COMP 3311 Assignment 3
Spring 2016

Due date: 6th Apr 2016 (23:55)

Assignment Rules:

1) This is an *individual* assignment; you are required to work on your own.
2) The assignment solution you submit must be solely your own work; Copying or letting others to copy are both considered cheating.
3) Please run the solution .sql file on Oracle before submitting, and make sure your script file name is correct, if it does not execute correctly, you **may NOT get any marks**.

Assignment description:

Part 1. Creating tables using the SQL Data Definition Language:

Create the following 10 tables with the given constraints. Use exactly the *same table names* (i.e. “course”, “prof”…etc.) and *attribute names* (i.e. “course_ID”, “course_name”… etc.) as listed below, otherwise marks will be deducted. The table order below is *not necessarily the correct order* for creating the tables; it is merely describing the column types and the different constraints of the tables.

- **course table**
    1. course_ID: varchar2(8), primary key.
    2. course_name: varchar2(80).
    3. credits: number(3).

- **prof table**
    1. staff_ID: number(8), primary key.
    2. last_name: varchar2(80).
    3. first_name: varchar2(80).

- **prof_phone table**
    1. staff_ID: number(8).
    2. phone_number: number(8).

    staff_ID foreign key referencing the prof() table, primary key (staff_ID, phone_number).

- **prerequisite table**
    1. main_course_ID: varchar2(8).
    2. prereq_course_ID: varchar2(8).

    main_course_ID foreign key referencing the course_ID column of the course() table, prereq_course_ID foreign key referencing the course_ID column of the course() table, primary key (main_course_ID, prereq_course_ID).

- **prof_teach table**
    1. staff_ID: number(8).
    2. course_ID: varchar2(8).
    3. offering_no: number(8).

    staff_ID foreign key referencing the prof() table, (course_ID, offering_no) foreign key referencing the offering() table, primary key (staff_ID, course_ID,offering_no).

- **pref_TA table**
    1. staff_ID: number(8).
    2. student_ID: number(8).

    staff_ID foreign key referencing the prof() table, student_ID foreign key referencing the TA() table, primary key (staff_ID, student_ID).

- **supervise table**
    1. staff_ID: number(8).
    2. student_ID: number(8).

    staff_ID foreign key referencing the prof() table, student_ID foreign key referencing the TA() table, primary key (staff_ID, student_ID).

- **pref_offering table**
    1. student_ID: number(8).
    2. course_ID: varchar2(8).

3. offering_no: number(8).

   student_ID foreign key referencing the TA() table, (course_ID,offering_no) foreign key referencing the offering() table, primary key (student_ID, course_ID, offering_no).

- **TA table**
  1. student_ID: number(8) , primary key.
  2. last_name: varchar2(80).
  3. first_name: varchar2(80).
  4. phone: number(8).
  5. course_ID: varchar2(8), NOT NULL.
  6. offering_no: number(8), NOT NULL.

  (course_ID, offering_no) foreign key referencing the offering table.

- **offering table**
  1. course_ID: varchar2(8).
  2. offering_no: number(8).
  3. YearSemester: varchar2(10).
  4. classroom: number(5).
  5. no_of_stds: number(5).
  6. staff_ID: number(8), NOT NULL.

  course_ID foreign key referencing the course() table, staff_ID foreign key referencing the prof() table, primary key (course_ID, offering_no).

<u>Part 2. Write the following SQL queries:</u>

1. Find the course_ID for courses with the highest number of credit.
2. Find the staff_ID, last_name, first_name of all the professors who have taught 'Comp3311' but not 'Comp4311'.
3. Find the student_ID, last_name, first_name of the TAs who were preferred by the most number of professors.
4. Find the staff_ID, last_name, first_name of all the professors who have NOT taught all the prerequisites of 'Comp3311'.
5. Find the staff_ID, last_name, first_name of each professor who has taught *all* the offerings of 'Comp3311'

Submission:

- Write SQL statements and put them into **two plain text files**:
  - create.txt for creating the tables given in part 1,
  - query.txt for the queries in part 2.

- The two files will be tested directly on SQLPlus by issuing

  @create.txt

  and

  @query.txt

- Grading process: This assignment is graded by two programs.

  For create.txt it will execute your script and then use *describe table_name;* command to display the schema of each table and then do a character by character comparison with the expected result.

  For query.txt it recreates all the tables using the correct script, insert some records into the tables, execute the script you submitted, and then compare the result with the expected result character by character.

  (The test cases will be posted on course website after grading.)

- No partial grade will be given based on the query if the execution result is wrong. (We want to make sure you will be able to use SQL in real life situations, otherwise we'll just ask you to write the queries in paper.)

- Put your name, ID as comment on the first line of each of your text files.

  Comments are enclosed by "/*" and "*/"

- **Marks will be deducted** if you create the files in other formats (i.e. doc, rtf, etc.), or if you do not put the name and ID information correctly.

- Zip the two text files into a zip file named “ass3.zip” and submit the zip file to the CASS.

- Submit the zip file to the CASS submission system:

  http://cssystem.cse.ust.hk/home.php?docbase=UGuides/cass&req_url=UGuides/cass/index.html

- No Late submission will be accepted!