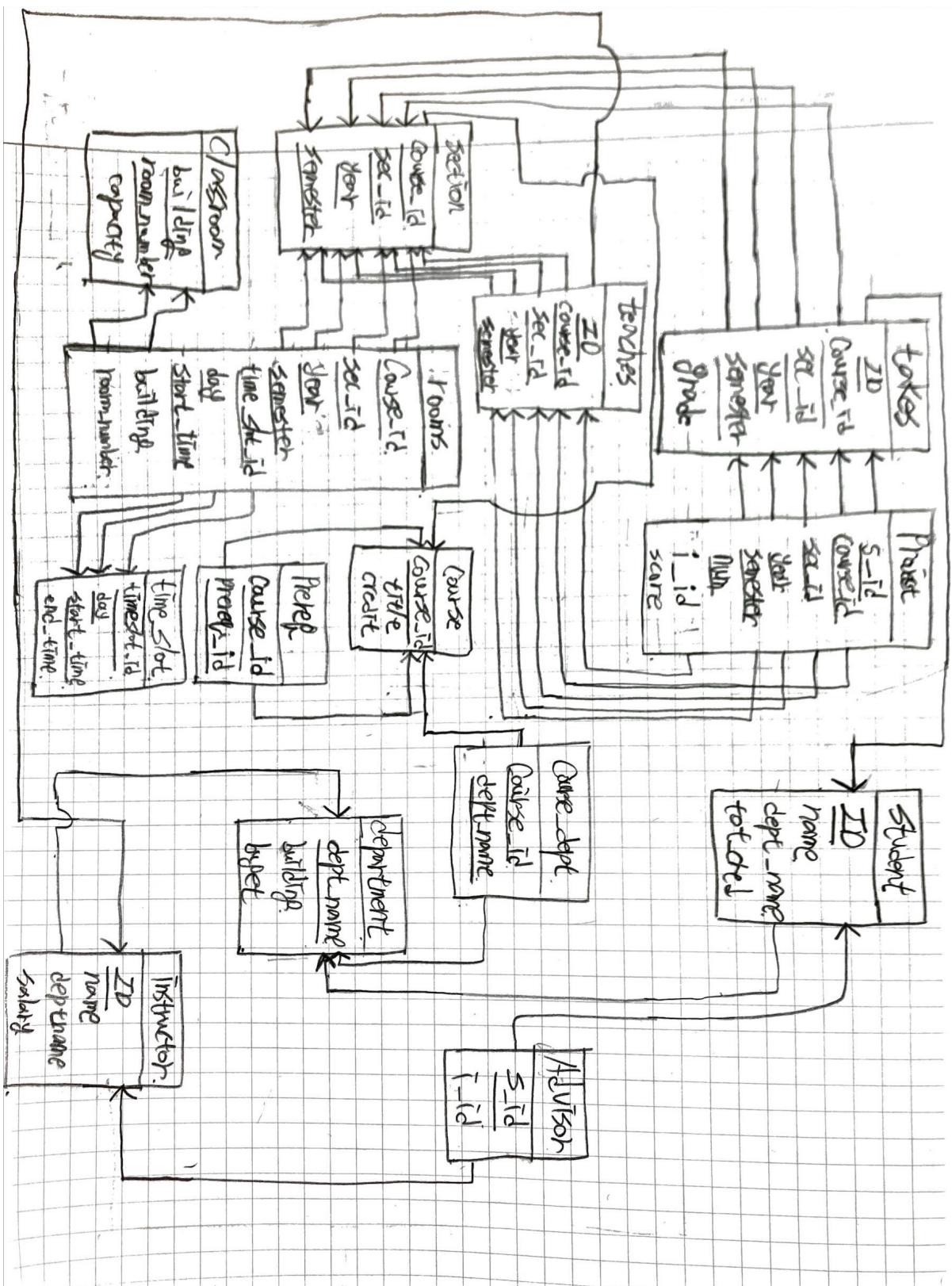




Database weak 13-14 assignment¹

20203361 장민석

1. 12주차 과제의 최종 결과로 제출한 RDB 스키마 다이어그램



2. create table문

다음과 같이 총 10개의 사용할 테이블을 생성하였다.

```
DROP TABLE IF EXISTS project, takes, student, instructor, course_dept, section, rooms,
course, time_slot;

-- Course table
CREATE TABLE course (
    course_id VARCHAR(8),
    title VARCHAR(50),
    credits NUMERIC(2,0),
    PRIMARY KEY (course_id)
);

-- Section table
CREATE TABLE section (
    course_id VARCHAR(8),
    sec_id VARCHAR(8),
    semester VARCHAR(6) CHECK (semester IN ('Fall', 'Winter', 'Spring', 'Summer')),
    year NUMERIC(4,0),
    PRIMARY KEY (course_id, sec_id, semester, year),
    FOREIGN KEY (course_id) REFERENCES course(course_id)
        ON DELETE CASCADE
);

-- Course_dept table
CREATE TABLE course_dept (
    course_id VARCHAR(8),
    dept_name VARCHAR(50),
    PRIMARY KEY (course_id, dept_name),
    FOREIGN KEY (course_id) REFERENCES course(course_id)
);

-- Time_slot table
CREATE TABLE time_slot (
    time_slot_id VARCHAR(4),
    day VARCHAR(3),
    start_time NUMERIC(2) CHECK (start_time >= 0 AND start_time < 24),
    end_time NUMERIC(2) CHECK (end_time >= 0 AND end_time < 24),
    PRIMARY KEY (time_slot_id, day, start_time)
);

-- Rooms table
CREATE TABLE rooms (
    course_id VARCHAR(8),
    sec_id VARCHAR(8),
    year NUMERIC(4,0),
    semester VARCHAR(6),
    time_slot_id VARCHAR(4),
    day VARCHAR(3),
    start_time NUMERIC(2),
    building VARCHAR(15),
    room_number VARCHAR(7),
```

```

    FOREIGN KEY (course_id, sec_id, semester, year) REFERENCES section(course_id, sec_id,
semester, year),
    FOREIGN KEY (time_slot_id, day, start_time) REFERENCES time_slot(time_slot_id, day,
start_time)
);

-- Student table
CREATE TABLE student (
    ID VARCHAR(5),
    name VARCHAR(20) NOT NULL,
    dept_name VARCHAR(50),
    tot_cred NUMERIC(3,0),
    PRIMARY KEY (ID)
);

-- Instructor table
CREATE TABLE instructor (
    ID VARCHAR(5),
    name VARCHAR(20),
    dept_name VARCHAR(50),
    salary NUMERIC(8,2),
    PRIMARY KEY (ID)
);

-- Takes table
CREATE TABLE takes (
    ID VARCHAR(5),
    course_id VARCHAR(8),
    sec_id VARCHAR(8),
    semester VARCHAR(6),
    year NUMERIC(4,0),
    grade VARCHAR(2),
    PRIMARY KEY (ID, course_id, sec_id, semester, year),
    FOREIGN KEY (course_id, sec_id, semester, year) REFERENCES section(course_id, sec_id,
semester, year),
    FOREIGN KEY (ID) REFERENCES student(ID)
);

-- Project table
CREATE TABLE project (
    s_id VARCHAR(5),
    course_id VARCHAR(8),
    sec_id VARCHAR(8),
    semester VARCHAR(6),
    year NUMERIC(4,0),
    num NUMERIC(2,0),
    name VARCHAR(50),
    i_id VARCHAR(5),
    max_score NUMERIC(3,0),
    score NUMERIC(3,0),
    PRIMARY KEY (s_id, course_id, sec_id, semester, year, num),
    FOREIGN KEY (s_id, course_id, sec_id, semester, year) REFERENCES takes(ID, course_id,
sec_id, semester, year)
);

```

3. SQL문

첫 번째 기능을 구현하기 위해 사용되는 SQL 구문은 다음과 같이 2개이다. 먼저 query1은 'course' 테이블에서 course_id, title, credits를 조회한다. 하지만 모든 강의를 조회하는 것이 아닌, 'section' 테이블과 'course_dept' 테이블에 있는 강의 중에서 주어진 year와 semester에 해당하는 강의들 중, 두 개 이상의 학과에서 개설한 과목, 즉 융합과목 만을 조회한다. 이를 위해 group by 구문과 having 구문을 사용했다. query2는 'course_dept' 테이블에서 query1에서 결과로 조회된 course_id에 해당하는 튜플을 검색한다. 이를 통해 융합과목의 dept_name 리스트를 모두 찾을 수 있다.

```
String query1 = """
    SELECT course_id, title, credits
    FROM course
    WHERE course_id IN (
        SELECT S.course_id
        FROM section as S, course_dept as CD
        WHERE (S.year, S.semester) = (?, ?)
        AND S.course_id = CD.course_id
        GROUP BY S.course_id
        HAVING COUNT(*) >= 2
    )""";
String query2 = """
    SELECT dept_name
    FROM course_dept
    WHERE course_id = ?
""";
```

두 번째 기능을 구현하기 위해 사용되는 SQL 구문은 다음과 같이 2개이다. query1은 'section' 테이블과 'course' 테이블을 조인하여, 특정 연도와 학기에 개설되는 모든 section을 찾고 이의 course_id, title, sec_id를 조회한다. query2는 위에서 찾은 section의 정보를 바탕으로 'rooms' 테이블과 'time_slot' 테이블을 조인하여, 해당 section의 요일 별 강의실의 위치 정보인 building, room_number와 수업 시간 정보인 day, start_time, end_time 정보를 찾는다.

```
String query1 = """
    SELECT S.course_id, C.title, S.sec_id
    FROM section as S, course as C
    WHERE (S.year, S.semester) = (?, ?)
    AND S.course_id = C.course_id
""";
String query2 = """
    SELECT T.day, T.start_time, T.end_time, R.building, R.room_number
    FROM rooms as R, time_slot as T
    WHERE (R.time_slot_id, R.day, R.start_time) = (T.time_slot_id, T.day,
T.start_time)
    AND (R.course_id, R.sec_id, R.year, R.semester) = (?, ?, ?, ?)
""";
```

세 번째 기능을 구현하기 위해서 총 4개의 쿼리를 실행한다. query1은 'student' 테이블에서 특정 학생 ID에 해당하는 학생의 ID, name, dept_name을 조회한다. query2는 'takes' 테이블과 'course' 테이블을 조인하여, 특정 학생이 특정 year와 semester에 수강한 section의 course_id, title, sec_id, 그리고 grade를 조회한다. query3은 query2의 결과를 바탕으로 'project' 테이블과 'instructor' 테이블을 조인하여, 특정 학생(s_id)이 들은 section(course_id, sec_id, year, semester)에서 수행한 프로젝트의 정보인 num, name, max_score, I.name, 그리고 score를 조회한다. query4는 'project' 테이블에서 집계함수를 통해 query2의 결과인 각 section에 대한 프로젝트의 총 개수(count)와 평균 점수(avg)를 계산한 결과를 검색한다.

```
String query1 = """
    SELECT ID, name, dept_name
    FROM student
    WHERE ID = ?
    """;

String query2 = """
    SELECT T.course_id, C.title, T.sec_id, T.grade
    FROM takes as T, course as C
    WHERE T.course_id = C.course_id
    AND (T.ID, T.year, T.semester) = (?, ?, ?)
    """;

String query3 = """
    SELECT P.num, P.name, P.max_score, I.name, P.score
    FROM project as P, instructor as I
    WHERE P.i_id = I.id
    AND (P.s_id, P.course_id, P.sec_id, P.year, P.semester) = (?, ?, ?, ?, ?, ?)
    """;

String query4 = """
    SELECT count(*) as count, avg(score) as avg
    FROM project
    WHERE (s_id, course_id, sec_id, year, semester) = (?, ?, ?, ?, ?, ?)
    """;
```

4. 프로그램의 로직

다음과 같다.

5. DB 데이터

다음과 같다.

6. Java 프로그램 실행 결과

다음과 같다.

7. 정확성 검증

다음과 같다.

8. Java 프로그램 소스코드

다음과 같다.

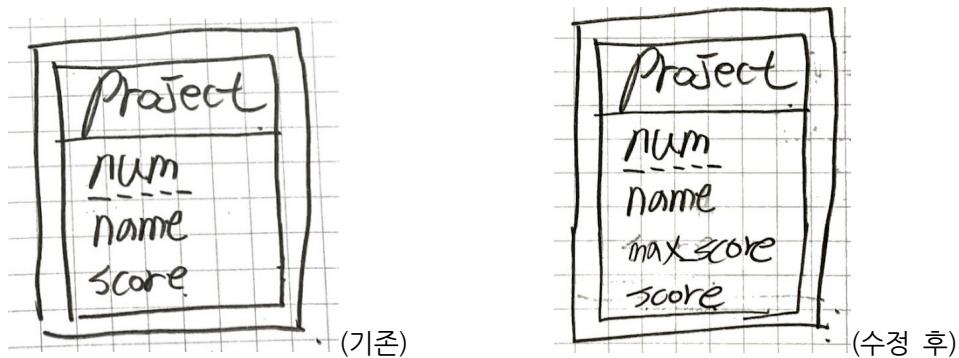
9. 기타

없음.

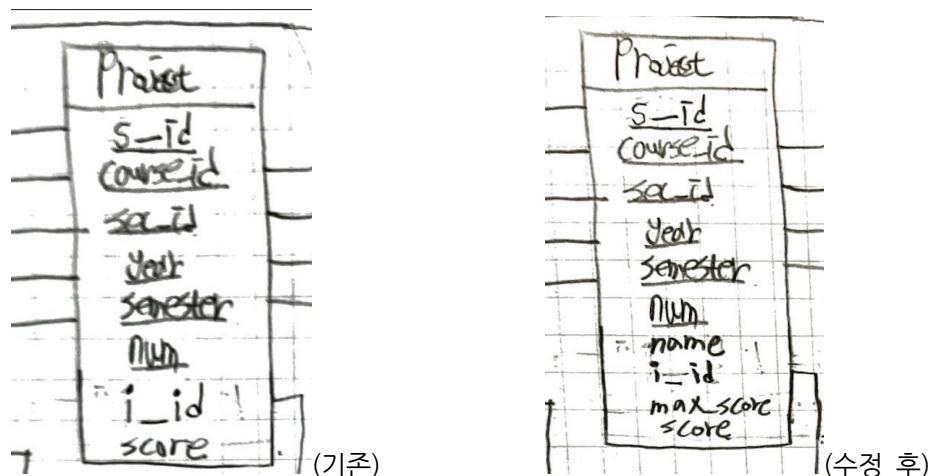
10. 설계 수정

10.1. 기능 구현 불가능의 이유: ERD의 project entity set을 테이블 스키마로 변환하는 과정에서, name과 max_score 속성을 옮기는 것이 누락되었다. 따라서, “3번 프로젝트 수행결과 조회” 과제를 수행함에 있어 해당 컬럼을 사용할 수 없게 되었고, 결과적으로 과제가 요구하는 프로젝트 이름과 배점을 출력하는 부분의 기능을 구현하는 것이 불가능하다.

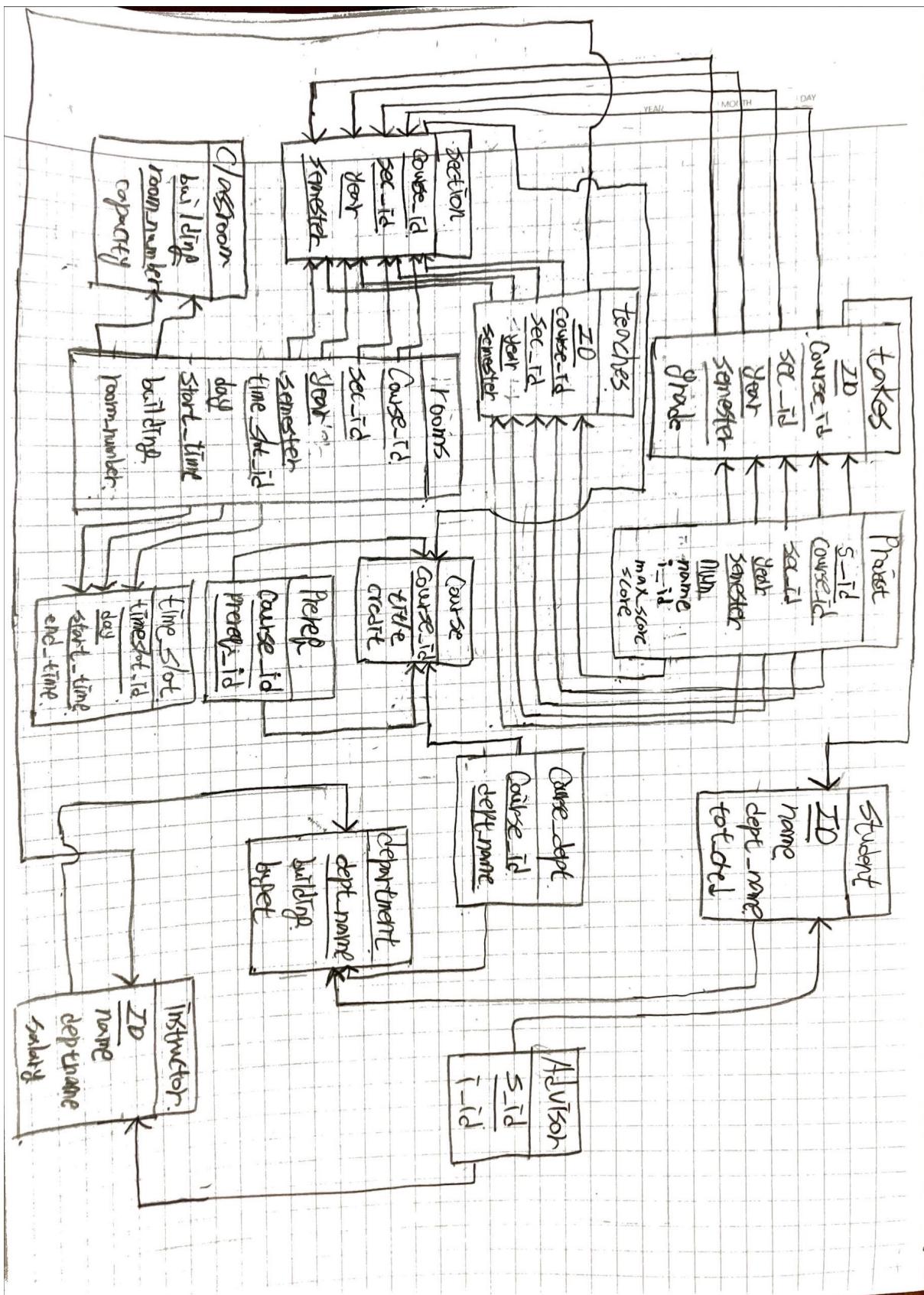
10.2. ER modeling의 수정: project entity set에 max_score(배점) 속성을 추가하였다. 기존 project entity set에서 해당 속성이 누락되어 최종 테이블 스키마에도 누락되었기에 다음과 같이 수정하였다.



10.3. 변환규칙에 의한 테이블 스키마 변환의 수정: weak entity set이었던 project를 새롭게 추가된 max_score 속성과 기존 변환에서 누락된 name 속성까지 모두 포함해서 변환한다.



10.4. 수정된 RDB 스키마 다이어그램: 다음 장의 이미지와 같다.



최종 스키마: 최종적으로 완성된 스키마는 다음과 같다.

Department (dept_name, building, budget)

Course (course_id, title, credits)

Classroom (building, room_number, capacity)

Time_slot (time_slot_id, day, start_time, end_time)

Instructor (ID, name, salary, dept_name)

foreign key (dept_name) references Department (dept_name)

Student (ID, name, tot_cred, dept_name)

foreign key (dept_name) references Department (dept_name)

Teaches (ID, course_id, sec_id, year, semester)

foreign key (ID) references Instructor (ID)

foreign key (course_id, sec_id, year, semester) references Section (course_id, sec_id, year, semester)

Takes (ID, course_id, sec_id, year, semester, grade)

foreign key (ID) references Student (ID)

foreign key (course_id, sec_id, year, semester) references Section (course_id, sec_id, year, semester)

Project (s_id, course_id, sec_id, year, semester, num, name, i_id, max_score, score)

foreign key (s_id, course_id, sec_id, year, semester) references Takes (ID, course_id, sec_id, year, semester)

foreign key (i_id, course_id, sec_id, year, semester) references Teaches (ID, course_id, sec_id, year, semester)

Section (course_id, sec_id, year, semester)

foreign key (course_id) references Course (course_id)

Course_dept (course_id, dept_name)

foreign key (course_id) references Course (course_id)

foreign key (dept_name) references Department (dept_name)

Prereq (course_id, prereq_id)

foreign key (course_id) references Course (course_id)

foreign key (Prereq_id) references Course (course_id)

Advisor (s_id, i_id)

foreign key (s_id) references Student (ID)

foreign key (i_id) references Instructor (ID)

Rooms (course_id, sec_id, year, semester, time_slot_id, day, start_time, building, room_number)

foreign key (building, room_number) references Classroom (building, room_number)