Due date: December 3, 2020 (Thursday)

Project Description:
You will be using the University Database introduced in Project 1 to solve the following problems:

Part 1: Views and Indexes

1. A view is a stored query that has been given a name and virtually saved in the database. Unlike ordinary tables in a relational database, a view is the dynamic result of one or more relational operations and it does not actually exist in the database but is produced upon request by a particular user, at the time of request.

   a. List the advantages of views over ordinary tables (base tables).

   b. Create the following three views (using the suggested names) for the university database.

      i. FacultyCourses: Join the Faculties, Courses tables using the faculty_id and display the values for the following variables: faculty_id, name (Faculties), level (Faculties), course_id, description (Courses), level (Courses).

      ii. Gradebook: Join the Courses, Enroll, and Students tables and display the values for the following variables: student_id, name (Students), level (Students), course_id, description (Courses), level (Courses), grade.

      iii. GradeLevel: For each student’s level (i.e. ugrad/grad), group and count the grades of the student’s enrolled courses (You may join the tables, Enroll and Student then group by two columns: level (Students) and grade (Enroll) then count; you can also use Gradebook). The View must display the following variables: level (Students), grade, Count.

2. Databases usually contain large amounts of data and a DBMS is required to read data from disk whenever a query is executed. A database index is a data structure that
improves the speed of data retrieval operations by enabling the DBMS to read only a subset of all data from disk when answering a query.

a. Create an index for the *Students* table on the *email* attribute in the university database and provide a point query for which the index is useful.

b. Create an index for the *Enroll* table on the *grade* attribute in the university database and provide a range query for which the index is useful.

**Part 2: Stored Procedure**

1. Write a stored procedure for the university database to output information (mentioned below) for a *Student*. The procedure will be named *GetStudentInfo*, will accept a student ID and generate the following information:
   - The Student’s name, level, email, date_of_birth, and address.
   - List all the courses for the student’s level in which the student has yet to enroll.
   - Using the *FacultyCourses* view, list all the courses for which the student enrolled along with the name of the instructor who taught the course.
   - Using the *Gradebook* view, display the information of the course in which the student got his/her highest grade.
   - Using the *GradeLevel* view, list the letter grades and counts correspondent to the student’s level.
   - Group and count the grades of all the courses taken by the student, ordered by grades.

2. Create a second stored procedure, *GetAllStudentInfo*, which will call *GetStudentInfo* and generate the information for all the Students in the university database.

**Submission**

For all the students, you need to save your implemented solutions as a SQL files (.sql) and upload it to Canvas.