NYU Tandon School of Engineering
CS-GY 6083, Principles of Database Systems, Fall 2024
Prof Phyllis Frankl
HOMEWORK #3

You may work alone or with one or two others. As usual, if you work with others, you should try to do each problem on your own and make sure you understand the solutions that your group arrives at.

As in HW2, you should hand this in via two GradeScope assignments:

- HW 3A: a plain text file with the SQL queries from which we can copy/paste/execute your queries. Use the SQLcomment delimiter -- to comment out everything other than the queries. Label which query is which with -- Problem X.y
- HW 3B: a pdf with all problems (marking which is which with GradeScope)

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 one or two classmates. If you work with others, use GradeScope's group handin feature and also make sure all of your names are in your file (near the top.)

Create and use VIEWs, if it will make your answers clearer.

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)
```

Use the University DB, unless otherwise noted. Write SQL queries (unless otherwise noted) for each of the following:

1. Find IDs and names of students who got a B in CS-101 and an A in CS-319

2. A. Create a table Gradepoint(grade,points) to associate letter grades with points and fill it with the appropriate values (’A’, 4.0), (’A-’ ,3.7), etc.

   B. Add a foreign key constraint to *another table*, referencing thie Gradepoint table. Note which table, write the constraint, including “on delete” and “on update” clauses and briefly justify your choices for those clauses.
   C. Define a VIEW GradePointAvg(ID, GPA) that lists each student’s ID and gradepoint average.

   Note: If all courses had the same number of credits, you could compute gradepoint averages with a query involving a natural join of takes and gradepoint, along with the AVG aggregation operator, grouping by ids.However, that solution doesn’t take account of different courses having different numbers of credits. Assume that all graded courses are included, even if a student repeats the same course.

3. Find the name, ID, and GPA of the Comp. Sci. student who has the highest GPA among all Comp. Sci. students. If several students are tied for the highest, your query should return them all. *Do not sort students by GPA.*

4. Find the ID and name of each Comp. Sci. student who has not taken any courses offered by the Math department.

5. Find ID, name, course_id of each student and each course they took in Fall 2009. Students who did not take any courses that semester should be listed with NULL as the course_id.

6. Find the ID, name, and Fall 2009 GPA for each student. Students who didn’t take any courses in Fall 2009 should be listed with GPA either NULL or zero.

7. Find the ID and name of each student who has taken every course taught by the instructor whose ID is 10101.
   a. Write an SQL query using checks for empty set differences
   b. Write an SQL query using comparison of sizes of sets
   c. Write a TRC query. Hint: it should involve universal quantification.

8. **Using Retailer DB:** Let’s define the “profitability” of a product to be the sum of the difference between priceEach and basePrice for all items ordered (taking quantity into account) divided by the total number of items of that product ordered.
   Write an SQL query to find the product name and product code of the product that has the highest profitability. (If there are ties, all such products should be listed. Do not use sorting.) Your solution should include WITH, VIEW, or TEMPORARY TABLE creations with comments to make them readable.

9. Consider the posted solution to the tennis tournament problem from HW 1. Derive schemas and CREATE TABLE definitions for relevant tables and write constraints expressing the following:
    a. A player cannot play a match against themself
    b. No more than one match can be played on a given court during a given time slot
    c. A player cannot play in more than one match during the same time slot

    Try to determine whether your DBMS allows you to write those constraints in SQL and, if so, whether they are enforced by the DBMS. Briefly explain how you investigated this and your findings.