

EE468: Database Systems

Design Manual

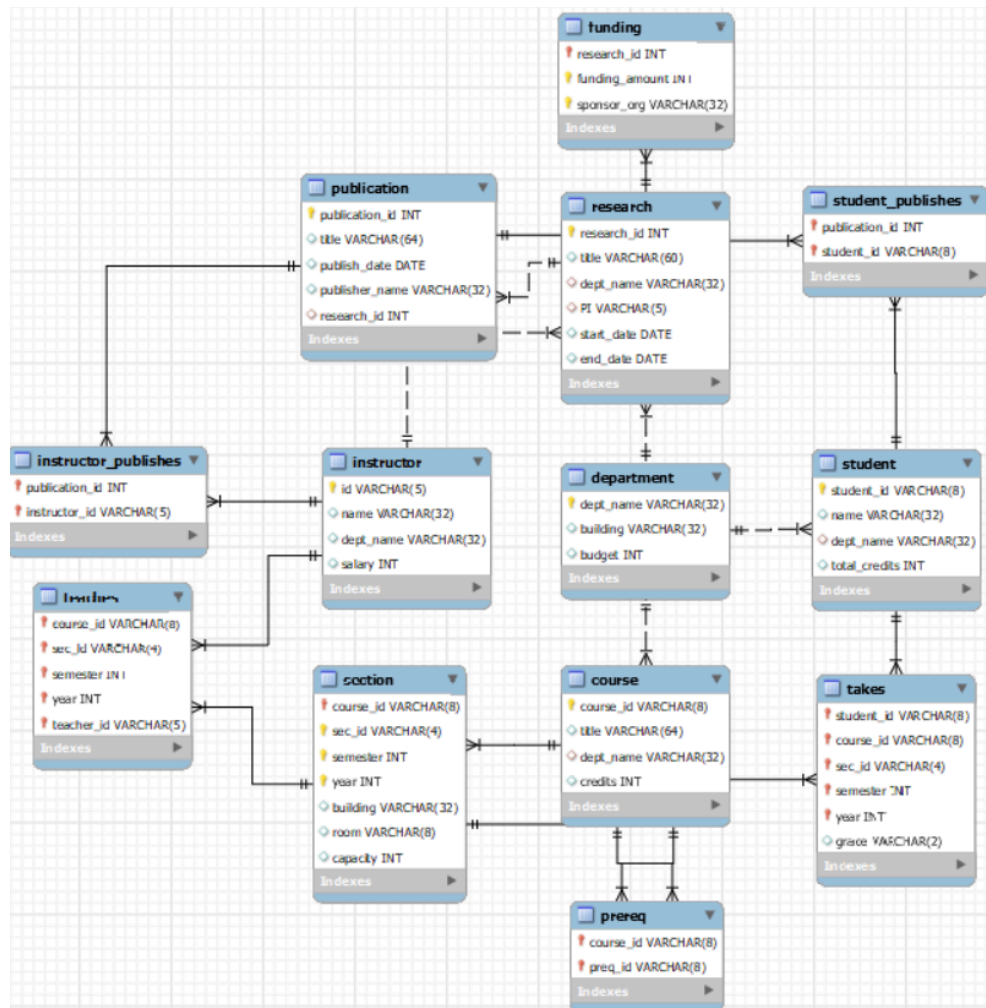


Colin Gasiewicz, Ryan Quirk, Jerye
Demers-St Hilaire, John Parker

1. Database Design

1.1. ER Diagram

The database used by the application is called University, and it contains 14 tables. These tables all have varying relationships allowing the application to perform a variety of tasks. The ER diagram of the University database can be seen below.



1.2. Tables

Instructor

This table contains necessary information about university instructors.

Relationships:

Many to one → Department

One to many → Teaches

One to many → Publication

Primary Key: dept_name

Department

This table contains information regarding each department in the university.

Relationships:

One to many → Instructor

One to many → Course

One to many → Section

Primary Key: id

Student

This table contains necessary information about university students.

Relationships:

One to many → takes

Primary Key: student_id

Course

This table contains basic information about courses offered by the university.

Relationships:

Many to one → Department

One to many → Section

One to many → Takes

One to many → Teaches

Primary Key: student_id

Prereq

This table stores the prerequisite courses for each course offered by the university.

Relationships:

Many to one → Course

Foreign Keys: course_id, preq_id (from Course)

Section

This table specifies the section of a given course, and contains more specific information regarding the class.

Relationships:

Many to one → Course

One to many → Teaches

One to many → Takes

Primary Keys: sec_id, semester, year

Foreign Keys: course_id (from Course)

Takes

This table connects students to what courses they are taking or have taken. Containing only identifying information about the student and course.

Relationships:

Many to one → Student

Many to one → Course

Many to one → Section

Foreign Keys: course_id (from Course) – student_id (from Student) –
sec_id, semester, year (from Section)

Teaches

This table connects professors to what courses they are teaching or have taught. Containing only identifying information about the professor and course.

Relationships:

Many to one → Instructor

Many to one → Course

Many to one → Section

Foreign Keys: course_id (from Course) – teacher_id (from Instructor) –
sec_id, semester, year (from Section)

Project

This table contains all projects that are in the system not every project has a publication. Has title dept_name, PI, start_date, and, end date

Primary Keys: Project_ID

Foreign Keys: dept_name, PI

Publication

This table contains information about various publications instructors have completed while working for the university.

Relationships:

One to one → Project_ID

Many to one → Instructor

Primary Keys: semester, year, title

Funding

Table contains funding sources for projects

Relationships:

Many to One → Project_ID

Primay Keys: project_id, funding_amount, sponsor_org

Instructor_Publishes

There can be multiple instructors on a publication

One to one → Project_ID

One to one → Instructor_ID

Primary Keys: Project_ID, Instructor ID

Student_Publishes

There can be multiple students on a publication

One to one → Project_ID

One to one → Student_ID

Primary Keys: Project_ID, Student_ID

User

Contains usernames and passwords (stored in hashes) for the web application

2. Application Overview

This section goes over the general use cases of the application. The application has three user groups: Admin, Professor, or Student. Each user has its own landing page and separate tables they are able to view. All queries used to populate tables can be found below with a short description of what the tables contain.

2.1. Admin Queries

2.1.1. FR 1

Admins will be able to create a list of professors, which can be sorted either by name, by department, or by salary. The order by clause will be determined by an option selected through the application interface

```
SELECT name, dept_name, salary FROM instructor
ORDER BY name;

SELECT name, dept_name, salary FROM instructor
ORDER BY dept_name;

SELECT name, dept_name, salary FROM instructor
ORDER BY salary;
```

2.1.2. FR 2

Admin users will also be able to create a table of min/max/average salaries by dept.

```
SELECT dept_name, MIN(salary) AS min_salary, MAX(salary) AS
max_salary, AVG(salary) AS average_salary FROM instructor GROUP BY
dept_name
```

2.1.3. FR 3

```
-- 3.1
SELECT COUNT(sec_id) FROM teaches t
INNER JOIN instructor i ON t.teacher_id = i.id
WHERE i.name = ? AND t.year = ? AND t.semester = ?;

-- 3.2
SELECT COUNT(DISTINCT takes.student_id) FROM takes
INNER JOIN teaches ON takes.course_id = teaches.course_id AND
takes.sec_id = teaches.sec_id
INNER JOIN instructor ON teaches.teacher_id = instructor.id
WHERE instructor.name = ? AND takes.year = ? AND takes.semester =
?;

-- 3.3
SELECT SUM(f.funding_amount) from funding f
NATURAL JOIN research r
INNER JOIN instructor i ON r.PI = i.id
WHERE i.name = ? AND r.end_date BETWEEN ? AND ?;

-- 3.4
SELECT COUNT(ip.publication_id) FROM publication p
NATURAL JOIN instructor_publishes ip
INNER JOIN instructor i ON ip.instructor_id = i.id
WHERE i.name = ? AND p.publish_date BETWEEN ? AND ?;
```

2.2. Professor Queries

2.2.1. FR 4

Professors using this application will be able to create the list of course sections and the number of students enrolled in each section that the professor taught in a chosen semester. They will be able to select which semester and year they are querying through the application interface.

```
select course_id, sec_id, semester, year,
(select count(student_id) from takes group by course_id, sec_id,
(semester, year) from section
natural join teaches
where semester = 1 and year = 2020 and teacher_id = 12345;
```


2.2.2. FR 5

Professors using this application will also be able to create the list of students enrolled in a chosen course section they taught in a chosen semester. They will be able to select the semester and year that they are querying through the application interface, as well as the section and course ids.

```
SELECT DISTINCT s.name, s.student_id, ta.grade FROM student s
NATURAL JOIN takes ta
NATURAL JOIN section sec
NATURAL JOIN teaches t
WHERE t.teacher_id = '12345' AND t.course_id = 'EE468' AND
t.sec_id = 01 AND t.semester = 1 AND t.year = 2020;
```

2.3. Student Query

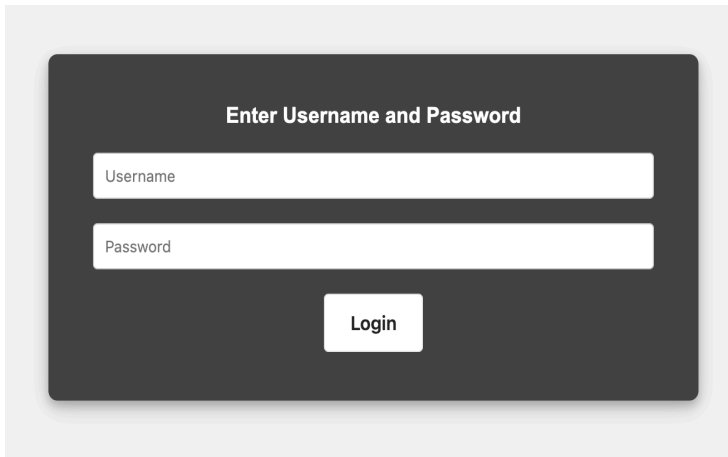
2.3.1. FR 6

Student users will have the ability to query the list of course sections offered by a chosen dept in a chosen year and semester. The 'MA', '2020', and '1' (semester) are all placeholders that the student users can change through the application interface.

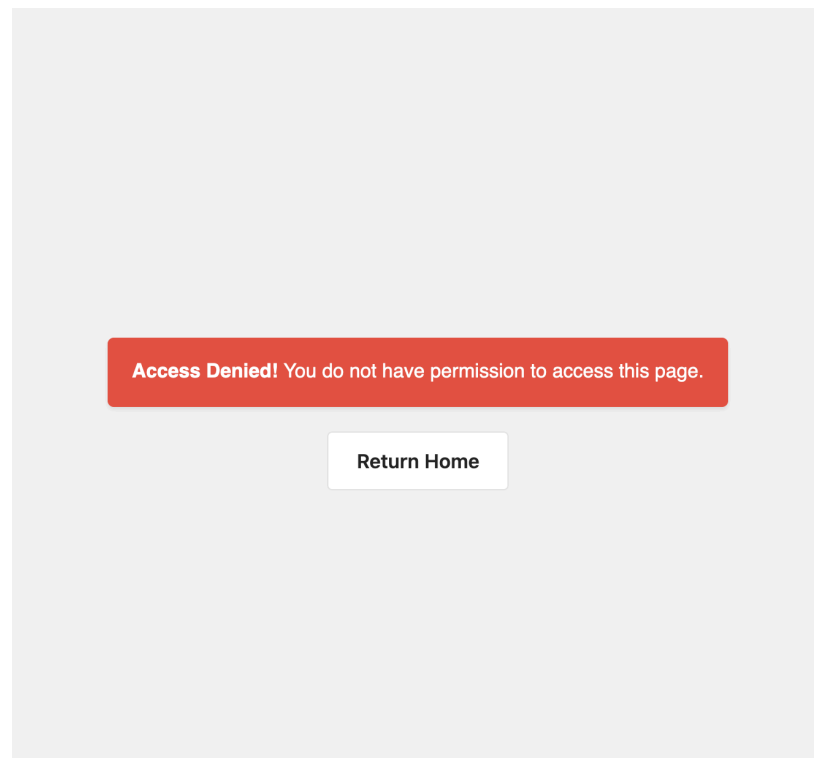
```
SELECT c.course_id, c.title, s.sec_id FROM section s
NATURAL JOIN course c
NATURAL JOIN department d
WHERE d.dept_name = 'MA' AND s.year = '2020' AND s.semester = 1;
```

3. Security

The security of the application is assured through a simple login portal that assigns users to 1 of the 3 permission groups. The login portal is the landing page of the application, and prompts the user for a username and password. Currently there are only 3 users that can be logged into; corresponding to the 3 different permission levels: admin, professor, and student. All account credentials are stored in the user table, and passwords are encrypted with SHA-256. After the user is authenticated by logging in, a global variable is set corresponding to what permission group the user is in. Admin is permission group 1, professor 2, student 3. The permission group is then checked whenever the user tries to access a different page on the website. If the page is outside the user's permission group the user will be served the permission denied page that can be seen below, and be prompted to return to its permission group's landing page.

A dark gray rectangular login portal with rounded corners. At the top, it says "Enter Username and Password". Below this, there are two white input fields: the first is labeled "Username" and the second is labeled "Password". At the bottom center, there is a white button with the text "Login".

Login Portal



Permission Denied Page