BBM473 Project - Group 2
Course Enrollment System
Phase 2

21627103 - Ayça Meriç Çelik
21526638 - Okan Alan
21527214 - Kaan Mersin

1. Functionalities

1.1. Procedures:

a. Base Procedures: Insert/Update/Delete

We have insert, delete and update procedures for each table. The formats of these can be seen below:

*insert*TableName**
*delete*TableName**
*update*TableName*(all attributes of the table)*

As the name suggests, we use insert procedures to add an entry to the table, delete procedures to remove an entry from the table, and update procedures to change the values of entries.

Update procedures take a parameter for each column of the table, and update each column value with the newer value.

b. AllInstructorInOneDepartment: Takes the departmentID as a parameter and lists all the instructors who belong to this department.

c. AllStudentInOneDepartment: Takes the departmentID as a parameter and lists all the students who belong to this department.

d. OneInstructorAllClasses: Takes the instructorID as a parameter and lists all the course-class-section entries which are taught by this instructor.

e. OneStudentAllGrade: Takes the studentID as a parameter and lists all the graded course-class-section entries of his/her.

f. OneStudentTookAllCourse: Takes the studentID as a parameter and lists all the course-class-section entries he/she takes currently (in his/her basket).

1.2. Views

a. JoinCourseClassSection: This view behaves like a natural join of three different tables: course, class, and section. We use this view to manage all the course-class-sections opened so far.

b. JoinInstructorDepartment: We use this view to manage all instructors in a department.
c. JoinStudentDepartment: We use this view to manage all students in a department.

d. StudentGradedCourse: We use this view to manage the students and all their gradings of courses.

e. StudentTakeCourse: We use this view to manage the students and all the courses they take.

f. JoinClassInstructor: We use this view to manage all classes with their instructor.

1.3. Triggers

a. StudentTakenCourse_AFTER_INSERT: This trigger decrements the remaining credit of the student by the credit of the course he/she took, and decrements the remaining quota of the section by 1.

b. StudentTakenCourse_AFTER_DELETE: This trigger increments the remaining credit of the student by the credit of the course he/she took, and increments the remaining quota of the section by 1.

c. Student_BEFORE_INSERT: This trigger sets the newly added student's ID by the number coming up next (The new maximum).

d. Instructor_BEFORE_INSERT: This trigger sets the newly added instructor's ID by the number coming up next (The new maximum).

e. Faculty_BEFORE_INSERT: This trigger sets the newly added faculty's ID by the number coming up next (The new maximum).

f. Department_BEFORE_INSERT: This trigger sets the newly added department's ID by the number coming up next (The new maximum).

g. Course_BEFORE_INSERT: This trigger sets the newly added course's ID by the number coming up next (The new maximum).

h. Class_BEFORE_INSERT: This trigger sets the newly added class's ID by the number coming up next (The new maximum).

2. Revision of The Tables

2.1 Adding “ClassID” Attribute To The Class Table:

In our initial design, the relational schema of the “class” table was like this:

Class(Course.ID, Term, Year)

So we define each class entity with course.id (foreign key from “course” table), class term and class year. It is still logically reasonable since a course has classes in different years/terms. We need to distinguish them in some way, and this design satisfies the needs. But, when we get the key of the “class” table as a foreign key for different tables (such as StudentTakenCourse or InstructorGivesCourse), this term and year attributes will be repeated unnecessarily. To eliminate this redundancy, we define a “classID” attribute for table “class”. With this design, each entity in the class table will be distinguishable with only one attribute.

2.2 Renaming The ID Attributes:

We renamed all the “ID” attributes as *TableName*ID to avoid confusion.