데이터베이스관리

13주차

담당교수: 김희숙
(jasmin11@hanmail.net)

SQL: 자체 조인(self join)

13주차 13-01

담당교수: 김희숙
(jasmin11@hanmail.net)

[실습] 자체조인(self_mysql.sql)


```
-- selfemp(empno,empname,manager,dno)
```

selfemp E

empno	empname	manager	dno
1003	조민희	3011	1
2106	김창섭	3426	2
3011	이수민	NULL	1
3426	박영권	3011	3
3427	최종철	2106	3

selfemp M

empno	empname	manager	dno
1003	조민희	3011	1
2106	김창섭	3426	2
3011	이수민	NULL	1
3426	박영권	3011	3
3427	최종철	2106	3

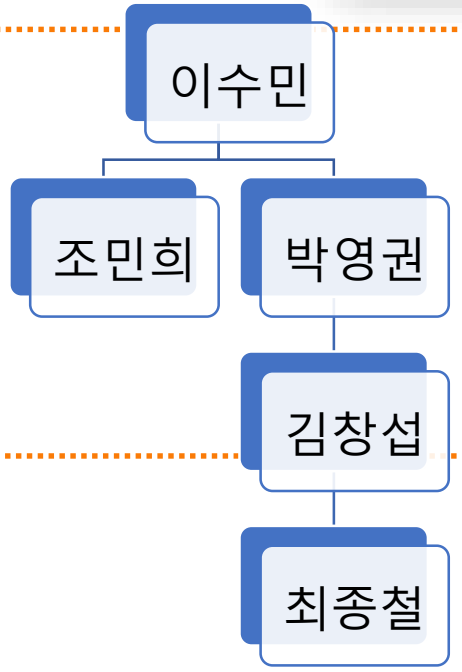
(테이블 1개)

selfemp 외래키

empno	empname	manager	dno
1003	조민희	3011	1
2106	김창섭	3426	2
3011	이수민		1
3426	박영권	3011	3
3427	최종철	2106	3

```
-- [자체조인 1]
-- selfemp(empno,empname,manager,dno)
-- 사원명, 관리자이름 을 검색하라

select  E.empname 사원명, M.empname 관리자이름
from    selfemp E, selfemp M
where   E.manager = M.empno;
```



사원명	관리자 이름
조민희	이수민
김창섭	박영권
박영권	이수민
최종철	김창섭

[실습] 자체조인(self_mysql.sql)


```
-- deptself(deptno,dname,college,loc)
```

```
-- deptself(deptno,dname,college,loc)
CREATE TABLE deptself (
  deptno char(3)      NOT NULL,
  dname  varchar(30) NOT NULL,
  college char(3),
  loc varchar(7),
  CONSTRAINT dept_no_pk PRIMARY KEY(deptno)
);
```

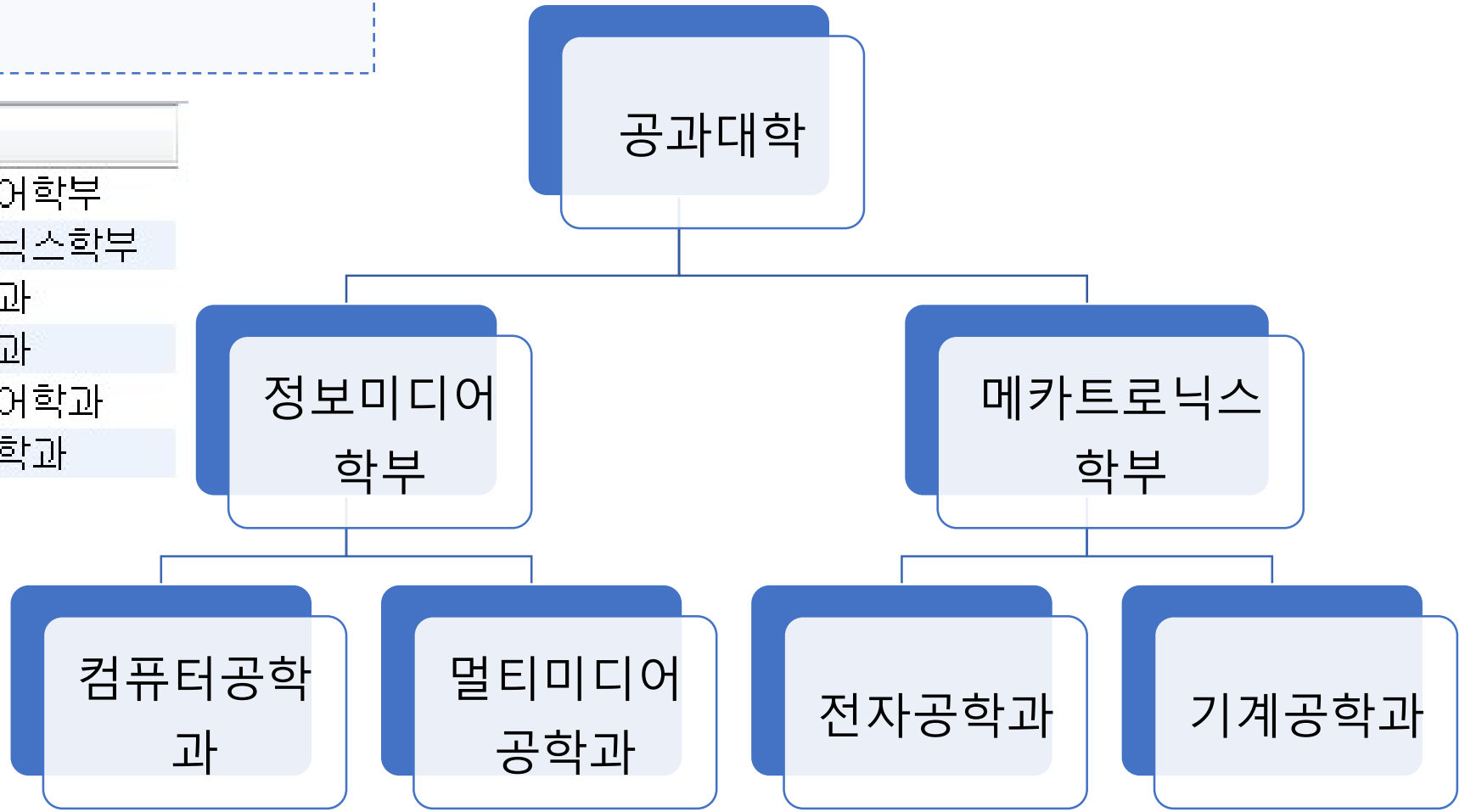
```
-- [자체조인 2]
insert into deptself values ('101', '컴퓨터공학과', '100', '1호관');
insert into deptself values ('102', '멀티미디어학과', '100', '2호관');
insert into deptself values ('201', '전자공학과', '200', '3호관');
insert into deptself values ('202', '기계공학과', '200', '4호관');
insert into deptself values ('100', '정보미디어학부', '10', NULL);
insert into deptself values ('200', '메카트로닉스학부', '10', NULL);
insert into deptself values ('10',  '공과대학', NULL, NULL);
```

[실습] 자체조인(self_mysql.sql)


```
-- deptself(deptno,dname,college,loc)
```

```
-- [자체조인 2]
```

dname	dname
공과대학	정보미디어학부
공과대학	메카트로닉스학부
메카트로닉스학부	기계공학과
메카트로닉스학부	전자공학과
정보미디어학부	멀티미디어학과
정보미디어학부	컴퓨터공학과



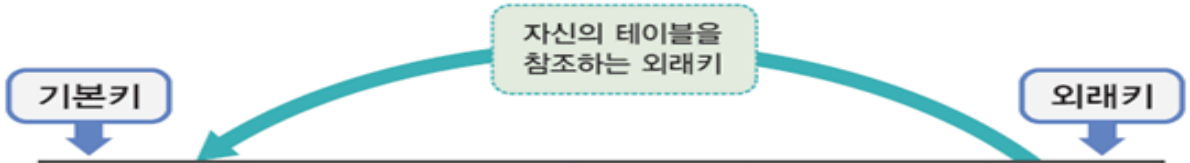
[실습] 자체조인(self_mysql.sql)

-- prof_self(교수번호,주민등록번호,이름,학과명,학과장)

```
-- [자체조인 3]
-- prof_self(교수번호,주민등록번호,이름,학과명,학과장)

-- 학과명, 학과장이름 을 검색하라
select   DISTINCT p1.학과명, p2.이름
from     prof_self p1, prof_self p2
where    p1.학과장 =  p2.교수번호;
```

교수(교수번호, 주민등록번호, 이름, 학과명, 학과장)



교수번호	주민등록번호	이름	학과명	학과장
92001	590327-1839240	이태규	컴퓨터공학과	92001
92002	690702-1350026	고희석	컴퓨터공학과	92001
92301	741011-2765501	최성희	산업공학과	92302
92302	750728-1102458	김태석	산업공학과	92302
92501	620505-1200546	박철재	전자공학과	NULL
92502	681006-1023456	강만희	전자공학과	NULL

(그림 출처: "Understanding of Database", 이상구외 공저, 이한, 2012)

[실습] 자체조인(self_mysql.sql)

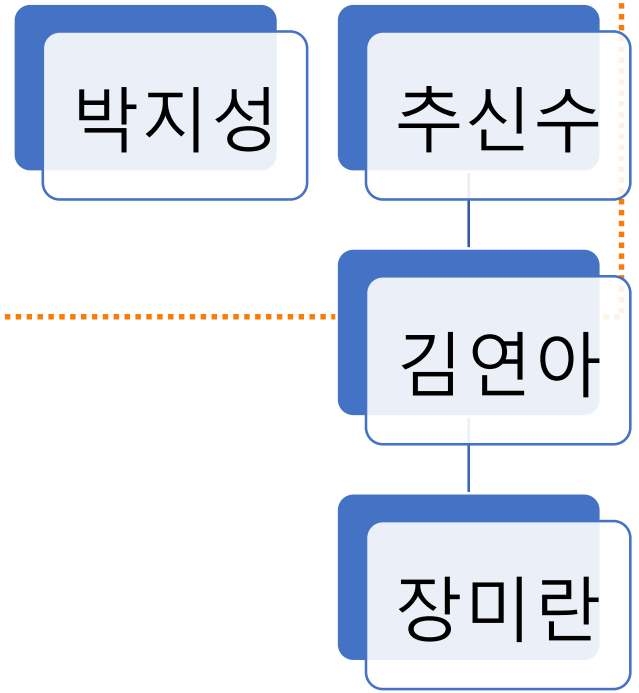
-- 멘토(선수번호, 이름, 주소, 멘토번호)

```
-- [자체조인 4]
-- 멘토(선수번호, 이름, 주소, 멘토번호)
CREATE TABLE 멘토 (
        선수번호              int NOT NULL,
        이름                  varchar(9) NULL,
        주소                  varchar(50) NULL,
        멘토번호              int ,
        PRIMARY KEY (선수번호) ,
        FOREIGN KEY (멘토번호)
            REFERENCES 멘토(선수번호)
);
```

-- [자체조인 4]
데이터입력 순서는?

멘토 외래키

선수번호	이름	주소	멘토번호
1	박지성	영국	
2	김연아	대한민국	3
3	장미란	대한민국	4
4	추신수	미국	



SQL: 하위 질의(sub query)

13주차 13-02

담당교수: 김희숙
(jasmin11@hanmail.net)

[요약] 조인

- ❖ 내부조인(theta join, equi join, natural join)
- ❖ 외부조인(left outer join, right outer join, full outer join)

1. 내부조인(동등조인)

1) SELECT .. FROM .. WHERE 방법

2) SELECT .. FROM .. ON 방법

2. 내부조인(자연조인)

1) SELECT .. FROM .. WHERE 방법

2) SELECT .. FROM .. ON 방법

[실습] 조인

동양미래대학교 DONGYANG MIRAE UNIVERSITY

[실습]

-- department(deptno, deptname, floor)

-- employee(empno, empname, title, manager, salary, dno)

deptno	deptname	floor
1	영업	8
2	기획	10
3	개발	9
4	총무	7

empno	empname	title	manager	salary	dno
1003	조민희	과장	4377	3000000	2
1365	김상원	사원	3426	1500000	1
2106	김창섭	대리	1003	2500000	2
3011	이수민	부장	4377	4000000	3
3426	박영권	과장	4377	3000000	1
3427	최종철	사원	3011	1500000	3
4377	이성래	이사	NULL	5000000	2

-- 17) 사원의 이름과 이 사원이 속한 부서이름을 검색하라 (조인)

-- 19) 사원에 대해서 부서이름, 사원이름, 직급, 급여를 검색하라.
-- 부서이름에 대해서 오름차순, 부서이름이 같을 경우에는 salary에 대해서 내림차순으로 정렬하라

[요약] 하위질의

- ❖ 하위질의(Sub query)
- ❖ 서브쿼리 (Sub query, 부질의)
- ❖ 중첩 질의
- ❖ 상관중첩질의

[실습]

-- department(deptno, deptname, floor)

-- employee(empno, empname, title, manager, salary, dno)

empno	empname	title	manager	salary	dno
1003	조민희	과장	4377	3000000	2
1365	김상원	사원	3426	1500000	1
2106	김창섭	대리	1003	2500000	2
3011	이수민	부장	4377	4000000	3
3426	박영권	과장	4377	3000000	1
3427	최종철	사원	3011	1500000	3
4377	이성래	이사	NULL	5000000	2

-- (하위질의)

-- 20) 박영권과 같은 직급을 갖는 모든 사원들의 이름과 직급을 검색하라

```
SELECT   empname, title
FROM     employee
WHERE    title   = (SELECT   title
                   FROM     employee
                   WHERE    empname = '박영권');
```

[요약] 하위질의

[실습]

-- department(deptno, deptname, floor)

-- employee(empno, empname, title, manager, salary, dno)

-- 21) 영업부나 개발부에 근무하는 사원들의 이름을 검색하라(IN 사용)

deptno	deptname	floor
1	영업	8
2	기획	10
3	개발	9
4	총무	7

empno	empname	title	manager	salary	dno
1003	조민희	과장	4377	3000000	2
1365	김상원	사원	3426	1500000	1
2106	김창섭	대리	1003	2500000	2
3011	이수민	부장	4377	4000000	3
3426	박영권	과장	4377	3000000	1
3427	최종철	사원	3011	1500000	3
4377	이성래	이사	NULL	5000000	2

```
SELECT   empname
FROM     employee
WHERE    dno  IN (SELECT  deptno
                  FROM    department
                  WHERE   deptname IN ('영업','개발'));
```

```
SELECT empname
FROM   employee, department
WHERE  employee.dno = department.deptno
  AND  (deptname = '영업' OR deptname = '개발');
```

[요약] 하위질의

-- 24) 사원들이 한 명도 소속되지 않은 부서명을 검색하라 (NOT EXISTS 사용)

[실습]

-- department(deptno, deptname, floor)

-- employee(empno, empname, title, manager, salary, dno)

deptno	deptname	floor
1	영업	8
2	기획	10
3	개발	9
4	총무	7

empno	empname	title	manager	salary	dno
1003	조민희	과장	4377	3000000	2
1365	김상원	사원	3426	1500000	1
2106	김창섭	대리	1003	2500000	2
3011	이수민	부장	4377	4000000	3
3426	박영권	과장	4377	3000000	1
3427	최종철	사원	3011	1500000	3
4377	이성래	이사	NULL	5000000	2

```
select   deptname
from    department D
where   NOT EXISTS (select *
                    from    employee E
                    where   D.deptno = E.dno);
```

[요약] 하위질의

-- stu(sno, sname, dept, sage)

-- pro(pno, pname, dept, page)

- ANY 서브쿼리의 여러 개의 결과 중 한가지만 만족해도 되며
- ALL 서브쿼리의 여러 개의 결과를 모두 만족시켜야 된다
- SOME 은 ANY 와 동일한 의미
- IN 는 =ANY 와 동일, NOT IN 은 <>ALL 과 동일

stu

sno	sname	dept	sage
s1	유준호	컴퓨터	23
s2	오정민	컴퓨터	34
s3	이태현	건축	22
s4	신현주	건축	21

prof

pno	pname	dept	page
p1	이정무	컴퓨터	36
p2	우태하	컴퓨터	32
p3	이성민	건축	45

```
-- (하위질의)
-- 3-1) 교수 테이블에서 이정무 의 학과와 같은 학생 이름, 학과, 나이를 검색하라
```

[요약] 하위질의

-- stu(sno, sname, dept, sage)

-- pro(pno, pname, dept, page)

- ANY 서브쿼리의 여러 개의 결과 중 한가지만 만족해도 되며
- ALL 서브쿼리의 여러 개의 결과를 모두 만족시켜야 된다
- SOME 은 ANY 와 동일한 의미
- IN 는 =ANY 와 동일, NOT IN 은 <>ALL 과 동일

stu

sno	sname	dept	sage
s1	유준호	컴퓨터	23
s2	오정민	컴퓨터	34
s3	이태현	건축	22
s4	신현주	건축	21

prof

pno	pname	dept	page
p1	이정무	컴퓨터	36
p2	우태하	컴퓨터	32
p3	이성민	건축	45

-- (하위질의)

-- 3-2) 교수 테이블에서 pno 가 p1 이거나 p2 인 학과와 같은 학생 이름, 학과, 나이를 검색하라

[요약] 하위질의

-- stu(sno, sname, dept, sage)

-- pro(pno, pname, dept, page)

- ANY 서브쿼리의 여러 개의 결과 중 한가지만 만족해도 되며
- ALL 서브쿼리의 여러 개의 결과를 모두 만족시켜야 된다
- SOME 은 ANY 와 동일한 의미
- IN 는 =ANY 와 동일, NOT IN 은 <>ALL 과 동일

stu

sno	sname	dept	sage
s1	유준호	컴퓨터	23
s2	오정민	컴퓨터	34
s3	이태현	건축	22
s4	신현주	건축	21

prof

pno	pname	dept	page
p1	이정무	컴퓨터	36
p2	우태하	컴퓨터	32
p3	이성민	건축	45

```
-- (하위질의)
-- 3-3) 학생 테이블과 교수 테이블에서 모든 학생들보다 나이가 많은 교수의
-- 교번, 이름, 나이를 ALL 구문을 이용하여 검색하라
select pno, pname,page
from   pro
where  page >ALL (select sage
                  from   stu);
```

[요약] 하위질의

-- stu(sno, sname, dept, sage)

-- pro(pno, pname, dept, page)

- ANY 서브쿼리의 여러 개의 결과 중 한가지만 만족해도 되며
- ALL 서브쿼리의 여러 개의 결과를 모두 만족시켜야 된다
- SOME 은 ANY 와 동일한 의미
- IN 는 =ANY 와 동일, NOT IN 은 <>ALL 과 동일

stu

sno	sname	dept	sage
s1	유준호	컴퓨터	23
s2	오정민	컴퓨터	34
s3	이태현	건축	22
s4	신현주	건축	21

prof

pno	pname	dept	page
p1	이정무	컴퓨터	36
p2	우태하	컴퓨터	32
p3	이성민	건축	45

```
-- (하위질의)
-- 3-4) 학생 테이블과 교수 테이블에서 한 명 이상 교수보다
-- 나이가 많은 학생이 있을 경우 학생의 학번, 이름, 나이를 SOME 구문으로 검색하라


select sno, sname, sage
from   stu
where  sage >SOME (select page
                   from   pro);
```

[요약] 하위질의

-- stu(sno, sname, dept, sage)

-- pro(pno, pname, dept, page)

- ANY 서브쿼리의 여러 개의 결과 중 한가지만 만족해도 되며
- ALL 서브쿼리의 여러 개의 결과를 모두 만족시켜야 된다
- SOME 은 ANY 와 동일한 의미
- IN 는 =ANY 와 동일, NOT IN 은 <>ALL 과 동일

stu

sno	sname	dept	sage
s1	유준호	컴퓨터	23
s2	오정민	컴퓨터	34
s3	이태현	건축	22
s4	신현주	건축	21

prof

pno	pname	dept	page
p1	이정무	컴퓨터	36
p2	우태하	컴퓨터	32
p3	이성민	건축	45

```
-- (하위질의)
-- 3-5) 가장 나이가 많은 학생 이름, 나이를 검색하라
select sname, dept, sage
from   stu
where  sage = (select MAX(sage)
               from   stu);
```

[요약] 하위질의

-- stu(sno, sname, dept, sage)

-- pro(pno, pname, dept, page)

- ANY 서브쿼리의 여러 개의 결과 중 한가지만 만족해도 되며
- ALL 서브쿼리의 여러 개의 결과를 모두 만족시켜야 된다
- SOME 은 ANY 와 동일한 의미
- IN 는 =ANY 와 동일, NOT IN 은 <>ALL 과 동일

stu

sno	sname	dept	sage
s1	유준호	컴퓨터	23
s2	오정민	컴퓨터	34
s3	이태현	건축	22
s4	신현주	건축	21

prof

pno	pname	dept	page
p1	이정무	컴퓨터	36
p2	우태하	컴퓨터	32
p3	이성민	건축	45

```
-- (하위질의)
-- 3-6) 각 학과마다 가장 나이가 많은 학생 이름, 학과, 나이를 검색하라
select sname, dept, sage
from   stu S1
where  sage IN (select MAX(sage)
                from   stu S2
                where  S1.dept = S2.dept);
```

[실습] 하위질의

```
-- stu(sno, sname, dept, sage)
-- pro(pno, pname, dept, page)

CREATE TABLE pro (
  pno    char(2),
  pname varchar(20),
  dept   varchar(20),
  page   int,
  primary key(pno)
);

CREATE TABLE stu (
  sno    char(2),
  sname varchar(20),
  dept   varchar(20),
  sage   int,
  primary key(sno)
);
```

[실습] 하위질의(subquery.sql)

```
insert into pro values('p1','이정무','컴퓨터',36);
insert into pro values('p2','우태하','컴퓨터',32);
insert into pro values('p3','이성민','건축',45);

insert into stu values('s1','유준호','컴퓨터',23);
insert into stu values('s2','오정민','컴퓨터',34);
insert into stu values('s3','이태현','건축',22);
insert into stu values('s4','신현주','건축',21);

select * from pro;
select * from stu;
```

SQL: 실습 예제(학사)

13주차 13-03

담당교수: 김희숙
(jasmin11@hanmail.net)

[실습] 실습 예제(ehan-mysql-stu.sql)

[실습]

-- department(dept_id, dept_name, office)

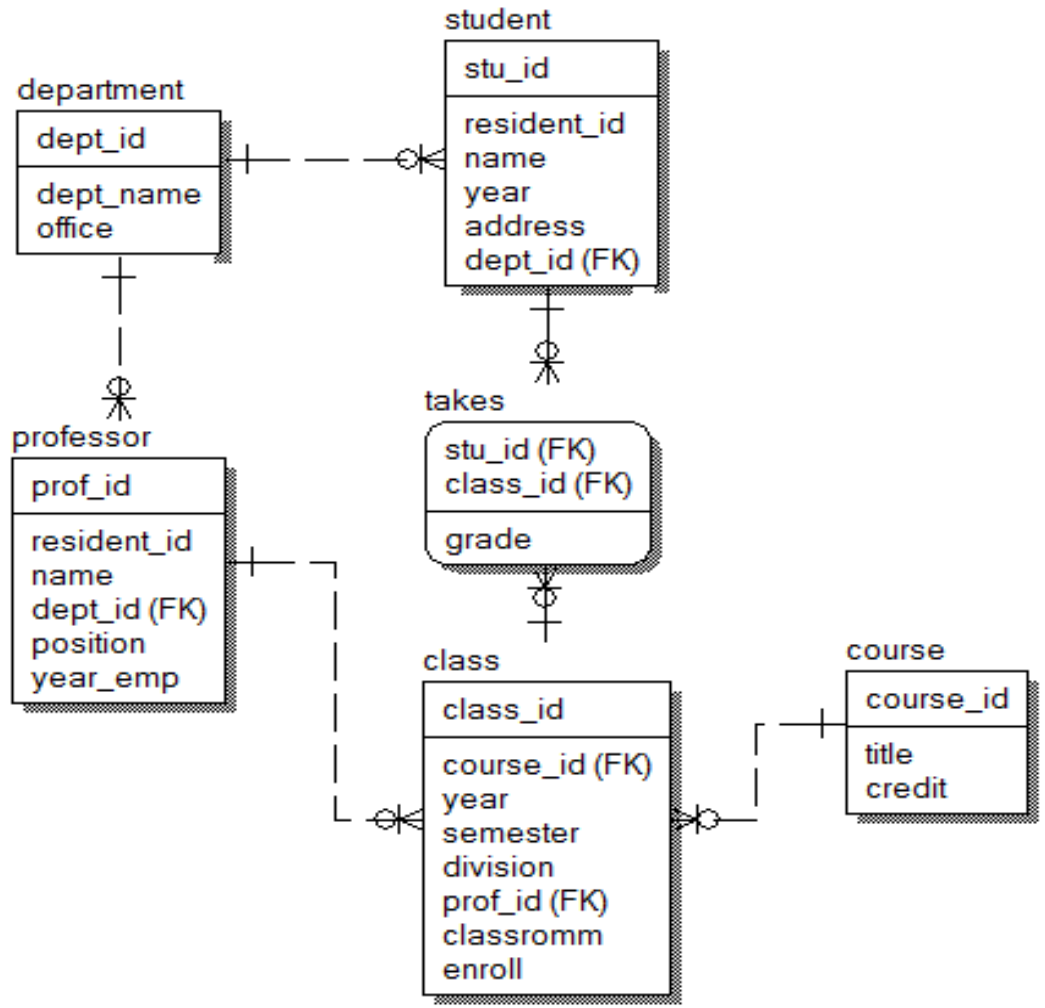
-- student(stu_id, resident_id, name, year, address, dept_id)

-- professor(prof_id, resident_id, name, dept_id, position, year_emp)

-- course(course_id, title, credit)

-- class(class_id, course_id, year, semester, division, prof_id, classroom, enroll)

-- takes(stu_id, class_id, grade)



-- [예제1] 다음에 대하여 SQL문법과 실행결과를 작성하라

-- 질의21) student 테이블에서 주소를 검색하라
-- 질의22) student 테이블에서 주소를 검색하라(중복 제거)
-- 질의23) student 테이블에서 모든 필드를 검색하라
-- 질의24) professor 테이블에서 교수이름, 재직연수를 검색하라(
-- 질의27) student 테이블에서 1, 2학년 학생들의 이름과 학번을 검색하라
-- 단, 학생이름의 오름차순 정렬하고, 같은 이름은 학번의 오름차순 정렬하라
-- 질의28) student 테이블에서 1, 2학년 학생들의 이름과 학번을 검색하라
-- 단, 학생이름의 내림차순 정렬하고, 같은 이름은 학번의 오름차순 정렬하라

[실습] 실습 예제(ehan-mysql-stu.sql)

동양미래대학교 DONGYANG MIRAE UNIVERSITY

[실습]

-- department(dept_id, dept_name, office)

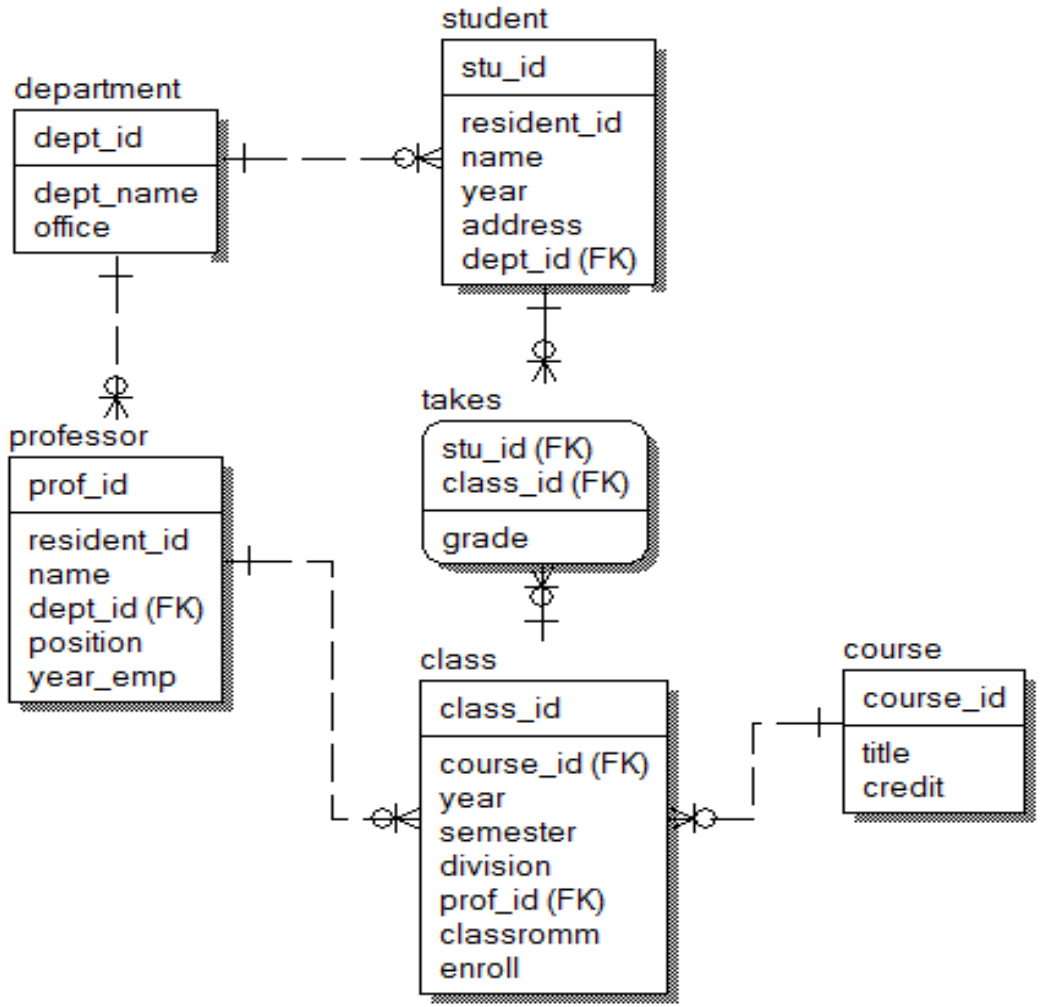
-- student(stu_id, resident_id, name, year, address, dept_id)

-- professor(prof_id, resident_id, name, dept_id, position, year_emp)

-- course(course_id, title, credit)

-- class(class_id, course_id, year, semester, division, prof_id, classroom, enroll)

-- takes(stu_id, class_id, grade)




```
-- [예제2] 조인, 집합연산

-- 질의20) 학생이름, 학과명을 검색하라
-- 방법1) SELECT ... FROM ... WHERE
-- 방법2) SELECT ... FROM ... ON

-- 질의25) 학생이름, 학번, 소속학과명을 검색하라
-- 질의26) 컴퓨터공학과 2학년 학생들의 학번을 검색하라
-- 질의30) student 테이블과 department 테이블을 조인하여 학생이름, 소속학과이름을 검색
--       단, student 테이블은 s로, department 테이블은 d 로 재명명하라
-- 질의31) student 테이블에서 김광식 학생과 주소가 같은 학생이름을 검색하라
```

[실습] 실습 예제(ehan-mysql-stu.sql)

[실습]

-- department(dept_id, dept_name, office)

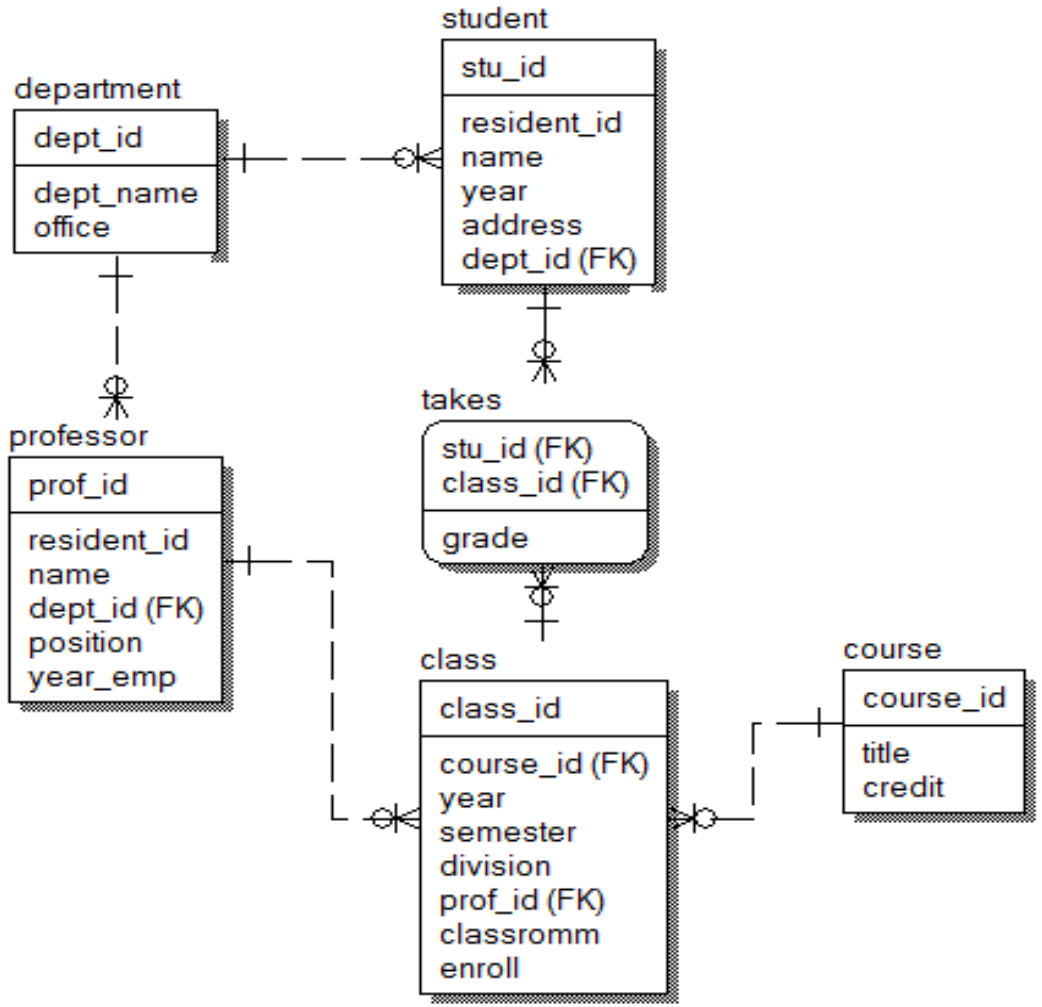
-- student(stu_id, resident_id, name, year, address, dept_id)

-- professor(prof_id, resident_id, name, dept_id, position, year_emp)

-- course(course_id, title, credit)

-- class(class_id, course_id, year, semester, division, prof_id, classroom, enroll)

-- takes(stu_id, class_id, grade)



-- [예제3] 조인, 집합연산

-- 질의36) student 테이블의 학생이름과 professor 테이블의 교수이름을 합집합하라

-- 질의40) 컴퓨터공학과 학생들 중에서 교과목에 상관없이 학점을 A+ 받은 학생의 학번검색

[실습] 실습 예제(ehan-mysql-stu.sql)

[실습]

-- department(dept_id, dept_name, office)

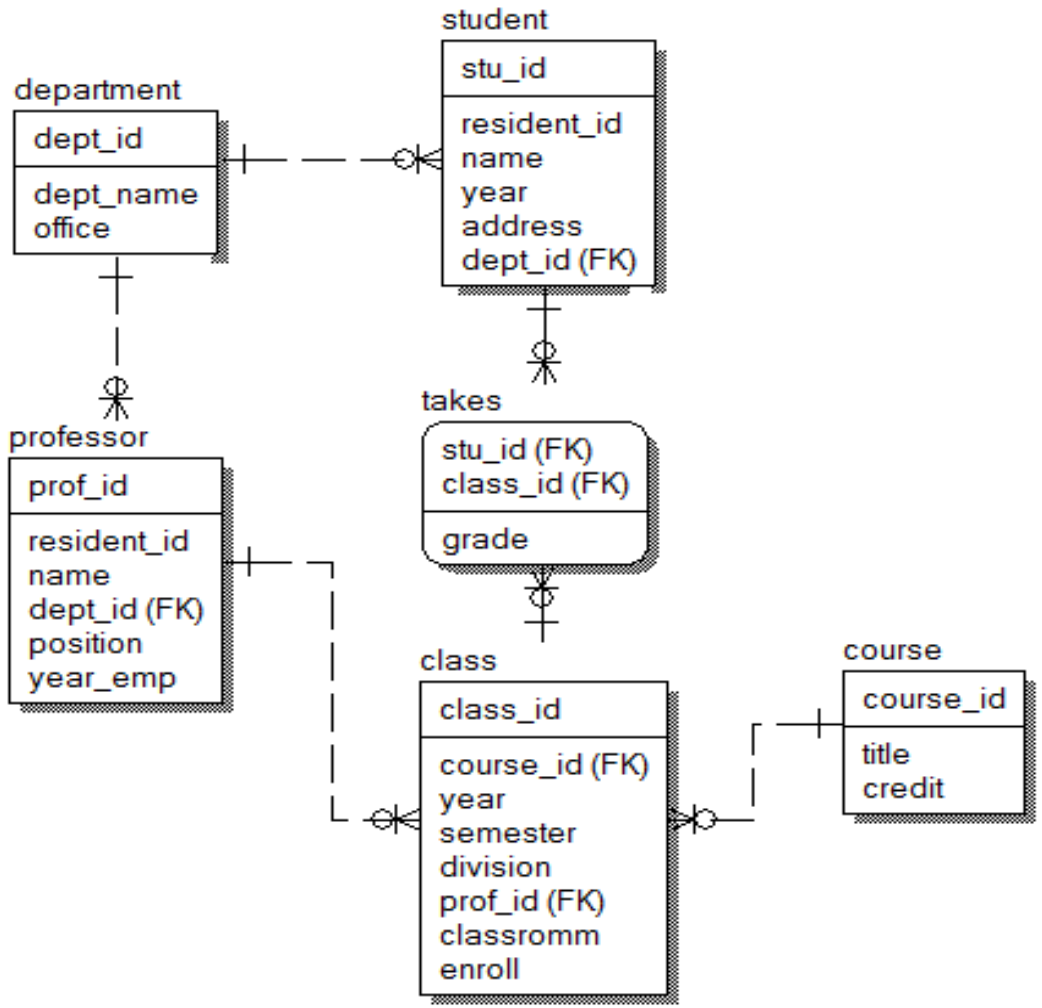
-- student(stu_id, resident_id, name, year, address, dept_id)

-- professor(prof_id, resident_id, name, dept_id, position, year_emp)

-- course(course_id, title, credit)

-- class(class_id, course_id, year, semester, division, prof_id, classroom, enroll)

-- takes(stu_id, class_id, grade)



-- [예제4] 외부조인

-- 질의42) 모든 교과목들에 대해 교과목명, 학점수, 개설연도, 개설학기를 검색하라

[실습] 실습 예제(ehan-mysql-stu.sql)

동양미래대학교 DONGYANG MIRAE UNIVERSITY

[실습]

-- department(dept_id, dept_name, office)

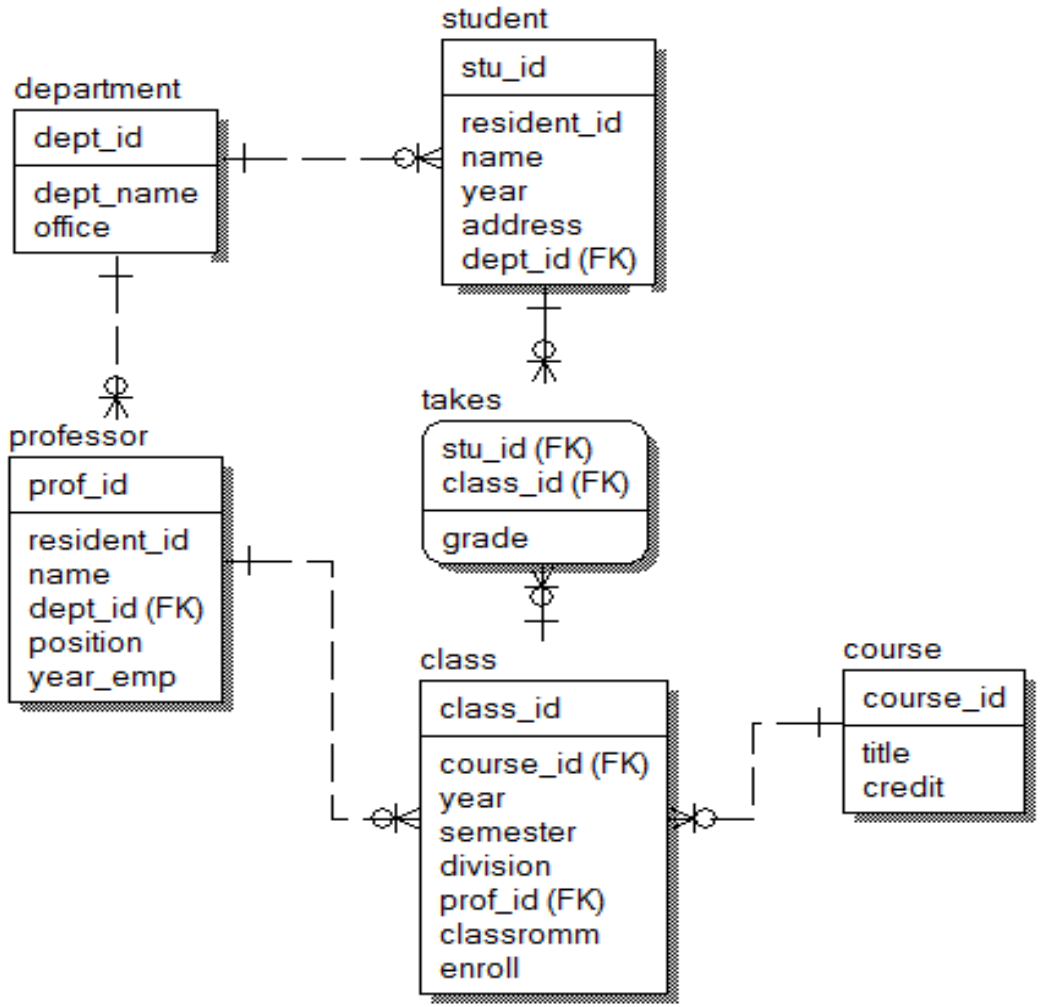
-- student(stu_id, resident_id, name, year, address, dept_id)

-- professor(prof_id, resident_id, name, dept_id, position, year_emp)

-- course(course_id, title, credit)

-- class(class_id, course_id, year, semester, division, prof_id, classroom, enroll)

-- takes(stu_id, class_id, grade)



-- [예제5] 집계함수, group by

-- 질의48) student 테이블에서 2학년 학생이 몇 명인지 검색하라
-- 질의49) student 테이블에서 dept_id 필드에 값이 몇 개인지 검색하라
-- 질의49) student 테이블에서 dept_id 필드에 값이 몇 개인지 검색하라 (중복 제거)
-- 질의50) 컴퓨터공학과의 학생 수를 검색하라
-- 질의51) 전체교수들의 재직연수 합을 구하라
-- 질의55) 전체 교수의 평균 재직연수를 출력하라

[실습] 실습 예제(ehan-mysql-stu.sql)

[실습]

-- department(dept_id, dept_name, office)

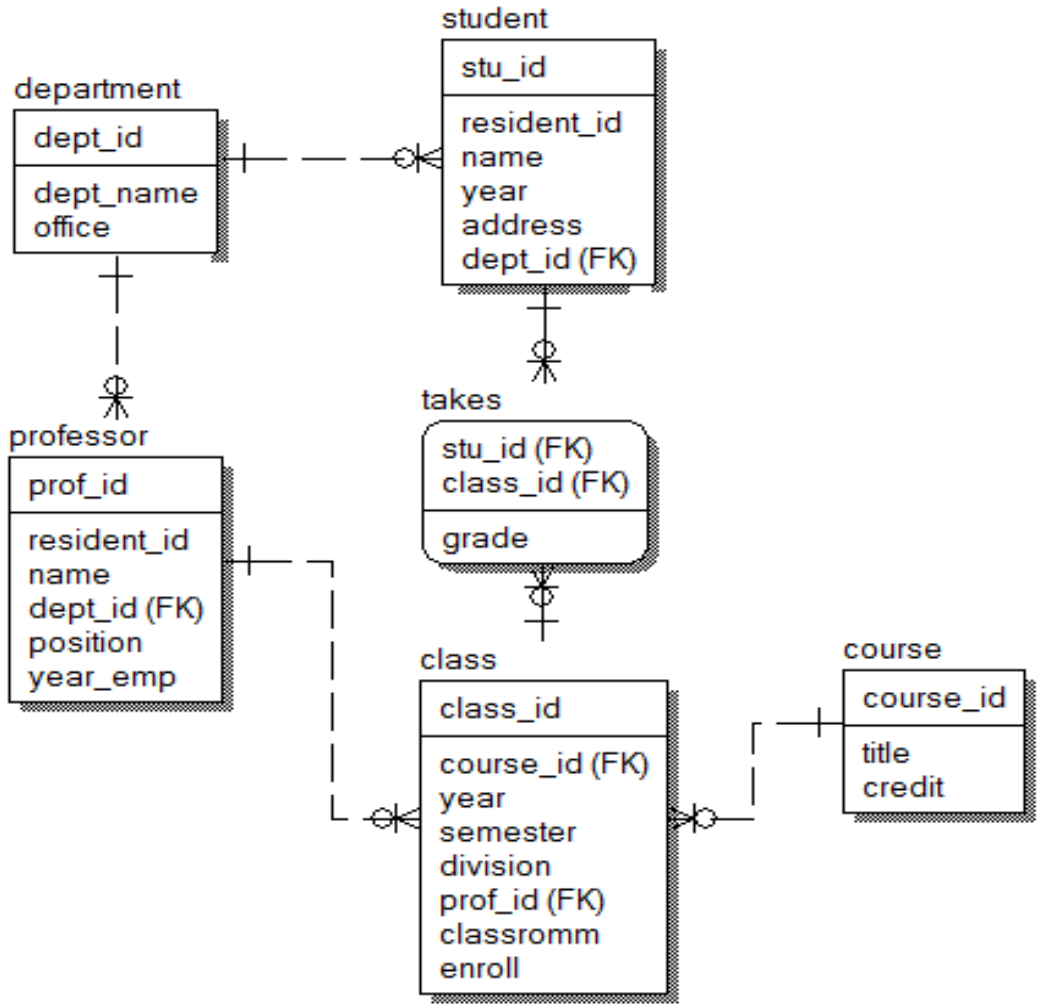
-- student(stu_id, resident_id, name, year, address, dept_id)

-- professor(prof_id, resident_id, name, dept_id, position, year_emp)

-- course(course_id, title, credit)

-- class(class_id, course_id, year, semester, division, prof_id, classroom, enroll)

-- takes(stu_id, class_id, grade)



-- [예제5] 집계함수, group by

-- 질의57) student 테이블에서 학과번호(dept_id) 별로 레코드의 개수를 검색하라

-- 질의58) 학과명 별로 레코드의 개수를 검색하라

-- 질의60) 학과별 교수 숫자와 평균 재직연수, 최대 재직연수를 검색하라

-- 질의62) 평균 재직연수가 10년 이상인 학과에 대해서만

-- 교수 수, 평균재직연수, 최대재직연수를 검색하라(having)

[실습] 실습 예제(ehan-mysql-stu.sql)

[실습]

-- department(dept_id, dept_name, office)

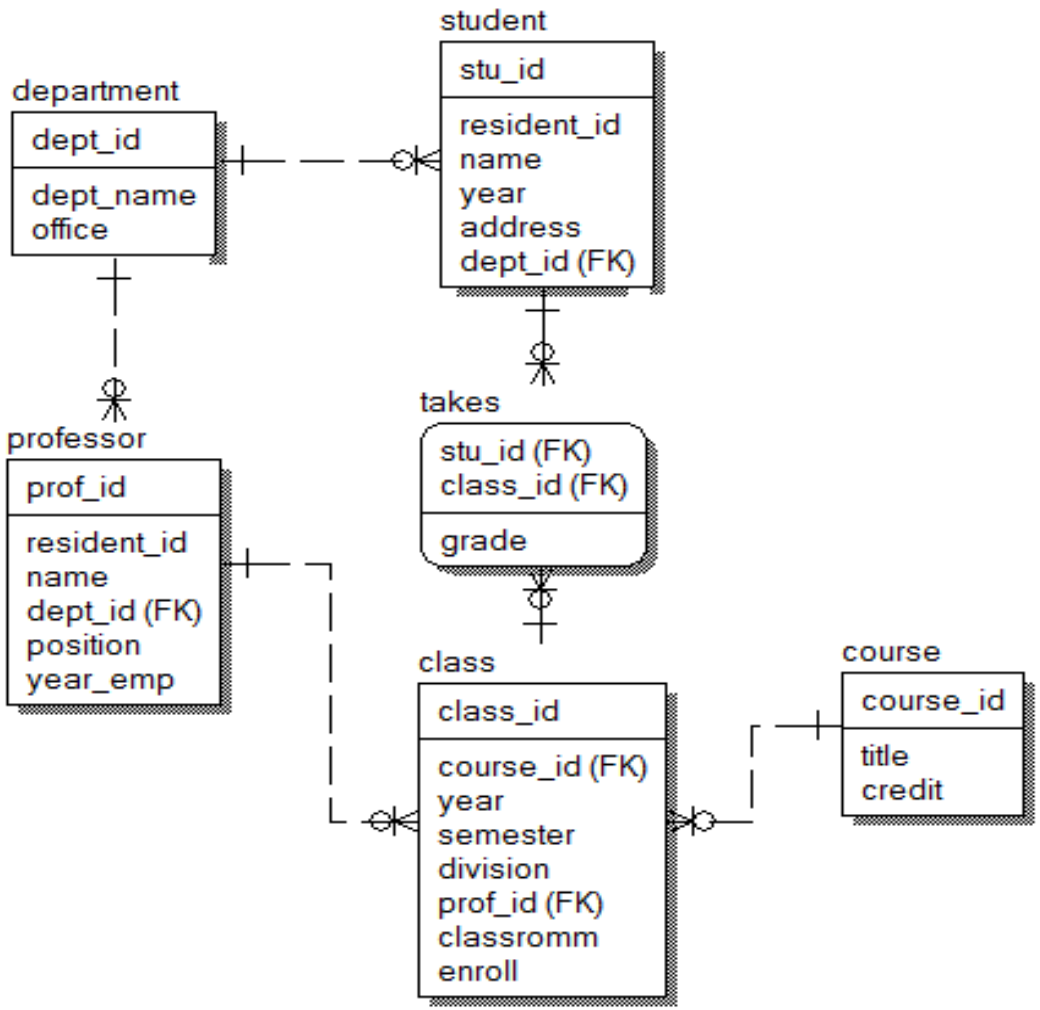
-- student(stu_id, resident_id, name, year, address, dept_id)

-- professor(prof_id, resident_id, name, dept_id, position, year_emp)

-- course(course_id, title, credit)

-- class(class_id, course_id, year, semester, division, prof_id, classroom, enroll)

-- takes(stu_id, class_id, grade)



-- [예제6] 하위질의

질의66) 301호 강의실에서 개설된 강좌의 과목명을 검색하라(하위질의)

질의67) 301호 강의실에서 개설된 강좌의 과목명을 검색하라(조인)

[실습] 실습 예제(ehan-mysql-stu.sql)

[실습]

-- department(dept_id, dept_name, office)

-- student(stu_id, resident_id, name, year, address, dept_id)

-- professor(prof_id, resident_id, name, dept_id, position, year_emp)

-- course(course_id, title, credit)

-- class(class_id, course_id, year, semester, division, prof_id, classroom, enroll)

-- takes(stu_id, class_id, grade)

-- [예제6] 하위질의

질의69) 가장 많은 수강인원을 가진 강좌를 검색하라