

01 관계대수 & 관계해석

※ [문 01~04.] 다음 <<고객>> 릴레이션에 대한 관계대수를 작성하시오.

<<고객>>

고객아이디	이름	나이	등급	직업
hoho	이순신	29	gold	교사
grace	홍길동	24	gold	학생
mango	삼돌이	27	silver	학생
juce	갑순이	31	gold	공무원
orange	강감찬	23	silver	군인

01. 고객 릴레이션에서 등급이 gold인 고객들을 검색하는 관계대수를 작성하시오.

답안 : σ 등급 = 'gold' (고객)

02. 고객 릴레이션에서 등급이 gold이고 나이가 25 이상인 고객들을 검색하는 관계대수를 작성하시오.

답안 : σ 등급 = 'gold' \wedge 나이 \geq 25 (고객)

03. 고객 릴레이션에서 고객아이디와 등급을 가져오는 관계대수를 작성하시오.

답안 : π 고객아이디, 등급 (고객)

04. 고객 릴레이션에서 등급이 gold인 고객의 고객아이디와 등급을 가져오는 관계대수를 작성하시오.

답안 : π 고객아이디, 등급 (σ 등급 = 'gold' (고객))

05. 다음 SQL문장과 동일한 관계대수를 작성하시오.

```
SELECT SNO, NAME
FROM STUDENT
WHERE AGE > 20;
```

답안 : π SNO, NAME (σ AGE>20 (STUDENT))

06. 다음 SQL문장과 동일한 관계대수를 작성하시오.

```
SELECT SNO, NAME
FROM T1, T2
ON T1.SNO = T2.SNO
```

답안 : π SNO, NAME (T1 \bowtie T1.SNO = T2.SNO T2)

07. 다음 관계대수식을 적용한 결과를 작성하시오.

<<직원>>

직원번호	이름	부서
10	김	B20
20	이	A10
30	박	A10
40	최	C30

<<부서>>

부서번호	부서명
A10	기획
B20	인사
C30	총무

<<정책>>

정책번호	정책명	제안자
100	인력양성	40
200	주택자금	20
300	친절교육	10
400	성과금	10
500	신규고용	20

π 이름, 부서명, 정책명 (부서 \bowtie 부서번호 = 부서 (π 정책명, 이름, 부서 (정책 \bowtie 제안자 = 직원번호 직원)))

답안 : 첫 번째로, 정책 릴레이션과 직원 릴레이션을 조인하여 제안자와 직원번호가 같은 정책명, 이름, 부서를 가져온다.

정책명	이름	부서
인력양성	최	C30
주택자금	이	A10
친절교육	김	B20
성과금	김	B20
신규고용	이	A10

두 번째로, 부서 릴레이션과 위의 결과값을 조인하여 이름, 부서명, 정책명을 가져온다.

이름	부서명	정책명
최	총무	인력양성
이	기획	주택자금
김	인사	친절교육
김	인사	성과금
이	기획	신규고용

08. 아래의 R과 S, 두 릴레이션에 대한 $R \div S$ 의 결과와 DIVISION 연산의 정의를 작성하시오.

<<R>>

D1	D2	D3
a	1	A
b	1	A
a	2	A
c	2	B

<<S>>

D2	D3
1	A

답안 : 릴레이션 S의 조건에 만족하는 릴레이션 R에서 튜플을 분리하는 관계대수

D1
a
b

09. 다음의 릴레이션 R1과 R2에 대한 관계대수 $R1 \div R2$ 의 결과 릴레이션을 작성하시오.

<<R1>>

C1	C2
1	A
2	C
1	E
1	B
3	J
4	R
3	B
2	B
5	R
3	A
4	A

<<R2>>

C2
A
B

답안 : C2 릴레이션에 있는 A와 B가 모두 있는 C1의 항목은 1과 3이다.

C1
1
3

10. 관계해석 연산자

구분	기호	설명
연산자	\vee	- OR 연산
	\wedge	- AND 연산
	\neg	- NOT 연산
정량자	\forall	- 모든 가능한 튜플 "For All"
	\exists	- 어떤 튜플 하나라도 존재

02 DDL(Data Definition Language)

※ CREATE

01. 테이블 생성 예제

```
// ORDER INFO 테이블을 생성
CREATE TABLE ORDER_INFO (
    ORDER_ID NUMBER NOT NULL,
    USER_ID NUMBER NOT NULL,
    USER_NM VARCHAR(20) NOT NULL,
    ADDRESS VARCHAR(200) NOT NULL,
    TOT_PRICE NUMBER DEFAULT '0', // 값이 들어오지 않을 경우 기본값 0
    GENDER CHAR(1) NOT NULL,
    ORDER_CI VARCHAR(100) NULL,

    // ORDER_ID 속성에 대해 기본키 지정(개체 무결성 제약조건)
    CONSTRAINTS ORDER_PK PRIMARY KEY (ORDER_ID),
    // ORDER_CI 속성에 대해 고유값 제약 지정
    CONSTRAINTS UNIQUE_CI UNIQUE (ORDER_CI),
    // GENDER 속성에 대해 M, F 값만 받도록 설정(도메인 무결성 제약조건)
    CONSTRAINTS CHK_GENDER CHECK(GENDER IN ('M', 'F') )
);

CREATE TABLE ORDER_ITEM (
    ITEM_ID NUMBER NOT NULL,
    ORDER_ID NUMBER NOT NULL,
    PRODUCT_ID NUMBER NOT NULL,
    PRICE NUMBER DEFAULT '0',

    // ITEM_ID 속성에 대해 기본키 지정(개체 무결성 제약조건)
    PRIMARY KEY(ITEM_ID),

    // ORDER_ID 속성에 대해 외래키 지정 (참조 무결성 제약조건)
    // 대상은 ORDER_INFO 테이블에 ORDER_ID 속성을 지정
    // ORDER_INFO 테이블 ORDER_ID 수정 시 제한속성, 삭제 시 연쇄삭제 지정
    CONSTRAINTS FK_ORDER_ID FOREIGN KEY(ORDER_ID)
        REFERENCES ORDER_INF(ORDER_ID)
        ON UPDATE RESTRICT ON DELETE CASCADE
);
```

02. INDEX 생성 예제

```
// ORDER_INFO테이블에 인덱스 생성
// 인덱스 테이블에 들어갈 속성은 USER_NM 컬럼과 GENDER 컬럼
// USER_NM 은 오름차순, GENDER 는 내림차순으로 인덱스 테이블에 추가됨
CREATE INDEX idx_order ON ORDER_INFO (USER_NM ASC, GENDER DESC);
```

03. VIEW 생성 예제

```
// 성별이 여성인 회원의 회원정보를 가져오는 VIEW 생성
CREATE OR REPLACE VIEW V_ORDER_INFO_F AS
SELECT
    ORDER_ID, USER_ID, USER_NM
FROM ORDER_INFO
WHERE GENDER = 'F'

// 성별이 남성인 회원의 회원정보를 가져오는 VIEW 생성
CREATE OR REPLACE VIEW V_ORDER_INFO_M AS
SELECT
    ORDER_ID, USER_ID, USER_NM
FROM ORDER_INFO
WHERE GENDER = 'M'
```

04. 트리거생성 예제

```
// INSERT 가 된 이후 TB_GOODS 의 nStock 개수와 새롭게 등록된 nStock 의 개수를 더해서 업데이트 해준다.
CREATE TRIGGER TRIGGER_GOODS_STOCK
AFTER INSERT ON TB_GOODS_STOCK FOR EACH ROW
BEGIN
    UPDATE TB_GOODS
    SET
        nStock = nStock + NEW.nStock
    WHERE idx = NEW.p_idx;
END
```

05. 프로시저 생성 예제

```
CREATE OR REPLACE PROCEDURE 프로시저명  
( 변수1 IN 변수타입, 변수2 OUT 변수타입, 변수3 IN OUT 변수타입.... )  
IS  
    변수 처리부  
BEGIN  
    처리내용  
EXCEPTION  
    예외처리부  
END;
```

06. 파티션 생성 예제

```
-- 범위 분할 파티션  
CREATE TABLE TB_USER (  
    id INT,  
    year INT  
)  
PARTITION BY RANGE (year) (  
    PARTITION U1 VALUES LESS THAN (2000),  
    PARTITION U2 VALUES LESS THAN (2010),  
    PARTITION U3 VALUES LESS THAN (2020)  
);  
  
-- 목록 분할 파티션  
CREATE TABLE TB_STUDENT (  
    id INT,  
    grade INT  
)  
PARTITION BY LIST (grade) (  
    PARTITION high_grade VALUES IN (1, 2, 3),  
    PARTITION low_grade VALUES IN (4, 5, 6)  
);
```

※ DROP

01. 테이블 삭제

```
// 회원 테이블 삭제  
DROP TABLE 회원;
```

02. 뷰 삭제

```
// USER_VIEW 삭제  
DROP VIEW USER_VIEW;
```

03. 인덱스 삭제

```
// USER_INDEX 삭제  
DROP INDEX USER_INDEX;
```

※ ALTER

01. 속성 추가

```
// 회원 테이블에 ADDR 속성 추가  
ALTER TABLE 회원 ADD ADDR VARCHAR(200) null;
```

02. 속성 변경

```
// 회원 테이블에 AGE 속성 INT 로 변경  
ALTER TABLE 회원 MODIFY AGE INT(11);
```

03. 속성 삭제

```
// 회원 테이블에 AGE 속성 삭제  
ALTER TABLE 회원 DROP COLUMN AGE;
```


04. INDEX 변경

```
// 회원명 INDEX를 성명으로 변경
ALTER INDEX 회원명 RENAME TO 성명
// INDEX의 속도가 저하되거나 깨졌을 경우 INDEX 재구성
ALTER INDEX 회원명 REBUILD
// INDEX 비활성화
ALTER INDEX 회원명 UNUSABLE
```

※ TRUNCATE

01. 테이블 내용 비우기

```
// 회원 테이블 내용 삭제
TRUNCATE [TABLE] 회원;
```

03 DCL(Data Control Language)

※ GRANT & REVOKE

01. DBA가 사용자 Park에게 테이블A의 데이터를 갱신할 수 있는 시스템 권한을 부여하고자 하는 SQL문을 작성하고자 한다. 다음에 주어진 SQL문의 빈칸에 알맞게 채우시오.

GRANT ㉠ ㉡ 테이블A To Park

답안 : ㉠ UPDATE

㉡ ON

02. STUDENT에 대한 권한을 부여하는 ㉠과 회수하는 ㉡을 SQL로 쓰시오.

㉠ 테이블 student에 대한 SELECT, INSERT 권한을 Kim과 Lee에게 부여한다.

㉡ 테이블 student에 대한 SELECT, INSERT 권한을 Lee로부터 회수한다.

답안 : ㉠ GRANT SELECT, INSERT ON student TO Kim, Lee;

㉡ REVOKE SELECT, INSERT ON student FROM Lee;

03. 사용자 X1에게 department 테이블에 대한 검색 연산을 회수하는 명령을 쓰시오.

답안 : REVOKE SELECT ON department FROM X1;

04. 관계 데이터베이스에서 테이블 조작을 위한 권한부여 명령을 다음과 같이 수행하였다.

명령을 수행한 후의 테이블에 대한 권한을 서술하시오.

```
[DBA] GRANT SELECT ON T1 TO USER1 WITH GRANT OPTION;
[USER1] GRANT SELECT ON T1 TO USER2 WITH GRANT OPTION;
[USER2] GRANT SELECT ON T1 TO USER3;
[USER1] REVOKE SELECT ON T1 FROM USER2 CASCADE;
```

답안 :

- ① DBA가 USER1에게 T1 테이블에 대한 SELECT 권한을 주면서 다른 사용자에게 권한을 부여할 수 있도록 설정
- ② USER1이 USER2에게 T1 테이블에 대한 SELECT 권한을 주면서 다른 사용자에게 권한을 부여할 수 있도록 설정
- ③ USER2가 USER3에게 T1 테이블에 대한 SELECT 권한을 부여
- ④ USER1이 USER2에게 부여한 SELECT 권한을 회수하면서, USER2가 USER3에게 부여한 권한도 같이 회수

04 DML(Data Manipulation Language)

※ INSERT

01. EMP 테이블 USER_NO, USER_NAME 컬럼에 각각 '001', 'USER1'을 삽입하는 SQL문을 작성하시오.

답안 : INSERT INTO EMP (USER_NO, USER_NAME) VALUES ('001', 'USER1');

02. 다음과 같은 SQL이 실행되었을 때 CSTUDENT에 삽입되지 않는 것을 고르시오.

```
CREATE TABLE STUDENT (  
    SNO INT NOT NULL,  
    NAME VARCHAR(10),  
    YEAR INT,  
    DEPT VARCHAR(10),  
    PRIMARY KEY (SNO)  
);  
INSERT INTO STUDENT VALUES (1001, 'KIM', 4, 'COMPUTER');  
INSERT INTO STUDENT VALUES (1002, 'LEE', 4, 'COMPUTER');  
  
CREATE VIEW CSTUDENT (SNO, NAME, YEAR)  
AS  
    SELECT SNO, NAME, YEAR  
    FROM STUDENT  
    WHERE DEPT = 'COMPUTER'
```

- ① INSERT INTO CSTUDENT VALUES (1003, 'PARK', 3);
- ② INSERT INTO CSTUDENT (SNO, NAME) VALUES (1003, 'PARK');
- ③ INSERT INTO CSTUDENT (SNO, YEAR) VALUES (1003, 3);
- ④ INSERT INTO CSTUDENT (NAME, YEAR) VALUES ('PARK', 3);
- ⑤ INSERT INTO CSTUDENT (SNO, NAME) VALUES (1002, 'CHO');

답안 : ④, ⑤

※ UPDATE

01. EMP 테이블 USER_NO가 1000인 고객의 USER_NAME을 '이흥직'으로 변경하는 SQL문을 작성하시오.

답안 : UPDATE EMP

SET

USER_NAME = '이흥직'

WHERE USER_NO = '1000'

02. STUDENT 테이블에서 GRADE를 1씩 증가하는 SQL문을 작성하시오.

답안 : UPDATE STUDENT

SET

GRADE = GRADE + 1

03. STUDENT 테이블에서 SNO가 1000인 학생의 GRADE를 1씩 증가하는 SQL문을 작성하시오.

답안 : UPDATE STUDENT

SET

GRADE = GRADE + 1

WHERE SNO = '1000'

※ DELETE

01. EMP 테이블 USER_NO가 1000인 고객을 삭제하는 SQL문을 작성하시오.

답안 : DELETE FROM EMP WHERE USER_NO = '1000'

02. STUDENT 테이블에서 GRADE가 6이상인 행들을 삭제하는 SQL문을 작성하시오.

답안 : DELETE FROM STUDENT WHERE GRADE >= '6'

※ SELECT

01. 다음 SQL문에서 사용된 BETWEEN 연산의 의미와 동일하게 AND를 이용하여 SQL문을 작성하시오.

```
SELECT *  
FROM 성적  
WHERE (점수 BETWEEN 90 AND 95)  
AND 학과 = '컴퓨터공학과'
```

답안 : 점수 ≥ 90 AND 점수 ≤ 95

02. 직원 테이블에서 10,000,000원 이상의 급여를 받는 직원의 이름과 급여를 검색하는 SQL문을 작성하시오.

답안 : SELECT 이름, 급여 FROM 직원
WHERE 급여 \geq '10000000'

03. 학적 테이블에서 전화번호가 NULL값인 학생의 학번과 학생명을 검색하는 SQL문을 작성하시오.

답안 : SELECT 학번, 학생명 FROM 학적
WHERE 전화번호 IS NULL

04. 학생 테이블에서 학과의 중복을 제거하고 검색하는 SQL문을 작성하시오.

답안 : SELECT DISTINCT 학과 FROM 학생

05. 다음 R1과 R2의 테이블에서 아래의 실행 결과를 얻기 위한 SQL문을 작성하시오.

[R1] 테이블

학번	이름	학년	학과	주소
1000	홍길동	1	컴퓨터공학	서울
2000	김철수	1	전기공학	경기
3000	강남길	2	전자공학	경기
4000	오말자	2	컴퓨터공학	경기
5000	장미화	3	전자공학	서울

[R2] 테이블

학번	과목번호	과목이름	학점	점수
1000	C100	컴퓨터구조	A	91
2000	C200	데이터베이스	A+	99
3000	C100	컴퓨터구조	B+	89
3000	C200	데이터베이스	B	85
4000	C200	데이터베이스	A	93
4000	C300	운영체제	B+	88
5000	C300	운영체제	B	82

[실행결과]

과목번호	과목이름
C100	컴퓨터구조
C200	데이터베이스

답안 : SELECT 과목번호, 과목이름 FROM R1, R2
 WHERE R1.학번 = R2. 학번
 AND R1.학과 = '전자공학'
 AND R1.이름 = '강남길';

06. 두 릴레이션 R1(A, B), R2(B, C)를 왼쪽 외부조인(Left Outer Join)을 한 결과를 쓰시오.
(이때, 중복된 열인 B는 한 열로 표시하시오.)

<<R1>>

A	B
1	2
4	2
7	8

<<R2>>

B	C
2	3
2	4
6	7

답안 : R1 릴레이션 첫 번째 행과 두 번째 행은 R2의 두 행씩 연결이 되므로, 각각 두 개의 행으로, 총 4개 행이 만들어진다.

R1 릴레이션의 마지막 행의 B 값은 8이고, R2에 8을 가진 데이터가 없기 때문에, R1 릴레이션의 값만 표현하고, C 열은 NULL값을 가지게 된다.

A	B	C
1	2	3
1	2	4
4	2	3
4	2	4
7	8	(NULL)

07. 두 릴레이션 R1(A, B), R2(B, C)를 오른쪽 외부조인(Right Outer Join)을 한 결과를 쓰시오.
(이때, 중복된 열인 B는 한 열로 표시하시오.)

<<R1>>

A	B
1	2
4	2
7	8

<<R2>>

B	C
2	3
2	4
6	7

답안 : R2 릴레이션 첫 번째 행과 두 번째 행은 R1의 두 행씩 연결이 되므로, 각각 두 개의 행으로, 총 4개 행이 만들어진다.

R2 릴레이션의 마지막 행의 B 값은 6이고, R1에 6을 가진 데이터가 없기 때문에, R2 릴레이션의 값만 표현하고, A 열은 NULL값을 가지게 된다.

A	B	C
1	2	3
1	2	4
4	2	3
4	2	4
(NULL)	6	7

08. 두 릴레이션 R1(A, B), R2(B, C)를 전체 외부조인(FULL Outer Join)을 한 결과를 쓰시오.
(이때, 중복된 열인 C는 한 열로 표시하시오.)

<<R1>>

A	B
1	2
4	2
7	8

<<R2>>

B	C
2	3
2	4
6	7

답안 : R1 릴레이션 첫 번째 행과 두 번째 행은 R2의 두 행씩 연결이 되므로, 각각 두 개의 행으로, 총 4개 행이 만들어진다.

R1 릴레이션의 마지막 행의 B 값은 8이고, R2에 8을 가진 데이터가 없기 때문에, R1 릴레이션의 값만 표현하고, C 열은 NULL값을 가지게 된다.

R2 릴레이션의 마지막 행의 B 값은 6이고, R1에 6을 가진 데이터가 없기 때문에, R2 릴레이션의 값만 표현하고, A 열은 NULL값을 가지게 된다.

A	B	C
1	2	3
1	2	4
4	2	3
4	2	4
(NULL)	6	7
7	8	(NULL)

09. 두 릴레이션 T1, T2를 이용하여 SQL을 실행하였을 때, 나타나는 결과를 쓰시오.

```
SELECT COUNT(*) AS CNT FROM T1 CROSS JOIN T2
WHERE T1.NAME LIKE T2.RULE
```

[T1]

NAME
ALLEN
SCOTT
SMITH

[T2]

RULE
S%
%T%

답안 : CROSS JOIN을 이용하게 되면, 모든 교차 조합을 리턴하게 된다.

조건절과 COUNT를 하지 않은 결과는 아래와 같다.

NAME	RULE
ALLEN	S%
ALLEN	%T%
SCOTT	S%
SCOTT	%T%
SMITH	S%
SMITH	%T%

조건에서 LIKE 연산을 사용하게 되면 ALLEN 은 S로 시작하지도 않고, T가 포함되지 않기 때문에 2개 행을 제외한 4개가 출력된다.

CNT
4

10. 두 릴레이션 R1(A, B), R2(B, C)를 CROSS JOIN 한 결과를 쓰시오.

<<R1>>		<<R2>>	
A	B	B	C
1	2	2	3
4	2	2	4
		6	7

답안 : CROSS JOIN을 이용하게 되면 R1과 R2의 모든 조합을 출력하게 된다.

A	B	B	C
1	2	2	3
1	2	2	4
1	2	6	7
4	2	2	3
4	2	2	4
4	2	6	7

11. 결과 값이 아래와 같을 때 SQL문을 작성하시오. (단, WHERE 조건에 LIKE연산자를 사용해야 합니다.)

[공급자] 테이블		
공급자 번호	공급자명	위치
16	대신공업사	수원
27	삼진사	서울
39	삼양사	인천
62	진아공업사	대전
70	신촌상사	서울

[결과]		
공급자 번호	공급자명	위치
16	대신공업사	수원
70	신촌상사	서울

답안 : SELECT * FROM 공급자
WHERE 공급자명 LIKE '%신%';

12. COMPANY 테이블에서 소재지가 '서울', '수원'인 회사의 NAME과 ADDR을 검색하는 SQL문을 작성하시오.

답안 : SELECT NAME, ADDR FROM COMPANY
WHERE 소재지 IN ('서울', '수원')

13. 다음 중 주어진 [학생] 테이블을 참조하여 아래의 SQL문을 실행한 결과를 작성하시오.

```
SELECT AVG(나이) AS 평균 FROM 학생
WHERE 전공 NOT IN ('수학', '회계');
```

[학생]

학번	전공	학년	나이
100	국사	4	21
150	회계	2	19
200	수학	3	30
250	국사	3	31
300	회계	4	25
350	수학	2	19
400	국사	1	23

답안 : 수학과 회계를 전공한 학생을 제외하면, 100, 250, 400 세 명의 학생이다.

세 학생의 나이를 더하면 $21+31+23=75$ 이고, 평균을 구하게 되면 25이다.

평균
25

14. 아래의 SQL문을 실행한 결과를 작성하시오.

```

SELECT 이름
FROM R1
WHERE 학번 IN
(
    SELECT 학번
    FROM R2
    WHERE 과목번호 = 'C100'
);

```

[R1 테이블]

학번	이름	학년	학과	주소
1000	홍길동	4	컴퓨터	서울
2000	김철수	3	전기	경기
3000	강남길	1	컴퓨터	경기
4000	오말자	4	컴퓨터	경기
5000	장미화	2	전자	서울

[R2 테이블]

학번	과목번호	성적	점수
1000	C100	A	91
1000	C200	A	94
2000	C300	B	85
3000	C400	A	90
3000	C500	C	75
3000	C100	A	90
4000	C400	A	95
4000	C500	A	91
4000	C100	B	80
4000	C200	C	74
5000	C400	B	85

답안 : 서브쿼리에서 가져온 학번은, 1000, 3000, 4000 이다.

해당 학번을 가진 학생의 이름은 홍길동, 강남길, 오말자이다.

이름
홍길동
강남길
오말자

15. 상품 테이블에서 B 제조사 단가보다 하나라도 높은 단가를 가진 제품을 모두 출력하는 SQL문을 작성하시오.

[상품] 테이블

제품번호	단가	제조사
100	1000	A
200	1500	B
300	3000	C
400	900	D
500	2000	B
600	1000	C

[결과]

제품번호	단가	제조사
300	3000	C
500	2000	B

답안 : SELECT * FROM 상품
 WHERE 단가 > ANY (
 SELECT 단가 FROM 상품
 WHERE 제조사 = 'B'
) ;

16. 상품 테이블에서 B 제조사 단가보다 모두 높은 단가를 가진 제품을 모두 출력하는 SQL문을 작성하시오.

[상품] 테이블

제품번호	단가	제조사
100	1000	A
200	1500	B
300	3000	C
400	900	D
500	2000	B
600	1000	C

[결과]

제품번호	단가	제조사
300	3000	C

답안 : SELECT * FROM 상품
 WHERE 단가 > ALL (
 SELECT 단가 FROM 상품
 WHERE 제조사 = 'B'
);

17. '갑' 테이블의 속성 A가 1, 2, 3, 4, 5의 도메인을 가지고 있고, '을' 테이블의 속성 A가 0, 2, 3, 4, 6의 도메인을 가지고 있다고 가정할 때 다음 SQL구문의 실행 결과는?

SELECT A FROM 갑 UNION SELECT A FROM 을;

답안 : 0, 1, 2, 3, 4, 5, 6

18. 다음 질의문 실행의 결과는?

[도서] 테이블

책번호	책명
1111	운영체제
2222	세계지도
3333	생활영어

[도서가격] 테이블

책번호	가격
1111	15000
2222	23000
3333	7000
4444	5000

```
SELECT 가격 FROM 도서가격
WHERE 책번호 =(SELECT 책번호 FROM 도서 WHERE 책명='운영체제');
```

답안 : 15000

19. 다음 SQL문의 실행 결과를 쓰시오.

[학생] 테이블

학번	이름	학년	학과	주소
1000	김철수	1	전산	서울
2000	고영준	1	전기	경기
3000	유진호	2	전자	경기
4000	김영진	2	전산	경기
5000	정현영	3	전자	서울

[성적] 테이블

학번	과목번호	과목이름	학점	점수
1000	A100	자료구조	A	91
2000	A200	DB	A+	99
3000	A100	자료구조	B+	88
3000	A200	DB	B	85
4000	A200	DB	A	94
4000	A300	운영체제	B+	89
5000	A300	운영체제	B	88

```

SELECT 과목이름
FROM 성적
WHERE EXISTS (
    SELECT 학번
    FROM 학생
    WHERE 학생.학번 = 성적.학번
    AND 학생.학과 IN ('전산', '전기')
    AND 학생.주소 = '경기'
);

```

답안 :

과목이름
DB
DB
운영체제

20. 다음의 성적 테이블에서 성명과 학생의 평균점수를 구하는 SQL문을 작성하시오.

성명	과목	점수
홍길동	국어	80
홍길동	영어	68
홍길동	수학	97
강감찬	국어	58
강감찬	영어	97
강감찬	수학	65

답안 : SELECT 성명, AVG(점수) FROM 성적 GROUP BY 성명;

21. 다음의 student 테이블을 이용하여 아래의 SQL을 수행하였을 때 결과를 쓰시오.

id	name	grade	subject	score
2017001	Ryu	2	math	60
2017002	Cho	1	kor	80
2019006	Kim	1	kor	55
2018002	Yang	3	eng	85
2018004	Park	2	math	45
2016003	Choi	3	eng	55
2016003	Kang	3	eng	60

```
SELECT count(*) FROM student
GROUP BY subject
HAVING count(*) > 2;
```

답안 : 3

22. 다음 EMP 테이블에 ㉠~㉤의 SQL문을 차례대로 모두 실행한 최종 결과를 쓰시오.

[EMP] 테이블

NAME	DEPT	SALARY
김직원	1	200
이직원	2	100
박직원	2	300

- ㉠ INSERT INTO EMP VALUES ('정직원', 2, 200);
 ㉡ UPDATE EMP SET DEPT = 1 WHERE NAME LIKE '박%';
 ㉢ INSERT INTO EMP VALUES ('최직원', 3, 400);
 ㉤ SELECT DEPT, AVG(SALARY) AS ASAL
 FROM EMP
 GROUP BY DEPT
 HAVING COUNT(*) >= 2
 ORDER BY DEPT ASC;

DEPT	ASAL
1	250
2	150

답안 :

23. [평균성적] 테이블에서 '평균' 필드 값이 90 이상인 학생들을 검색하여 '학년' 필드를 기준으로 내림차순, '반' 필드를 기준으로 오름차순 정렬하여 표시하고자 한다. 다음 중 아래 SQL문의 각 괄호를 채워 넣으시오.

SELECT 학년, 반, 이름 FROM 평균성적 WHERE 평균 >= 90 (㉠) 학년 (㉡) 반 (㉢);

답안 : ㉠ ORDER BY

- ㉡ DESC
 ㉢ ASC

이 자료는 대한민국 저작권법의 보호를 받습니다.

작성된 모든 내용의 권리는 작성자에게 있으며, 작성자의 동의 없는 사용이 금지됩니다.

본 자료의 일부 혹은 전체 내용을 무단으로 복제/배포하거나 2차적 저작물로 재편집하는 경우,

5년 이하의 징역 또는 5천만 원 이하의 벌금과 민사상 손해배상을 청구합니다.