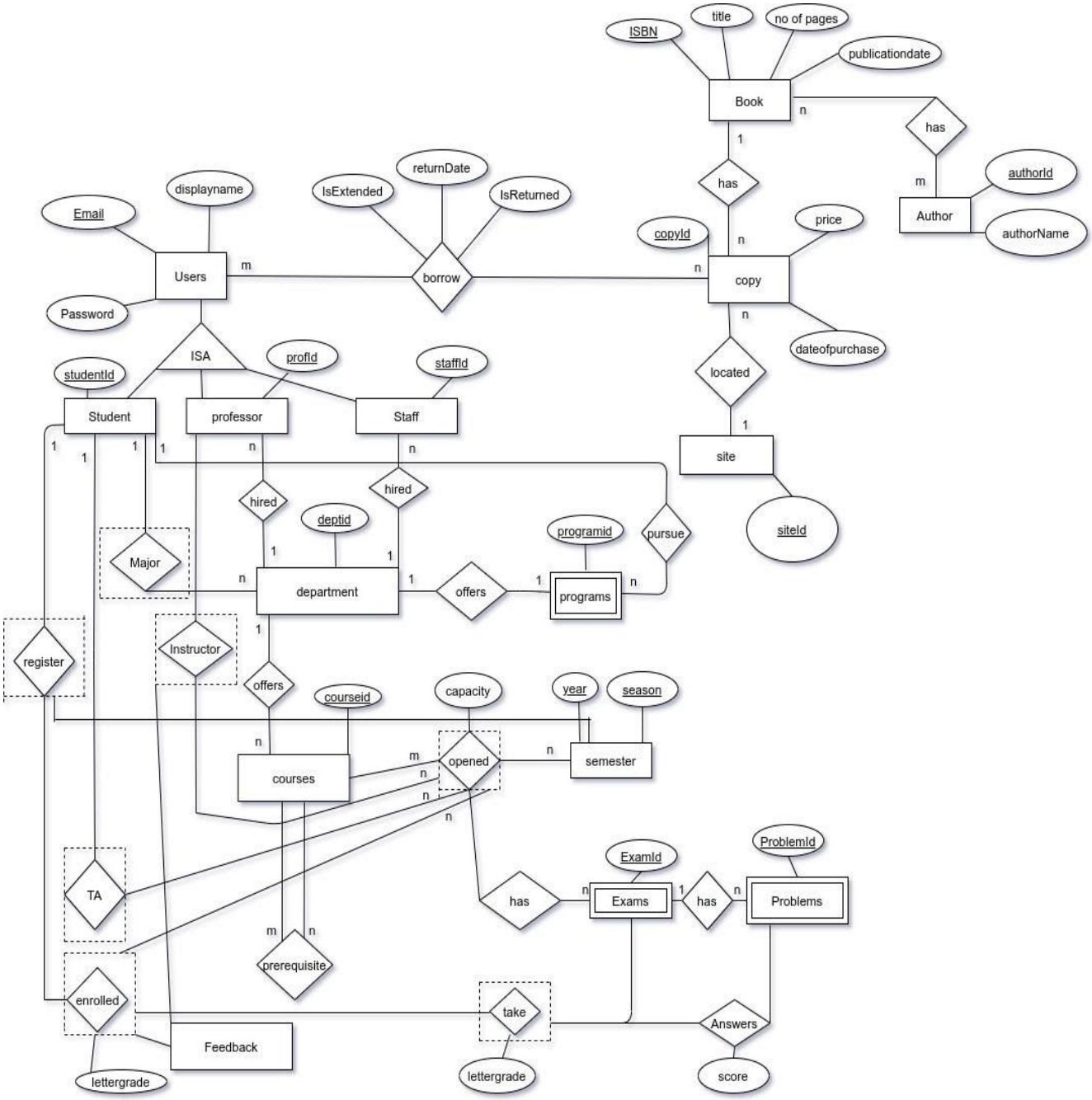
CSE 560 - Project 1

Shreyas Shirish Pimpalkar - spimpalk
Mandeep Singh - msingh42
Rohith Reddy Kolla - rkolla

ER Schema:
ER schema for Tinyhub is shown below.



Introduction ER Schema:

In the above schema we see that the user entity has three types namely: students, professors and staff. students are registered for a semester and registered students can enroll in the courses opened in that semester. Since both registered and enrolled can be entity and relationship they are defined by the aggregation of entity and relationship in the diagram. Only the students enrolled for a particular open course can get the grades for that opened courses. Each open course has some students employed as TA which is represented by the TA entity.

Every User must belong to some department therefore the entities student, professors and staff are connected to the department via student_major, hired and hired relationship respectively.

Department and course entities are connected to each other via the offers relationship.

Courses and semester entities are connected via the opened relationship which is also an aggregate entity with the attribute capacity (representing the courses opened in a semester).

As courses have some prerequisite courses it is a recursive relationship.

Opened entity is connected to exams with has-relationship and exams entity is connected with the problem entity with has-relationship. Both exams and problems are weak entities.

Also enrolled students take exams and they get letter grades. Therefore, take is an aggregate entity-relationship.

For the library management system, Users can borrow copies of the book from the library therefore, the User entity is connected to the copy entity with the borrow relationship. Also books have multiple copies therefore, the copy entity is connected to the book entity with has-relationship.

Book has author/authors which is a separate entity connected to books via has-relationship.

Also copy and site are connected by n-to-1 relationship.

Relational Database Schema:

Relational Database schema for tinyhub is shown below.

departments (department_id, department_name):
PK (department_id)

program (program_id, department_id)
PK(program_id)
FK (program.department_id , departments.department_id)

user (email, password, displayname, accounttype):
PK (email)

students (student_id)
PK(student_id)
FK (student_id, user.email)

professor (professor_email, department_id)
PK(professor_email)
FK (professor_email, user.email)

FK (professor.department_id, departments.department_id)

staff (staff_email, department_id)
PK(staff_email)
FK (staff_email, user.email)
FK (staff.department_id, departments.department_id)

student_major (student_email, department_id)
PK (student_email, department_id)
FK (student_major.student_email, student.student_id)
FK (student_major.department_id,departments.department_id)

pursued_programs (student_email, program_id)
PK (student_email, program_id)
FK (pursued_programs.student_email, student.student_id)
FK (pursued_programs.program_id, program.program_id)

courses (course_id, department_id)
PK (course_id)
FK (course.department_id, departments.department_id)

opened_courses (opened_course_id, course_id, year, season, capacity, instructor)
PK (opened_course_id)
FK (opened_courses.year, semester.year)
FK (opened_courses.season, semester.season)
FK (opened_courses.course_id, courses.course_id)
FK (opened_courses.Instructor, professor.professor_email)

teaching_assistants (opened_course_id, student)
PK (opened_course_id,student)
FK (teaching_assistants.opened_course_id, Opened_Course.opened_course_id)
FK (teaching_assistants.student, students.student_id)

semester (Year, Season)
PK (Year, Season)

sem_registration (student_id, year, season)
PK (student_id, year, season)
FK (sem_registration.student_id, students.student_id)
FK (sem_registration.year, semester.year)
FK (sem_registration.season, semester.season)

prerequisite_course (course_id, prerequisite_course_id)
PK (course_id,prerequisite_course_id)
FK (prerequisite_course.course_id, courses.course_id)
FK (prerequisite_course.prerequisite_course_id, courses.course_id)

enrolled (student_id, opened_course, grade)
PK (registered_student, opened_course)
FK (enrolled.student_id , sem_registration.student_id)
FK (enrolled.opened_course , opened_courses.opened_course_id)

exam (exam_id, opened_course)
PK (exam_id)
FK (Exam.opened_course, opened_courses.opened_course_id)

exam_result (exam_id, enrolled_student, grade):
PK (exam_id)
FK (exam.enrolled_student, student.student_id)

problems (problem_id, exam_id, problem)
PK (problem_id)
FK (problem.exam_id, exam.exam_id)

problem_score (problem_id, enrolled_student, score)
PK (problem_id, enrolled_student)
FK (problem_score.enrolled_student, student.student_id)

instructor_feedback (enrolled_student, opened_course, feedback)
PK (opened_course,enrolled_student)
FK (instructor_feedback.enrolled_student, enrolled.student_id)
FK (instructor_feedback.opened_course, opened_courses.opened_course_id)

book (ISBN, title, num_pages,pub_date)
PK (ISBN)

author (author_id, author_name)
PK (author_id)

book_author (book_id, author_id)
PK (book_id, author_id)
FK (book_author.book_id, book.ISBN)
FK (book_author.author_id, author.author_id)

copy (copy_id, book_id, price, date_of_purchase, site_id)
PK (copy_id)
FK (copy.book_id, book.ISBN)
FK (copy.site_id,site.site_id)

site (site_id)
PK (site_id)

borrowed_books (user_id, copy_id, return_date, is_returned, is_extended)
PK (user_id, copy_id)
FK (borrowed_books.user_id,user. user_id)
FK (borrowed_books.copy_id, copy.copy_id)

ER Schema to relational schema:

For the conversion, Entity and a weak entity in ER Model is changed into tables or we can say for every Entity in ER model, a table is created in the Relational Model. And the attributes of the Entity gets converted to columns of the table. Primary key specified for the entity in the ER model, will become the primary key for the table in the relational model. Relationships based on the requirement are either converted to a relational table or just added as a separate attribute with a foreign key constraint.

The User management part of the ER schema is mapped to the relational schema using four tables. user, student, professor and staff where the studenID, professorID and staffID refer to the email used to create a unique user. The display name and password are only in the User table.
In the user management part the entity user is converted into a user table and since the user entity has ISA relationship with student,professor and staff. Three separate tables are created for students, professor and staff.

The Department Management part is mapped by adding four new tables – department, program, student_major and pursued_programs. The Department table contains the department_id which is referred to by the program and student_major tables. The Program table contains the program_id which is referred to by the pursued_programs table. The student_major and pursued_programs tables also refer to the student ID to indicate the departments the students are majoring in and the programs they are pursuing.

The Course Management part is mapped by adding the tables courses, opened_courses, semester, teaching_assistants, instructors, prerequisite_courses, sem_registration.
The student- Course management part is mapped by adding enrolled, instructor_feedback.

Similarly, the library management part of the ER model has the book, copy, author, site entities converted to tables and the entire library system connected to the user management system using booked relationship.

Requirement Satisfaction:

1. User Management.

- Users can sign up with an email address which is the primary key and therefore is used to identify each unique account.

- The user account can be of three types student, Professor or Staff as is reflected in the accounttype column for the user. The tables student, Professor and Staff refer to the email address from the User table as the primary key.

- Since the password column under users is set be not null and therefore has to be set for account creation.

- The column for display name requires each row to be unique or null since two users cannot have the same display name.

- The email, password and display name are of the string datatype.

2. Department Management

- Department has an identifier department_id which is the primary key.

- Since each professor and staff are hired by one department, there is a column called department for the Professor and Staff tables which indicates it.

- Departments can offer different programs making the program entity a weak entity represented in a separate table with department id (department_id) as one of the columns.

- A separate table called student_majors is used to indicate the department each student is majoring in. Each student is capable of having multiple departments as majors.

- Similarly, a separate table is made to indicate the programs each student is pursuing. A student can pursue multiple programs and the student_major table can be referred to ensure that the student is majoring in the same department as the program being pursued.

3. Course Management

- Departments can offer different courses and the department for each course is shown in the department_id column in the Course table.

- The opened_courses table requires the course_id and semester in which it is being opened to indicate the course and the semester it is being opened in.

- The semester table contains Year and Season and is identified by them since these two columns form the primary key.

- A course does not have to be opened every semester and only those courses opened will be in the opened_courses table.

- Since every opened course has TAs, a table called TAs is made where one column requires the ID of the student and the other column required the opened_course_id of the opened course. Furthermore, since each opened course requires one instructor, there is a column Instructor in the opened_course table which refers to the Professor ID.

- Since there may be multiple opened courses from the same course in a semester, the primary key for the Opened_Course table is not the columns course_id and Semester but instead is a separate column called opened_course_id.

- The instructor column in Opened_Course refers to the ID column in the Professor table so only the professors can be instructors.

- Similarly, the student column in the teaching_assistants table refers to the ID column in the student table so only the students can be TAs.

- prerequisite_course table indicates the prerequisite courses required for each course.

- The sem_registration table contains 3 columns – student_id which refers to the ID of the student table and Year and Season which refers to the Year and Season of the Semester table which allows students to register in different semesters.

4. student- Course Management

- The Enrolled table has the registered_student and opened_course_id as the primary key where the registered_student refers to the ID in the student table and the opened_course_id refers to the ID of the opened_course table allowing a student to enroll in different opened courses.

  The sem_registration table could be checked to make sure the student is registered in the semester as the course being offered. The Enrolled Table contains the grade of student enrolled in a course and could be checked along with the prerequisite_course table to verify if the student is qualified. The student_major table could be checked along with the Courses table to verify that the student is majoring in the same department as the course being offered. Lastly, the capacity column in the opened_course table could be checked along with the Enrolled table to see whether it is full.

- The Enrolled table has a column for grade which shows the grade received by a student in the enrolled course.

- The instructor_feedback table contains the feedback posted by a student for the instructor of the course they enrolled in.

- The Exam table contains the exam_id and the course_id where each course can have one or more exam. The exam_result table contains the student ID and the exam_id along with the lettergrade they received for that exam.

- The Problem table contains the problem_id and exam_id where each exam can have a number of problems. The problem_score table containing the student ID and the problem_id along with the score they received for that problem.

5. Library Management

- The Book table contains an ISBN column as the primary key along with the columns title, num_pages and publication_date.

- Since an author may write multiple books and a book may have more than one author, there is a table Author which contains the author_id as primary key and Author_Name along with a table book_author which contains book_id which refers to ISBN in the book table and Author_ID in the Author table to indicate which books have been written by which authors.

- Since there may be more than one copy of a book in the library, there is a table called Copy which contains a column copy_id as the primary key and a column Book_ID which indicates what book it is.

- There is a table called Site which has a column site_id as the primary key to represent the different physical sites of the campus library.

- Since copies of the books may be located in different sites of the library the Copy table has a column site_id which refers to the site_id from the Site table in order to identify the location of each copy.

- Since different copies of the same book may have different dates of purchase and prices, the Copy table contains the columns date_of_purchase and price to indicate them.

- There exists a table borrowed_books in which the column ID referring to the email ID from the Users table and the column copy_id referring to the copy_id from the Copy table form the primary key. There is a column called return_date which indicates the date before which the copy has to be returned (2 weeks from the borrowed date).

- There is also a boolean column Is_Extended which changes to True if the user requests an extension. This can be used to modify the return_date by 1 week.

- There is another column Is_Returned which indicated if the user has returned the book in time. Failing to do so would prevent them from borrowing another book.

- Since the borrowed_books table contains the copy_id and not the Book_ID, the copies borrowed and the copies returned could be verified to be the same before accepting the return.

Advantages of given ER model

- Clear distinction between teacher, professor and staff by creating separate entity for each one.
- It accommodates instructor without creating a separate entity.
- The borrowing of books is able to function with only 3 attributes for the Borrowed entity.

Disadvantages of given ER model

- There are composite primary keys in some tables which makes joins slow.