



SQL (4)

컴퓨터과학과 정재화

학습목차

- ① 중첩 질의의 이해
- ② 조인 질의의 이해
- ③ 뷰의 사용



01

중첩 질의의 이해

- 중첩 질의의 개념
- 중첩 질의의 형식
- 사용 예



중첩 질의의 개념

- ◇ SELECT 문 내부에서 독립적으로 실행 가능한 또 다른 SELECT 문이 내포되어 있는 질의
 - ⊕ 일반적으로 내부 질의의 처리결과를 외부 질의에서 재사용하여 처리하는 과정
- ◇ 중첩 질의의 종류
 - ⊕ FROM 절에서의 중첩 질의 활용
 - FROM 절에서의 결과 집합을 SELECT 문에서 재검색
 - ⊕ WHERE 절에서의 중첩 질의 활용
 - WHERE 절에서의 결과 집합을 활용하여 외부 질의에서 레코드의 출력 여부를 결정
 - IN, NOT IN, EXISTS, NOT EXISTS 사용

☆ 중첩 질의의 형식



```
SELECT 컬럼1, 컬럼2, ..., 컬럼n
      FROM (SELECT 컬럼1, 컬럼2, ..., 컬럼m
            FROM 테이블
            WHERE 조건 )
      WHERE 조건
```



```
SELECT 컬럼1, 컬럼2, ..., 컬럼n
      FROM 테이블1
      WHERE 컬럼i 연산자 (SELECT 컬럼j
                        FROM 테이블2
                        WHERE 조건)
```

☆ 중첩 질의의 사용 1

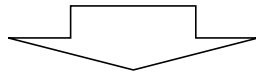
?

학과별 교수의 평균 연봉이 70,000,000 미만인 학과 중 가장 높은 평균 연봉을 출력하시오.

교수번호	교수이름	직위	소속학과	연봉
186432-760829	최우성	조교수	생활과학과	52000000
201547-634895	현경석	정교수	생활과학과	66000000
189414-790829	한용운	조교수	법학과	45000000
191924-730620	이동휘	부교수	행정학과	51000000
194634-810228	김규식	정교수	컴퓨터과학과	70000000
194834-760517	정재화	부교수	컴퓨터과학과	53000000
201216-158465	정용제	조교수	국어국문학과	55000000
210315-549413	황지수	부교수	유아교육과	52000000

중첩 질의의 사용 1

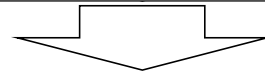
SELECT 소속학과, AVG(연봉) AS 평균연봉
FROM 교수
GROUP BY 소속학과



소속학과	평균연봉
국어국문학과	55000000.00
법학과	45000000.00
생활과학과	59000000.00
유아교육과	52000000.00
컴퓨터과학과	61500000.00

☆ 중첩 질의의 사용 1

SELECT MAX(d.평균연봉) AS 평균연봉
FROM (SELECT 소속학과, AVG(연봉) AS 평균연봉
FROM 교수
GROUP BY 소속학과) AS d
WHERE d.평균연봉 < 70000000

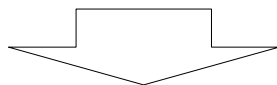


평균연봉
61500000.00

중첩 질의의 사용 2



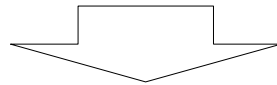
직위가 '부교수' 이고 이수구분이 '교양' 인 과목을 강의하는 교수의 이름과 소속학과를 출력하시오.



과목코드	과목명	학점	선수과목	이수구분	교수번호
COM11	컴퓨터의 이해	3		교양	...
COM12	파이썬 프로그래밍 기초	3		교양	...
COM24	자료구조	3	COM12	전공필수	...
COM31	데이터베이스 시스템	3	COM24	전공필수	...
COM34	알고리즘	3	COM24	일반선택	...
COM44	클라우드 컴퓨팅	3		전공필수	...
ECE24	놀이지도	3		전공필수	...
ECE31	유아언어교육	3	ECE31	전공필수	...
⋮	⋮	⋮	⋮	⋮	⋮

☆ 중첩 질의의 사용 2

SELECT 교수번호 FROM 과목
WHERE 이수구분 = '교양'

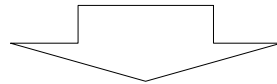


교수번호
...
...
...

☆ 중첩 질의의 사용 2



```
SELECT A.교수이름, A.소속학과  
FROM 교수 AS A  
WHERE A.직위 = '부교수' AND  
A.교수번호 IN (SELECT B.교수번호 FROM 과목 B  
WHERE B.이수구분 = '교양')
```

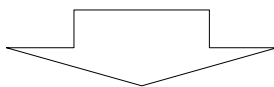


교수이름	소속학과
정재화	컴퓨터과학과

☆ 중첩 질의의 사용 2



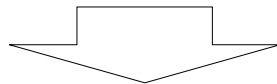
'생활과학과' 소속 학생 중 수강신청을 하지 않은 학생의
학생번호를 출력하시오.



과목코드	학생번호	신청시각
KO03	201831-331215	2019-02-11 13:31:45
COM34	201831-331215	2020-02-20 13:54:22
COM24	201831-331215	2019-08-22 12:23:31
COM12	201831-331215	2019-08-21 23:25:25
LAW21	201834-021216	2016-02-11 08:21:22
KO03	201834-021216	2016-11-12 02:16:51
HE25	201834-021216	2017-08-01 01:24:54
:	:	:

☆ 중첩 질의의 사용 2

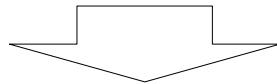
SELECT 학생번호
FROM 전공
WHERE 학과이름 = '생활과학과'



학생번호
201926-880215
202031-354516

중첩 질의의 사용 2

```
SELECT A.학생번호  
FROM 전공 AS A  
WHERE A.학과이름 = '생활과학과' AND  
      NOT EXISTS (SELECT B.학생번호 FROM 수강 B  
                  WHERE A.학생번호 = B.학생번호)
```



학생번호
202031-354516

02

조인 질의의 이해

- 데이터베이스 언어의 필요
- SQL의 개요
- SQL의 구성

☆ **조인 질의**

?

나이가 30세 이상인 학생의 학생이름과 나이, 그리고 그 학생이 소유한 계좌의 계좌번호, 잔액을 출력하시오.

- ▷ 테이블 간의 관련성을 이용하여 두 개 이상의 테이블에서 데이터를 검색하는 질의 기법
- ▷ ER 모델링 및 정규화 기법으로 여러 테이블로 분리된 정보를 일시적으로 하나의 레코드로 통합
- ▷ 조인 질의의 종류
 - + 내부조인
 - + 외부조인

☆ 내부 조인

- ▷ 두 개 이상의 테이블에서 조인 조건을 만족하는 레코드만 결합하여 출력 결과에 포함시키는 연산
- ▷ 조인 조건은 WHERE 절이 아닌 ON 절에 기록
- ▷ ANSI SQL 표준과 사실상의 표준인 Oracle사가 제안한 조인 형식이 사용



```
SELECT 컬럼1, 컬럼2, ..., 컬럼m,  
      FROM 테이블1 INNER JOIN 테이블2  
      ON 조인 조건1  
      [WHERE 조건]
```

☆ 내부 조인의 사용 1



나이가 30세 이상인 학생의 학생이름과 나이, 그리고 그 학생이 소유한 계좌의 계좌번호, 잔액을 출력하시오.



학생이름	나이	...	학생번호
유관순	118	...	201834-021216
지청천	32	...	201926-880215
안창호	42	...	201931-781109
박은식	61	...	201934-080621
안중근	41	...	201934-790902
손병희	59	...	201978-610408
윤봉길	112	...	202031-816515

학생번호	...	잔액
201831-331215	...	800000
201834-021216	...	600000
201978-610408	...	400000
201931-781109	...	400000
201926-880215	...	300000
201934-790902	...	100000
201934-080621	...	300000
202034-596541	...	1200000
⋮	⋮	⋮

내부 조인의 사용 1

```
SELECT 학생이름, 나이, 학생번호  
FROM 학생  
WHERE 나이 >= 30
```

학생이름	나이	...	학생번호	...	잔액
유관순	118	...	201834-021216	...	600000
지청천	32	...	201926-880215	...	300000
안창호	42	...	201931-781109	...	400000
박은식	61	...	201934-080621	...	300000
안중근	41	...	201934-790902	...	100000
손병희	59	...	201978-610408	...	400000
윤봉길	112	...	202031-816515	...	150000

☆ 내부 조인의 사용 1

```
SELECT 학생.학생이름, 학생.나이,  
       계좌.계좌번호, 계좌.잔액  
FROM 학생 INNER JOIN 계좌  
ON 학생.학생번호 = 계좌.학생번호  
WHERE 학생.나이 >= 30
```

```
SELECT 학생.학생이름, 학생.나이,  
       계좌.계좌번호, 계좌.잔액  
FROM 학생, 계좌  
WHERE 학생.학생번호 = 계좌.학생번호  
      AND 학생.나이 >= 30
```

내부 조인의 사용 2



'컴퓨터과학과' 소속의 교수가 강의하는 과목에 대해 과목별 수강하는 학생수를 과목코드와 함께 출력하시오.

교수	교수번호	교수이름	직위	소속학과	연봉	
과목	과목코드	과목명	학점	선수과목	이수구분	교수번호
수강	과목코드	학생번호	신청시각			

☆ 내부 조인의 사용 2

```
SELECT *  
  FROM 교수 INNER JOIN 과목  
    ON 교수.교수번호 = 과목.교수번호  
      INNER JOIN 수강  
    ON 과목.과목코드 = 수강.과목코드
```

☆ 내부 조인의 사용 2

```
SELECT *  
  FROM 교수 INNER JOIN 과목  
    ON 교수.교수번호 = 과목.교수번호  
      INNER JOIN 수강  
    ON 과목.과목코드 = 수강.과목코드  
 WHERE 교수.소속학과 = '컴퓨터과학과'  
 GROUP BY 과목.과목코드
```

☆ 내부 조인의 사용 2

```
SELECT 과목.과목코드, COUNT(수강.학생번호)
FROM 교수 INNER JOIN 과목
    ON 교수.교수번호 = 과목.교수번호
    INNER JOIN 수강
    ON 과목.과목코드 = 수강.과목코드
WHERE 교수.소속학과 = '컴퓨터과학과'
GROUP BY 과목.과목코드
```


자연 조인

- ▶ 두 개 이상의 테이블을 하나의 테이블로 결합하는 내부 조인과 매우 유사한 기능
- ▶ 두 테이블에 동일한 이름의 컬럼에 대해 값이 같은 레코드를 결합하는 내부 조인



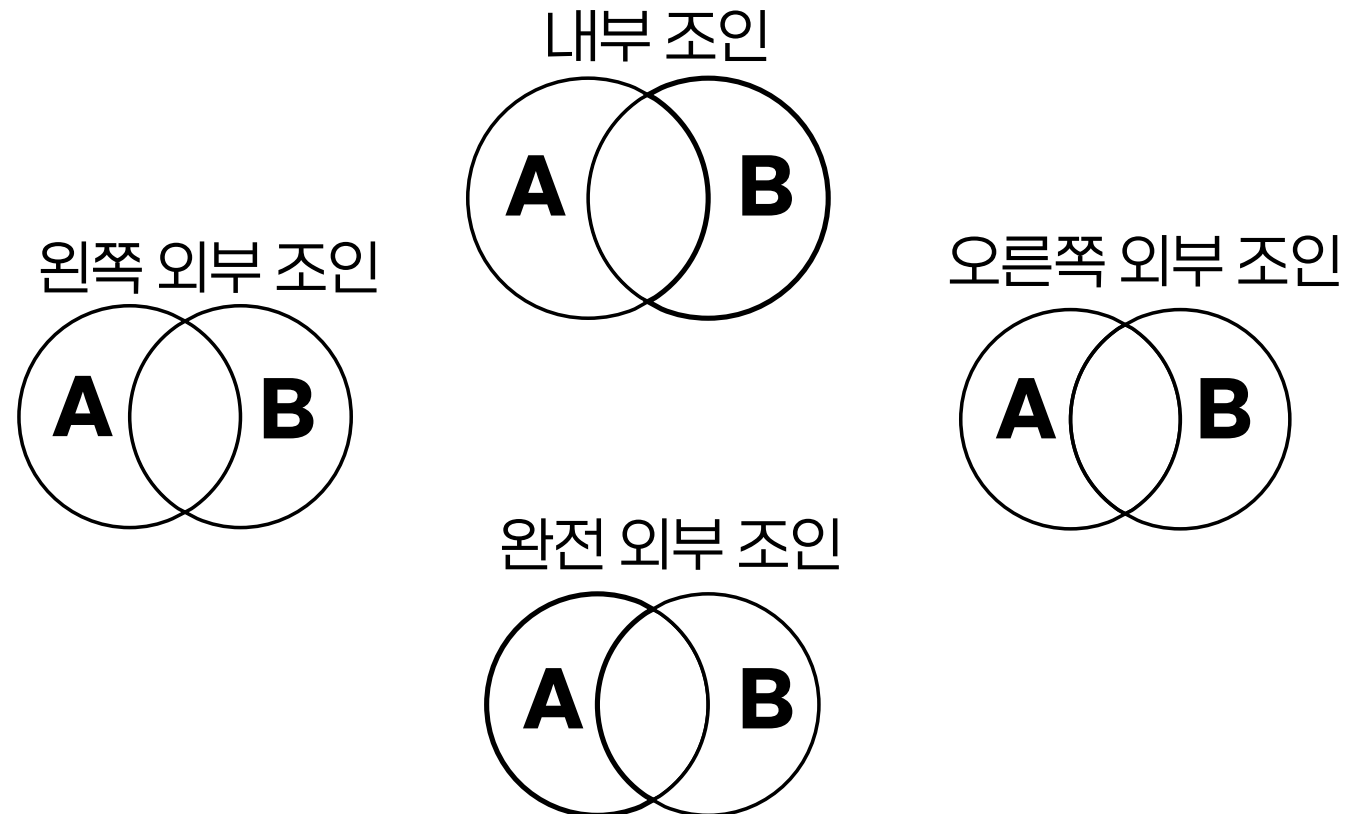
```
SELECT 컬럼1, 컬럼2, ..., 컬럼m,  
       FROM 테이블1 NATURAL JOIN 테이블2  
       [WHERE 조건]
```



외부 조인의 개념

- ▷ 내부 조인(inner join)은 조인조건에 일치하는 레코드만 결합하여 결과를 생성
 - ⊕ 조인 결과에 정보의 손실이 발생
- ▷ 외부 조인은 조인조건에 맞지 않는 레코드도 질의의 결과에 포함시키는 질의
- ▷ 외부 조인의 종류
 - ⊕ 왼쪽 외부 조인(left outer join)
 - ⊕ 오른쪽 외부 조인(right outer join)
 - ⊕ 완전 외부 조인(full outer join)

외부조인의 개념



외부 조인 구문형식



```
SELECT 별칭1.컬럼1, 별칭1.컬럼2, ..., 별칭1.컬럼m,  
        별칭2.컬럼1, 별칭2.컬럼2, ..., 별칭2.컬럼n,  
FROM 테이블1 AS 별칭1  
      LEFT|RIGHT [OUTER] JOIN  
      테이블2 AS 별칭2  
ON 별칭1.컬럼i=별칭2.컬럼j  
[WHERE 절]
```



외부 조인의 사용



학생의 학생번호, 학생이름과 그 학생이 수강신청한 과목의 과목코드, 신청시각을 출력하시오.
단, 수강신청을 하지 않은 학생도 결과에 포함시키고 과목코드를 기준으로 오름차순 정렬한다.

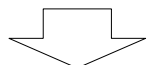
과목코드	학생번호	신청시각
COM11	201934-790902	2019-02-28 08:32:54
COM11	202026-590930	2019-02-20 16:00:21
COM11	202078-080621	2019-02-21 15:21:54
COM12	201831-331215	2019-08-21 23:25:25
COM12	201931-781109	2018-08-02 03:25:16
COM12	201978-610408	2015-02-24 10:25:40
⋮	⋮	⋮

외부 조인의 사용

```
SELECT A.학생번호, A.학생이름, B.과목코드, B.신청시각  
FROM 학생 AS A LEFT OUTER JOIN 수강 AS B  
ON A.학생번호 = B.학생번호  
ORDER BY B.과목코드 ASC
```

과목코드	학생번호	신청시각
COM11	201934-790902	2019-02-28 08:32:54
COM11	202026-590930	2019-02-20 16:00:21
COM11	202078-080621	2019-02-21 15:21:54
COM12	201831-331215	2019-08-21 23:25:25
COM12	201931-781109	2018-08-02 03:25:16
COM12	201978-610408	2015-02-24 10:25:40
⋮	⋮	⋮

외부 조인의 실행과정



학생

학생이름	...	학생번호
강신영	...	202078-080621
안중근	...	201934-790902
조중대	...	202031-354516
⋮		⋮
정용호	...	202034-596541

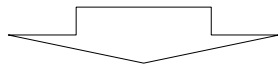
수강

학생번호	...	과목코드
202078-080621	...	COM11
201934-790902	...	COM11
201931-781109	...	COM12
⋮		⋮
202034-596541	...	COM24

강신영	...	202078-080621	...	COM11
안중근	...	201934-790902	...	COM11
조중대	...	202031-354516	...	NULL
⋮		⋮		⋮
정용호	...	202034-596541	...	COM24

외부조인의 사용

```
SELECT A.학생번호, A.학생이름, B.과목코드, B.신청시각  
FROM 학생 AS A LEFT OUTER JOIN 수강 AS B  
ON A.학생번호 = B.학생번호  
ORDER BY B.과목코드 ASC
```



학생번호	학생이름	과목코드	신청시각
202031-354516	조중대	NULL	NULL
201934-080621	박은식	NULL	NULL
201934-790902	안중근	COM11	2019-02-28 08:32:54
202078-080621	강신영	COM11	2019-02-21 15:21:54
202026-590930	정용민	COM11	2019-02-20 16:00:21
201931-781109	안창호	COM12	2018-08-02 03:25:16
⋮	⋮	⋮	⋮

셀프 조인

- ▷ 한 테이블이 자기 자신과 조인되는 질의
- ▷ 동일한 테이블에 대한 조인이므로 반드시 테이블 이름에 대한 별칭이 의무적으로 사용

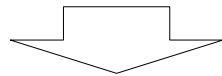


```
SELECT 별칭1.컬럼1, 별칭1.컬럼2, ..., 별칭1.컬럼m,  
       별칭2.컬럼1, 별칭2.컬럼2, ..., 별칭2.컬럼n,  
FROM 테이블1 AS 별칭1  
      INNER|OUTER JOIN 테이블2 AS 별칭2  
ON 조인조건  
[WHERE 절]
```

셀프조인의 사용



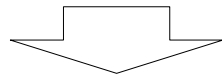
과목의 과목코드, 과목명 그리고 그 과목의 선수과목의 과목코드, 과목명을 모두 출력하시오.
단, 선수과목이 없는 과목도 결과에 포함시킨다.



과목코드	과목명	학점	선수과목	이수구분	교수번호
COM11	컴퓨터의 이해	3		교양	...
COM12	파이썬 프로그래밍 기초	3		교양	...
COM24	자료구조	3	COM12	전공필수	...
COM31	데이터베이스 시스템	3	COM24	전공필수	...
COM34	알고리즘	3	COM24	일반선택	...
COM44	클라우드 컴퓨팅	3		전공필수	...
⋮	⋮	⋮	⋮	⋮	⋮

셀프조인의 사용

```
SELECT B.과목명, B.과목코드
       A.과목명 AS 선수과목명, A.과목코드 AS 선수과목코드
FROM 과목 AS A RIGHT OUTER JOIN 과목 AS B
ON A.과목코드 = B.선수과목
```



과목명	과목코드	선수과목명	선수과목코드
컴퓨터의 이해	COM11		
파이썬 프로그래밍 기초	COM12		
자료구조	COM24	인터넷과 정보사회	COM12
데이터베이스 시스템	COM31	자료구조	COM24
알고리즘	COM34	자료구조	COM24
클라우드 컴퓨팅	COM44		
⋮	⋮	⋮	⋮

03

뷰의 사용

- 뷰의 개념
- 뷰 생성, 수정, 삭제
- 뷰를 이용한 검색 및 수정

뷰의 개념

- ▷ 데이터를 저장하고 있는 하나 이상의 테이블을 유도하여 생성하는 가상의 테이블(virtual table)
 - ⊕ 데이터 독립성: 원본 테이블의 구조가 바뀌어도 뷰를 이용한 작업은 정의만 변경되어 응용 프로그램에 영향이 없음
 - ⊕ 데이터 보안: 사용자에게 원본 테이블의 일부 컬럼에 대한 접근을 허용하여 보안 효과를 향상
 - ⊕ 다양한 구조의 테이블 사용: 사용자의 요구사항에 맞는 테이블의 구조를 제공
 - ⊕ 작업의 단순화: 복잡한 질의문을 뷰로 단순화
 - ⊕ 데이터 무결성: WITH CHECK OPTION을 이용하여 뷰 생성에 위배되는 수정작업을 거부

뷰의 생성

- ▷ 생성되는 뷰의 구조는 SELECT 문의 결과로 결정



```
CREATE VIEW 뷰이름 AS  
    ( SELECT 컬럼1, 컬럼2, ..., 컬럼n  
      FROM 테이블  
      [WHERE 조건] )  
[WITH CHECK OPTION]
```

뷰의 수정 및 삭제

- ▷ 뷰의 수정은 생성과 동일하게 새로운 SELECT 문의 결과로 변경



```
ALTER VIEW 뷰이름(컬럼1, 컬럼2, ..., 컬럼n) AS  
    ( SELECT 컬럼1, 컬럼2, ..., 컬럼n  
      FROM 테이블  
      [WHERE 조건] )
```

- ▷ 뷰의 삭제는 일반적인 데이터베이스 객체 삭제와 동일



```
DROP VIEW 뷰이름
```


뷰 생성의 예

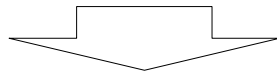
?


컴퓨터과학과 소속의 학생정보와 학과이름 및 이수학점을 출력하는 '컴퓨터과학과_학생' 뷰를 생성하시오.

학생번호	학생이름	성별	생년월일	나이	전화번호	학과이름	이수학점
201831-331215	김마리아	여	1991-06-18	30	010-0000-0002	컴퓨터과학과	39
201934-790902	안중근	남	1979-09-02	42	010-0000-0006	컴퓨터과학과	
202026-590930	정용민	남	2003-05-19	18	010-0000-0012	컴퓨터과학과	137
202031-816515	윤봉길	남	1908-06-21	113	010-0000-0009	컴퓨터과학과	12
202034-596541	정용호	남	2000-01-23	21	010-0000-0008	컴퓨터과학과	117
202078-080621	강신영	남	1991-06-26	30	010-0000-0011	컴퓨터과학과	96

뷰 생성의 예

 **SELECT** 학생.*, 전공.학과이름, 전공.이수학점
FROM 학생 **NATURAL JOIN** 전공
WHERE 전공.학과이름 = '컴퓨터과학과'



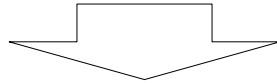
 **CREATE VIEW** 컴퓨터과학과_학생 AS
(**SELECT** 학생.*, 전공.학과이름, 전공.이수학점
FROM 학생 **NATURAL JOIN** 전공
WHERE 전공.학과이름 = '컴퓨터과학과')

뷰를 이용한 데이터 검색

- ▷ 뷰는 가상의 테이블이므로 데이터 조작은 테이블 조작과 동일하게 수행



```
SELECT 컬럼1, ..., 컬럼2 FROM 뷰이름  
WHERE 조건
```



```
SELECT 컬럼1, ..., 컬럼2 FROM 뷰이름  
WHERE 조건 AND 뷰 정의 조건
```



뷰를 이용한 데이터 삽입

- ▷ 뷰에 대한 INSERT 문은 원본 테이블에서 실행
- ▷ INSERT 문 실행이 불가능한 경우
 - + PRIMARY KEY, NOT NULL 등의 제약사항이 위배되는 경우 삽입이 불가능
 - + 원본 테이블에 존재하는 컬럼이지만 뷰에는 없는 컬럼에 삽입하는 경우 실행 불가능
 - + 조인 질의 또는 그룹 질의가 적용된 뷰는 데이터 삽입 및 수정이 불가능
 - + WITH CHECK OPTION이 적용된 뷰는 위배되는 사항은 없지만 뷰에 맞지 않는 조건일 경우 실행 불가능



다음 시간



정규화