WHERE predicate2))
```

Is equivalent to

```
(SELECT a1, a2  FROM T1 WHERE predicate1)
EXCEPT
(SELECT a1, a2 FROM T2 WHERE predicate2)
```

Problem 1 Write SQL queries for each of the following:

1. Find IDs of students who took lectures in classroom 101 and in classroom 120
2. Find IDs and names of students who took course under instructor "Kim" and "Katz"
3. Find IDs and names of students who took CS-315 and got an A in all prerequisite for CS-315 and, but did not get an A in CS-315.
4. Find IDs and names of students who got an A in CS-101 and but not under instructor "Katz".
5. Find IDs of students who have repeated a course (i.e. taken the same course more than once.)
6. Create a table gradepoint (letter,points) to associate letter grades with points and fill it with the appropriate values ('A', 4.0), ('A-' ,3.7), etc. Use it to find average grade of entire class for all courses taught in summer 2010.
7. When the database has the gradepoint table an additional foreign key constraint would be useful on one of the other tables. What is that constraint and which table does it belong on? (Optionally, you may add the constraint to your database.)
8. Create a view called deanslist with the ids and names of students whose gradepoint averages are over 3.0.

   Note: Unfortunately, If your GPA calculation involves a subquery in the where clause, MySQL won't let you use it to define a view. If necessary, re-write the view definition so it doesn't involve a subquery in the where clause.
9. Use the deanslist view to find the number of deanslist students for each advisor.

10. Find the ID and name of the 'Comp Sci' student with the highest grade point average (of any Comp Sci student).
11. Find IDs of students who got a higher grade in CS-101 than they got in CS-347
12. Find the IDs and names of students with the highest number of total credit.
13. Find the IDs and names of students with highest number of 'A' grades in 'Comp Sci' courses (i.e. no other students have gotten more A's in Comp Sci courses than these students)
    Note: There may be more than one such students in case of a tie for most A's.
14. Find IDs of students who never got grades higher than B+ (in other words, they never got A- or A in any course for which they have a grade).
15. Find the course id of each course that has been offered two years in a row.
    Hint: Find the course ids of courses taught in year s.year and in year s.year +1, where s is a correlation variable representing a tuple of the section table.
16. Find instructors who have taught all 'Comp Sci' courses.
    Hint: Model this after the problem we did in class to find students who've taken all 'Biology' courses.
17. Find id and name of each student who has name LIKE '%an%' and taken a course with course id LIKE 'CS%' every semester that he or she has been enrolled (i.e. every semester that he/she has taken any course.)
    Hint:
    - Write a subquery to find all the (semester, year) pairs student s has taken any courses. (Here, s represents a correlation variable.)
    - Write a subquery to find all the (semester, year) pairs when student s has taken a course with id LIKE 'CS%'.
    - Nest these subqueries inside a main query involving the correlation variable, a set difference operation, and a test for an empty relation. (Remember that MySQL doesn't have except operator, so the set difference operation requires an additional level of nesting.)
18. List all courses and the IDs of instructors who've taught them. Include courses that haven't been taught, with NULL in the ID column for such courses.

Problem 2 Consider the table definitions in the `university DDL.sql` file used in HW 1.

1. What tables could change and how would they change if a Course is deleted from the `COURSE` table? Explain.
2. Now suppose each `ON DELETE SET NULL` were changed to `ON DELETE CASCADE`. What tables could change and how would they change if a Course is deleted from the `COURSE` table? Explain.
3. Add a constraint to assure that a student cannot take a section of a course that no one teaches. (If you want to try this on MySQL you might need to create an additional index.)
4. Consider the following foreign key constraints on the `section` table:

```
a) foreign key (building) references classroom (building)
b) foreign key (room_number) references classroom (room_number)
c) foreign key (building, room_number) references classroom (building,
   room_number)
```

   Give a small example of data that satisfy constraint a and constraint b, but do not satisfy constraint c.