

[데이터베이스시스템]

Term Project



소 속 : 충북대학교 소프트웨어학과
과 목 명 : 데이터베이스시스템
이 름 : 조준화
학 번 : 2020039071
교 수 : 노서영 교수님
제 출 일 : 24.12.10

0. Github URL

<https://github.com/jjj5306/ClubManagementSystem>

1. ER 다이어그램 수정 보완 및 요구사항 재정의

1. 동아리는 다른 동아리와 중복되지 않는 동아리 이름과 동아리 번호, 활동 분야, 부원 수를 가진다.
2. 동아리는 부원의 학번, 이름, 연락처, 가입일, 학과를 등록하고 관리한다.
3. 모든 학생은 최대 하나의 동아리에만 가입할 수 있다.
4. 학생은 동아리에서 회장, 부회장, 임원, 일반 부원의 역할을 가진다.
5. 동아리는 동아리 회장을 가진다.
6. 동아리는 매 학기 예산 정보를 관리한다. 예산 정보에는 사용처별 영수증, 사용 분야, 날짜, 금액이 포함되어야 한다.
7. 동아리는 한 명의 지도교수를 가져야 하고 지도교수의 이름, 소속 학과, 연락처를 관리한다.
8. 교수는 동아리를 담당하지 않아도 되고, 한 번에 하나의 동아리만 담당할 수 있다.
9. 동아리는 동아리 주요 활동을 가지고 주요 활동 별 이름, 날짜, 참여 인원, 수상 유무 정보를 관리한다.
10. 신규 동아리가 등록될 수 있다. 신규 동아리는 10명 이상의 부원을 포함해야 한다.
11. 동아리는 부원이 10명 미만이면 해체된다.
12. 동아리는 반드시 웹페이지를 가진다. 웹페이지에는 동아리 소개를 포함해야 한다.
13. 동아리는 매 학기 프로젝트를 하나 이상 진행해야 한다.
14. 프로젝트는 참여인원(수), 프로젝트 목적, 프로젝트 관리 도구, 프로젝트 주제, 프로젝트 이름을 포함해야 한다.
15. 프로젝트는 웹사이트에 게시될 수 있다.
16. 웹페이지는 동아리 상태(활동, 휴면, 해체)에 따른 웹페이지 표시 여부도 함께 관리된다.

2. 둘 다 total인 경우 table merge

동아리 - 가짐 - 웹페이지

3. 둘 다 total이 아닌 경우

X

4. 1:N Relationship Types 매핑

1 쪽의 PK를 N쪽의 FK로 사용

동아리 가입함 학생(N)

동아리 관리함 주요 활동(N)

웹페이지 게시함 프로젝트(N) ⇒ 웹페이지는 동아리에 merge 되었으므로 동아리 번호를 사용하면 됨

동아리 진행함 프로젝트(N)

5. M:N Relationship Types 매핑

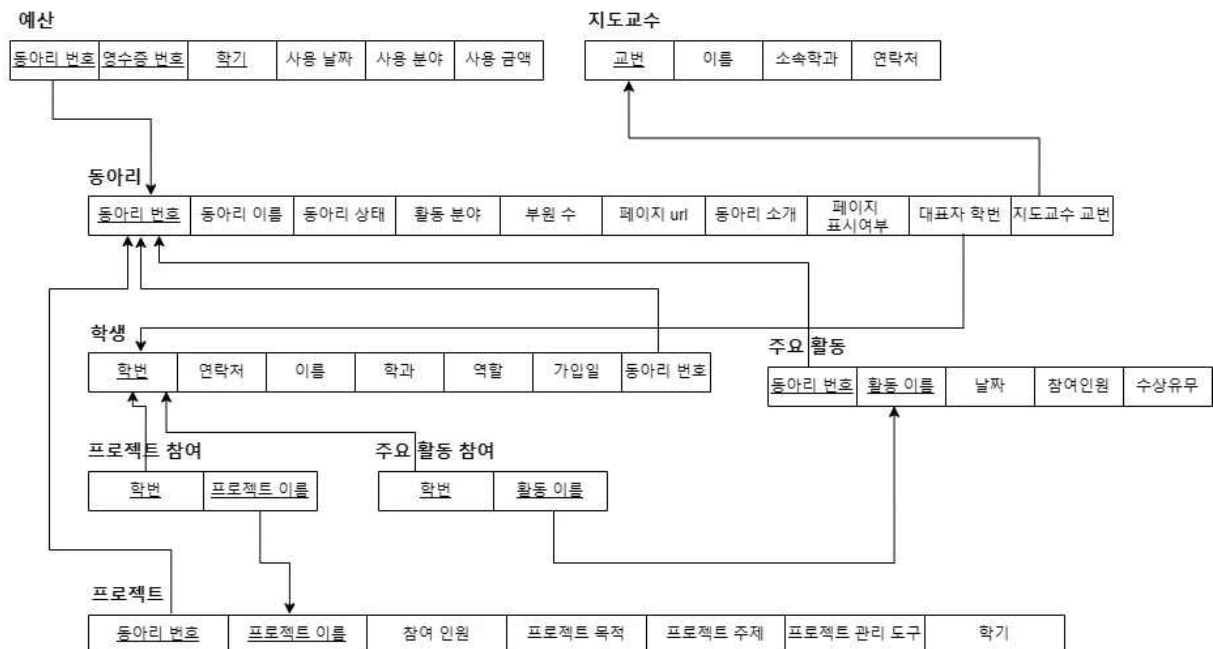
학생 - 참여함 - 프로젝트

학생 - 참여함 - 주요 활동

6. Multivalued attributes 매핑

7. N-ary Relationship Types

존재하지 않음



3. 정규화 과정

1. 1NF

이미 만족함

2. 2NF

2NF를 위해 key 및 FD 검출

1. 예산 테이블

key {동아리번호, 영수증번호}, candidate key 존재하지 않음

FD {동아리번호, 영수증 번호} → 사용 날짜, 사용 분야, 사용 금액

2NF 만족

2. 지도교수 테이블

key {교번}, candidate key {연락처}

FD 교번 → 이름, 소속학과, 연락처

FD 연락처 → 교번, 이름, 소속학과

2NF 만족

3. 학생 테이블

key {학번}, candidate key {연락처}

FD 학번 → 연락처, 이름, 학과, 역할, 가입일, 동아리번호

FD 연락처 → 학번, 이름, 학과, 역할, 가입일, 동아리번호

2NF 만족

4. 동아리 테이블

key {동아리 번호}, candidate key {동아리 이름, 지도교수 교번, 대표자 학번, 페이지 url}

FD 동아리 번호 → 동아리 이름, 동아리 상태, 활동 분야, 부원 수, ... 모든 속성

FD 동아리 이름 → 동아리 번호, 동아리 상태, 활동 분야, 부원 수, ... 모든 속성

FD 대표자 학번 → 동아리 이름, 동아리 번호, 활동 분야, 부원 수, ... 모든 속성

FD 지도교수 학번 → 동아리 이름, 동아리 번호, 활동 분야, 부원 수, ... 모든 속성

FD 페이지 url → 동아리 이름, 동아리 번호, 동아리 상태, 활동 분야, ... 모든 속성

2NF 만족

5. 참여 테이블

key {프로젝트 이름, 학번}

FD X

6. 프로젝트 테이블

key {동아리 번호, 프로젝트 이름}

FD {동아리 번호, 프로젝트 이름} → 모든 속성

2NF 만족

7. 주요 활동

key {동아리번호, 활동 이름}

FD {동아리 번호, 활동 이름} → 모든 속성

2NF 만족

3. 3NF

3NF는 FD $X \rightarrow A$ 에 대해

X가 superkey

A가 prime attribute

두 조건중 하나를 만족해야 함.

1. 예산 테이블

3NF 만족

2. 지도교수 테이블

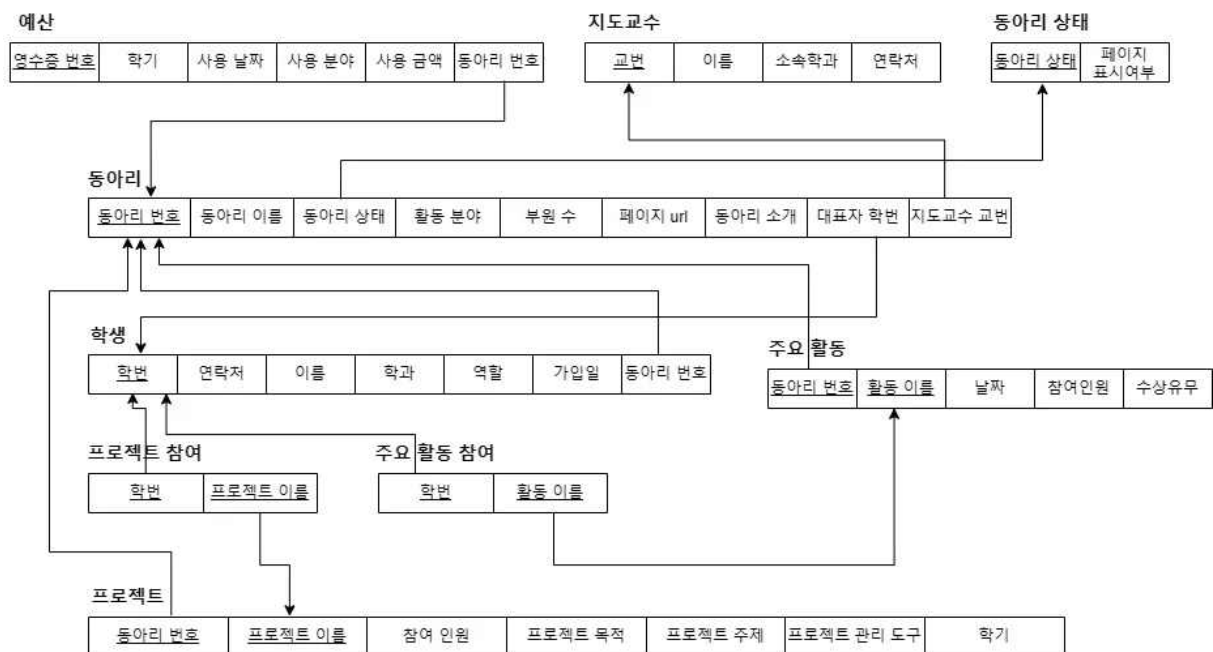
3NF 만족

3. 동아리 테이블

FD 동아리 상태 → 페이지 표시 여부에 대해 동아리 상태는 superkey가 아니고 페이지 표시 여부도 prime attribute가 아니다. 따라서 3NF의 대상이고 쪼개야 한다.

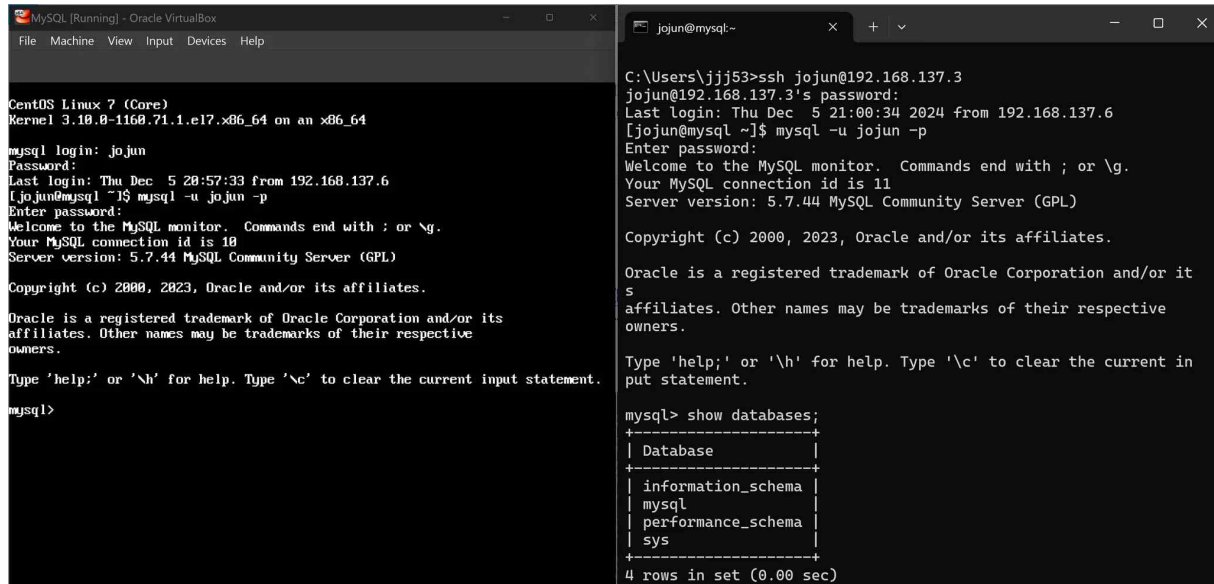
4. 다른 모든 테이블도 마찬가지로 3NF를 만족한다.

위와 같은 과정을 거쳐 정규화한 결과는 아래와 같다.



4. MySQL DBMS Database 생성

기본 환경은 가상머신에 CentOS + MySQL 설치이므로 기존 실습 환경에서 사용하던 이미 설치되어있는 환경을 사용함.



```
CentOS Linux 7 (Core)
Kernel 3.10.0-1160.71.1.el7.x86_64 on an x86_64

mysql login: jojun
Password:
Last login: Thu Dec 5 20:57:33 from 192.168.137.6
[jojun@mysql ~]$ mysql -u jojun -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 10
Server version: 5.7.44 MySQL Community Server (GPL)

Copyright (c) 2000, 2023, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql>

C:\Users\jjj53>ssh jojun@192.168.137.3
jojun@192.168.137.3's password:
Last login: Thu Dec 5 21:00:34 2024 from 192.168.137.6
[jojun@mysql ~]$ mysql -u jojun -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 11
Server version: 5.7.44 MySQL Community Server (GPL)

Copyright (c) 2000, 2023, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> show databases;
+-----+
| Database |
+-----+
| information_schema |
| mysql |
| performance_schema |
| sys |
+-----+
4 rows in set (0.00 sec)
```

5. 프로그램을 통해 데이터베이스에 접속하여 사용자가 요청 수행

1. 데이터베이스 생성

```
C:\Users\jjj53>ssh 192.168.137.3
C:\Users\jjj53>ssh jojun@192.168.137.3
jojun@192.168.137.3's password:
Last login: Thu Dec 5 21:01:03 2024 from 192.168.137.6
[jojun@mysql ~]$ mysql -u jojun -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 2
Server version: 5.7.44 MySQL Community Server (GPL)

Copyright (c) 2000, 2023, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql>
```

```
mysql> CREATE DATABASE club_management;
Query OK, 1 row affected (0.00 sec)

mysql> SHOW DATABASES;
+-----+
| Database |
+-----+
| information_schema |
| club_management |
| mysql |
| performance_schema |
| sys |
+-----+
5 rows in set (0.00 sec)
```

2. 데이터베이스 연결

application.properties 에서 연결 정보 설정 후 DatabaseConnection.java에서 연결
(아래는 application.properties 코드입니다. github에 올라가있지 않아서 따로 명시하였습니다.)

url=jdbc:mysql://192.168.137.3:4567/club_management?useSSL=false

username=jojun

password=whwnsgkh9326@

3. 기본 프롬프트(사용자 화면) 설정

4. 테이블 생성

1. 예산 테이블

- PK: 영수증번호
- FK: 동아리번호
- 일반속성: 사용날짜, 사용분야, 사용금액, 학기
- ON UPDATE : CASCADE
- ON DELETE : CASCADE

2. 지도교수 테이블

- PK: 교번
- 일반속성: 이름, 소속학과, 연락처(unique)

3. 동아리상태 테이블

- PK: 동아리상태
- 일반속성: 페이지표시여부

4. 동아리 테이블

- PK: 동아리번호
- FK: 대표자학번, 지도교수교번, 동아리상태
- 일반속성: 동아리이름(unique), 활동분야, 부원수, 페이지url(unique), 동아리소개
- 대표자 학번 ON DELETE : RESTRICT (대표자는 탈퇴할 수 없음 대표자 변경 후 탈퇴해야함)
- ON UPDATE : CASCADE
- 지도교수 교번 : ON DELETE : RESTRICT
- ON UPDATE : CASCADE
- 동아리 상태 : ON DELETE : RESTRICT
- ON UPDATE : CASCADE

5. 학생 테이블

- PK: 학번
- FK: 동아리번호
- 일반속성: 연락처(unique), 이름, 학과, 역할, 가입일 (모두 학번에 종속)
- ON DELETE : SET NULL (동아리가 삭제되더라도 학생 정보는 삭제되면 안됨)
- ON UPDATE : CASCADE

6. 프로젝트 테이블

- PK: (동아리번호, 프로젝트이름)
- FK : 동아리번호
- 일반속성: 참여인원(int), 프로젝트목적, 프로젝트주제, 프로젝트관리도구, 학기
- ON DELETE: CASCADE
- ON UPDATE: CASCADE

7. 프로젝트참여 테이블

- PK: (학번, 동아리번호, 프로젝트이름)
- FK: 학번, (동아리번호, 프로젝트이름)
- ON DELETE: CASCADE
- ON UPDATE: CASCADE

8. 주요활동 테이블

- PK: (동아리번호, 활동이름)
- FK: 동아리번호
- 일반속성: 날짜, 참여인원(int), 수상유무
- ON DELETE: CASCADE
- ON UPDATE: CASCADE

9. 주요활동참여 테이블

- PK: (학번, 동아리번호, 활동이름) - 복합키
- FK: 학번, (동아리번호, 활동이름)
- ON DELETE: CASCADE
- ON UPDATE: CASCADE

위의 방식대로 테이블을 생성하였고 SQL 쿼리를 직접 날려 생성하였다. 아래는 테이블 생성에 사용한 SQL문이다.

```
CREATE DATABASE IF NOT EXISTS club_management;  
USE club_management;
```

-- 지도교수 테이블

```
CREATE TABLE professors  
(  
    prof_id    VARCHAR(20) PRIMARY KEY,  
    name       VARCHAR(50)      NOT NULL,  
    department VARCHAR(50)      NOT NULL,  
    contact    VARCHAR(20) UNIQUE NOT NULL  
);
```

-- 동아리상태 테이블

```
CREATE TABLE club_status  
(  
    status_name    VARCHAR(20) PRIMARY KEY,  
    is_page_visible BOOLEAN NOT NULL  
);
```

-- 학생 테이블 (동아리 FK는 나중에 추가)

```
CREATE TABLE students
(
    student_id VARCHAR(20) PRIMARY KEY,
    contact    VARCHAR(20) UNIQUE NOT NULL,
    name       VARCHAR(50)        NOT NULL,
    department VARCHAR(50)        NOT NULL,
    role       VARCHAR(50),
    join_date  DATE,
    club_id    VARCHAR(20)
);
```

-- 동아리 테이블

```
CREATE TABLE clubs
(
    club_id          VARCHAR(20) PRIMARY KEY,
    president_id     VARCHAR(20)        NOT NULL,
    prof_id          VARCHAR(20)        NOT NULL,
    status_name      VARCHAR(20)        NOT NULL,
    club_name        VARCHAR(100) UNIQUE NOT NULL,
    activity_field   VARCHAR(50)        NOT NULL,
    member_count     INT                NOT NULL,
    page_url         VARCHAR(200) UNIQUE,
    club_info        TEXT,
    FOREIGN KEY (president_id) REFERENCES students (student_id)
        ON DELETE RESTRICT
        ON UPDATE CASCADE,
    FOREIGN KEY (prof_id) REFERENCES professors (prof_id)
        ON DELETE RESTRICT
        ON UPDATE CASCADE,
    FOREIGN KEY (status_name) REFERENCES club_status (status_name)
        ON DELETE RESTRICT
        ON UPDATE CASCADE
);
```

-- 학생 테이블 생성 (동아리 FK는 나중에 추가)

```
CREATE TABLE students
(
    id          VARCHAR(20) PRIMARY KEY,
    name        VARCHAR(50) NOT NULL,
    dept        VARCHAR(50) NOT NULL,
    contact     VARCHAR(20) NOT NULL,
    role        VARCHAR(50) NOT NULL,
    join_date   DATE          NOT NULL
);
```

-- 학생 테이블에 FK 추가

```
ALTER TABLE students
    ADD FOREIGN KEY (club_id) REFERENCES clubs (club_id)
        ON DELETE SET NULL
        ON UPDATE CASCADE;
```

-- 예산 테이블

```
CREATE TABLE budgets
(
    receipt_no  VARCHAR(20) PRIMARY KEY,
    club_id     VARCHAR(20)   NOT NULL,
    use_date    DATE          NOT NULL,
    use_field   VARCHAR(50)   NOT NULL,
    amount      DECIMAL(10, 2) NOT NULL,
    semester    VARCHAR(20)   NOT NULL,
    FOREIGN KEY (club_id) REFERENCES clubs (club_id)
        ON DELETE CASCADE
        ON UPDATE CASCADE
);
```

-- 프로젝트 테이블

```
CREATE TABLE projects
(
    club_id      VARCHAR(20),
    project_name  VARCHAR(100),
    member_count  INT          NOT NULL,
```

```

project_purpose TEXT          NOT NULL,
project_topic  TEXT          NOT NULL,
management_tool VARCHAR(50) NOT NULL,
semester      VARCHAR(20) NOT NULL,
PRIMARY KEY (club_id, project_name),
FOREIGN KEY (club_id) REFERENCES clubs (club_id)
    ON DELETE CASCADE
    ON UPDATE CASCADE
);

```

-- 프로젝트참여 테이블

```

CREATE TABLE project_participants
(
    student_id  VARCHAR(20),
    club_id     VARCHAR(20),
    project_name VARCHAR(100),
    PRIMARY KEY (student_id, club_id, project_name),
    FOREIGN KEY (student_id) REFERENCES students (student_id)
        ON DELETE CASCADE
        ON UPDATE CASCADE,
    FOREIGN KEY (club_id, project_name) REFERENCES projects (club_id, project_name)
        ON DELETE CASCADE
        ON UPDATE CASCADE
);

```

-- 주요활동 테이블

```

CREATE TABLE activities
(
    club_id      VARCHAR(20),
    activity_name VARCHAR(100),
    activity_date DATE    NOT NULL,
    member_count INT      NOT NULL,
    has_award    BOOLEAN NOT NULL,
    PRIMARY KEY (club_id, activity_name),
    FOREIGN KEY (club_id) REFERENCES clubs (club_id)
        ON DELETE CASCADE
        ON UPDATE CASCADE
);

```

```
);
```

```
-- 주요활동참여 테이블
```

```
CREATE TABLE activity_participants
```

```
(
```

```
    student_id    VARCHAR(20),
```

```
    club_id       VARCHAR(20),
```

```
    activity_name VARCHAR(100),
```

```
    PRIMARY KEY (student_id, club_id, activity_name),
```

```
    FOREIGN KEY (student_id) REFERENCES students (student_id)
```

```
        ON DELETE CASCADE
```

```
        ON UPDATE CASCADE,
```

```
    FOREIGN KEY (club_id, activity_name) REFERENCES main_activities (club_id,  
activity_name)
```

```
        ON DELETE CASCADE
```

```
        ON UPDATE CASCADE
```

```
);
```

```
mysql> show tables;  
+-----+  
| Tables_in_club_management |  
+-----+  
| activities                 |  
| activity_participants     |  
| budgets                   |  
| club_status               |  
| clubs                     |  
| professors                |  
| project_participants      |  
| projects                  |  
| students                  |  
+-----+  
9 rows in set (0.00 sec)
```

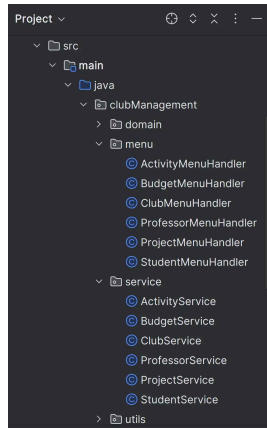
6. 사용자 요청 정의 및 구현

사용자는 조교로 상정하였고, 각 테이블별로 요청 정의와 동시에 구현을 하였다.

1. 데이터베이스 연결

```
2020039071 조준화 Database System Term Project - Club Management System  
  
-----  
1. Database Connection  2. Manage Students  
3. Manage Clubs        4. Manage Professors  
5. Manage Budgets      6. Manage Projects  
7. Manage Activities    8. Quit  
-----  
Select menu: 1  
Database connection successful!
```

2. domain 패키지 하위에 테이블 정보를 저장할 객체 생성 및 패키지 구조 정의
enum, local date 정도만 복잡하고 나머지는 간단하게 정보를 저장하는 객체
service 패키지에서 데이터베이스와의 작업을 수행하고 menu 패키지에서 service 모듈 사용 및
사용자 메뉴 정의



3. 학생 기능 구현

1. 학생 등록

- 등록 시 clubID, role, joinDate 모두 null로 설정

2. 학생 전체 조회, 특정 학생 조회

3. 학생 삭제 - FK 예외 고려하여 구현

4. 학생 동아리 등록

- 학생 유효성 검사(존재하는지, 소속 동아리 없는지)
- 동아리 유효성 검사(존재하는지, 활동중인지)
- 학생 정보 업데이트
- 동아리 회원 수 증가

5. 학생 정보 수정

- 학번, 소속 동아리, 가입일, 역할은 수정할 수 없음

6. 학생 동아리 정보 수정

- 가입된 동아리가 있어야만 변경할 수 있음
- 역할, 가입일, 소속 동아리 변경 가능

7. 학생 동아리 탈퇴 처리

```
7. Manage Activities      8. Quit
-----
Select menu: 2

===== 학생 관리 =====
1. 학생 등록
2. 전체 학생 조회
3. 특정 학생 조회
4. 학생 기본 정보 수정
5. 동아리 가입 처리
6. 학생 동아리 정보 수정
7. 학생 삭제
8. 동아리 탈퇴 처리
9. 뒤로 가기
=====
메뉴 선택: 2

===== 전체 학생 목록 =====
학번: 1
이름: test1
학과: test
연락처: 1
역할: president
가입일: 2024-12-09
소속 동아리: 2
-----
```

```
=====
메뉴 선택: 1

===== 학생 등록 =====
학번: 2020039071
이름: 조준화
학과: 소프트웨어
연락처: 010
학생이 성공적으로 등록되었습니다.
```

```
===== 동아리 가입 처리 =====
학번: 2020039071
가입할 동아리 ID: 1
error : 현재 활동 중인 동아리가 아닙니다.
```

```
===== 동아리 가입 처리 =====
학번: 2020039071
가입할 동아리 ID: 2
동아리 가입이 완료되었습니다.
```

```
조회할 학생의 학번을 입력하세요: 2020039071

===== 학생 정보 =====
학번: 2020039071
이름: 조준화
학과: 소프트웨어
연락처: 010
역할: null
가입일: null
소속 동아리: null
```

```
===== 동아리 탈퇴 처리 =====
학번: 2020039071

현재 동아리 정보:
소속 동아리: 2
역할: member
가입일: 2024-12-10

정말로 탈퇴하시겠습니까? (y/n): y
동아리 탈퇴가 완료되었습니다.
```

```
메뉴 선택: 7

삭제할 학생의 학번을 입력하세요: 2020039071
정말로 삭제하시겠습니까? (y/n): y
학생이 성공적으로 삭제되었습니다.
```

```
메뉴 선택: 3

조회할 학생의 학번을 입력하세요: 2020039071
해당 학번의 학생을 찾을 수 없습니다.
```

4. 지도교수

- 지도교수 등록
- 지도교수 정보 수정 - 교번은 안됨
- 지도교수 삭제

```
Select menu: 4

===== 지도교수 관리 =====
1. 지도교수 등록
2. 전체 지도교수 조회
3. 특정 지도교수 조회
4. 지도교수 정보 수정
5. 지도교수 삭제
6. 뒤로 가기
=====
메뉴 선택: 2

===== 전체 지도교수 목록 =====
교번: 1
이름: test
학과: test
연락처: 1
-----
```

5. Club Status

- Club Status 테이블은 관리용이므로 따로 조작할 수 없음
- 쿼리

```
INSERT INTO club_status (status_name, is_page_visible) VALUES  
( 'pending', false),  
( 'active', true),  
( 'closed', false);
```

6. 동아리

- 동아리 등록(회장, 지도교수 필수)
 - 회장, 지도교수 필수
 - page_url, club_info는 필수가 아님
 - 최소 10명의 부원 필요
 - 동아리 초기 상태는 pending
- page_url, club_info 추가
- 동아리 정보 변경
- 동아리 삭제 - closed로 바꾸고 회원들을 동아리 탈퇴 처리함
- 동아리 조회
- 특정 동아리 조회
- 동아리 상태 변경 - closed 동아리는 다른 상태로 변경 불가능
- 특정 동아리 명단 조회

```
===== 동아리 관리 =====  
1. 동아리 등록  
2. 전체 동아리 조회  
3. 특정 동아리 조회  
4. 동아리 정보 수정  
5. 동아리 상태 변경  
6. 동아리 삭제  
7. 상태별 동아리 조회  
8. 동아리 회원 조회  
9. 뒤로 가기  
=====   
메뉴 선택:
```

```
메뉴 선택: 2  
  
===== 전체 동아리 목록 =====  
동아리 ID: 1  
동아리 이름: 1  
회장 학번: 1  
지도교수 ID: 1  
동아리 상태: closed  
페이지 공개 여부: 비공개  
활동 분야: 1  
회원 수: 0  
페이지 URL: 1  
동아리 소개: 1  
  
-----  
동아리 ID: 2  
동아리 이름: test  
회장 학번: 1  
지도교수 ID: 1  
동아리 상태: active  
페이지 공개 여부: 공개  
활동 분야: 1  
회원 수: 10  
페이지 URL: 없음  
동아리 소개: 없음  
  
-----
```


7. 예산

- 예산 추가, 수정, 삭제 모든 필드가 not null
- 동아리별 예산 조회
- 동아리별 학기별 예산 조회

```
===== 예산 관리 메뉴 =====
1. 예산 등록
2. 예산 수정
3. 예산 삭제
4. 특정 동아리의 학기별 예산 조회
5. 특정 동아리의 전체 예산 조회
6. 뒤로 가기
=====
메뉴를 선택하세요: 1
동아리 ID: 2
영수증 번호: 33
사용 분야: test
금액: 10000
예산이 성공적으로 등록되었습니다.
```

```
===== 예산 관리 메뉴 =====
1. 예산 등록
2. 예산 수정
3. 예산 삭제
4. 특정 동아리의 학기별 예산 조회
5. 특정 동아리의 전체 예산 조회
6. 뒤로 가기
=====
메뉴를 선택하세요: 5
동아리 ID: 2

=== 전체 예산 내역 ===

2024-2 학기:
영수증 번호: 33, 사용 분야: test, 금액: 10000.00원, 사용일: 2024-12-10

2024-2 학기 총액: 10000.00원

총 지출액: 10000.00원
```

8. 프로젝트

- 프로젝트 추가, 수정, 삭제
- 프로젝트에 참여하는 학생 추가, 삭제
- 동아리별 프로젝트 조회
- 프로젝트 전체 조회
- 프로젝트 참여자 조회

```
-----
Select menu: 6

===== 프로젝트 관리 메뉴 =====
1. 프로젝트 등록
2. 프로젝트 수정
3. 프로젝트 삭제
4. 전체 프로젝트 조회
5. 프로젝트 참여자 추가
6. 프로젝트 참여자 삭제
7. 프로젝트 참여자 조회
8. 동아리별 프로젝트 조회
9. 이전 메뉴로
=====
메뉴를 선택하세요: 4

=== 전체 프로젝트 목록 ===

2024-2 학기:

-----
동아리: 2
프로젝트명: testp
목적: t
주제:
사용 도구: t
```

9. 주요활동

- 주요활동 추가, 수정, 삭제
- 주요활동에 참여하는 학생 추가, 삭제
- 동아리별 주요활동 조회
- 주요활동 전체 조회
- 주요활동 참여자 조회
- 특정 주요활동 수상 표시

Select menu: 7

===== 주요활동 관리 메뉴 =====

1. 주요활동 등록
2. 주요활동 수정
3. 주요활동 삭제
4. 전체 활동 조회
5. 동아리별 활동 조회
6. 활동 수상 여부 변경
7. 활동 참여자 추가
8. 활동 참여자 삭제
9. 활동 참여자 조회
0. 이전 메뉴로

=====

메뉴를 선택하세요: 4

=== 전체 활동 목록 ===

동아리: 2

활동명: t

활동일: 2024-12-09

참여 인원: 1명

수상 여부: 0
