# CS348: Introduction to Database Systems

(Fall 2019)

**Assignment 2** (due Monday October 21 by 5pm via submit)

**Overview:** For this assignment, you must use your Unix accounts and DB2 to compose and evaluate a number of SQL queries over the same enrollment database as was used in your first assignment. The visualization of the database schema for this database is reproduced below. All submissions must also use the SQL DDL code for this database given in the contents of a file downloadable from the course web page.

As with the first assignment, you are given a requirement for each query in English, and your task is to write source code in the SQL query language that implements the requirement over the same database schama as for Assignment 1 (a SQL declarations are provided on the assignment web page in `a2schema.db2` file. To determine the *department code*s for `COURSE`s use `substring` built-in functoin as follows:

```
@ubuntu1804-004% db2 "select cnum,cname,substring(cnum,1,2) as dept from course"

CNUM   CNAME                                              DEPT
------ -------------------------------------------------- ------
CS348  Intro to DB                                        CS

  1 record(s) selected.
```

**Assignment submission:** By the assignment due date, you must have used the `submit` command to submit files containing SQL queries that implement each of the queries below. For the online submission, put the queries in separate files named `q1.sql`,...,`q10.sql` and then to submit your assignment, use "`submit cs348 a2 .`". Answers to your solutions must adhere to the specification provided at the end of each information request, for exanple, the answers produced by qurey 1 should be binary tuples with the first and second columns named "snum" and "sname", respectively).

Note that some of the requirements stipulate conditions on what features of SQL may be used in your source code, e.g., that `group by` clauses and aggregate functions may not be used. Part of the grading for your answers in these cases relate to these conditions. The rest of grading will be based on two additional criteria: (1) correctness, the query implements the requirement, and (2) readability. Consequently, the efficiency of your source code, e.g., as determined by DB2, will not be a factor in any grading.

## Queries that may *not* use aggregation in SQL

1. The student number and name of each 3rd or 4th year student who has obtained a grade of at least 85 in CS240 and in CS245. (snum, sname)
2. The number and name of professors who are not in the pure math (PM) department, and who are teaching CS245 for the first time. (pnum, pname)
3. The number, name and year of each student who has obtained a grade in CS240 that is within 3 marks of the highest ever grade recorded for that course. (snum, sname, year)
4. The number and name of students who have completed two years, who have a final grade of at least 85 in every computer science course that they have taken, and who have always been taught by a professor in the computer science (CS) department. (snum, sname)
5. The number and name of each student who is not in his or her first year, who has a final grade of at most 70 in every course that she/he has completed and who was not taught by a professor in the philosophy (PH) department. (snum, sname)

## Queries that *may* use aggregation in SQL

6. The number and name of each professor who has taught a CS 245 class in which exactly three students received the highest grade and in which every other student received a grade within 20 marks of the highest grade. (pnum, pname)
7. A count of the number of different students in each term for any course that has never been taught by either a computer science (CS) or applied math (AM) professor. Each result should identify the course, the term and said count, and should be sorted in descending order by the said count. (cnum, term, numstudents)
8. For each term, the total number of enrollments in classes for any course that has never been taught by either a computer science (CS) or combinatorics and optimization (CO) professor. The result should be in chronological order of the term. (cnum, term, total)
9. The student number and name of each fourth year student, together with three measures: the average grade of the student's completed CS courses, the average grade of all completed courses by the student, and the percentage of all completed courses that were CS courses by the student. The result should be sorted in descending order of the first measure, and then by descending order of the third measure. (snum, sname, avgcs, avgall, percentage)
10. The percentage of professors who have never taught in the past, during the same term, at least two classes for two different courses, and for whom this is the case with the current term. Note that a percentage should be a number between 0 and 100. You can assume there is at least one professor. (percent)