Database Systems
Project 1: Writing SQL Queries in Oracle

(Due by 10pm on March 28, 2022 to Brightspace Project 1 Submission Folder.)

Please remember to include the following statement at the beginning of your submitted assignment and SIGN it. Your assignment won't be graded without the signed 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."

The following normalized tables from the Student Registration System will be used in this project (Note 1: Attributes Faculty_B# and TA_B# are not kept for relation Classes because relations TAs and Faculty are not used for this project. Attributes start_time and end_time are also not kept for relation Classes as they will not be used in this project. Note 2: To simplify, relation G_Students is not used in this project. We will let G_B# to be a foreign key referencing Students.B# directly.)

Students(B#, first_name, last_name, level, gpa, email, bdate)
Courses(dept_code, course#, title)
Course_credit(course#, credits)
Classes(classid, dept_code, course#, sect#, year, semester, limit, size, room)
G_Enrollments(G_B#, classid, score)
Score_Grade(score, 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_script22.txt as proj1_tables_script22.sql in your harveyv account.

To run the above script from your Oracle account, use

```
SQL> start proj1_tables_script22
```

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 in Brightspace. 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   /* this allows both queries and their results to be captured by the spool file */
SQL> spool project1.txt
SQL> start query1
SQL> start query2
.......
SQL> start query20
SQL> spool off
```

Please check if the spool file project1.txt in your harveyv account contains all of your queries and their results. 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) Submit project1.txt to the project 1 submission folder in Brightspace by 10:00pm on March 28, 2022.

3. Query Statements

The following are the 20 query statements:

1. Find the B#, the first name and last name of every master's 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 the graduate student with B# = B00000003, find the dept_code, course# and title of every class this student took in Spring 2021. For all queries in this project, you can assume that every student whose B# is in G_Enrollments is a graduate student.

3. Find the B#, first name and birth date of every graduate student who has taken at least one CS class (i.e., dept_code = 'CS' in Classes). Your SQL query should not have duplicate results. 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".)

4. Find the B#, first name, last name and GPA of each graduate 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 duplicates.

5. Find the B#, first name and last name of each graduate 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 graduate student who has taken at least one course and received an A for every class he/she has taken. Count only classes for which the student 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 graduate class (i.e., course# >= 500) that was offered in Spring 2021. 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 graduate student who has taken at least one class (i.e., who appears in G_Enrollments).

9. Find the dept_code and course# of the course that has been attended by the largest number of students. If a course has been offered multiple times, the numbers of students for all these offerings (i.e., classes) should be added up for the course. Note that it is possible for multiple courses to 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 graduate 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 every class taken by all graduate students whose GPA is not null and whose last name starts with a capital B.

12. Find the B#, first name and last name of every graduate student who has taken all CS classes with exactly 13 students.

13. Find the B#, first name and last name of every graduate student who has taken all courses (note: not classes) whose title contains "database". Note: if a student has taken any class of a course, the student is considered to have taken the course.

14. Find the B#s, first names and GPA of the top three master students with the highest GPAs. Furthermore, only students who have taken at least one course with a non-null score are considered. (Note: only three students will be output no matter how many students may be tied in the third place.)

15. Find the B#, first name and last name of every graduate student who has taken at least one CS class but has not taken any Math class.

16. List the dept_code, course# and title of each class student B00000003 has taken. For each such class, also list the letter grade the student received. If no score is assigned for a class for a student (i.e., when the score is null), output "missing grade" as the grade information for the class of the student. (It is possible to use select to display any character string, e.g., select 'missing grade' from … will display "missing grade". To display "missing grade" properly, you can use `SQL> column lgrade format a13` before running your query.)

17. Display the g_enrollments table with tuples with higher scores displayed first. Do not display tuples with null scores. Also find the letter grade for each score and display the letter 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 graduate student who has taken at least one course taken by student B00000004. Note that here we are talking about taking the same course, not necessarily the same class. (Clearly, student B00000004 satisfies the condition and this student's B#, first name and last name should be included in the output.)

19. Find the average total credits earned by graduate students (the average is over all graduate students and only one value should be computed as the output) who have taken at least one course. Don’t include the credits of a class when the score is null for a class. (Example: If the total credits of student s1 is x and the total credits of student s2 is y, then the average total credits of the two students is (x+y)/2.)

20. Find dept_code, course# and the average score for each course (note: not class). The average score for a course is computed as follows: for all students who have taken any class of the course and received a non-null score, compute the average of the scores earned by these students for all classes of the course. (Example: Consider a course with two classes and 5 students have taken the two classes and received scores. Then the average score for this course is the average score over the five scores received by the five students for the two classes.)