Technical University of Crete
Department of Electrical and Computer Engineering

LABORATORY OF DISTRIBUTED INFORMATION SYSTEMS AND APPLICATIONS

DATABASES
A Phase of Project – SPRING SEMESTER 2021-2022

I. General Description

The implementation of the database of a polytechnic department is requested. The database will contain all data related to the department and its operation according to the following specifications. For this database you are given the ER diagram, the implementation of a large part of the ER in a relational PostgreSQL database and initial data. The relational database will be given to you in a backup file which you can restore to a PostgreSQL system installed on your personal computer. This database contains data on the departments, laboratories, courses taught in the department and random records for faculty, laboratory staff and students in the department.

As part of the project you will need to extend the relational base to meet the need for the diploma and thesis and to support the creation of student groups for laboratory work and exercises, as described below.

Appropriate personal data should be maintained for all members of the department (faculty, laboratory staff and students). This data includes: Social Security Number (AMKA) which is unique for each person, first name, last name, patronymic, email address. For students in particular, information should also be available about the registration number, date of registration.

Professors belong to different ranks (regular, substitute, assistant, lecturer). The laboratory staff is also divided into ranks (A, B, C, D). The department is organised research-wise in specific areas (identified by code, title and description). Laboratories have also been set up in the department to support the educational processes. Each laboratory belongs to a single domain, has a unique code, specific title, description and is staffed by teachers and laboratory staff. Each laboratory is headed by a professor, who must necessarily be at the highest level and covers one or more subject areas which are recorded in the form of three-letter codes (e.g. PLH, HRY, THL...*).

All courses are semester-long. For each course there is a unique code and information is kept related to title, description, credits, weekly teaching hours, tutorial support and laboratory work and they have an indicative semester of execution (typical year of study - winter, spring / typical year, typical season). In each semester of the academic year it is decided which courses will be taught. One or more lecturers are appointed for each semester course and, if the course is laboratory-based, the laboratory in which the work is to be carried out and the laboratory staff to support them are appointed beforehand.

Each laboratory course requires the completion of one or more assignments (LabModule) by one or more students. Each lab assignment has a set of rules that define the maximum number of participants in the groups and the percentage of the grade of the assignment in the final lab grade of the course. Students should be able to register to participate either individually or in a group for the laboratory assignments of course.

When the completion of the assignments is assessed, the corresponding grade for each group assignment is entered into the database.

***greek subjects acronyms.

Course grading rules

For each semester course, the grading rules are defined from which each student's final grade is derived. The rules include:

1. The percentage of the written examination in the final grade. If the course is not a laboratory course, the percentage of participation is 100%.
2. If the course is a laboratory course and the student is required to have a laboratory grade above a minimum threshold, that threshold is recorded; otherwise, that minimum threshold is zero.
3. If the course is a laboratory course and requires the student to have a written exam grade above a minimum, then that threshold is recorded. Otherwise, this minimum threshold is zero.

The final laboratory grade for a laboratory course is calculated based on the individual course assignment grades and the percentage of the assignment grade in the final laboratory grade for the course.

The final grade for a course is as follows:

1. If the course is not a laboratory course, the final grade is equal to the written examination grade as the grade is calculated with 100% participation as already mentioned.
2. If the course is a laboratory course and the laboratory grade is strictly less than the relevant minimum, then the final grade is automatically set to zero (0) even if an update to a non-zero value is requested.
3. If the course is a laboratory course and the written examination grade is strictly less than the relevant minimum threshold, then the final grade is the written examination grade (the laboratory grade is disregarded).
4. In all other cases, the percentage of the written examination is applied to combine the laboratory and written examination grades in arriving at the final grade.

In order for a student to be considered successful in a course and for it to be registered, the student must have a final grade of five (5) or higher.

Courses are divided into two categories; (a) mandatory courses, and (b) optional required courses. The first category includes core courses which provide basic knowledge and must all be successfully completed by a student without exception in order to be eligible for graduation. The second category comprises a large number of specialised courses, of which each student is required to select and successfully complete a minimum number in order to graduate. Each course may have no or more prerequisite courses. Also, for each course there are no or more recommended courses, which it is desirable (but not required) that the student should know in order to be more fluent in the course.

In each semester of the academic year, students apply for registration (status requested) to attend semester courses. Requests are generated as proposed (status proposed) by the system. Each request is checked by the system that it meets the required conditions and is approved (status approved) or rejected (status rejected).

In order to obtain a diploma, students are required to complete a thesis for which a title and grade are recorded. In this context, a committee of professors is appointed, one of whom is the supervisor of the thesis. The number of members of the committee is determined by the school's regulations, which may be updated every academic year.

All the Diplomas are recorded in the system, with a unique serial number, the final diploma grade and the date of graduation. The diploma grade is calculated from the average of the grades of all courses required for the diploma with a weighting factor of 80% and the grade of the diploma thesis with a weighting factor of 20%. To calculate the average of the grades of the courses, the grade of each course is multiplied by the weighting factor of the course and the sum of the individual products is divided by the sum of the weighting factors of all the courses. The weightings are calculated according to the number of credits in each course, as shown in the table below:

Credits (ECTS)	1-2	3-4	5
Weighting Factor	1	1.5	2

If a student has successfully completed more courses than the minimum number of courses required for the diploma, the optional compulsory courses with the lowest passing grades are not calculated for the final diploma grade, provided that all the requirements for the diploma are fully met by the remaining courses.

II. Implementation of the required functionality

Create a database in the PostgreSQL database management system and restore the data from the backup file provided. Then implement the following functionality:

1. Convert the part of the ER model enclosed in the diagram by the green polygon into a relational shape. Implement the new tables in the base you have already created.

2. Data management (implementation using postgreSQL functions).

2.1. In the database given to you there is a table DiplomaTitles containing the titles of the theses and dissertations that have been written in our School. Create a function that utilizes the above table to import random theses for students who are in their 4th year of study or older, have enrolled in at least one course in the current semester, and have not been assigned a thesis.

2.2. Create a function to import student work groups for a specific laboratory project (LabModule). The function will take as input the job code and the number of groups to be created. The members of the groups will be randomly selected from the registered students of the course keeping the maximum number of participants defined for the assignment.

2.3. Create a function to import grades for enrolled students in courses of a given semester which is given as a parameter. A random integer number from 1 to 10 will be entered as a written exam grade. If grades already exist for some students, they will not be updated. For lab grades, the grade for the most recent semester (in which the student was enrolled) will be entered if it exists and is greater than or equal to 5. Otherwise, a random integer from 1 to 10 will be entered.

3. Data retrieval and calculations (implementation using postgreSQL functions)

3.1. Retrieval of name and social security number of professors and laboratory staff belonging to laboratories of a specific domain for which the domain code is given.

3.2. Retrieval of the courses along with the grades (the desired grade category will be given, i.e. written exam, lab grade or final grade) for the current semester and for a specific student (the student's social security number is given).

3.3. Retrieval of all optional compulsory courses (code and title) that are scheduled to be taught in the current semester but are not taught.

3.4. Retrieval of the codes of all laboratory exercises of the current semester with the indication YES or NO depending on whether a specific student for whom the student's registration number is given participates in a group implementing the exercise.

3.5. Retrieval of the maximum grade for each course in a given semester of the curriculum. The semester as well as the desired grade category, i.e. written exam, lab grade or final grade, will be given as input. Results to be displayed in descending order of grades.

3.6. Identification of the field or fields where most graduate theses were produced. The field of elaboration is derived from the laboratory in which the supervising professor is affiliated.

3.7. Determination of the workload of all laboratory staff in the current semester. The workload is calculated as the sum of the laboratory hours for the courses supported by each member of the laboratory staff plus one hour per working group participating in a laboratory exercise of the course. The result will be a table with columns: (SSN(AMKA), surname, first name, sum of hours). Each tuple of this table corresponds to one laboratory staff member. The result shall show all laboratory staff members, even if they have a zero workload.

3.8. **(*)** Identification of all thesis titles where all members of the committee work in the same laboratory.

3.9. **(*)** Retrieval of all courses that are prerequisites or recommended, directly or indirectly, for a particular course whose code is given. The result is tuples of the form: (course code, course title).

4. Functionality with triggers in postgreSQL

5.1. Automatic control to ensure that the maximum number of members in a thesis committee and in a working group for a course lab is not exceeded.

5.2. In case of insertion or update in the Semester table, the generated attributes academic_year and academic_season should be calculated appropriately.

5.3. Automatic calculation of final course grades and new status (pass/fail) of student enrollment in courses when necessary individual grades are completed or changed. For this purpose, all the grading rules described above in section 1 will have to be taken into account. It will not be allowed to enter/update enrolments in 'pass' or 'fail' status as the status will be automatically determined by the grades.

5.4. Automatic verification of student enrollment in a semester course to ensure that the prerequisite course restrictions are met and that the total number of credits of courses taken by the student along with that course does not exceed 20 credits or the number of courses does not exceed 6. The check is triggered when new 'requested' records are entered or when updating from 'proposed' to 'requested'. If the conditions are met, the status immediately becomes 'approved', if the check fails then the status is set to 'rejected'. Neither inserting nor updating records to/from 'approved' or 'rejected' status is allowed because this will only be done via the trigger as already described.

5.5. When entering a new future semester (future status), semester courses should be automatically entered. A semester course (CourseRun) is created for each course (Course) that has typical_season equal to the academic_season of that semester. The grading rules (grade_rules), instructors and (for lab courses) the lab staff and lab (Lab) are entered randomly as long as the subject matter of the course is covered by the lab used for the course or owned by the instructors for non-lab courses.

6. Functionality using VIews

6.1. **(*)** Presentation of the supervisor and committee members for the theses of students who have not yet graduated. The view will have two fields (SSN(AMKA), Committee) where the second field will have the format: <lastname1> <name1>, <lastname2> <name2>, ..., <lastnameN> <nameN>. The first name corresponds to the supervisor and the rest to the committee members.

6.2. **(*)** Retrieve the number of students per year of enrolment for the last 10 years who meet the graduation requirements and have not yet completed a thesis. The result will be a table with columns: (year, number). If for any year this number is zero (0), the corresponding tuple will be displayed in the result with a value in the column 'number' equal to zero.

III. Deliverable

The deliverable of phase A of the laboratory work includes the database that you have implemented (backup file). Your deliverable must be accompanied by a summary report in which you will indicate the names of the stored functions that implement each task of the work.

All functions marked with (*) will be delivered with the second phase of the laboratory project.

Phase A contributes 70% to the final lab grade. The remaining 30% is for phase B of the assignment.

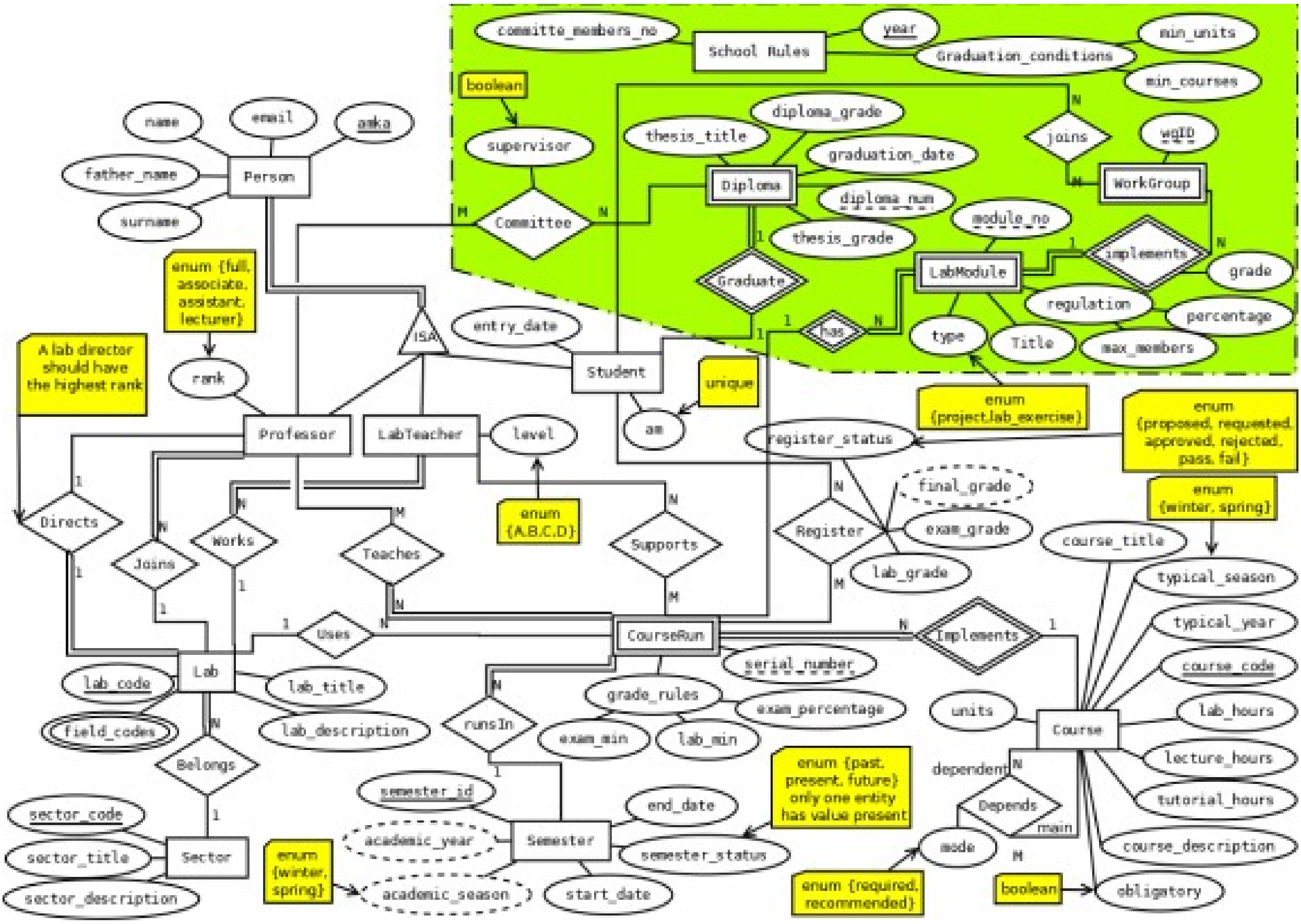
Delivery date: 9 May 2022

IV. PostgreSQL and pgAdmin version

To install the PostgreSQL Database Management System and the pgAdmin GUI to be used in the lab project, you can download the relevant installation file fromhttps://www.enterprisedb.com/downloads/postgres postgresql-downloads **Select version 13.6.**

V. Entity Relationship (ER) diagram of the database

The following is the entity relationship diagram for the database that you will use in the lab project. In addition to the tables that implement the entity types and association types in this diagram, the database also contains the Name, Surname tables with names and surnames in Greek that you can use to create new records for faculty, students and lab staff. It also contains the DiplomaTitles table which contains the titles of diploma theses produced in our School that you can use to import random diploma theses.


committe_members_no
School Rules
year
Graduation_conditions
min_units
min_courses
boolean
supervisor
thesis_title
diploma_grade
graduation_date
Diploma
diploma_num
thesis_grade
joins
wgID
WorkGroup
module_no
implements
LabModule
grade
regulation
percentage
max_members
type
Title
Committee
Graduate
has
name
email
amka
father_name
Person
surname
enum {full, associate, assistant, lecturer}
rank
ISA
entry_date
Student
unique
am
A lab director should have the highest rank
Professor
LabTeacher
level
enum {A,B,C,D}
enum {project,lab_exercise}
register_status
enum {proposed, requested, approved, rejected, pass, fail}
enum {winter, spring}
final_grade
exam_grade
lab_grade
Register
Directs
Joins
Works
Teaches
Supports
Uses
CourseRun
Implements
course_title
typical_season
typical_year
course_code
Lab
lab_code
lab_title
field_codes
lab_description
serial_number
grade_rules
exam_percentage
exam_min
lab_min
runsIn
units
Course
lab_hours
lecture_hours
tutorial_hours
course_description
obligatory
Belongs
Sector
sector_code
sector_title
sector_description
semester_id
academic_year
Semester
end_date
semester_status
start_date
academic_season
enum {winter, spring}
enum {past, present, future} only one entity has value present
dependent
Depends
main
mode
enum {required, recommended}
boolean
M
N
1