Database Systems
Project 1: Writing SQL Queries in Oracle
(Due: October 26, 2018. Hard copy due at the start of the class; soft copy submitted to MyCourses by 5pm on Oct 26)

The following normalized tables from the Student Registration System (some tables have been simplified) will be used in this project:

Students(B#, first_name, last_name, status, gpa, email, bdate, deptname)
TAs(B#, ta_level, office)
Courses(dept_code, course#, title)
Course_credit(course#, credits)
Classes(classid, dept_code, course#, sect#, year, semester, limit, class_size, room, TA_B#)
Enrollments(B#, classid, lgrade)

As a general clarification, we assume that no student takes the same course (including different sections of the same course) more than once. If you have questions about these tables, please contact the instructor for clarification.

1. Preparation

Save the sql script file proj1_tables_script18.txt as proj1_tables_script18.sql in your bingsuns account.

To run the above script from your Oracle account, use

SQL> start proj1_tables_script18

Then check whether all tables are created correctly in your Oracle account.

2. Project Requirements

There are 20 statements (see Section 3 below) in this project and you are asked to write one or more SQL queries for each statement. Your query should take into consideration that the tuples currently in the database may change. In other words, your query must be correct for any valid state of every table. It is very important that you understand each query statement correctly. If you have any doubt about the correct interpretation of a statement, please ask the instructor for clarification by posting your questions in the Project 1 thread in the discussion forum on blackboard. The query (or queries) for each statement is worth 5 points. Partial credit may not be given in general.

It is suggested that you first test each query individually and save each query in a different file (with extension .sql). After all queries have been tested to your satisfaction, you can run all the queries in a sequence and save the entire session in a spool file. Suppose you have saved your queries in files query1.sql, ..., query20.sql. Follow the steps below to generate the spool file after you have logged on Oracle:

```
SQL> set echo on
SQL> spool project1.txt
SQL> start query1
.......
SQL> start query20
SQL> spool off
```

To prepare your submission of Project 1, follow the steps below:

(1) **Edit file project1.txt** by **adding your name** at the top of the file and **adding the following honesty statement**: "I have done this assignment completely on my own. I have not copied it, nor have I given my solution to anyone else. I understand that if I am involved in plagiarism or cheating I will have to sign an official form that I have cheated and that this form will be stored in my official university record. I also understand that I will receive a grade of 0 for the involved assignment and my grade will be reduced by one level (e.g., from A to A- or from B+ to B) for my first offense, and that I will receive a grade of "F" for the course for any additional offense of any kind." No other changes to the file are permitted.

(2) Print the edited file project1.txt, **sign your name** next to the above statement. **Your printout must list each SQL query followed by the result of executing the query.**

(3) Hand in the hardcopy printout at the start of the class on October 26, 2018.

(4) Submit a softcopy of project1.txt to the project 1 submission folder in MyCourses by 5:00pm on October 26, 2018.

3. Query Statements

The following are the 20 query statements:

1. Find the B#, the first name and last name of every CS student whose GPA is higher than 3.5. In the output, first name and last name of each student should be concatenated with a space in between under a new column header name.

2. For each TA from the CS department, find his/her B#, first name, last name and birth date. The header of the last column needs to be "birth date" (without the quotes and there is a space between birth and date). To not have header "birth date" truncated in the output, run SQL> column "birth date" format a10 before you run your query, here 10 is the new column/header width in character for "birth date".)

3. For each class that has a PhD student TA, find its classid, dept_code and course# as well as the TA's name and email. Here a TA's name is its first name followed by its last name with a space in between. Furthermore, in the output, dept_code and course# of each class are concatenated under a new column header course_id. Make sure that no column header in the output is truncated without manually editing the spool file.

4. Find the B#, first name, last name and GPA of each student who has taken at least one CS course and at least one math course. Write an SQL query that uses neither intersect nor distinct but the results have no duplicate.

5. Find the B#, first name and last name of each student who has taken at least one course but has never received an A for any course he/she has taken. Write a query with an uncorrelated subquery and write another query with a correlated subquery.

6. Find the B#, first name and last name of each student who has taken at least one course and received an A for every class he/she has taken. Count only classes for which he/she received a non-null grade. For this query, do not use the GPA information. Also do not use any aggregate (set) function.

7. Find the classid, dept_code and course# of each undergraduate class (i.e., course# < 500) that was offered in Spring 2017. For each such class, also list the number of seats available (computed by limit – class_size) under the header "seats_available".

8. Find the B# and the total number of credits of each student.

9. Find the dept_code and course# of the course that has been attended by most students. If a course has been offered multiple times, the numbers of students for all these offerings should be added for the course. Note that it is possible that multiple courses may have the same highest attendance; in this case, all such courses should be output.

10. Find the B#, first name and last name of every student who has taken at least 2 classes. Also output the number of classes each such student has taken.

11. Find the classid, dept_code and course# of each class that has been taken by all juniors.

12. Find the B#, first name and last name of every student who has taken all CS classes offered in Spring 2017.

13. Find the B#, first name and last name of every student who has taken at least one course offered by a department different from his/her own department. Write a nested query with a correlated subquery.

14. Find the B#, first name and last name of every student who has taken at least one course but has not taken any course offered by a department different from his/her own department.

15. List the dept_code, course# and title of each course student B003 has taken. For each such course, also list the grade the student received. If no grade has been assigned for a course, output "grade missing" as the grade information for the course.

16. Find the dept_code, course# and title of each course whose title contains "systems" and that has been taken by all students whose GPA is 4.0. Note that even though a qualified course is required to be taken by all students whose GPA is 4.0, the course may also be taken by some students whose GPA is lower than 4.0.

17. Display the enrollment table with tuples with higher grades displayed first. Do not display tuples with null or incomplete (I) grades. Also convert letter grade (lgrade) to number grade (ngrade) based on A → 4, A- → 3.7, B+ → 3.3, B → 3, B- → 2.7, C+ → 2.3, C → 2, C- → 1.7 and D → 1. Display the converted number grade for each row. Make sure no column header is truncated in the output.

18. Find the B#, first name and last name of every student who has taken at least one course taken by student B005. Note that here we are talking about taking the same course, not just the same class. (Clearly, student B005 satisfies the condition and his first name should be included in the output.) Write two queries for this question, one uses only uncorrelated subqueries and the other uses only correlated subqueries.

19. Find the average total number of credits earned by students (the average is over all students and only one value should be computed) who have taken at least one course. Don't count the credits when the grade is null for a class.

20. Find the average total number of credits earned by students in each department (the average is over all students in each department and one value should be computed for each department). For this query, we assume that at least one student from each department has taken a course. Don't count the credits when the grade is null for a class. The output should have 2 columns: deptname, average_total_credits.