Databases Programming Project: **Proposal Report**

이름 (Name): 허치영, 하현진

학과 (Department): 정보컴퓨터공학부, 전기컴퓨터공학부 정보컴퓨터공학전공

학번 (Student ID): 202055623, 201924604

1. **프로젝트 주제 (Project topic)**

|  |
| --- |
| ※ 프로젝트의 주제(topic)에 대한 설명. 참신성 및 프로젝트와의 적합성을 고려하여 어떤 단체에 대한 데이터베이스인지, 구현하려는 시스템에 대한 설명 등에 대해 기술.  연구실과 연구실에 진학하고자 하는 학생을 연결하는 SNS와 컨택 관리 시스템이 결합된 플랫폼을 제작한다.  플랫폼에서 제공할 기능   * 연구실이 작성한 게시글과 그에 대한 댓글, 좋아요 등 SNS 기능을 제공한다. * 학생이 연구실에 컨택하는 과정에서 연구실에게는 학생의 학부, 전공 등과 같은 데이터를 제공하고 컨택 과정을 편리하게 관리할 수 있는 기능을 제공한다.   위 플랫폼에서 사용할 데이터베이스 애플리케이션을 제작한다. |

1. **사용자 (역할) (Users / Roles)**

|  |
| --- |
| ※ 해당 단체에 어떤 사용자(역할)들이 존재할 수 있는지 설명. 최소 3종류 (2인팀은 5종류) 이상의 역할을 포함. 구현할 시스템에서 각 사용자들이 어떤 역할을 하는지 등에 대해 기술.   * 독자 (reader) :   SNS 기능 전반 (게시글 조회, 좋아요, 댓글 작성 / 삭제 )   * 학부생 (student) :   일반 독자 + 프로필 등록 / 수정 (학교, 전공), 연구실 컨택 제안   * 연구자(researcher) :   게시글 작성, 학생 프로필 조회   * 연구실 관리자(lab\_manager) :   연구실 프로필 작성, 편집, 삭제, 컨택 수락, 컨택 거절, 소속 연구실 등록   * 관리자 계정(admin) :   부정 사용자 정지, 글/댓글 삭제 권한 |

1. **기능 (Functions)**

|  |
| --- |
| ※ 어떤 기능들이 구현될지에 대한 설명. 각 기능들이 어떤 사용자에게 적합한지, 각 기능들을 구현하는 데에 어떤 SQL feature (DML, Transaction 등) 가 필요할지 등. 사용자 역할 에 따라 적게는 3가지, 많게는 5가지 이상의 기능을 구현하고, 해당 기능을 사용할 때에 유용한 세부 기능 (ex. Sorting, Filtering by condition 등)을 포함.  사용자의 역할 (reader, student, researcher, lab\_manager, admin)  독자 (reader)   * 게시글을 조건에 따라 조회하고 댓글, 좋아요를 등록함. * 게시글 작성자(연구자 or 연구실)의 학교, 전공 등의 조건에 따라 게시글을 조회한다. * 학교, 전공 등의 조건에 따라 사용자의 프로필을 조회한다. * 학교, 전공 등의 조건에 따라 연구실의 프로필을 조회한다. * 게시글에 댓글, 좋아요를 등록한다. * 특정 연구실/연구자를 구독한다. * 본인 프로필을 관리한다. * 본인 프로필에 학교, 전공을 등록하는 순간 student가 된다.   학부생 (student)   * 독자의 하위 클래스 + 본인 학교, 전공 등록 + 연구실 컨택 가능   + 본인의 학교, 전공을 수정한다.   + 특정 연구실에 컨택한다.   + 본인 프로필에 학사 상위의 학위 과정을 등록하는 순간 researcher가 된다.   연구자 (researcher)   * 학부생의 하위 클래스 + 본인 학위 과정(석사 이상) 등록 + 현재 소속 연구실 등록 + 게시글 작성 가능   + 본인의 학위 과정(석사 과정, 박사 과정, 교수)을 등록한다.   + 현재 소속 연구실을 등록한다.   + 본인의 게시글을 작성 / 수정 / 삭제 한다.   + 연구실 관리자 권한을 획득하면 lab\_manager가 된다.   연구실 관리자 (lab\_manager)   * 연구자의 하위 클래스 + 본인 연구실 소속 인원 관리 + 연구실의 게시글 작성 + 연구실 프로필 관리 + 학생의 컨택 요청 관리   + 본인 연구실의 소속인원(researcher)을 추가 / 제거 한다.   + 연구실의 게시글을 작성 / 수정 / 삭제 한다.   + 연구실의 프로필을 관리한다.   + 컨택 수락 / 거절 한다.   + 컨택의 동봉된 포트폴리오를 다운받는다.   관리자 계정 (admin)   * SNS 게시글 관리, 사용자 정지 권한   + 모든 게시글에 대한 관리(수정 / 삭제)를 한다.   + 특정 유저의 role을 lab\_manager로 변경한다.   + 특정 유저의 role을 banned로 변경한다. |

1. **데이터베이스 스키마 (Database schema)**

|  |
| --- |
| ※ 데이터베이스 스키마에 대해 작성 (필요한 경우 Schema diagram을 포함). 각 relation의 constraint와 key, authorization 등에 대한 설명.  CREATE TABLE Users (  id INT PRIMARY KEY,  username VARCHAR(50) NOT NULL,  password VARCHAR(50) NOT NULL,  role ENUM('reader', 'student', 'researcher', 'lab\_manager', 'admin', ‘banned’) NOT NULL  );  Authorization :   * role은 admin만 수정할 수 있다. * username, password는 본인만 수정할 수 있다.   CREATE TABEL Schools (  id INT PRIMARY KEY,  school\_name VARCHAR(50) NOT NULL,  )  CREATE TABEL Majors (  id INT PRIMARY KEY,  major\_name VARCHAR(50) NOT NULL,  school\_id INT,  FOREIGN KEY (school\_id) REFERENCES Schools(id),  )  CREATE TABLE Profiles (  user\_id INT PRIMARY KEY,  name VARCHAR(50),  school\_id INT,  major\_id INT,  lab\_id INT,  degree ENUM('undergraduate', 'master', 'doctoral', 'professor'),  FOREIGN KEY (user\_id) REFERENCES Users(id),  FOREIGN KEY (school\_id) REFERENCES Schools(id),  FOREIGN KEY (major\_id) REFERENCES Majors(id),  FOREIGN KEY (lab\_id) REFERENCES Labs(id)  );  Authorization :   * name,school,major,degree는 본인만 수정할 수 있다. * researcher의 소속 lab\_id는 해당 연구실의 lab\_manager만 추가해줄 수 있다.   CREATE TABLE Labs (  id INT PRIMARY KEY,  name VARCHAR(50) NOT NULL,  description TEXT,  manager\_id INT,  FOREIGN KEY (manager\_id) REFERENCES Users(id)  );  Authorization :   * name, description은 manager\_id의 lab\_manager만 수정할 수 있다.   CREATE TABLE Posts (  id INT PRIMARY KEY,  author\_id INT,  lab\_id INT,  content TEXT,  date\_posted DATETIME DEFAULT CURRENT\_TIMESTAMP,  FOREIGN KEY (author\_id) REFERENCES Users(id),  FOREIGN KEY (lab\_id) REFERENCES Labs(id)  );  Authorization :   * author\_id가 일치하는 유저 혹은 admin만 해당 tuple을 삭제 할 수 있다. * content는 author\_id를 가진 유저 혹은 admin만 수정할 수 있다. * lab\_id는 lab\_manager가 개인이 아닌 연구실 입장으로 쓴 글에만 존재한다.   CREATE TABLE Comments (  id INT PRIMARY KEY,  post\_id INT,  author\_id INT,  content TEXT,  date\_posted DATETIME DEFAULT CURRENT\_TIMESTAMP,  FOREIGN KEY (post\_id) REFERENCES Posts(id),  FOREIGN KEY (author\_id) REFERENCES Users(id)  );  Authorization :   * author\_id가 일치하는 유저 혹은 admin만 해당 tuple을 삭제 할 수 있다. * content는 author\_id를 가진 유저 혹은 admin만 수정할 수 있다.   CREATE TABLE Likes (  post\_id INT,  user\_id INT,  date\_liked DATETIME DEFAULT CURRENT\_TIMESTAMP,  PRIMARY KEY (post\_id, user\_id),  FOREIGN KEY (post\_id) REFERENCES Posts(id),  FOREIGN KEY (user\_id) REFERENCES Users(id)  );  Authorization :   * author\_id가 일치하는 유저 혹은 admin만 해당 tuple을 삭제 할 수 있다.   CREATE TABLE Contacts (  student\_id INT,  lab\_id INT,  portfolio\_path VARCHAR(255),  date\_contacted DATETIME DEFAULT CURRENT\_TIMESTAMP,  PRIMARY KEY (student\_id, lab\_id),  FOREIGN KEY (student\_id) REFERENCES Users(id),  FOREIGN KEY (lab\_id) REFERENCES Labs(id)  );  Authorization :   * student\_id와 lab\_id가 일치하는 유저와 연구실만 조회가 가능하다.   CREATE TABLE Subscriptions (  user\_id INT,  lab\_id INT,  date\_subscribed DATETIME DEFAULT CURRENT\_TIMESTAMP,  PRIMARY KEY (user\_id, lab\_id),  FOREIGN KEY (user\_id) REFERENCES Users(id),  FOREIGN KEY (lab\_id) REFERENCES Labs(id)  );  Authorization :   * 구독의 당사자, user\_id가 일치하는 유저만 해당 튜플을 삭제할 수 있다. |