

▼ Homework - SQL1


1. Use Google Colabortory and do your homework.
2. (In Google colaboratory) Before you submit your homework, restart kernel and run every cell!
3. Save (File->Save) the file
4. Submit your homework (this file) in Google classroom
5. **Don't forget to click "제출" button** ("Submit", "완료로 표시", 또는 "제출" 버튼을 누르지 않으면 제출된 것이 아님)
6. No late homeworks will be accepted for any reason!

To edit this cell, double click here

```
이름 : 이준용
학번 : 201904458
학과 : 컴퓨터 전자 시스템 공학과
제출일 : 20211114
```

- SQL 작성시, 문제에 주어지지 않은 상수를 사용하는 경우 (cheating), **마이너스 점수**를 받는다.
- **점수: 1번은 20점 나머지는 10점**

아래는 학생과 수업에 관련된 데이터베이스 테이블이다.

 image.png

```
%load_ext sql
```

```
!pip install pymysql
```

```
Requirement already satisfied: pymysql in /usr/local/lib/python3.7/dist-packages (1.0.2)
```

▼ Use your host, id, and password

```
import getpass
user = 's201904458'
password = getpass.getpass()
host='dm.hufs.ac.kr'
database = 's201904458db'
connection_string = f'mysql+pymysql://{user}:{password}@{host}:3306/{database}'
```

```
%sql $connection_string
```

```
.....
'Connected: s201904458@s201904458db'
```

▼ Use your s학번db database

```
%sql use s201904458db;
```

```
* mysql+pymysql://s201904458:***@dm.hufs.ac.kr:3306/s201904458db
0 rows affected.
[]
```

1. 관계형 데이터베이스 테이블 스키마(create table)을 생성하고, MySQL DBMS를 이용하여 테이블을 생성하시오.

- 테이블은 적절한 컬럼 데이터 타입, Primary Key, Foreign Key (on delete, on update rule 포함, restrict, set null, cascade), 필요하다면 Unique, NOT NULL 등이 표시되어야 한다.
- 조건: 선수과목 정보는 course 정보 때문에 유지하는 정보이다. 즉, Course가 삭제되면, 불필요한 정보이다.
- 스키마 작성시 9번 문제가 성공할 수 있도록 스키마를 작성하여야 한다.
- Semester은 enum type을 사용하여, 봄, 여름, 가을, 겨울 순서가 되게 한다.
- Grade는 enum type을 사용하여 A, B, C, D, F 순서가 되게 한다.

```
%%sql
```

```
SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0;
SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS, FOREIGN_KEY_CHECKS=0;
```

```
DROP TABLE IF EXISTS STUDENT;
DROP TABLE IF EXISTS COURSE;
DROP TABLE IF EXISTS PREREQUISITE;
DROP TABLE IF EXISTS SECTION;
DROP TABLE IF EXISTS GRADE_REPORT;
```

```
CREATE TABLE STUDENT (Name VARCHAR(30) NOT NULL,
Student_number INTEGER NOT NULL,
Class CHAR NOT NULL,
Major CHAR(4),
PRIMARY KEY (Student_number));
```

```
CREATE TABLE COURSE ( Course_name VARCHAR(30) NOT NULL,
Course_number CHAR(8) NOT NULL,
Credit_hours INTEGER,
Department CHAR(4),
PRIMARY KEY (Course_number),
UNIQUE (Course_name));
```

```
CREATE TABLE PREREQUISITE ( Course_number CHAR(8) NOT NULL,
Prerequisite_number CHAR(8) NOT NULL,
PRIMARY KEY (Course_number, Prerequisite_number),
FOREIGN KEY (Course_number) REFERENCES
COURSE (Course_number) ON DELETE CASCADE ON UPDATE CASCADE,
FOREIGN KEY (Prerequisite_number) REFERENCES
COURSE (Course_number) ON DELETE CASCADE ON UPDATE CASCADE);
```

```
CREATE TABLE SECTION ( Section_identifier INTEGER NOT NULL,
Course_number CHAR(8) NOT NULL,
Semester ENUM('Spring', 'Summer', 'Fall', 'Winter'),
Year CHAR(4) NOT NULL,
Instructor VARCHAR(15),
PRIMARY KEY (Section_identifier),
FOREIGN KEY (Course_number) REFERENCES
COURSE (Course_number) ON DELETE CASCADE ON UPDATE CASCADE);
```

```
CREATE TABLE GRADE_REPORT ( Student_number INTEGER NOT NULL,
Section_identifier INTEGER NOT NULL,
Grade ENUM('A', 'B', 'C', 'D', 'F'),
```

[illegible]

```
* mysql+pymysql://s201904458:***@dm.hufs.ac.kr:3306/s201904458db
2 rows affected.
4 rows affected.
6 rows affected.
6 rows affected.
3 rows affected.
[]
```

▼ Run the following cell to show that insert is correctly done

```
%sql select * from STUDENT;
```

```
* mysql+pymysql://s201904458:***@dm.hufs.ac.kr:3306/s201904458db
2 rows affected.
```

Name Student_number Class Major

Brown	8	2	CS
Smith	17	1	CS

```
%sql select * from COURSE;
```

```
* mysql+pymysql://s201904458:***@dm.hufs.ac.kr:3306/s201904458db
4 rows affected.
```

Course_name	Course_number	Credit_hours	Department
Intro to Computer Science	CS1310	4	CS
Data Structures	CS3320	4	CS
Database	CS3380	3	CS
Discrete Mathematics	MATH2410	3	MATH

```
%sql select * from SECTION;
```

```
* mysql+pymysql://s201904458:***@dm.hufs.ac.kr:3306/s201904458db
6 rows affected.
```

Section_identifier Course_number Semester Year Instructor

85	MATH2410	Fall	07	King
92	CS1310	Fall	07	Anderson
102	CS3320	Spring	08	Knuth
112	MATH2410	Fall	08	Chang
119	CS1310	Fall	08	Anderson
135	CS3380	Fall	08	Stone

```
%sql select * from GRADE_REPORT;
```

```
* mysql+pymysql://s201904458:***@dm.hufs.ac.kr:3306/s201904458db
6 rows affected.
```

Student_number Section_identifier Grade

8	85	A
8	92	A
8	102	B
8	135	A
17	112	B
17	119	C

```
%sql select * from PREREQUISITE;
```

```
* mysql+pymysql://s201904458:***@dm.hufs.ac.kr:3306/s201904458db
3 rows affected.
```

Course_number Prerequisite_number

CS3320	CS1310
CS3380	CS3320
CS3380	MATH2410

▼ 3. Database 과목을 수강한 학생들의 이름을 나열하시오.

```
%%sql
```

```
SELECT S.Name
FROM STUDENT S, GRADE_REPORT G, SECTION SE, COURSE C
WHERE C.Course_name = 'Database' and S.Student_number = G.Student_number and
      G.Section_identifier=SE.Section_identifier and
      SE.Course_number= C.Course_number;
```

```
* mysql+pymysql://s201904458:***@dm.hufs.ac.kr:3306/s201904458db
1 rows affected.
```

Name

Brown

4. Grade_report 테이블에 <8, 112, NULL> 행을 삽입하시오. 동일한 과목(course)을 두 번 이상 수강한 학생의 이름을 찾으시오.

```
%sql insert into GRADE_REPORT values (8, 112, NULL);
```

```
* mysql+pymysql://s201904458:***@dm.hufs.ac.kr:3306/s201904458db
1 rows affected.
[]
```

```
%%sql
```

```
SELECT S.NAME
FROM STUDENT S, GRADE_REPORT G, SECTION SE
WHERE S.Student_number=G.Student_number AND G.Section_identifier= SE.Section_identifier
AND NOT EXISTS (SELECT Section_identifier,Course_number FROM SECTION );
```

```
* mysql+pymysql://s201904458:***@dm.hufs.ac.kr:3306/s201904458db
0 rows affected.
```

NAME

5. Database 과목을 듣기 위해 (직접 또는 간접적으로: directly or indirectly) 미리 들어야 하는

(prerequisite) 과목(Course_name)들을 모두 나열하시오. (Prerequisite는 Acyclic graph이고 path의 최대 길이는 3이라고 가정함. 길이 3: A-B-C-D)

```
%%sql
# (SELECT CE.Course_name
# FROM COURSE C, PREREQUISITE P, PREREQUISITE P2, COURSE CE
# WHERE C.Course_number=P.Course_number and P2.Course_number=P.Prerequisite_number and C.Course_name='database' and
# P2.Prerequisite_number=CE.Course_number
# GROUP BY C.Course_name, P.Prerequisite_number, P2.Prerequisite_number,CE.Course_name)
# UNION
# (SELECT CE.Course_name
# FROM COURSE C, PREREQUISITE P, COURSE CE
# WHERE C.Course_number=P.Course_number and C.Course_name='database' and P.Prerequisite_number=CE.Course_number
# GROUP BY C.Course_name, P.Prerequisite_number,CE.Course_name);
(SELECT CE.Course_name
FROM COURSE C, PREREQUISITE P, PREREQUISITE P2, COURSE CE
WHERE C.Course_number=P.Course_number and P2.Course_number=P.Prerequisite_number and C.Course_name='database' and
P2.Prerequisite_number=CE.Course_number)
UNION
(SELECT CE.Course_name
FROM COURSE C, PREREQUISITE P, COURSE CE
WHERE C.Course_number=P.Course_number and C.Course_name='database' and P.Prerequisite_number=CE.Course_number);
```

```
* mysql+pymysql://s201904458:***@dm.hufs.ac.kr:3306/s201904458db
3 rows affected.
```

Course_name

Intro to Computer Science
Data Structures
Discrete Mathematics

6. 전공으로 수강한 과목들만 나열하는 MAJOR_GRADE_REPORT 테이블을 만드시오. 테이블을 생성하고, GRADE_REPORT 테이블에서 전공으로 수강한 경우에 해당하는 행들만 아래 테이블에 삽입하시오.
- 전공으로 수강하였다는 것은 학생의 전공(Major)과 COURSE의 개설 Department가 같으면 전공으로 수강한 과목이다.

```
MAJOR_GRADE_REPORT(Student_number, Section_identifier, Grade)
```

```
%%sql
```

```
DROP TABLE IF EXISTS MAJOR_GRADE_REPORT;
```

```
CREATE TABLE MAJOR_GRADE_REPORT(Student_number INTEGER NOT NULL,
Section_identifier INTEGER NOT NULL,
Grade ENUM('A','B','C','D','F'),
FOREIGN KEY (Student_number) REFERENCES
STUDENT (Student_number) ON DELETE CASCADE ON UPDATE CASCADE,
FOREIGN KEY (Section_identifier) REFERENCES
SECTION (Section_identifier) ON DELETE CASCADE ON UPDATE CASCADE
);
```

```
INSERT INTO MAJOR_GRADE_REPORT
VALUES(8,92, 'A'),(8,102, 'B'),(8,135, 'A'),(17,119, 'C');
```

```
* mysql+pymysql://s201904458:***@dm.hufs.ac.kr:3306/s201904458db
0 rows affected.
0 rows affected.
4 rows affected.
[]
```

- Run the following cell to show that insert is correctly done

```
%sql select * from MAJOR_GRADE_REPORT;
```

```
* mysql+pymysql://s201904458:***@dm.hufs.ac.kr:3306/s201904458db
4 rows affected.
```

Student_number	Section_identifier	Grade
8	92	A
8	102	B
8	135	A
17	119	C

7. 모든 학생들의 성적들을 다음과 같이 출력하려 한다. 적절한 SQL문을 작성하여 실행하시오.
- 결과 테이블은 Major, Student_number, Year, Semester, Course_number 순서로 정렬되어야 한다.
 - 결과 테이블 애트리뷰트 이름 및 순서: (Major, Student_number, Student_name, Year, Semester, Course_number, Course_name, Grade)

```
%%sql
```

```
SELECT S.Major, S.Student_number, S.Name, SE.Year, SE.Semester, C.Course_number, C.Course_name, G.Grade
```

```
SELECT S.Major, S.Student_number, S.Name, SE.Year, SE.Semester, C.Course_number, C.Course_name, G.Grade
FROM GRADE_REPORT G, SECTION SE, STUDENT S, COURSE C
WHERE S.Student_number=G.Student_number and G.Section_identifier=SE.Section_identifier and SE.Course_number=C.Course_number
ORDER BY Major, Student_number, Year, Semester, Course_number;
```



* mysql+pymysql://s201904458:***@dm.hufs.ac.kr:3306/s201904458db
7 rows affected.

Major	Student_number	Name	Year	Semester	Course_number	Course_name	Grade
CS	8	Brown	07	Fall	CS1310	Intro to Computer Science	A
CS	8	Brown	07	Fall	MATH2410	Discrete Mathematics	A
CS	8	Brown	08	Spring	CS3320	Data Structures	B
CS	8	Brown	08	Fall	CS3380	Database	A
CS	8	Brown	08	Fall	MATH2410	Discrete Mathematics	None
CS	17	Smith	08	Fall	CS1310	Intro to Computer Science	C
CS	17	Smith	08	Fall	MATH2410	Discrete Mathematics	B

▼ 8. Brown이 수강한 Database 과목의 학점을 F로 수정하시오.

```
%%sql
```

```
UPDATE GRADE_REPORT
SET Grade='F'
WHERE Section_identifier IN (SELECT Section_identifier FROM SECTION SE, COURSE C
WHERE C.Course_name='Database' and C.Course_number=SE.Course_number );

# SELECT *
# FROM GRADE_REPORT
# WHERE Section_identifier IN (SELECT G.Section_identifier FROM SECTION SE, COURSE C, GRADE_REPORT G
# WHERE C.Course_name='Database' and C.Course_number=SE.Course_number and SE.Section_identifier=G.Section_identifier
```

* mysql+pymysql://s201904458:***@dm.hufs.ac.kr:3306/s201904458db
1 rows affected.
0 rows affected.
[]

▼ Run the following cell to show that update is correctly done

```
%sql select * from GRADE_REPORT where Student_number = 8 and Section_identifier = 135;
```

* mysql+pymysql://s201904458:***@dm.hufs.ac.kr:3306/s201904458db
1 rows affected.

Student_number	Section_identifier	Grade
8	135	F

▼ 9. Brown의 학번을 8번에서 9번으로 수정하시오. 반드시 update문 하나만 있어야 하며, update 문은 반드시 성공하여야 한다. 테이블의 Foreign Key 선언문을 삭제하면 안됨.

```
%%sql
```

```
UPDATE STUDENT
SET Student_number=9
WHERE Name='Brown'
```

* mysql+pymysql://s201904458:***@dm.hufs.ac.kr:3306/s201904458db

1 rows affected.
[]

▼ Run the following cell to show that update is correctly done

```
%sql select * from STUDENT where name = 'Brown';
```

* mysql+pymysql://s201904458:***@dm.hufs.ac.kr:3306/s201904458db
1 rows affected.

Name	Student_number	Class	Major
Brown	9	2	CS