# CMPE 351
# Database Systems
# Project

21st November, 2017-23rd November, 2017

**Due.** Monday December 11th , 23:59

## 1 Project Specifications

You are given 4 main tasks for this project. In addition to solution of each task, you are required to create a README document. This document will be used to execute your code-queries on a different system(a different computer with any OS). Therefore, pay attention to explain in detail; what each file is used for, how Python files will be run and what results will be displayed/produced.

### 1.1 Part 1: Design Database(10p)

In this part, you are given the conceptual details about *CMPE351* database. Your task is to create the corresponding ER diagram using Crow's Foot notation. You can use this online tool or simply pen and paper to create the diagram. Remember to visualize entities(with data types of attributes), relationships, cardinality and participation information all together.

> Each *STUDENT* has unique student ID. We keep track of registration type, registration date and department of the students. We keep records of student submissions in this database. Each *SUBMISSION* has a submission grade, deadline and submission date. *COURSE GRADE* of the students are determined by midterm,final and lab grades.
> A student can make zero or more submissions throughout the semester. Given studentID and assignmentID, we should be able to access the submissions of any student. Students are graded for each course component midterm,final,lab- at the end of the semester without any exemption.

## 1.2 Part 2:Create Database (20p)

In this part, you are given a sample Python file `sampleDB.py`. Your task is to modify and extend it. Save it as `cmpe351db.py` at the end of this part. Having each step implemented correctly, this script should create and populate all tables in *CMPE351* when it is run.

(a) Add `CREATE TABLE` statements for all tables into `cmpe351db.py`. You may want to use export feature of LucidChart instead of writing each query from scratch. Remember to add keys and other types of constraints(not null, unique etc) when necessary.

(b) Read and format data from student.csv. You will need to import external libraries for this task[1]. Format data so that each student record is a tuple. Let us call this structure `studentData`.

(c) Iterate over `studentData` to execute `INSERT INTO` statements for each tuple. These tuples should be inserted into *student* table.

(d) For each student record, generate a tuple *grades* with 3 elements (<midterm>,<final> ,<lab>). e.g. (95,null ,null). Get the subset [studentID] from`studentData` and append the randomly generated grade tuples to it. Let us call this structure `gradeData`.

(e) Iterate over `gradeData` to execute `INSERT INTO` statements for each tuple. These tuples should be inserted into *grade* table.

(f) Repeat similar steps as in (d)-(e) for *submission* table. You can create random dates for deadline. `subgrade,subDate` columns. Pay attention to `studentID` and `assignmentID` columns; together they make a composite key.

## 1.3 Part 3: View and Update Database (10p)

In this part, you are going to execute variations of `SELECT` statements. You may prefer to use MySQL in batch mode, or you may prefer to stick with Python and add your `SELECT` queries in `cmpe351db.py`. Remember for some of the following tasks, you will need to `JOIN` tables.

(a) Add quiz attribute to grade table.

(b) Calculate overall course grades of students using the formula $overall = (mid * 25 + quiz * 15 + lab * 20 + final * 40)/100$ and display as `overall_grade` (with corresponding student IDs).

(c) Add `bonus` attribute with default value 10 to submission table. Increase submission grade of students by `bonus` who submit their assignments before deadline.

(d) View students with their course grades (midterm-final-lab).

(e) Sort student with respect to their midterm grades.

[1] csv(click for documentation) or pandas (click for documentation) is useful.
[1] source <queries>.sql − > how we execute queries in batch mode

### 1.4 Part 4: Normalize and View Database(20p)

In this part, you are going to inspect cmpe351DB to check for anomalies. Your task is to avoid the anomalies by applying normalization. We have three normal forms to take care: First normal form (1NF), second normal form (2NF) and third normal form (3NF). Use pen and paper to formulate the changes first and then do the actual implementation.

(a) Inspect each table and convert to 1NF if it violates the rules of 1NF.

(b) Inspect each table and convert to 2NF if it violates the rules of 2NF.

(c) Inspect each table and convert to 3NF if it violates the rules of 3NF.

(d) View each table after normalization.

(e) Save your queries in `normalization.sql`. Note that the order of (a)-(b)-(c) is important to follow.

## 2 Regulations

- This project requires individual work.
- If there is an unclear part in your code, prepare yourself to explain it during presentation.
- Presentations will take place in weeks 12- 13. Projects without presentation will be graded out of 70.
- Late submission is **not** allowed.
- This project will be graded out of 100. Readme file(10p) and presentation(30p) points will be added to your points from solution.
- You must follow announcements regularly to check for possible edit/correction on questions. If you detect a mistake, e-mail the TA as soon as possible.
- ZERO tolerance policy for cheating.

# 3 Submission

You should submit your solution as [name_surname_project].zip file to Blackboard. Compressed folder should include the following files.

1. er.png: Database schema, solution of Part 1.
2. `cmpe351db.py`: Python source file with solution to Part 2 (and optionally to Part 3) .
3. `queries.sql`: Text file with solution to Part 3(If you chose to use MySQL in batch-mode for Part 3).
4. `normalization.sql`Text file with solution to Part 4.
5. results.png: Screenshots of resulting tables from Part 3 and Part 4.
6. README.xx : Readme document of any format(Word,latex, .md, .txt)

**Resources** Format of this Project Description document is adapted from open source course materials found in ceng.metu.edu.tr.
https://www.py4e.com/lectures3/
https://www.py4e.com/code3/