

Chapter

00

# 연습문제 해답

---



### 01. 데이터 베이스와 DBMS

- 데이터 베이스의 목표는 무엇인지 서술해 보세요.

#### 데이터 베이스의 목표

지속적인 데이터의 관리 및 보호가 가능해야 합니다.  
안전성을 보장할 수 있어야 합니다.  
무결성을 보장 받을 수 있어야 합니다.

- 데이터 베이스의 특징은 무엇인지, 3가지 이상 서술해 보세요.

#### 데이터 베이스의 특징

실시간 접근이 가능해야 합니다.  
동적인 변화에 유연하게 대처 가능해야 합니다.  
모든 사용자에게 동시에 공유 가능(public)해야 합니다.  
데이터들은 실제 데이터에 의한 참조가 가능합니다.

- 주요 SQL 구문을 보고 적절한 이름을 작성해 보세요.

구문	이름
DDL	데이터 정의어
DML	데이터 조작어
DCL	데이터 제어문
DQL	데이터 질의어
TCL	트랜잭션

## 03. SQL 구문

- 다음 각 구문에서 사용할 수 있는 명령어(문법 요소)들을 열거해 보세요.

구문	문법 요소
DQL	select
DML	insert, update, delete
DDL	create, alter, drop
DCL	grant, revoke
TCL	commit, rollback

## 04. 사용자 관리

- 다음 그림과 같이 신규 사용자 계정들을 생성해 보세요. 사용자들의 계정 정보는 다음과 같습니다.

사용자	아이디	비밀 번호
곰돌이	gomdori	omygod
순돌이	sundori	abc123

```
create user gomdori identified by omygod account unlock ;
alter user gomdori default tablespace users quota unlimited on users;
grant connect, resource to gomdori ;

create user sundori identified by abc123 account unlock ;
alter user sundori default tablespace users quota unlimited on users;
grant connect, resource to sundori ;
```

## 05. 데이터 정의어(DDL)

- 다음 빈칸에 알맞은 데이터 타입을 작성해 보세요.

데이터 타입	설명
number	숫자형 데이터 <정수 / 실수 가능>
varchar2	가변 길이 문자열 타입. 최대 4000 바이트 까지 가능.
date	날짜 데이터 타입. 시스템 설정에 따라 출력 형태가 다르다.

● 다음 빈칸에 알맞은 용어를 작성해 보세요.

(기본 키)는 반드시 입력(not null)이 되어야 하고, 절대로 다른 행(row)과 중복(unique)이 되어서는 안되는 컬럼을 의미합니다.

create table 구문은 (테이블)을 생성하고자 할 때 사용하는 구문입니다.

insert ... into 구문은 테이블에 (행)을 추가 하고자 할 때 사용하는 구문입니다.

● 다음 빈칸에 알맞은 키워드(keyword)를 작성해 보세요.

기능	명령어
컬럼 추가하기	alter table 테이블_이름 add(컬럼_이름 데이터_타입);
컬럼 구조 변경하기	alter table 테이블_이름 modify(컬럼_이름 데이터_타입);
컬럼 이름 변경하기	alter table 테이블_이름 rename column 이전_컬럼_이름 to 새로운_컬럼_이름 ;
컬럼 제거하기	alter table 테이블_이름 drop column ( 컬럼_이름 );
테이블 이름 변경하기	rename 이전_테이블_이름 to 신규_ 테이블_이름 ;

● '곰돌이' 사용자를 이용하여 다음과 같은 테이블을 생성하고, 데이터를 추가하세요.

'Books' 테이블의 내용은 다음과 같습니다.

컬럼	데이터 타입	길이	기본 값	널 허용	코멘트
bid	number			no	서적의 아이디입니다.
bname	varchar2	40		yes	서적의 이름입니다.
publisher	varchar2	40		yes	출판사의 이름입니다.
price	number			yes	서적의 단가입니다.
inputdate	date		sysdate	yes	서적의 입고 일자입니다.

```
-- drop table books purge ;
create table books (
    bid varchar2(30) primary key,
    bname varchar2(40),
    publisher varchar2(40),
    price number,
    inputdate date default sysdate
);

insert into books values('java', '자바 프로그래밍', '대한 출판사', 7000, '70/08/28');
insert into books values('jsp', 'jsp 프로그래밍', '대한 Books', 8000, '80/08/15');
insert into books values('python', '파이썬 프로그래밍', '민국 미디어', 9000, '90/07/17');

commit ;
select * from books ;
```

	BID	BNAME	PUBLISHER	PRICE	INPUTDATE
1	java	자바 프로그래밍	대한 출판사	7000	70/08/28
2	jsp	jsp 프로그래밍	대한 Books	8000	80/08/15
3	python	파이썬 프로그래밍	민국 미디어	9000	90/07/17

## 요구 사항

기본 값이 50인 적립 포인트 숫자형 컬럼(mpoint)을 추가해보세요.

다음 데이터가 추가 되도록 구조를 변경해 주세요.  
insert into books values(8, '동해물과 백두산이 마르고 닳도록 ...이 책은 엄청나게 재미있는 책입니다.  
자바 프로그래밍', '대한 출판사', 7000, sysdate, 10);

컬럼 inputdate을 regdate으로 이름 변경해 주세요.  
price 컬럼을 제거해 보세요.  
books 테이블을 oldbook으로 변경해 보세요.  
내가 보유한 테이블 목록을 조회해 보세요.  
oldbook 테이블을 영구 삭제하세요.

## 정답

-- 기본 값이 50인 적립 포인트 숫자형 컬럼(mpoint)을 추가해보세요.

```
alter table books add (mpoint number default 50) ;
```

-- 다음 데이터가 추가되도록 구조를 변경해 주세요.

```
insert into books values('nation', '동해물과 백두산이 마르고 닳도록 ...이 책은 엄청나게 재미있는  
책입니다. 자바 프로그래밍', '대한 출판사', 7000, sysdate, 10);
```

```
alter table books modify(bname varchar2(255));
```

```
insert into books values('nation', '동해물과 백두산이 마르고 닳도록 ...이 책은 엄청나게 재미있는  
책입니다. 자바 프로그래밍', '대한 출판사', 7000, sysdate, 10);
```

-- 컬럼 inputdate을 regdate으로 이름 변경해 주세요.

```
alter table books rename column inputdate to regdate ;
```

-- price 컬럼을 제거해 보세요.

```
alter table books drop column price ;
```

-- books 테이블을 oldbook으로 변경해 보세요.

```
rename books to oldbook ;
```

-- 내가 보유한 테이블 목록을 조회해 보세요.

```
select * from tab ;
```

-- oldbook 테이블을 영구 삭제하세요.

```
drop table oldbook purge ;
```

- '곰돌이' 사용자에게 고객 테이블 'customers'을 다음과 같이 생성하세요.

❖ COLUMN_NAME	❖ DATA_TYPE	❖ NULLABLE	❖ DATA_DEFAULT	❖ COLUMN_ID	❖ COMMENTS
1 ID	VARCHAR2 (20 BYTE)	No	(null)	1 (null)	
2 NAME	VARCHAR2 (40 BYTE)	Yes	(null)	2 (null)	
3 ADDRESS	VARCHAR2 (50 BYTE)	Yes	(null)	3 (null)	
4 PHONE	VARCHAR2 (20 BYTE)	Yes	(null)	4 (null)	
5 MANAGER	VARCHAR2 (20 BYTE)	Yes	(null)	5 (null)	
6 SALARY	NUMBER	Yes	0	6 (null)	

-- 고객 테이블을 다음과 같이 만들어 보세요.

-- drop table customers purge ;

```
create table customers (
    id varchar2(20) primary key,
    name varchar2(40),
    address varchar2(50),
    phone varchar2(20),
    manager varchar2(20),
    salary number default 0
);
```

- '곰돌이' 사용자에게 주문 테이블 'orders'을 다음과 같이 생성하세요.

❖ COLUMN_NAME	❖ DATA_TYPE	❖ NULLABLE	❖ DATA_DEFAULT	❖ COLUMN_ID	❖ COMMENTS
1 OID	NUMBER (2, 0)	No	(null)	1 (null)	
2 CID	VARCHAR2 (20 BYTE)	Yes	(null)	2 (null)	
3 BID	VARCHAR2 (30 BYTE)	Yes	(null)	3 (null)	
4 SALEPRICE	NUMBER (8, 0)	Yes	(null)	4 (null)	
5 SALEDATE	DATE	Yes	(null)	5 (null)	

-- 주문 테이블을 다음과 같이 만들어 보세요.

-- drop table orders purge ;

```
create table orders (
    oid number(2) primary key,
    cid varchar2(20),
    bid varchar2(30),
    saleprice number(8) ,
    saledate date
);
```

## 06. 데이터 조작어(DML)

- 다음 빈칸에 DML과 관련된 명령어를 작성해 보세요.

명령어	의미
insert	테이블에 새로운 행을 추가합니다.
update	이미 들어 있는 테이블의 행에 대한 정보를 수정합니다.
delete	테이블의 특정 행 정보를 삭제합니다.

- 다음의 간략한 의미를 보고 이에 적절한 연산자 기호를 작성해 보세요.

연산자	의미
=	같습니다.
◇	같지 않습니다.
>	좌측이 큼니다.
>=	좌측이 크거나 같습니다.
<	좌측이 작습니다.
<=	좌측이 작거나 같습니다.

- '곰돌이' 사용자에서 다음 테이블을 생성하도록 합니다.

COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1 BID	VARCHAR2 (30 BYTE)	No	(null)	1 (null)	
2 BNAME	VARCHAR2 (40 BYTE)	Yes	(null)	2 (null)	
3 PUBLISHER	VARCHAR2 (40 BYTE)	Yes	(null)	3 (null)	
4 PRICE	NUMBER	Yes	(null)	4 (null)	
5 INPUTDATE	DATE	Yes	sysdate	5 (null)	

```
-- drop table books purge ;
create table books (
    bid varchar2(30) primary key,
    bname varchar2(40),
    publisher varchar2(40),
    price number,
    inputdate date default sysdate
);
```

- '곰돌이' 사용자의 books 테이블에 데이터를 추가하도록 해주세요.

```
insert into books values('java', '자바 프로그래밍', '대한 출판사', 7000, '70/08/28');
insert into books values('jsp', 'jsp 프로그래밍', '대한 Books', 8000, '80/08/15');
insert into books values('python', '파이썬 프로그래밍', '민국 미디어', 9000, '90/07/17');
insert into books values('r', 'R 프로그래밍', '민국 미디어', 35000, '75/11/11');
insert into books values('database', '데이터 베이스 개론', '대한 출판사', 8000, '78/03/01');
insert into books values('mybatis', '마이바티스', '대한 출판사', 7500, '45/08/15');
insert into books values('spring', '스프링 Framework', '소셜 미디어', 20000, '50/06/25');
insert into books values('xml', 'xml 프로그래밍', '소셜 미디어', 13000, '70/06/06');
insert into books values('story', '프로그래밍 이야기', '지한 출판사', 6000, '70/09/09');
insert into books values('timeseries', '시계열 예측', '지한 출판사', 13000, '50/06/25');

commit ;

select * from books ;
```

BID	BNAME	PUBLISHER	PRICE	INPUTDATE
1 java	자바 프로그래밍	대한 출판사	7000	70/08/28
2 jsp	jsp 프로그래밍	대한 Books	8000	80/08/15
3 python	파이썬 프로그래밍	민국 미디어	9000	90/07/17
4 r	R 프로그래밍	민국 미디어	35000	75/11/11
5 database	데이터 베이스 개론	대한 출판사	8000	78/03/01
6 mybatis	마이바티스	대한 출판사	7500	45/08/15
7 spring	스프링 Framework	소셜 미디어	20000	50/06/25
8 xml	xml 프로그래밍	소셜 미디어	13000	70/06/06
9 story	프로그래밍 이야기	지한 출판사	6000	70/09/09
10 timeseries	시계열 예측	지한 출판사	13000	50/06/25

- 모든 책들의 단가를 1000원으로 변경해 보세요.

```
update books set price = 1000 ;
select * from books ;
```

BID	BNAME	PUBLISHER	PRICE	INPUTDATE
1 java	자바 프로그래밍	대한 출판사	1000	70/08/28
2 jsp	jsp 프로그래밍	대한 Books	1000	80/08/15
3 python	파이썬 프로그래밍	민국 미디어	1000	90/07/17
4 r	R 프로그래밍	민국 미디어	1000	75/11/11
5 database	데이터 베이스 개론	대한 출판사	1000	78/03/01
6 mybatis	마이바티스	대한 출판사	1000	45/08/15
7 spring	스프링 Framework	소셜 미디어	1000	50/06/25
8 xml	xml 프로그래밍	소셜 미디어	1000	70/06/06
9 story	프로그래밍 이야기	지한 출판사	1000	70/09/09
10 timeseries	시계열 예측	지한 출판사	1000	50/06/25

- 모든 출판사의 이름을 '가나 출판사'로 변경해 보세요.

```
update books set publisher = '가나 출판사' ;
select * from books ;
```



BID	BNAME	PUBLISHER	PRICE	INPUTDATE
1 java	자바 프로그래밍	가나 출판사	1000	70/08/28
2 jsp	jsp 프로그래밍	가나 출판사	1000	80/08/15
3 python	파이썬 프로그래밍	가나 출판사	1000	90/07/17
4 r	R 프로그래밍	가나 출판사	1000	75/11/11
5 database	데이터 베이스 개론	가나 출판사	1000	78/03/01
6 mybatis	마이바티스	가나 출판사	1000	45/08/15
7 spring	스프링 Framework	가나 출판사	1000	50/06/25
8 xml	xml 프로그래밍	가나 출판사	1000	70/06/06
9 story	프로그래밍 이야기	가나 출판사	1000	70/09/09
10 timeseries	시계열 예측	가나 출판사	1000	50/06/25

- 모든 출판사의 이름을 '대한 출판사'로 단가를 5000으로 변경해 보세요.

```
update books set price = 5000, publisher = '대한 출판사';
select * from books ;
```

BID	BNAME	PUBLISHER	PRICE	INPUTDATE
1 java	자바 프로그래밍	대한 출판사	5000	70/08/28
2 jsp	jsp 프로그래밍	대한 출판사	5000	80/08/15
3 python	파이썬 프로그래밍	대한 출판사	5000	90/07/17
4 r	R 프로그래밍	대한 출판사	5000	75/11/11
5 database	데이터 베이스 개론	대한 출판사	5000	78/03/01
6 mybatis	마이바티스	대한 출판사	5000	45/08/15
7 spring	스프링 Framework	대한 출판사	5000	50/06/25
8 xml	xml 프로그래밍	대한 출판사	5000	70/06/06
9 story	프로그래밍 이야기	대한 출판사	5000	70/09/09
10 timeseries	시계열 예측	대한 출판사	5000	50/06/25

- 데이터를 롤백하시고, 서적 아이디가 'java'인 서적의 단가를 4000으로 변경해 보세요.

```
rollback ;
update books set price = 4000 where bid = 'java' ;
select * from books where bid = 'java' ;
```

BID	BNAME	PUBLISHER	PRICE	INPUTDATE
1 java	자바 프로그래밍	대한 출판사	4000	70/08/28

- 모든 서적들에 대하여 값을 10%씩 인상해 보세요.

```
update books set price = 1.1 * price ;
select * from books ;
```

BID	BNAME	PUBLISHER	PRICE	INPUTDATE
1 java	자바 프로그래밍	대한 출판사	4400	70/08/28
2 jsp	jsp 프로그래밍	대한 Books	8800	80/08/15
3 python	파이썬 프로그래밍	민국 미디어	9900	90/07/17
4 r	R 프로그래밍	민국 미디어	38500	75/11/11
5 database	데이터 베이스 개론	대한 출판사	8800	78/03/01
6 mybatis	마이바티스	대한 출판사	8250	45/08/15
7 spring	스프링 Framework	소셜 미디어	22000	50/06/25
8 xml	xml 프로그래밍	소셜 미디어	14300	70/06/06
9 story	프로그래밍 이야기	지한 출판사	6600	70/09/09
10 timeseries	시계열 예측	지한 출판사	14300	50/06/25

- 아이디가 'database'이거나 'xml'인 책의 단가를 2배수로 만들어 보세요.

```
update books set price = 2 * price
where bid = 'database' or bid = 'xml' ;
select * from books where bid = 'database' or bid = 'xml' ;
```

BID	BNAME	PUBLISHER	PRICE	INPUTDATE
1 database	데이터 베이스 개론	대한 출판사	17600	78/03/01
2 xml	xml 프로그래밍	소셜 미디어	28600	70/06/06

- 서적 아이디가 'database'인 서적의 단가를 100만큼 뺄셈하세요.

```
update books set price = price - 100 where bid = 'database' ;
select * from books where bid = 'database' ;
```

BID	BNAME	PUBLISHER	PRICE	INPUTDATE
1 database	데이터 베이스 개론	대한 출판사	17500	78/03/01

- 서적 아이디가 'java'인 서적의 단가가 50원 인상이 되었고, 이름은 '자바'로 변경이 되었습니다.

```
update books set price = price + 50, bname = '자바' where bid = 'java' ;
select * from books where bid = 'java' ;
```

BID	BNAME	PUBLISHER	PRICE	INPUTDATE
1 java	자바	대한 출판사	4450	70/08/28

- 서적 아이디가 'python'인 서적의 이름을 '파이썬'으로 변경하세요.

```
update books set bname = '파이썬' where bid = 'python' ;
select * from books where bid = 'python' ;
```

BID	BNAME	PUBLISHER	PRICE	INPUTDATE
1 python	파이썬	민국 미디어	9900	90/07/17

- 서적 아이디가 'python'인 서적의 단가를 600으로 변경하세요.

```
update books set price = 600 where bid = 'python' ;
select * from books where bid = 'python' ;
```

BID	BNAME	PUBLISHER	PRICE	INPUTDATE
1 python	파이썬	민국 미디어	600	90/07/17

- 단가가 5,000원 이하인 항목들의 단가를 3,000원으로 변경하세요.

```
update books set price = 3000 where price <= 10000 ;
select * from books ;
```

BID	BNAME	PUBLISHER	PRICE	INPUTDATE
1 java	자바	대한 출판사	3000	70/08/28
2 jsp	jsp 프로그래밍	대한 Books	3000	80/08/15
3 python	파이썬	민국 미디어	3000	90/07/17
4 r	R 프로그래밍	민국 미디어	38500	75/11/11
5 database	데이터 베이스 개론	대한 출판사	17500	78/03/01
6 mybatis	마이바티스	대한 출판사	3000	45/08/15
7 spring	스프링 Framework	소셜 미디어	22000	50/06/25
8 xml	xml 프로그래밍	소셜 미디어	28600	70/06/06
9 story	프로그래밍 이야기	지한 출판사	3000	70/09/09
10 timeseries	시계열 예측	지한 출판사	14300	50/06/25

- 단가가 10,000원 이상이고 서적 아이디가 'java' 또는 'xml'인 서적들의 단가를 13000으로 변경하세요.

```
update books set price = 13000 where (bid = 'java' or bid = 'xml') and price >= 10000 ;
select * from books ;
```

BID	BNAME	PUBLISHER	PRICE	INPUTDATE
1 java	자바	대한 출판사	3000	70/08/28
2 jsp	jsp 프로그래밍	대한 Books	3000	80/08/15
3 python	파이썬	민국 미디어	3000	90/07/17
4 r	R 프로그래밍	민국 미디어	38500	75/11/11
5 database	데이터 베이스 개론	대한 출판사	17500	78/03/01
6 mybatis	마이바티스	대한 출판사	3000	45/08/15
7 spring	스프링 Framework	소셜 미디어	22000	50/06/25
8 xml	xml 프로그래밍	소셜 미디어	13000	70/06/06
9 story	프로그래밍 이야기	지한 출판사	3000	70/09/09
10 timeseries	시계열 예측	지한 출판사	14300	50/06/25

- 서적 아이디가 'java' 또는 'spring'인 서적의 단가를 450으로 변경해 보세요.

```
update books set price = 450 where bid = 'java' or bid = 'spring' ;
select * from books where bid = 'java' or bid = 'spring' ;
```

BID	BNAME	PUBLISHER	PRICE	INPUTDATE
1 java	자바	대한 출판사	450	70/08/28
2 spring	스프링 Framework	소셜 미디어	450	50/06/25

- 서적 아이디가 'java' 또는 'spring'인 서적의 단가를 450으로 변경해 보세요.

in 연산자를 이용하여 다시 풀어 보세요.

```
update books set price = 450 where bid in('java', 'spring') ;
select * from books where bid in('java', 'spring') ;
```

BID	BNAME	PUBLISHER	PRICE	INPUTDATE
1 java	자바	대한 출판사	450	70/08/28
2 spring	스프링 Framework	소셜 미디어	450	50/06/25

● 롤백하세요.

출판사 '민국 미디어'를 '영문 출판사'로 변경해 보세요.

```
rollback ;
update books set publisher = '영문 출판사' where publisher = '민국 미디어' ;
select * from books where publisher = '영문 출판사' ;
```

BID	BNAME	PUBLISHER	PRICE	INPUTDATE
1	python	파이썬 프로그래밍	영문 출판사	9000 90/07/17
2	r	R 프로그래밍	영문 출판사	35000 75/11/11

● 위의 사항들을 롤백하시오.

모든 변경이 되었던 내용에 대하여 다시 원래 데이터로 복귀했는지 확인하도록 합니다.

```
rollback ;
select * from books ;
```

BID	BNAME	PUBLISHER	PRICE	INPUTDATE
1	java	자바 프로그래밍	대한 출판사	7000 70/08/28
2	jsp	jsp 프로그래밍	대한 Books	8000 80/08/15
3	python	파이썬 프로그래밍	민국 미디어	9000 90/07/17
4	r	R 프로그래밍	민국 미디어	35000 75/11/11
5	database	데이터 베이스 개론	대한 출판사	8000 78/03/01
6	mybatis	마이바티스	대한 출판사	7500 45/08/15
7	spring	스프링 Framework	소셜 미디어	20000 50/06/25
8	xml	xml 프로그래밍	소셜 미디어	13000 70/06/06
9	story	프로그래밍 이야기	지한 출판사	6000 70/09/09
10	timeseries	시계열 예측	지한 출판사	13000 50/06/25

● 이후의 모든 실습후 각각 롤백을 수행하면서 다음 실습을 진행하도록 합니다.

모든 서적을 삭제한 다음 조회를 수행해 보세요.

```
delete from books ;
select * from books ;

-- 조회되는 데이터가 없습니다.
```

● 서적 아이디가 'mybatis'인 서적을 삭제해 보세요.

```
rollback ;
delete from books where bid = 'mybatis' ;
select * from books ;
```

BID	BNAME	PUBLISHER	PRICE	INPUTDATE
1	java	자바 프로그래밍	대한 출판사	7000 70/08/28
2	jsp	jsp 프로그래밍	대한 Books	8000 80/08/15
3	python	파이썬 프로그래밍	민국 미디어	9000 90/07/17
4	r	R 프로그래밍	민국 미디어	35000 75/11/11
5	database	데이터 베이스 개론	대한 출판사	8000 78/03/01
6	spring	스프링 Framework	소셜 미디어	20000 50/06/25
7	xml	xml 프로그래밍	소셜 미디어	13000 70/06/06
8	story	프로그래밍 이야기	지한 출판사	6000 70/09/09
9	timeseries	시계열 예측	지한 출판사	13000 50/06/25

- 출판사가 '민국 미디어'인 서적을 삭제해 보세요.

```
rollback ;
delete from books where publisher = '민국 미디어' ;
select * from books ;
```

BID	BNAME	PUBLISHER	PRICE	INPUTDATE
1 java	자바 프로그래밍	대한 출판사	7000	70/08/28
2 jsp	jsp 프로그래밍	대한 Books	8000	80/08/15
3 database	데이터 베이스 개론	대한 출판사	8000	78/03/01
4 mybatis	마이바티스	대한 출판사	7500	45/08/15
5 spring	스프링 Framework	소셜 미디어	20000	50/06/25
6 xml	xml 프로그래밍	소셜 미디어	13000	70/06/06
7 story	프로그래밍 이야기	지한 출판사	6000	70/09/09
8 timeseries	시계열 예측	지한 출판사	13000	50/06/25

- 출판사가 '민국 미디어'와 '지한 출판사'인 서적들을 삭제해 보세요.

```
rollback ;
delete from books where publisher in ('민국 미디어', '지한 출판사') ;
select * from books ;
```

BID	BNAME	PUBLISHER	PRICE	INPUTDATE
1 java	자바 프로그래밍	대한 출판사	7000	70/08/28
2 jsp	jsp 프로그래밍	대한 Books	8000	80/08/15
3 database	데이터 베이스 개론	대한 출판사	8000	78/03/01
4 mybatis	마이바티스	대한 출판사	7500	45/08/15
5 spring	스프링 Framework	소셜 미디어	20000	50/06/25
6 xml	xml 프로그래밍	소셜 미디어	13000	70/06/06

- 출판사가 '소셜 미디어'인 서적 중에서 단가가 10,000이상인 서적들을 삭제해 보세요.

```
rollback ;
delete from books where publisher in ('소셜 미디어') and price >= 10000 ;
select * from books ;
```

BID	BNAME	PUBLISHER	PRICE	INPUTDATE
1 java	자바 프로그래밍	대한 출판사	7000	70/08/28
2 jsp	jsp 프로그래밍	대한 Books	8000	80/08/15
3 python	파이썬 프로그래밍	민국 미디어	9000	90/07/17
4 r	R 프로그래밍	민국 미디어	35000	75/11/11
5 database	데이터 베이스 개론	대한 출판사	8000	78/03/01
6 mybatis	마이바티스	대한 출판사	7500	45/08/15
7 story	프로그래밍 이야기	지한 출판사	6000	70/09/09
8 timeseries	시계열 예측	지한 출판사	13000	50/06/25

- 단가가 8,000원 미만이거나 13,000원 이상인 서적들을 삭제해 보세요.

```
rollback ;
delete from books where price < 8000 or price >= 13000 ;
select * from books ;
```

BID	BNAME	PUBLISHER	PRICE	INPUTDATE
1 jsp	jsp 프로그래밍	대한 Books	8000	80/08/15
2 python	파이썬 프로그래밍	민국 미디어	9000	90/07/17
3 database	데이터 베이스 개론	대한 출판사	8000	78/03/01

- 출판사가 '지한 출판사'와 '소셜 미디어'를 제외한 다른 출판사의 서적들을 모두 삭제하세요.  
이번 예시에서는 in 키워드와 not 키워드를 같이 사용하는 not in 구문을 사용하면 합니다.

```
rollback ;
delete from books where publisher not in ('지한 출판사', '소셜 미디어') ;
select * from books ;
```

BID	BNAME	PUBLISHER	PRICE	INPUTDATE
1 spring	스프링 Framework	소셜 미디어	20000	50/06/25
2 xml	xml 프로그래밍	소셜 미디어	13000	70/06/06
3 story	프로그래밍 이야기	지한 출판사	6000	70/09/09
4 timeseries	시계열 예측	지한 출판사	13000	50/06/25

- 서적 아이디 'jsp'가 절판이 되었습니다.

```
rollback ;
delete from books where bid = 'jsp' ;
select * from books ;
```

BID	BNAME	PUBLISHER	PRICE	INPUTDATE
1 java	자바 프로그래밍	대한 출판사	7000	70/08/28
2 python	파이썬 프로그래밍	민국 미디어	9000	90/07/17
3 r	R 프로그래밍	민국 미디어	35000	75/11/11
4 database	데이터 베이스 개론	대한 출판사	8000	78/03/01
5 mybatis	마이바티스	대한 출판사	7500	45/08/15
6 spring	스프링 Framework	소셜 미디어	20000	50/06/25
7 xml	xml 프로그래밍	소셜 미디어	13000	70/06/06
8 story	프로그래밍 이야기	지한 출판사	6000	70/09/09
9 timeseries	시계열 예측	지한 출판사	13000	50/06/25

- 서적 아이디 'java', 'spring'이 절판되었습니다.

```
rollback ;
delete from books where bid in( 'java', 'spring' ) ;
select * from books ;
```

BID	BNAME	PUBLISHER	PRICE	INPUTDATE
1 jsp	jsp 프로그래밍	대한 Books	8000	80/08/15
2 python	파이썬 프로그래밍	민국 미디어	9000	90/07/17
3 r	R 프로그래밍	민국 미디어	35000	75/11/11
4 database	데이터 베이스 개론	대한 출판사	8000	78/03/01
5 mybatis	마이바티스	대한 출판사	7500	45/08/15
6 xml	xml 프로그래밍	소셜 미디어	13000	70/06/06
7 story	프로그래밍 이야기	지한 출판사	6000	70/09/09
8 timeseries	시계열 예측	지한 출판사	13000	50/06/25

● 위의 사항들을 롤백 하시오.

```
rollback ;
select * from books ;
```

BID	BNAME	PUBLISHER	PRICE	INPUTDATE
1 java	자바 프로그래밍	대한 출판사	7000	70/08/28
2 jsp	jsp 프로그래밍	대한 Books	8000	80/08/15
3 python	파이썬 프로그래밍	민국 미디어	9000	90/07/17
4 r	R 프로그래밍	민국 미디어	35000	75/11/11
5 database	데이터 베이스 개론	대한 출판사	8000	78/03/01
6 mybatis	마이바티스	대한 출판사	7500	45/08/15
7 spring	스프링 Framework	소셜 미디어	20000	50/06/25
8 xml	xml 프로그래밍	소셜 미디어	13000	70/06/06
9 story	프로그래밍 이야기	지한 출판사	6000	70/09/09
10 timeseries	시계열 예측	지한 출판사	13000	50/06/25

● savepoint를 실습합니다.

샘플용 테이블 mybook을 다음과 같이 생성하도록 합니다.

```
-- drop table mybook purge ;

create table mybook
as
select bid, bname, price from books where price < 10000 ;

select * from mybook ;
```

BID	BNAME	PRICE
1 java	자바 프로그래밍	7000
2 jsp	jsp 프로그래밍	8000
3 python	파이썬 프로그래밍	9000
4 database	데이터 베이스 개론	8000
5 mybatis	마이바티스	7500
6 story	프로그래밍 이야기	6000

- savepoint point01을 다음과 같이 생성합니다. 그리고, 현재 상황은 데이터가 총 6건임을 확인합니다.

```
savepoint point01 ;
delete from mybook where price < 8000 ;
select * from mybook ;
```

BID	BNAME	PRICE
1 jsp	jsp 프로그래밍	8000
2 python	파이썬 프로그래밍	9000
3 database	데이터 베이스 개론	8000

- savepoint point02을 다음과 같이 생성합니다. 그리고, 현재 상황에서 단가가 8000미만인 서적은 삭제되었습니다.

```
savepoint point02 ;
```

- 모든 서적의 단가를 10,000원으로 수정합니다.  
현재 상황) 모든 서적의 단가가 10000원입니다.

```
update mybook set price = 10000 ;
select * from mybook ;
```

BID	BNAME	PRICE
1 jsp	jsp 프로그래밍	10000
2 python	파이썬 프로그래밍	10000
3 database	데이터 베이스 개론	10000

- point02 지점으로 이동하면, 여자들의 급여가 이전 상태로 돌아갑니다.  
모든 서적들의 단가가 이전의 값으로 되돌아 갈 것이라고 기대됩니다.

```
rollback to point02 ;
select * from mybook ;
```

BID	BNAME	PRICE
1 jsp	jsp 프로그래밍	8000
2 python	파이썬 프로그래밍	9000
3 database	데이터 베이스 개론	8000

- point01 지점으로 이동하면, 8000 이하인 서적들도 보여야 합니다.  
8000원짜리 이하의 서적 목록도 보입니다.

```
rollback to point01 ;
select * from mybook ;
```

BID	BNAME	PRICE
1 java	자바 프로그래밍	7000
2 jsp	jsp 프로그래밍	8000
3 python	파이썬 프로그래밍	9000
4 database	데이터 베이스 개론	8000
5 mybatis	마이바티스	7500
6 story	프로그래밍 이야기	6000



- 다음과 같이 고객 테이블에 다음과 같이 데이터를 추가해 주세요.

ID	NAME	ADDRESS	PHONE	MANAGER	SALARY
1 yoon	윤봉길	중국 상하이	000-1111-1111	shin	100
2 yusin	김유신	신라 경주	000-2222-2222	shin	200
3 shin	신사임당	백제 공주	000-3333-3333	(null)	300
4 an	안중근	조선 강원도	000-4444-4444	hong	400
5 hong	홍범도	일제 시대 간도	(null)	(null)	500

```
insert into customers(id, name, address, phone, manager, salary)
values('yoon', '윤봉길', '중국 상하이', '000-1111-1111', 'shin', 100);
insert into customers(id, name, address, phone, manager, salary)
values('yusin', '김유신', '신라 경주', '000-2222-2222', 'shin', 200);
insert into customers(id, name, address, phone, manager, salary)
values('shin', '신사임당', '백제 공주', '000-3333-3333', null, 300);
insert into customers(id, name, address, phone, manager, salary)
values('an', '안중근', '조선 강원도', '000-4444-4444', 'hong', 400);
insert into customers(id, name, address, phone, manager, salary)
values('hong', '홍범도', '일제 시대 간도', null, null, 500);
commit ;

select * from customers ;
```

- 다음과 같이 주문 테이블에 다음과 같이 데이터를 추가해 주세요.

OID	CID	BID	SALEPRICE	SALEDATE
1	1 yoon	java	6000	21/07/01
2	2 yoon	python	21000	21/07/03
3	3 yusin	database	8000	21/07/03
4	4 shin	jsp	6000	21/07/04
5	5 an	spring	20000	21/07/05
6	6 yoon	jsp	12000	21/07/07
7	7 an	jsp	13000	21/08/15
8	8 shin	java	12000	21/03/01
9	9 yusin	java	7000	21/12/25
10	10 shin	python	13000	21/07/10

```
insert into orders(oid, cid, bid, saleprice, saledate)
values(1, 'yoon', 'java', 6000, to_date('2021-07-01', 'yyyy-mm-dd'));
insert into orders(oid, cid, bid, saleprice, saledate)
values(2, 'yoon', 'python', 21000, to_date('2021-07-03', 'yyyy-mm-dd'));
insert into orders(oid, cid, bid, saleprice, saledate)
values(3, 'yusin', 'database', 8000, to_date('2021-07-03', 'yyyy-mm-dd'));
insert into orders(oid, cid, bid, saleprice, saledate)
values(4, 'shin', 'jsp', 6000, to_date('2021-07-04', 'yyyy-mm-dd'));
insert into orders(oid, cid, bid, saleprice, saledate)
values(5, 'an', 'spring', 20000, to_date('2021-07-05', 'yyyy-mm-dd'));
insert into orders(oid, cid, bid, saleprice, saledate)
values(6, 'yoon', 'jsp', 12000, to_date('2021-07-07', 'yyyy-mm-dd'));
insert into orders(oid, cid, bid, saleprice, saledate)
values(7, 'an', 'jsp', 13000, to_date('2021-08-15', 'yyyy-mm-dd'));
insert into orders(oid, cid, bid, saleprice, saledate)
values(8, 'shin', 'java', 12000, to_date('2021-03-01', 'yyyy-mm-dd'));
insert into orders(oid, cid, bid, saleprice, saledate)
values(9, 'yusin', 'java', 7000, to_date('2021-12-25', 'yyyy-mm-dd'));
insert into orders(oid, cid, bid, saleprice, saledate)
values(10, 'shin', 'python', 13000, to_date('2021-07-10', 'yyyy-mm-dd'));
commit ;

select * from orders ;
```

## 7. 데이터 질의어(DQL)

- '곰돌이' 사용자의 테이블 'Books'에 대하여 다음 물음에 대한 sql 구문을 작성해 보세요  
다음 실행을 위하여 다음 구문을 수행하도록 합니다.

```
update books set price = null where bid in('r', 'database', 'mybatis') ;  
commit ;
```

- 모든 서적들의 정보를 출력해 보세요.

```
select * from books ;
```

BID	BNAME	PUBLISHER	PRICE	INPUTDATE
1 java	자바 프로그래밍	대한 출판사	7000	70/08/28
2 jsp	jsp 프로그래밍	대한 Books	8000	80/08/15
3 python	파이썬 프로그래밍	민국 미디어	9000	90/07/17
4 r	R 프로그래밍	민국 미디어	(null)	75/11/11
5 database	데이터 베이스 개론	대한 출판사	(null)	78/03/01
6 mybatis	마이바티스	대한 출판사	(null)	45/08/15
7 spring	스프링 Framework	소셜 미디어	20000	50/06/25
8 xml	xml 프로그래밍	소셜 미디어	13000	70/06/06
9 story	프로그래밍 이야기	지한 출판사	6000	70/09/09
10 timeseries	시계열 예측	지한 출판사	13000	50/06/25

- 모든 서적들의 아이디, 서적 이름, 단가를 조회해 보세요.

```
select bid, bname, price from books ;
```

BID	BNAME	PRICE
1 java	자바 프로그래밍	7000
2 jsp	jsp 프로그래밍	8000
3 python	파이썬 프로그래밍	9000
4 r	R 프로그래밍	(null)
5 database	데이터 베이스 개론	(null)
6 mybatis	마이바티스	(null)
7 spring	스프링 Framework	20000
8 xml	xml 프로그래밍	13000
9 story	프로그래밍 이야기	6000
10 timeseries	시계열 예측	13000

- 모든 서적들의 서적 이름, 단가, 아이디를 조회해 보세요.

```
select bname, price, bid from books ;
```

BNAME	PRICE	BID
1 자바 프로그래밍	7000	java
2 jsp 프로그래밍	8000	jsp
3 파이썬 프로그래밍	9000	python
4 R 프로그래밍	(null)	r
5 데이터 베이스 개론	(null)	database
6 마이바티스	(null)	mybatis
7 스프링 Framework	20000	spring
8 xml 프로그래밍	13000	xml
9 프로그래밍 이야기	6000	story
10 시계열 예측	13000	timeseries

- 서적의 이름과 10권을 판매한 금액을 출력해 보세요.

```
select bname, 10 * price from books ;
```

⚡ BNAME	⚡ 10*PRICE
1 자바 프로그래밍	70000
2 jsp 프로그래밍	80000
3 파이썬 프로그래밍	90000
4 R 프로그래밍	(null)
5 데이터 베이스 개론	(null)
6 마이바티스	(null)
7 스프링 Framework	200000
8 xml 프로그래밍	130000
9 프로그래밍 이야기	60000
10 시계열 예측	130000

- 서적의 이름과 가격과 10 퍼센트 할인된 가격 정보를 출력해 보세요.

```
select bname, price, (1 - 0.1) * price from books ;
```

⚡ BNAME	⚡ PRICE	⚡ (1-0.1)*PRICE
1 자바 프로그래밍	7000	6300
2 jsp 프로그래밍	8000	7200
3 파이썬 프로그래밍	9000	8100
4 R 프로그래밍	(null)	(null)
5 데이터 베이스 개론	(null)	(null)
6 마이바티스	(null)	(null)
7 스프링 Framework	20000	18000
8 xml 프로그래밍	13000	11700
9 프로그래밍 이야기	6000	5400
10 시계열 예측	13000	11700

- 출판사 이름만 조회해 보세요.

출판사 이름만 조회하되, 중복된 데이터는 하나만 출력되도록 해주세요.

```
select publisher from books ;
select distinct publisher from books ;
```

⚡ PUBLISHER	⚡ PUBLISHER
1 대한 출판사	1 소셜 미디어
2 대한 Books	2 대한 Books
3 민국 미디어	3 민국 미디어
4 민국 미디어	4 대한 출판사
5 대한 출판사	5 지한 출판사
6 대한 출판사	
7 소셜 미디어	
8 소셜 미디어	
9 지한 출판사	
10 지한 출판사	

- 서적의 이름과 단가와 100원 인상된 가격을 출력해 보세요.

```
select bname, price, price + 100 from books ;
```

BNAME	PRICE	PRICE+100
1 자바 프로그래밍	7000	7100
2 jsp 프로그래밍	8000	8100
3 파이썬 프로그래밍	9000	9100
4 R 프로그래밍	(null)	(null)
5 데이터 베이스 개론	(null)	(null)
6 마이바티스	(null)	(null)
7 스프링 Framework	20000	20100
8 xml 프로그래밍	13000	13100
9 프로그래밍 이야기	6000	6100
10 시계열 예측	13000	13100

- 서적의 이름과 단가와 50% 인상된 가격을 출력해 보세요.

```
select bname, price, 1.5*price from books ;
```

BNAME	PRICE	1.5*PRICE
1 자바 프로그래밍	7000	10500
2 jsp 프로그래밍	8000	12000
3 파이썬 프로그래밍	9000	13500
4 R 프로그래밍	(null)	(null)
5 데이터 베이스 개론	(null)	(null)
6 마이바티스	(null)	(null)
7 스프링 Framework	20000	30000
8 xml 프로그래밍	13000	19500
9 프로그래밍 이야기	6000	9000
10 시계열 예측	13000	19500

- 서적의 이름과 단가와 50% 인상된 가격을 출력하되 별칭을 적절히 사용해 보세요.

```
select bname, price, 1.5*price as upprice from books ;
```

BNAME	PRICE	UPPRICE
1 자바 프로그래밍	7000	10500
2 jsp 프로그래밍	8000	12000
3 파이썬 프로그래밍	9000	13500
4 R 프로그래밍	(null)	(null)
5 데이터 베이스 개론	(null)	(null)
6 마이바티스	(null)	(null)
7 스프링 Framework	20000	30000
8 xml 프로그래밍	13000	19500
9 프로그래밍 이야기	6000	9000
10 시계열 예측	13000	19500

- 서적 이름과 단가를 출력하되, 별칭 '서적\_이름', '가격'으로 출력해 보세요.

```
select bname as 서적_이름, price 가격 from books ;
```

서적_이름	가격
1 자바 프로그래밍	7000
2 jsp 프로그래밍	8000
3 파이썬 프로그래밍	9000
4 R 프로그래밍	(null)
5 데이터 베이스 개론	(null)
6 마이바티스	(null)
7 스프링 Framework	20000
8 xml 프로그래밍	13000
9 프로그래밍 이야기	6000
10 시계열 예측	13000

- 문자열 연결 연산자를 사용하여 다음과 같이 출력해 보세요.

자바 프로그래밍의 가격은 7000원입니다.

```
select bname || '의 가격은 ' || price || '원입니다.' as result
from books ;
```

RESULT
1 자바 프로그래밍의 가격은 7000원입니다.
2 jsp 프로그래밍의 가격은 8000원입니다.
3 파이썬 프로그래밍의 가격은 9000원입니다.
4 R 프로그래밍의 가격은 원입니다.
5 데이터 베이스 개론의 가격은 원입니다.
6 마이바티스의 가격은 원입니다.
7 스프링 Framework의 가격은 20000원입니다.
8 xml 프로그래밍의 가격은 13000원입니다.
9 프로그래밍 이야기의 가격은 6000원입니다.
10 시계열 예측의 가격은 13000원입니다.

- 서적 이름 : 자바 프로그래밍, 출판사 : 대한 출판사, 가격 : 7000

```
select '서적 이름 : ' || bname || ', 출판사 : ' || publisher || ', 가격 : ' || price as 결과
from books ;
```

결과
1 서적 이름 : 자바 프로그래밍, 출판사 : 대한 출판사, 가격 : 7000
2 서적 이름 : jsp 프로그래밍, 출판사 : 대한 Books, 가격 : 8000
3 서적 이름 : 파이썬 프로그래밍, 출판사 : 민국 미디어, 가격 : 9000
4 서적 이름 : R 프로그래밍, 출판사 : 민국 미디어, 가격 : 35000
5 서적 이름 : 데이터 베이스 개론, 출판사 : 대한 출판사, 가격 : 8000
6 서적 이름 : 마이바티스, 출판사 : 대한 출판사, 가격 : 7500
7 서적 이름 : 스프링 Framework, 출판사 : 소설 미디어, 가격 : 20000
8 서적 이름 : xml 프로그래밍, 출판사 : 소설 미디어, 가격 : 13000
9 서적 이름 : 프로그래밍 이야기, 출판사 : 지한 출판사, 가격 : 6000
10 서적 이름 : 시계열 예측, 출판사 : 지한 출판사, 가격 : 13000

## 행의 제한

- 서적의 아이디가 python인 서적을 조회해 보세요.

```
select * from books where bid = 'python' ;
```

BID	BNAME	PUBLISHER	PRICE	INPUTDATE
1	python	파이썬 프로그래밍	민국 미디어	9000 90/07/17

- 서적 제목이 '파이썬 프로그래밍'인 서적들을 출력해 보세요.  
오라클에서 문자는 외따옴표로 사용해야 합니다.

```
select * from books where bname = '파이썬 프로그래밍' ;
```

BID	BNAME	PUBLISHER	PRICE	INPUTDATE
1	python	파이썬 프로그래밍	민국 미디어	9000 90/07/17

- 서적의 단가가 10,000원 이상인 서적을 조회해 보세요.

```
select * from books where price >= 10000 ;
```

BID	BNAME	PUBLISHER	PRICE	INPUTDATE
1	spring	스프링 Framework	소셜 미디어	20000 (null)
2	xml	xml 프로그래밍	소셜 미디어	13000 70/06/06
3	timeseries	시계열 예측	지한 출판사	13000 50/06/25

- 출판사의 이름이 '지한 출판사'인 행들만 조회해 보세요.

```
select * from books where publisher = '지한 출판사' ;
```

BID	BNAME	PUBLISHER	PRICE	INPUTDATE
1	story	프로그래밍 이야기	지한 출판사	6000 70/09/09
2	timeseries	시계열 예측	지한 출판사	13000 50/06/25

- 서적의 단가가 9,000원이상이고, 13,000원 이하인 서적들을 조회해 보세요.

```
select * from books where price >= 9000 and price <= 13000 ;
```

BID	BNAME	PUBLISHER	PRICE	INPUTDATE
1	python	파이썬 프로그래밍	민국 미디어	9000 90/07/17
2	xml	xml 프로그래밍	소셜 미디어	13000 70/06/06
3	timeseries	시계열 예측	지한 출판사	13000 50/06/25

- 서적의 단가가 30,000원 이상이거나 7,000원 이하인 서적들을 조회해 보세요.

```
select * from books where price >= 30000 or price <= 7000 ;
```

BID	BNAME	PUBLISHER	PRICE	INPUTDATE
1	java	자바 프로그래밍 대한 출판사	7000	70/08/28
2	story	프로그래밍 이야기 지한 출판사	6000	70/09/09

- 서적의 아이디가 'python' 또는 'database'인 서적의 서적 이름과 단가를 조회해 보세요.

```
select bname, price from books where bid = 'python' or bid = 'database' ;
```

BNAME	PRICE
1 데이터 베이스 개론	(null)
2 파이썬 프로그래밍	9000

- 서적의 단가가 8,000원이거나 13,000원인 서적들을 조회해 보세요.

```
select bname, price from books where price = 8000 or price = 13000 ;
```

BNAME	PRICE
1 jsp 프로그래밍	8000
2 xml 프로그래밍	13000
3 시계열 예측	13000

- 서적의 단가가 없는 서적들을 출력해 보세요.

```
select * from books where price is null ;
```

BID	BNAME	PUBLISHER	PRICE	INPUTDATE
1	r	R 프로그래밍	(null)	75/11/11
2	database	데이터 베이스 개론 대한 출판사	(null)	78/03/01
3	mybatis	마이바티스 대한 출판사	(null)	45/08/15

- 단가가 존재하는 서적을 출력해 보세요.

```
select * from books where price is not null ;
```

BID	BNAME	PUBLISHER	PRICE	INPUTDATE
1	java	자바 프로그래밍 대한 출판사	7000	70/08/28
2	jsp	jsp 프로그래밍 대한 Books	8000	80/08/15
3	python	파이썬 프로그래밍 민국 미디어	9000	90/07/17
4	spring	스프링 Framework 소셜 미디어	20000	50/06/25
5	xml	xml 프로그래밍 소셜 미디어	13000	70/06/06
6	story	프로그래밍 이야기 지한 출판사	6000	70/09/09
7	timeseries	시계열 예측 지한 출판사	13000	50/06/25

- 모든 서적들의 서적 이름과 단가와 10권 판매된 금액을 조회해 보세요.

```
select bname, price, 10*price from books ;
```

BNAME	PRICE	10*PRICE
1 자바 프로그래밍	7000	70000
2 jsp 프로그래밍	8000	80000
3 파이썬 프로그래밍	9000	90000
4 R 프로그래밍	(null)	(null)
5 데이터 베이스 개론	(null)	(null)
6 마이바티스	(null)	(null)
7 스프링 Framework	20000	200000
8 xml 프로그래밍	13000	130000
9 프로그래밍 이야기	6000	60000
10 시계열 예측	13000	130000

- 서적의 단가가 9,000원 이상, 13,000 이하인 서적들을 조회해 보세요.

```
select * from books
where price between 9000 and 13000 ;
```

BID	BNAME	PUBLISHER	PRICE	INPUTDATE
1 python	파이썬 프로그래밍	민국 미디어	9000	90/07/17
2 xml	xml 프로그래밍	소셜 미디어	13000	70/06/06
3 timeseries	시계열 예측	지한 출판사	13000	50/06/25

- 서적 이름의 초성이 'ㅅ'으로 시작하고 'ㅈ'으로 끝나는 서적들을 조회해 보세요.

```
select * from books
where bname between 'ㅅ' and 'ㅈ' ;
```

BID	BNAME	PUBLISHER	PRICE	INPUTDATE
1 spring	스프링 Framework	소셜 미디어	20000	50/06/25
2 timeseries	시계열 예측	지한 출판사	13000	50/06/25

- 출판사 이름이 '소셜 미디어'이거나 '대한 출판사'인 서적들을 조회해 보세요.

```
select * from books where publisher in ('소셜 미디어', '대한 출판사') ;
```

BID	BNAME	PUBLISHER	PRICE	INPUTDATE
1 java	자바 프로그래밍	대한 출판사	7000	70/08/28
2 database	데이터 베이스 개론	대한 출판사	(null)	78/03/01
3 mybatis	마이바티스	대한 출판사	(null)	45/08/15
4 spring	스프링 Framework	소셜 미디어	20000	50/06/25
5 xml	xml 프로그래밍	소셜 미디어	13000	70/06/06



- in 절(clause)을 사용하여 서적의 아이디가 'java', 'python', 'database'인 행을 출력해 보세요.

```
select * from books where bid in('java', 'python', 'database') ;
```

BID	BNAME	PUBLISHER	PRICE	INPUTDATE
1 database	데이터 베이스 개론	대한 출판사	(null)	78/03/01
2 java	자바 프로그래밍	대한 출판사	7000	70/08/28
3 python	파이썬 프로그래밍	민국 미디어	9000	90/07/17

- 서적 제목이 'R 프로그래밍'이거나 '스프링 Framework'인 서적들은?

```
select * from books where bname in('R 프로그래밍', '스프링 Framework') ;
```

BID	BNAME	PUBLISHER	PRICE	INPUTDATE
1 r	R 프로그래밍	민국 미디어	(null)	75/11/11
2 spring	스프링 Framework	소셜 미디어	20000	50/06/25

## like 연산자

- 서적 이름이 '자'로 시작하는 서적들을 출력해 보세요.

```
select * from books where bname like '자%';
```

BID	BNAME	PUBLISHER	PRICE	INPUTDATE
1 java	자바 프로그래밍	대한 출판사	7000	70/08/28

- 글자 '파'로 시작하는 모든 서적들을 출력해 보세요.

```
select * from books where bname like '파%';
```

BID	BNAME	PUBLISHER	PRICE	INPUTDATE
1 python	파이썬 프로그래밍	민국 미디어	9000	90/07/17

- 서적 이름에 '프로'자가 들어가는 서적들을 출력해 보세요.

```
select * from books where bname like '%프로%';
```

BID	BNAME	PUBLISHER	PRICE	INPUTDATE
1 java	자바 프로그래밍	대한 출판사	7000	70/08/28
2 jsp	jsp 프로그래밍	대한 Books	8000	80/08/15
3 python	파이썬 프로그래밍	민국 미디어	9000	90/07/17
4 r	R 프로그래밍	민국 미디어	(null)	75/11/11
5 xml	xml 프로그래밍	소셜 미디어	13000	70/06/06
6 story	프로그래밍 이야기	지한 출판사	6000	70/09/09

- 두 번째 글자가 반드시 '이'인 서적들을 출력해 보세요.

```
select * from books where bname like '_이%';
```

BID	BNAME	PUBLISHER	PRICE	INPUTDATE
1	python 파이썬 프로그래밍	민국 미디어	9000	90/07/17
2	database 데이터 베이스 개론	대한 출판사	(null)	78/03/01
3	mybatis 마이바티스	대한 출판사	(null)	45/08/15

- 출판사 이름이 '사'로 끝나는 모든 서적들을 출력해 보세요.

```
select * from books where publisher like '%사';
```

BID	BNAME	PUBLISHER	PRICE	INPUTDATE
1	java 자바 프로그래밍	대한 출판사	7000	70/08/28
2	database 데이터 베이스 개론	대한 출판사	(null)	78/03/01
3	mybatis 마이바티스	대한 출판사	(null)	45/08/15

- 서적 이름이 '프'로 시작하거나, '론'으로 끝나는 모든 서적들을 출력해 보세요.

```
select * from books where bname like '프%' or bname like '%론';
```

BID	BNAME	PUBLISHER	PRICE	INPUTDATE
1	database 데이터 베이스 개론	대한 출판사	(null)	78/03/01
2	story 프로그래밍 이야기	지한 출판사	6000	70/09/09

## 부정어 사용해보기

- 출판사 이름이 '대한 출판사'와 '지한 출판사'가 아닌 모든 서적들을 출력해 보세요.

```
select * from books where publisher not in('대한 출판사', '지한 출판사');
```

BID	BNAME	PUBLISHER	PRICE	INPUTDATE
1	jsp jsp 프로그래밍	대한 Books	8000	80/08/15
2	python 파이썬 프로그래밍	민국 미디어	9000	90/07/17
3	r R 프로그래밍	민국 미디어	(null)	75/11/11
4	spring 스프링 Framework	소셜 미디어	20000	50/06/25
5	xml xml 프로그래밍	소셜 미디어	13000	70/06/06

- 서적의 아이디가 'r', 'database', 'mybatis'가 아닌 서적들을 출력해 보세요.

```
select * from books where bid not in('r', 'database', 'mybatis');
```

BID	BNAME	PUBLISHER	PRICE	INPUTDATE
1 java	자바 프로그래밍	대한 출판사	7000	70/08/28
2 jsp	jsp 프로그래밍	대한 Books	8000	80/08/15
3 python	파이썬 프로그래밍	민국 미디어	9000	90/07/17
4 spring	스프링 Framework	소셜 미디어	20000	50/06/25
5 xml	xml 프로그래밍	소셜 미디어	13000	70/06/06
6 story	프로그래밍 이야기	지한 출판사	6000	70/09/09
7 timeseries	시계열 예측	지한 출판사	13000	50/06/25

- 단가가 9,000원과 13,000원 사이에 있지 않은 서적들을 출력해 보세요.

```
select * from books where price not between 9000 and 13000 ;
```

BID	BNAME	PUBLISHER	PRICE	INPUTDATE
1 java	자바 프로그래밍	대한 출판사	7000	70/08/28
2 jsp	jsp 프로그래밍	대한 Books	8000	80/08/15
3 spring	스프링 Framework	소셜 미디어	20000	50/06/25
4 story	프로그래밍 이야기	지한 출판사	6000	70/09/09

- 서적의 이름이 '프로그래밍'으로 끝나지 않은 서적들을 출력해 보세요.

```
select * from books where bname not like '%프로그래밍' ;
```

BID	BNAME	PUBLISHER	PRICE	INPUTDATE
1 database	데이터 베이스 개론	대한 출판사	(null)	78/03/01
2 mybatis	마이바티스	대한 출판사	(null)	45/08/15
3 spring	스프링 Framework	소셜 미디어	20000	50/06/25
4 story	프로그래밍 이야기	지한 출판사	6000	70/09/09
5 timeseries	시계열 예측	지한 출판사	13000	50/06/25

- 서적의 단가가 결정이 된 서적들을 출력해 보세요.

```
select * from books where price is not null ;
```

BID	BNAME	PUBLISHER	PRICE	INPUTDATE
1 java	자바 프로그래밍	대한 출판사	7000	70/08/28
2 jsp	jsp 프로그래밍	대한 Books	8000	80/08/15
3 python	파이썬 프로그래밍	민국 미디어	9000	90/07/17
4 spring	스프링 Framework	소셜 미디어	20000	50/06/25
5 xml	xml 프로그래밍	소셜 미디어	13000	70/06/06
6 story	프로그래밍 이야기	지한 출판사	6000	70/09/09
7 timeseries	시계열 예측	지한 출판사	13000	50/06/25

## 연산자의 복합적 사용

- 출판사 이름이 '대한 출판사'이면서 단가가 10,000원 이하인 서적들을 출력해 보세요.

```
select * from books where publisher not in('대한 출판사') and price <= 10000 ;
```

BID	BNAME	PUBLISHER	PRICE	INPUTDATE
1	jsp	jsp 프로그래밍 대한 Books	8000	80/08/15
2	python	파이썬 프로그래밍 민국 미디어	9000	90/07/17
3	story	프로그래밍 이야기 지한 출판사	6000	70/09/09

- 서적의 이름이 '프로그래밍'으로 끝나는 서적 중에서 단가가 8,000원 이하인 서적들을 출력해 보세요.

```
select * from books where bname like '%프로그래밍' and price <= 8000 ;
```

BID	BNAME	PUBLISHER	PRICE	INPUTDATE
1	java	자바 프로그래밍 대한 출판사	7000	70/08/28
2	jsp	jsp 프로그래밍 대한 Books	8000	80/08/15

- 서적의 단가가 결정이 된 서적 중에서 단가가 8,000원과 13,000원 사이에 있지 않은 서적들을 출력해 보세요.

```
select * from books where (price is not null) and (price not between 8000 and 13000) ;
```

BID	BNAME	PUBLISHER	PRICE	INPUTDATE
1	java	자바 프로그래밍 대한 출판사	7000	70/08/28
2	spring	스프링 Framework 소셜 미디어	20000	50/06/25
3	story	프로그래밍 이야기 지한 출판사	6000	70/09/09

## 데이터 정렬하기

- 서적의 단가가 낮은 것부터 출력해 보세요.

```
select * from books order by price ;
```

BID	BNAME	PUBLISHER	PRICE	INPUTDATE
1	story	프로그래밍 이야기 지한 출판사	6000	70/09/09
2	java	자바 프로그래밍 대한 출판사	7000	70/08/28
3	jsp	jsp 프로그래밍 대한 Books	8000	80/08/15
4	python	파이썬 프로그래밍 민국 미디어	9000	90/07/17
5	xml	xml 프로그래밍 소셜 미디어	13000	70/06/06
6	timeseries	시계열 예측 지한 출판사	13000	50/06/25
7	spring	스프링 Framework 소셜 미디어	20000	50/06/25
8	mybatis	마이바티스 대한 출판사	(null)	45/08/15
9	r	R 프로그래밍 민국 미디어	(null)	75/11/11
10	database	데이터 베이스 개론 대한 출판사	(null)	78/03/01

- 서적의 이름에 대하여 오름차순으로 출력해 보세요.

```
select * from books order by bname asc ;
```

BID	BNAME	PUBLISHER	PRICE	INPUTDATE
1 r	R 프로그래밍	민국 미디어	(null)	75/11/11
2 jsp	jsp 프로그래밍	대한 Books	8000	80/08/15
3 xml	xml 프로그래밍	소셜 미디어	13000	70/06/06
4 database	데이터 베이스 개론	대한 출판사	(null)	78/03/01
5 mybatis	마이바티스	대한 출판사	(null)	45/08/15
6 spring	스프링 Framework	소셜 미디어	20000	50/06/25
7 timeseries	시계열 예측	지한 출판사	13000	50/06/25
8 java	자바 프로그래밍	대한 출판사	7000	70/08/28
9 python	파이썬 프로그래밍	민국 미디어	9000	90/07/17
10 story	프로그래밍 이야기	지한 출판사	6000	70/09/09

- 서적의 이름에 대하여 내림차순으로 출력해 보세요.

```
select * from books order by bname desc ;
```

BID	BNAME	PUBLISHER	PRICE	INPUTDATE
1 story	프로그래밍 이야기	지한 출판사	6000	70/09/09
2 python	파이썬 프로그래밍	민국 미디어	9000	90/07/17
3 java	자바 프로그래밍	대한 출판사	7000	70/08/28
4 timeseries	시계열 예측	지한 출판사	13000	50/06/25
5 spring	스프링 Framework	소셜 미디어	20000	50/06/25
6 mybatis	마이바티스	대한 출판사	(null)	45/08/15
7 database	데이터 베이스 개론	대한 출판사	(null)	78/03/01
8 xml	xml 프로그래밍	소셜 미디어	13000	70/06/06
9 jsp	jsp 프로그래밍	대한 Books	8000	80/08/15
10 r	R 프로그래밍	민국 미디어	(null)	75/11/11

- 입고 일자가 빠른 것 부터 정렬해 보세요.

```
select * from books order by inputdate ;
```

BID	BNAME	PUBLISHER	PRICE	INPUTDATE
1 timeseries	시계열 예측	지한 출판사	13000	50/06/25
2 spring	스프링 Framework	소셜 미디어	20000	50/06/25
3 xml	xml 프로그래밍	소셜 미디어	13000	70/06/06
4 java	자바 프로그래밍	대한 출판사	7000	70/08/28
5 story	프로그래밍 이야기	지한 출판사	6000	70/09/09
6 r	R 프로그래밍	민국 미디어	(null)	75/11/11
7 database	데이터 베이스 개론	대한 출판사	(null)	78/03/01
8 jsp	jsp 프로그래밍	대한 Books	8000	80/08/15
9 python	파이썬 프로그래밍	민국 미디어	9000	90/07/17
10 mybatis	마이바티스	대한 출판사	(null)	45/08/15

- 출판사 이름으로 내림차순 정렬 후, 단가가 낮은 서적부터 정렬해 보세요.

```
select * from books order by publisher desc, price asc ;
```

BID	BNAME	PUBLISHER	PRICE	INPUTDATE
1 story	프로그래밍 이야기	지한 출판사	6000	70/09/09
2 timeseries	시계열 예측	지한 출판사	13000	50/06/25
3 xml	xml 프로그래밍	소셜 미디어	13000	70/06/06
4 spring	스프링 Framework	소셜 미디어	20000	50/06/25
5 python	파이썬 프로그래밍	민국 미디어	9000	90/07/17
6 r	R 프로그래밍	민국 미디어	(null)	75/11/11
7 java	자바 프로그래밍	대한 출판사	7000	70/08/28
8 mybatis	마이바티스	대한 출판사	(null)	45/08/15
9 database	데이터 베이스 개론	대한 출판사	(null)	78/03/01
10 jsp	jsp 프로그래밍	대한 Books	8000	80/08/15

## 8. 집합 연산자

- '곰돌이' 사용자의 테이블 'Books'와 CTAS 기법을 사용하여 다음과 같은 데이터를 만들어 보세요.  
이 테이블을 생성한 다음 물음에 답하세요.
- 출판사의 이름이 '대한'으로 시작하는 도서의 목록을 구하여 테이블 branchA를 생성하세요.

```
create table branchA as
select * from books
where publisher like '대한%';

select * from branchA ;
```

BID	BNAME	PUBLISHER	PRICE	INPUTDATE
1 java	자바 프로그래밍	대한 출판사	7000	70/08/28
2 jsp	jsp 프로그래밍	대한 Books	8000	80/08/15
3 database	데이터 베이스 개론	대한 출판사	(null)	78/03/01
4 mybatis	마이바티스	대한 출판사	(null)	45/08/15

- 서적의 이름이 '그래밍'으로 끝나는 도서의 목록을 구하여 테이블 branchB를 생성하세요.

```
create table branchB as
select * from books
where bname like '%그래밍';

select * from branchB ;
```

BID	BNAME	PUBLISHER	PRICE	INPUTDATE
1 java	자바 프로그래밍	대한 출판사	7000	70/08/28
2 jsp	jsp 프로그래밍	대한 Books	8000	80/08/15
3 python	파이썬 프로그래밍	민국 미디어	9000	90/07/17
4 r	R 프로그래밍	민국 미디어	(null)	75/11/11
5 xml	xml 프로그래밍	소셜 미디어	13000	70/06/06

- 2개 도서의 정보를 합친 결과를 출력하되, 중복된 행은 하나로 출력해 보세요.

```
select * from branchA union select * from branchB ;
```

BID	BNAME	PUBLISHER	PRICE	INPUTDATE
1 database	데이터 베이스 개론	대한 출판사	(null)	78/03/01
2 java	자바 프로그래밍	대한 출판사	7000	70/08/28
3 jsp	jsp 프로그래밍	대한 Books	8000	80/08/15
4 mybatis	마이바티스	대한 출판사	(null)	45/08/15
5 python	파이썬 프로그래밍	민국 미디어	9000	90/07/17
6 r	R 프로그래밍	민국 미디어	(null)	75/11/11
7 xml	xml 프로그래밍	소셜 미디어	13000	70/06/06

- 두 지점의 서적 목록을 합쳐서 출력해 주세요. 단, 중복된 데이터는 모두 출력되어야 합니다.

```
select * from branchA union all select * from branchB ;
```

BID	BNAME	PUBLISHER	PRICE	INPUTDATE
1 java	자바 프로그래밍	대한 출판사	7000	70/08/28
2 jsp	jsp 프로그래밍	대한 Books	8000	80/08/15
3 database	데이터 베이스 개론	대한 출판사	(null)	78/03/01
4 mybatis	마이바티스	대한 출판사	(null)	45/08/15
5 java	자바 프로그래밍	대한 출판사	7000	70/08/28
6 jsp	jsp 프로그래밍	대한 Books	8000	80/08/15
7 python	파이썬 프로그래밍	민국 미디어	9000	90/07/17
8 r	R 프로그래밍	민국 미디어	(null)	75/11/11
9 xml	xml 프로그래밍	소셜 미디어	13000	70/06/06

- 2개 도서의 정보를 합친 결과를 출력하되, 서적 이름 순으로 정렬하여 보여 주세요.

order by 구문을 사용하고자 할 때는, 모든 컬럼을 일일이 열거해 줘야 합니다.

```
select bid, bname, publisher, price from branchA
union
select bid, bname, publisher, price from branchB
order by bname asc ;
```

BID	BNAME	PUBLISHER	PRICE
1 r	R 프로그래밍	민국 미디어	(null)
2 jsp	jsp 프로그래밍	대한 Books	8000
3 xml	xml 프로그래밍	소셜 미디어	13000
4 database	데이터 베이스 개론	대한 출판사	(null)
5 mybatis	마이바티스	대한 출판사	(null)
6 java	자바 프로그래밍	대한 출판사	7000
7 python	파이썬 프로그래밍	민국 미디어	9000

- 2개 도서의 정보를 합친 결과를 출력하되, 할인된 도서 가격 정보도 같이 조회해 보세요.

- BranchA 지점은 10% 할인된, BranchB 지점은 20% 할인된 정보를 출력하도록 한다.

```
select bid, bname, publisher, price, (1-0.1) * price as saleprice from branchA
union
select bid, bname, publisher, price, (1-0.2) * price as saleprice from branchB
order by bname asc ;
```

BID	BNAME	PUBLISHER	PRICE	SALEPRICE
1 r	R 프로그래밍	민국 미디어	(null)	(null)
2 jsp	jsp 프로그래밍	대한 Books	8000	6400
3 jsp	jsp 프로그래밍	대한 Books	8000	7200
4 xml	xml 프로그래밍	소셜 미디어	13000	10400
5 database	데이터 베이스 개론	대한 출판사	(null)	(null)
6 mybatis	마이바티스	대한 출판사	(null)	(null)
7 java	자바 프로그래밍	대한 출판사	7000	5600
8 java	자바 프로그래밍	대한 출판사	7000	6300
9 python	파이썬 프로그래밍	민국 미디어	9000	7200

- 2개의 지점에서 동일하게 보유하고 있는 도서 목록을 출력해 보세요.

```
select bid, bname, publisher, price from branchA
intersect
select bid, bname, publisher, price from branchB ;
```

BID	BNAME	PUBLISHER	PRICE
1 java	자바 프로그래밍	대한 출판사	7000
2 jsp	jsp 프로그래밍	대한 Books	8000

- BranchA 지점에서만 보유하고 있는 도서 목록을 출력해 보세요.

```
select * from branchA minus select * from branchB ;
```

BID	BNAME	PUBLISHER	PRICE	INPUTDATE
1 database	데이터 베이스 개론	대한 출판사	(null)	78/03/01
2 mybatis	마이바티스	대한 출판사	(null)	45/08/15

- BranchB 지점에서만 보유하고 있는 도서 목록을 출력해 보세요.

```
select * from branchB minus select * from branchA ;
```

BID	BNAME	PUBLISHER	PRICE	INPUTDATE
1 python	파이썬 프로그래밍	민국 미디어	9000	90/07/17
2 r	R 프로그래밍	민국 미디어	(null)	75/11/11
3 xml	xml 프로그래밍	소셜 미디어	13000	70/06/06



- 2개의 지점에서 동일하게 보유하고 있지 않는 도서 목록을 출력해 보세요.
- 방법1 : 합집합과 교집합을 구한 다음 차집합을 이용하면 됩니다.

```
(select * from branchA union select * from branchB)
minus
(select * from branchA intersect select * from branchB) ;
```

BID	BNAME	PUBLISHER	PRICE	INPUTDATE
1 database	데이터 베이스 개론	대한 출판사	(null)	78/03/01
2 mybatis	마이바티스	대한 출판사	(null)	45/08/15
3 python	파이썬 프로그래밍	민국 미디어	9000	90/07/17
4 r	R 프로그래밍	민국 미디어	(null)	75/11/11
5 xml	xml 프로그래밍	소셜 미디어	13000	70/06/06

- 방법2 : 각각의 차집합을 구한 다음 합쳐도 됩니다.

```
(select * from branchA minus select * from branchB)
union
(select * from branchB minus select * from branchA) ;
```

BID	BNAME	PUBLISHER	PRICE	INPUTDATE
1 database	데이터 베이스 개론	대한 출판사	(null)	78/03/01
2 mybatis	마이바티스	대한 출판사	(null)	45/08/15
3 python	파이썬 프로그래밍	민국 미디어	9000	90/07/17
4 r	R 프로그래밍	민국 미디어	(null)	75/11/11
5 xml	xml 프로그래밍	소셜 미디어	13000	70/06/06

- 서적의 이름과 단가만 출력해 주세요. 단, 중복된 데이터는 하나로 출력되어야 합니다.

```
select bname, price from branchA union select bname, price from branchB ;
```

	BNAME	PRICE
1	R 프로그래밍	(null)
2	jsp 프로그래밍	8000
3	xml 프로그래밍	13000
4	데이터 베이스 개론	(null)
5	마이바티스	(null)
6	자바 프로그래밍	7000
7	파이썬 프로그래밍	9000

## 9. 함수 다루기

- '곰돌이' 사용자의 'Books' 테이블에 대하여 적절한 함수를 사용하여 풀어 보세요

문자열 함수부터 먼저 풀어 보도록 하겠습니다.

서적의 이름을 대문자, 소문자 등으로 변경해 보세요.

```
select bname, upper(bname), lower(bname), initcap(bname) from books ;
```

⚡ BNAME	⚡ UPPER(BNAME)	⚡ LOWER(BNAME)	⚡ INITCAP(BNAME)
1 자바 프로그래밍	자바 프로그래밍	자바 프로그래밍	자바 프로그래밍
2 jsp 프로그래밍	JSP 프로그래밍	jsp 프로그래밍	Jsp 프로그래밍
3 파이썬 프로그래밍	파이썬 프로그래밍	파이썬 프로그래밍	파이썬 프로그래밍
4 R 프로그래밍	R 프로그래밍	r 프로그래밍	R 프로그래밍
5 데이터 베이스 개론	데이터 베이스 개론	데이터 베이스 개론	데이터 베이스 개론
6 마이바티스	마이바티스	마이바티스	마이바티스
7 스프링 Framework	스프링 FRAMEWORK	스프링 framework	스프링 Framework
8 xml 프로그래밍	XML 프로그래밍	xml 프로그래밍	Xml 프로그래밍
9 프로그래밍 이야기	프로그래밍 이야기	프로그래밍 이야기	프로그래밍 이야기
10 시계열 예측	시계열 예측	시계열 예측	시계열 예측

- 반드시 concat 함수를 사용하여 다음 예시와 같이 출력되도록 하세요.

제목 : 자바 프로그래밍, 출판사 : 대한 출판사

```
select concat(concat('제목 : ', bname), concat(', 출판사 : ', publisher))
from books ;
```

⚡ CONCAT(CONCAT('제목:',BNAME),CONCAT(',출판사:',PUBLISHER))
1 제목 : 자바 프로그래밍, 출판사 : 대한 출판사
2 제목 : jsp 프로그래밍, 출판사 : 대한 Books
3 제목 : 파이썬 프로그래밍, 출판사 : 민국 미디어
4 제목 : R 프로그래밍, 출판사 : 민국 미디어
5 제목 : 데이터 베이스 개론, 출판사 : 대한 출판사
6 제목 : 마이바티스, 출판사 : 대한 출판사
7 제목 : 스프링 Framework, 출판사 : 소셜 미디어
8 제목 : xml 프로그래밍, 출판사 : 소셜 미디어
9 제목 : 프로그래밍 이야기, 출판사 : 지한 출판사
10 제목 : 시계열 예측, 출판사 : 지한 출판사

- 서적 이름의 5번째 글자부터 끝까지 추출해 보세요.

```
select bname, substr(bname, 5) from books ;
```

⚡ BNAME	⚡ SUBSTR(BNAME,5)
1 자바 프로그래밍	로그래밍
2 jsp 프로그래밍	프로그래밍
3 파이썬 프로그래밍	프로그래밍
4 R 프로그래밍	그래밍
5 데이터 베이스 개론	베이스 개론
6 마이바티스	스
7 스프링 Framework	Framework
8 xml 프로그래밍	프로그래밍
9 프로그래밍 이야기	밍 이야기
10 시계열 예측	예측

- 서적 이름의 앞 3글자만 추출해 보세요.

```
select bname, substr(bname, 1, 3) from books ;
```

BNAME	SUBSTR(BNAME,1,3)
1 자바 프로그래밍	자바
2 jsp 프로그래밍	jsp
3 파이썬 프로그래밍	파이썬
4 R 프로그래밍	R 프
5 데이터 베이스 개론	데이터
6 마이바티스	마이바
7 스프링 Framework	스프링
8 xml 프로그래밍	xml
9 프로그래밍 이야기	프로그
10 시계열 예측	시계열

- 서적과 출판사의 이름과 이름의 길이를 조회해 보세요.

```
select bname, length(bname), publisher, length(publisher) from books ;
```

BNAME	LENGTH(BNAME)	PUBLISHER	LENGTH(PUBLISHER)
1 자바 프로그래밍	8	대한 출판사	6
2 jsp 프로그래밍	9	대한 Books	8
3 파이썬 프로그래밍	9	민국 미디어	6
4 R 프로그래밍	7	민국 미디어	6
5 데이터 베이스 개론	10	대한 출판사	6
6 마이바티스	5	대한 출판사	6
7 스프링 Framework	13	소셜 미디어	6
8 xml 프로그래밍	9	소셜 미디어	6
9 프로그래밍 이야기	9	지한 출판사	6
10 시계열 예측	6	지한 출판사	6

- 서적 이름이 9자리 이상인 항목만 조회해 보세요.

```
select * from books where length(bname) >= 9;
```

BID	BNAME	PUBLISHER	PRICE	INPUTDATE
1 jsp	jsp 프로그래밍	대한 Books	8000	80/08/15
2 python	파이썬 프로그래밍	민국 미디어	9000	90/07/17
3 database	데이터 베이스 개론	대한 출판사	(null)	78/03/01
4 spring	스프링 Framework	소셜 미디어	20000	50/06/25
5 xml	xml 프로그래밍	소셜 미디어	13000	70/06/06
6 story	프로그래밍 이야기	지한 출판사	6000	70/09/09

- instr 함수()를 사용하여 책의 이름에 '프로그래밍'라는 단어가 들어있는 행만 조회하세요.

```
select * from books where instr(bname, '프로그래밍') > 0 ;
```

BID	BNAME	PUBLISHER	PRICE	INPUTDATE
1 java	자바 프로그래밍	대한 출판사	7000	70/08/28
2 jsp	jsp 프로그래밍	대한 Books	8000	80/08/15
3 python	파이썬 프로그래밍	민국 미디어	9000	90/07/17
4 r	R 프로그래밍	민국 미디어	(null)	75/11/11
5 xml	xml 프로그래밍	소셜 미디어	13000	70/06/06
6 story	프로그래밍 이야기	지한 출판사	6000	70/09/09

- lpad 함수를 사용하여 price 컬럼을 전체 10자리 형식으로 채워 주세요.

모자란 공간은 '\*' 기호로 채우도록 하세요.

```
select bid, bname, publisher, lpad(price, 10, '*')
from books ;
```

BID	BNAME	PUBLISHER	LPAD(PRICE,10,'*')
1 java	자바 프로그래밍	대한 출판사	*****7000
2 jsp	jsp 프로그래밍	대한 Books	*****8000
3 python	파이썬 프로그래밍	민국 미디어	*****9000
4 r	R 프로그래밍	민국 미디어	(null)
5 database	데이터 베이스 개론	대한 출판사	(null)
6 mybatis	마이바티스	대한 출판사	(null)
7 spring	스프링 Framework	소셜 미디어	*****20000
8 xml	xml 프로그래밍	소셜 미디어	*****13000
9 story	프로그래밍 이야기	지한 출판사	*****6000
10 timeseries	시계열 예측	지한 출판사	*****13000

- price 컬럼의 값과 (책이름의 문자열 길이)의 제곱 값을 더한 결과를 출력해보세요.

예를 들어서 price=8000이고, 책이름이 "자바 프로그래밍"(8자리)이면 640이므로 최종 결과는 80640이 되어야 합니다.

```
select bid, bname, publisher, price, price + length(bname) * length(bname)
from books ;
```

BID	BNAME	PUBLISHER	PRICE	PRICE+LENGTH(BNAME)*LENGTH(BNAME)
1 java	자바 프로그래밍	대한 출판사	7000	7064
2 jsp	jsp 프로그래밍	대한 Books	8000	8081
3 python	파이썬 프로그래밍	민국 미디어	9000	9081
4 r	R 프로그래밍	민국 미디어	(null)	(null)
5 database	데이터 베이스 개론	대한 출판사	(null)	(null)
6 mybatis	마이바티스	대한 출판사	(null)	(null)
7 spring	스프링 Framework	소셜 미디어	20000	20169
8 xml	xml 프로그래밍	소셜 미디어	13000	13081
9 story	프로그래밍 이야기	지한 출판사	6000	6081
10 timeseries	시계열 예측	지한 출판사	13000	13036

- 서적 이름에 '프로그래밍'을 'programming'으로 치환해 보세요.

```
select bname, replace(bname, '프로그래밍', 'programming') from books ;
```

BNAME	REPLACE(BNAME,'프로그래밍','PROGRAMMING')
1 자바 프로그래밍	자바 programming
2 jsp 프로그래밍	jsp programming
3 파이썬 프로그래밍	파이썬 programming
4 R 프로그래밍	R programming
5 데이터 베이스 개론	데이터 베이스 개론
6 마이바티스	마이바티스
7 스프링 Framework	스프링 Framework
8 xml 프로그래밍	xml programming
9 프로그래밍 이야기	programming 이야기
10 시계열 예측	시계열 예측

- rtrim() 함수를 사용하여 서적 이름에 '프로그래밍'이라는 단어를 삭제해 보세요.

```
select bname, rtrim(bname, ' 프로그래밍 ') from books ;
```

BNAME	RTRIM(BNAME, '프로그래밍')
1 자바 프로그래밍	자바
2 jsp 프로그래밍	jsp
3 파이썬 프로그래밍	파이썬
4 R 프로그래밍	R
5 데이터 베이스 개론	데이터 베이스 개론
6 마이바티스	마이바티스
7 스프링 Framework	스프링 Framework
8 xml 프로그래밍	xml
9 프로그래밍 이야기	프로그래밍 이야기
10 시계열 예측	시계열 예측

- 서적의 아이디의 길이가 3의 배수인 행들만 조회해 보세요.

```
select * from books where mod(length(bid), 3) = 0 ;
```

BID	BNAME	PUBLISHER	PRICE	INPUTDATE
1 jsp	jsp 프로그래밍	대한 Books	8000	80/08/15
2 python	파이썬 프로그래밍	민국 미디어	9000	90/07/17
3 spring	스프링 Framework	소셜 미디어	20000	50/06/25
4 xml	xml 프로그래밍	소셜 미디어	13000	70/06/06

- 다음은 customers 테이블에서 실습하도록 합니다.  
주소 컬럼에서 국가 이름만 출력해 보세요.

```
select name, substr(address, 1, instr(address, ' ')) from customers ;
```

NAME	SUBSTR(ADDRESS,1,INSTR(ADDRESS,' '))
1 윤봉길	중국
2 김유신	신라
3 신사임당	백제
4 안중근	조선
5 홍범도	일제

- instr 함수()를 사용하여 전화 번호에 '3333'이 들어 있는 행만 조회해 보세요.

```
select * from customers where instr(phone, '3333') > 0 ;
```

ID	NAME	ADDRESS	PHONE	MANAGER	SALARY
1 shin	신사임당	백제 공주	000-3333-3333	(null)	300

- substr 함수()를 사용하여, 성씨를 제외하고, 이름만 추출해 보세요.

```
select name, substr(name, 2) from customers ;
```

NAME	SUBSTR(NAME,2)
1 윤봉길	봉길
2 김유신	유신
3 신사임당	사임당
4 안중근	중근
5 홍범도	범도

- 숫자형 함수 실습을 위하여 다음 문장을 우선 실행하도록 합니다.

```
update books set price = 1.1 * price ;
update books set price = -123.4567 where bid in('r', 'database', 'python') ;
commit ;
```

- 단가에 대한 절대 값을 구해 보세요.

```
select bid, bname, price, abs(price) from books ;
```

BID	BNAME	PRICE	ABS(PRICE)
1 java	자바 프로그래밍	7700	7700
2 jsp	jsp 프로그래밍	8800	8800
3 python	파이썬 프로그래밍	-123.4567	123.4567
4 r	R 프로그래밍	-123.4567	123.4567
5 database	데이터 베이스 개론	-123.4567	123.4567
6 mybatis	마이바티스	(null)	(null)
7 spring	스프링 Framework	22000	22000
8 xml	xml 프로그래밍	14300	14300
9 story	프로그래밍 이야기	6600	6600
10 timeseries	시계열 예측	14300	14300

- 서적의 아이디가 'r', 'database', 'mybatis', 'spring'인 서적에 대하여 가격을 올림/버림해 보세요.

```
select bid, bname, price, ceil(price), floor(price) from books
where bid in('r', 'database', 'mybatis', 'spring');
```

BID	BNAME	PRICE	CEIL(PRICE)	FLOOR(PRICE)
1 database	데이터 베이스 개론	-123.4567	-123	-124
2 mybatis	마이바티스	(null)	(null)	(null)
3 r	R 프로그래밍	-123.4567	-123	-124
4 spring	스프링 Framework	22000	22000	22000

- 단가들을 정수 값으로 반올림 및 절삭해보세요.

```
select bid, bname, price, round(price), trunc(price) from books ;
```

BID	BNAME	PRICE	ROUND(PRICE)	TRUNC(PRICE)
1 java	자바 프로그래밍	7700	7700	7700
2 jsp	jsp 프로그래밍	8800	8800	8800
3 python	파이썬 프로그래밍	-123.4567	-123	-123
4 r	R 프로그래밍	-123.4567	-123	-123
5 database	데이터 베이스 개론	-123.4567	-123	-123
6 mybatis	마이바티스	(null)	(null)	(null)
7 spring	스프링 Framework	22000	22000	22000
8 xml	xml 프로그래밍	14300	14300	14300
9 story	프로그래밍 이야기	6600	6600	6600
10 timeseries	시계열 예측	14300	14300	14300

- 모든 단가에 대하여 절대 값으로 변경하고, 루트를 씌운 결과를 출력하세요.

```
select bname, sqrt(abs(price)), round(sqrt(abs(price)), 3) from books ;
```

BNAME	SQRT(ABS(PRICE))	ROUND(SQRT(ABS(PRICE)),3)
1 자바 프로그래밍	87.74964387392122060406388307416309560875	87.75
2 jsp 프로그래밍	93.80831519646859109131260227088932561176	93.808
3 파이썬 프로그래밍	11.11110705555481541639651584896590489796	11.111
4 R 프로그래밍	11.11110705555481541639651584896590489796	11.111
5 데이터 베이스 개론	11.11110705555481541639651584896590489796	11.111
6 마이바티스	(null)	(null)
7 스프링 Framework	148.32396974191325897422794881601426122	148.324
8 xml 프로그래밍	119.58260743101398021129840756195616614	119.583
9 프로그래밍 이야기	81.24038404635960360459883568266040348504	81.24
10 시계열 예측	119.58260743101398021129840756195616614	119.583

- 서적 이름의 길이가 5의 배수인 서적들을 출력해 보세요.

```
select * from books where mod(length(bname), 5) = 0 ;
```

BID	BNAME	PUBLISHER	PRICE	INPUTDATE
1 database	데이터 베이스 개론	대한 출판사	-123.4567	78/03/01
2 mybatis	마이바티스	대한 출판사	(null)	45/08/15

- 서적 이름의 길이를 3제곱하는 문장을 만들어 보세요.

```
select bid, bname, power(length(bname), 3) from books ;
```

BID	BNAME	POWER(LENGTH(BNAME),3)
1 java	자바 프로그래밍	512
2 jsp	jsp 프로그래밍	729
3 python	파이썬 프로그래밍	729
4 r	R 프로그래밍	343
5 database	데이터 베이스 개론	1000
6 mybatis	마이바티스	125
7 spring	스프링 Framework	2197
8 xml	xml 프로그래밍	729
9 story	프로그래밍 이야기	729
10 timeseries	시계열 예측	216

- 단가에서 서적 이름의 길이의 2제곱을 뺄셈하는 문장을 만들어 보세요.

```
select bname, length(bname), price, price - power(length(bname), 2) from books ;
```

BNAME	LENGTH(BNAME)	PRICE	PRICE-POWER(LENGTH(BNAME),2)
1 자바 프로그래밍	8	7700	7636
2 jsp 프로그래밍	9	8800	8719
3 파이썬 프로그래밍	9	-123.4567	-204.4567
4 R 프로그래밍	7	-123.4567	-172.4567
5 데이터 베이스 개론	10	-123.4567	-223.4567
6 마이바티스	5	(null)	(null)
7 스프링 Framework	13	22000	21831
8 xml 프로그래밍	9	14300	14219
9 프로그래밍 이야기	9	6600	6519
10 시계열 예측	6	14300	14264

- 서적의 단가에 부호 함수를 적용해 보세요.

```
select bname, price, sign(price) from books ;
```

BNAME	PRICE	SIGN(PRICE)
1 자바 프로그래밍	7700	1
2 jsp 프로그래밍	8800	1
3 파이썬 프로그래밍	-123.4567	-1
4 R 프로그래밍	-123.4567	-1
5 데이터 베이스 개론	-123.4567	-1
6 마이바티스	(null)	(null)
7 스프링 Framework	22000	1
8 xml 프로그래밍	14300	1
9 프로그래밍 이야기	6600	1
10 시계열 예측	14300	1



- 절대 값 함수인 abs를 사용하지 않고 서적의 단가에 대하여 절대값으로 변경해 보세요.

```
select bname, price, price*sign(price) from books ;
```

◆ BNAME	◆ PRICE	◆ PRICE*SIGN(PRICE)
1 자바 프로그래밍	7700	7700
2 jsp 프로그래밍	8800	8800
3 파이썬 프로그래밍	-123.4567	123.4567
4 R 프로그래밍	-123.4567	123.4567
5 데이터 베이스 개론	-123.4567	123.4567
6 마이바티스	(null)	(null)
7 스프링 Framework	22000	22000
8 xml 프로그래밍	14300	14300
9 프로그래밍 이야기	6600	6600
10 시계열 예측	14300	14300

## 날짜 함수

- 서적들의 이름과 입고 날짜부터 경과한 일수를 조회해 보세요.

```
alter session set nls_date_format = 'yyyy-mm-dd hh:mi:ss' ;
select bname, inputdate, round(sysdate - inputdate, 3) as "입고 경과 날짜" from books ;
```

◆ BNAME	◆ INPUTDATE	◆ 입고 경과 날짜
1 자바 프로그래밍	1970-08-28 12:00:00	18552.72
2 jsp 프로그래밍	1980-08-15 12:00:00	14912.72
3 파이썬 프로그래밍	1990-07-17 12:00:00	11289.72
4 R 프로그래밍	1975-11-11 12:00:00	16651.72
5 데이터 베이스 개론	1978-03-01 12:00:00	15810.72
6 마이바티스	2045-08-15 12:00:00	-8828.28
7 스프링 Framework	1950-06-25 12:00:00	25921.72
8 xml 프로그래밍	1970-06-06 12:00:00	18635.72
9 프로그래밍 이야기	1970-09-09 12:00:00	18540.72
10 시계열 예측	1950-06-25 12:00:00	25921.72

- 서적들의 이름과 입고 날짜부터 경과한 월수를 소수점 3째자리까지 보여 주세요.

```
select bname, inputdate, round(months_between(sysdate, inputdate), 3) as passdate from books ;
```

	BNAME	INPUTDATE	PASSDATE
1	자바 프로그래밍	1970-08-28 12:00:00	609.539
2	jsp 프로그래밍	1980-08-15 12:00:00	489.959
3	파이썬 프로그래밍	1990-07-17 12:00:00	370.894
4	R 프로그래밍	1975-11-11 12:00:00	547.088
5	데이터 베이스 개론	1978-03-01 12:00:00	519.41
6	마이바티스	2045-08-15 12:00:00	-290.041
7	스프링 Framework	1950-06-25 12:00:00	851.636
8	xml 프로그래밍	1970-06-06 12:00:00	612.249
9	프로그래밍 이야기	1970-09-09 12:00:00	609.152
10	시계열 예측	1950-06-25 12:00:00	851.636

- 서적의 아이디가 'r', 'database'인 서적의 입고 일자를 3개월 전으로 변경하세요.  
변경 전의 데이터와 변경 후의 데이터를 동시에 출력해 봅니다.

```
update books set inputdate = add_months(inputdate, -3) where bid in('r', 'database');
select * from books where bid in('r', 'database');
```

BID	BNAME	PUBLISHER	PRICE	INPUTDATE
1 database	데이터 베이스 개론	대한 출판사	-123.4567	1977-12-01 12:00:00
2 r	R 프로그래밍	민국 미디어	-123.4567	1975-08-11 12:00:00

- 서적의 아이디가 'python', 'spring'인 서적의 입고 일자를 5일전으로 변경해 보세요.  
변경 전의 데이터와 변경 후의 데이터를 동시에 출력해 봅니다.

```
update books set inputdate = inputdate - 5 where bid in('python', 'spring');
select * from books where bid in('python', 'spring');
```

BID	BNAME	PUBLISHER	PRICE	INPUTDATE
1 python	파이썬 프로그래밍	민국 미디어	-123.4567	1990-07-12 12:00:00
2 spring	스프링 Framework	소셜 미디어	22000	1950-06-20 12:00:00

- 아이디가 'python', 'spring'인 서적의 입고 일자를 돌아오는 금요일로 변경해 보세요.

```
update books set inputdate = next_day(sysdate, '금') where bid in('python', 'spring');
select * from books where bid in('python', 'spring');
```

BID	BNAME	PUBLISHER	PRICE	INPUTDATE
1	python	파이썬 프로그래밍	민국 미디어	-123.4567 2021-06-18 05:20:51
2	spring	스프링 Framework	소셜 미디어	22000 2021-06-18 05:20:51

## 타입 변환 함수

- 서적의 입고 일자를 '월/일' 형식으로 출력해 보세요.

서적의 입고 일자를 'yyyy年 mm월 dd일' 형식으로 출력해 보세요.

```
select bname, to_char(inputdate, 'mm/dd') from books ;
select bname, to_char(inputdate, 'yyyy"年" mm"월" dd"일"') from books ;
```

BNAME	TO_CHAR(INPUTDATE, 'MM/DD')	BNAME	TO_CHAR(INPUTDATE, 'YYYY"年"MM"월"DD"일"')
1 자바 프로그래밍	08/28	1 자바 프로그래밍	1970年 08월 28일
2 jsp 프로그래밍	08/15	2 jsp 프로그래밍	1980年 08월 15일
3 파이썬 프로그래밍	05/14	3 파이썬 프로그래밍	2021年 05월 14일
4 R 프로그래밍	08/11	4 R 프로그래밍	1975年 08월 11일
5 데이터 베이스 개론	12/01	5 데이터 베이스 개론	1977年 12월 01일
6 마이바티스	08/15	6 마이바티스	2045年 08월 15일
7 스프링 Framework	05/14	7 스프링 Framework	2021年 05월 14일
8 xml 프로그래밍	06/06	8 xml 프로그래밍	1970年 06월 06일
9 프로그래밍 이야기	09/09	9 프로그래밍 이야기	1970年 09월 09일
10 시계열 예측	06/25	10 시계열 예측	1950年 06월 25일

- 다음 문장을 실행하고, 그 결과를 예측해 보세요.

```
select
to_char(price, '$999,999.00') as price01,
to_char(price, '$000,000.00') as price02,
to_char(price, 'L999,999.00') as price03,
to_char(price, '000,000.00') as price04
from books ;
```

	PRICE01	PRICE02	PRICE03	PRICE04
1	\$7,700.00	\$007,700.00	₩7,700.00	007,700.00
2	\$8,800.00	\$008,800.00	₩8,800.00	008,800.00
3	-\$123.46	-\$000,123.46	-₩123.46	-000,123.46
4	-\$123.46	-\$000,123.46	-₩123.46	-000,123.46
5	-\$123.46	-\$000,123.46	-₩123.46	-000,123.46
6	(null)	(null)	(null)	(null)
7	\$22,000.00	\$022,000.00	₩22,000.00	022,000.00
8	\$14,300.00	\$014,300.00	₩14,300.00	014,300.00
9	\$6,600.00	\$006,600.00	₩6,600.00	006,600.00
10	\$14,300.00	\$014,300.00	₩14,300.00	014,300.00

- null 처리 함수 실습을 위하여 다음 문장을 우선 실습해 주세요.

```
update books set inputdate = null where bid in ('python', 'spring') ;
commit ;
```

- 서적의 이름과 입고 일자를 출력해 보세요.

입고 일자가 존재하지 않는 데이터는 현재 시각으로 출력해보세요.

```
select bname, inputdate, nvl(inputdate, sysdate) newdata from books;
```

	BNAME	INPUTDATE	NEWDATA
1	자바 프로그래밍	1970-08-28 12:00:00	1970-08-28 12:00:00
2	jsp 프로그래밍	1980-08-15 12:00:00	1980-08-15 12:00:00
3	파이썬 프로그래밍	(null)	2021-05-07 02:19:56
4	R 프로그래밍	1975-08-11 12:00:00	1975-08-11 12:00:00
5	데이터 베이스 개론	1977-12-01 12:00:00	1977-12-01 12:00:00
6	마이바티스	2045-08-15 12:00:00	2045-08-15 12:00:00
7	스프링 Framework	(null)	2021-05-07 02:19:56
8	xml 프로그래밍	1970-06-06 12:00:00	1970-06-06 12:00:00
9	프로그래밍 이야기	1970-09-09 12:00:00	1970-09-09 12:00:00
10	시계열 예측	1950-06-25 12:00:00	1950-06-25 12:00:00

- 서적의 이름과 단가를 출력해 보세요.

단가가 존재하지 않는 데이터는 1,000원으로 출력해보세요.

```
select bname, price, nvl(price, 1000) newdata from books ;
```

	BNAME	PRICE	NEWDATA
1	자바 프로그래밍	8470	8470
2	jsp 프로그래밍	9680	9680
3	파이썬 프로그래밍	-123.4567	-123.4567
4	R 프로그래밍	-123.4567	-123.4567
5	데이터 베이스 개론	-123.4567	-123.4567
6	마이바티스	(null)	1000
7	스프링 Framework	24200	24200
8	xml 프로그래밍	15730	15730
9	프로그래밍 이야기	7260	7260
10	시계열 예측	15730	15730

- 서적들의 이름과 단가를 조화해 보세요.

단가의 존재 여부에 대하여 코멘트('단가 존재함', '단가가 null임')를 작성해 보세요.

```
select bname, price, nvl2(price, '단가 존재함', '단가가 null임') from books ;
```

	BNAME	PRICE	NVL2(PRICE, '단가 존재함', '단가가 null임')
1	자바 프로그래밍	8470	단가 존재함
2	jsp 프로그래밍	9680	단가 존재함
3	파이썬 프로그래밍	-123.4567	단가 존재함
4	R 프로그래밍	-123.4567	단가 존재함
5	데이터 베이스 개론	-123.4567	단가 존재함
6	마이바티스	(null)	단가가 null임
7	스프링 Framework	24200	단가 존재함
8	xml 프로그래밍	15730	단가 존재함
9	프로그래밍 이야기	7260	단가 존재함
10	시계열 예측	15730	단가 존재함

- 서적의 이름과 출판사를 조회하시오.

단, 출판사 '소셜 미디어'인 출판사는 null 값으로 대체 하세요.

```
select bname, publisher, nullif(publisher, '소셜 미디어') as result from books ;
```

BNAME	PUBLISHER	RESULT
1 자바 프로그래밍	대한 출판사	대한 출판사
2 jsp 프로그래밍	대한 Books	대한 Books
3 파이썬 프로그래밍	민국 미디어	민국 미디어
4 R 프로그래밍	민국 미디어	민국 미디어
5 데이터 베이스 개론	대한 출판사	대한 출판사
6 마이바티스	대한 출판사	대한 출판사
7 스프링 Framework	소셜 미디어	(null)
8 xml 프로그래밍	소셜 미디어	(null)
9 프로그래밍 이야기	지한 출판사	지한 출판사
10 시계열 예측	지한 출판사	지한 출판사

- 출판사의 이름이 '민국 미디어'이면 'm'을, '대한 출판사'이면 'd'를, '지한 출판사'이면 's'를 나머지는 'x'를 출력하는 코드를 작성해 보세요.

```
select bid, bname, publisher, decode(publisher, '민국 미디어', 'm', '대한 출판사', 'd', '지한 출판사', 's', 'x') as result from books ;
```

BID	BNAME	PUBLISHER	RESULT
1 java	자바 프로그래밍	대한 출판사	d
2 jsp	jsp 프로그래밍	대한 Books	x
3 python	파이썬 프로그래밍	민국 미디어	m
4 r	R 프로그래밍	민국 미디어	m
5 database	데이터 베이스 개론	대한 출판사	d
6 mybatis	마이바티스	대한 출판사	d
7 spring	스프링 Framework	소셜 미디어	x
8 xml	xml 프로그래밍	소셜 미디어	x
9 story	프로그래밍 이야기	지한 출판사	s
10 timeseries	시계열 예측	지한 출판사	s

- 위의 문제를 case 표현식으로 다시 풀어 보세요.

```
select bid, bname, publisher,
case publisher
when '민국 미디어' then 'm'
when '대한 출판사' then 'd'
when '지한 출판사' then 's'
else 'x'
end
as result
from books ;
```

BID	BNAME	PUBLISHER	RESULT
1 java	자바 프로그래밍	대한 출판사	d
2 jsp	jsp 프로그래밍	대한 Books	x
3 python	파이썬 프로그래밍	민국 미디어	m
4 r	R 프로그래밍	민국 미디어	m
5 database	데이터 베이스 개론	대한 출판사	d
6 mybatis	마이바티스	대한 출판사	d
7 spring	스프링 Framework	소셜 미디어	x
8 xml	xml 프로그래밍	소셜 미디어	x
9 story	프로그래밍 이야기	지한 출판사	s
10 timeseries	시계열 예측	지한 출판사	s

- 서적들의 아이디, 이름, 출판사, 단가 및 파생 결과를 출력해보세요. 파생 결과는 출판사가 '민국 미디어'이면, 단가의 10% 인상액, 출판사가 '대한 출판사'이면, 20%, 출판사가 '지한 출판사'이면, 30%, 나머지는 원래 단가를 그대로 출력해야 합니다. 단, 단가가 null이면 기본 값으로 100원을 지정하도록 합니다.

```
select bid, bname, publisher, price,
case publisher
when '민국 미디어' then 1.1*nvl(price, 100)
when '대한 출판사' then 1.2*nvl(price, 100)
when '지한 출판사' then 1.3*nvl(price, 100)
else nvl(price, 100) end as newprice
from books ;
```

BID	BNAME	PUBLISHER	PRICE	NEWPRICE
1 java	자바 프로그래밍	대한 출판사	7700	9240
2 jsp	jsp 프로그래밍	대한 Books	8800	8800
3 python	파이썬 프로그래밍	민국 미디어	-123.4567	-135.80237
4 r	R 프로그래밍	민국 미디어	-123.4567	-135.80237
5 database	데이터 베이스 개론	대한 출판사	-123.4567	-148.14804
6 mybatis	마이바티스	대한 출판사	(null)	120
7 spring	스프링 Framework	소셜 미디어	22000	22000
8 xml	xml 프로그래밍	소셜 미디어	14300	14300
9 story	프로그래밍 이야기	지한 출판사	6600	8580
10 timeseries	시계열 예측	지한 출판사	14300	18590

## orders 테이블

- 아래의 실습을 하기 위하여 다음 문장을 수행하세요.

```
update orders set saleprice = null where oid in(1, 3, 5) ;
update orders set saledate = null where oid in(1, 2, 3) ;
commit ;
```

- orders 테이블에 대하여 다음과 같이 출력이 되는 sql 구문을 작성해 보세요.

cid 컬럼과 case 구문을 사용하여 고객 아이디와 이름을 출력해 보세요.

단, 판매가가, 존재하지 않으면 10,000원으로 출력하세요.

판매가는 21,000일때, ₩21,000.00의 형식으로 출력하세요.

단, 주문 일자가, 존재하지 않으면 현재 시각으로 출력하세요.

주문 일자는 '2014年 07月 13日'의 형식으로 출력하세요.

OID	CID	RESULT	BID	SALEPRICE	SALEDATE
1	1 yoon	윤종길	java	₩10,000.00	2021年 05月 07日
2	2 yoon	윤종길	python	₩21,000.00	2021年 05月 07日
3	3 yusin	김유신	database	₩10,000.00	2021年 05月 07日
4	4 shin	신사임당	jsp	₩06,000.00	2021年 07月 04日
5	5 an	안중근	spring	₩10,000.00	2021年 07月 05日
6	6 yoon	윤종길	jsp	₩12,000.00	2021年 07月 07日
7	7 an	안중근	jsp	₩13,000.00	2021年 08月 15日
8	8 shin	신사임당	java	₩12,000.00	2021年 03月 01日
9	9 yusin	김유신	java	₩07,000.00	2021年 12月 25日
10	10 shin	신사임당	python	₩13,000.00	2021年 07月 10日

```

select oid, cid,
case cid
when 'yoon' then '윤봉길'
when 'yusin' then '김유신'
when 'hong' then '홍범도'
when 'shin' then '신사임당'
when 'an' then '안중근'
else '기타' end as result,
bid,
to_char(nvl(saleprice, 10000), 'L00,000.00') as saleprice,
to_char(nvl(saledate, sysdate), 'yyyy"年" mm"月" dd"日"') as saledate
from orders ;

```

## 10. 그룹(집계) 함수

- books 테이블에 대하여 다음과 같은 집계 함수 사용법에 대한 물음에 필요한 sql 구문을 작성해 보세요. 차후 실습을 위하여 다음 문장을 우선 수행하도록 합니다.

```

update books set price = abs(price) ;
commit ;

```

- 총 서적은 몇 권인가요?

```
select count(*) from books;
```

COUNT(*)
1 10

- 다음은 단가가 정해진 서적은 총 몇 권인가요?

```
select count(price) from books;
```

COUNT(PRICE)
1 9

c

- ount(\*)와 count(price)를 사용하여 단가가 null인 행 개수를 구해 보세요.

```
select (count(*) - count(price)) as result from books;
```

RESULT
1 1

- 출판사의 갯수는 총 몇 개인가요?

```
select count(distinct publisher) as 출판사_개수 from books;
```

	출판사_개수
1	5

- 모든 서적들의 단가의 평균 값을 소수점 2째자리까지 구해 보세요.

```
select round(avg(price), 2) as result from books;
```

	RESULT
1	8230.04

- 단가의 평균을 구하되, null인 서적은 100으로 치환하여 계산하세요.

```
select round(avg(nvl(price, 100)), 2) as result from books;
```

	RESULT
1	7417.04

- 단가의 총합을 구해 보세요.

```
select sum(price) from books;
```

	SUM(PRICE)
1	74070.3701

- 단가의 총금액을 구하되, null인 서적은 100으로 치환하여 계산하세요.

```
select sum(nvl(price, 100)) from books;
```

	SUM(NVL(PRICE,100))
1	74170.3701

- 책의 최저 단가와 최고 단가는 각각 얼마인가?

```
select max(price), min(price) from books ;
```

	MAX(PRICE)	MIN(PRICE)
1	22000	123.4567



- 출판사 이름을 기준으로 오름차순 정렬시 가장 먼저 조회되는 출판사는 어디인가요?  
출판사 이름을 기준으로 오름차순 정렬시 가장 나중에 조회되는 출판사는 어디인가요?

```
select min(publisher), max(publisher) from books;
```

MIN(PUBLISHER)	MAX(PUBLISHER)
1 대한 Books	지한 출판사

- 출판사별로 책들의 단가의 총합을 구해 보세요.

```
select publisher, sum(price)
from books
group by publisher ;
```

PUBLISHER	SUM(PRICE)
1 소셜 미디어	36300
2 대한 Books	8800
3 민국 미디어	246.9134
4 대한 출판사	7823.4567
5 지한 출판사	20900

- 출판사별로 각각 몇 권의 서적을 가지고 있는지 확인해 보세요.

```
select publisher, count(*) as cnt from books
group by publisher order by publisher desc ;
```

PUBLISHER	CNT
1 지한 출판사	2
2 소셜 미디어	2
3 민국 미디어	2
4 대한 출판사	3
5 대한 Books	1

- 출판사별로 책들의 단가의 총합과 최소 값을 구해보세요.  
단, 단가가 null이면 10,000원으로 치환하여 계산해야 합니다.

```
select publisher, sum(nvl(price, 10000)) as sumsal, min(nvl(price, 10000)) as minsal
from books group by publisher ;
```

PUBLISHER	SUMSAL	MINSAL
1 소셜 미디어	36300	14300
2 대한 Books	8800	8800
3 민국 미디어	246.9134	123.4567
4 대한 출판사	17823.4567	123.4567
5 지한 출판사	20900	6600

- 출판사별로 책들의 단가의 총합을 구하되, '민국 미디어'는 제외하도록 하고, 단가의 총합이 큰 것부터 정렬하세요.

```
select publisher, sum(price)
from books
where publisher not in('민국 미디어')
group by publisher
order by sum(price) desc ;
```

PUBLISHER	SUM(PRICE)
1 소셜 미디어	36300
2 지한 출판사	20900
3 대한 Books	8800
4 대한 출판사	7823.4567

- 출판사별로 책들의 단가의 총합을 구해 보세요.  
단, 단가의 총합이 20,000이상인 항목만 조회하세요.

```
select publisher, sum(price) from books
group by publisher having sum(price) >= 20000 ;
```

PUBLISHER	SUM(PRICE)
1 소셜 미디어	36300
2 지한 출판사	20900

- 출판사별로 책들의 단가의 총합을 구해 보세요.  
단, 단가의 총합이 20,000이상이고, 30,000이하인 항목만 조회하세요.

```
select publisher, sum(price) from books
group by publisher having sum(price) >= 20000 and sum(price) <= 30000 ;
```

PUBLISHER	SUM(PRICE)
1 지한 출판사	20900

- 시퀀스 실습을 위하여 주문(orders) 테이블의 데이터를 모두 삭제합니다.

시퀀스 seqorder를 생성하고, 해당 시퀀스를 사용하여 주문(orders) 테이블을 다시 생성하세요.

[illegible]

	OID	CID	BID	SALEPRICE	SALEDATE
1	1	yoon	java	6000	21/07/01
2	2	yoon	python	21000	21/07/03
3	3	yusin	database	8000	21/07/03
4	4	shin	jsp	6000	21/07/04
5	5	an	spring	20000	21/07/05
6	6	yoon	jsp	12000	21/07/07
7	7	an	jsp	13000	21/08/15
8	8	shin	java	12000	21/03/01
9	9	yusin	java	7000	21/12/25
10	10	shin	python	13000	21/07/10

## 12. 조인(Join)

- 고객의 이름과 주문한 책의 아이디를 조회해 보세요.

```
select customers.name, orders.bid
from customers join orders
on customers.id = orders.cid
order by customers.name, orders.bid ;
```

	NAME	BID
1	김유신	database
2	김유신	java
3	신사임당	java
4	신사임당	jsp
5	신사임당	python
6	안중근	jsp
7	안중근	spring
8	윤봉길	java
9	윤봉길	jsp
10	윤봉길	python

- 고객의 이름과 주문한 책의 아이디를 조회해 보세요.  
단, 고객의 이름이 '안중근', '윤봉길'인 사람들만 조회하도록 합니다.

```
select e.name, b.bid
from customers e join orders b
on e.id = b.cid
and e.name in('안중근', '윤봉길')
order by e.name, b.bid ;
```

	NAME	BID
1	안중근	jsp
2	안중근	spring
3	윤봉길	java
4	윤봉길	jsp
5	윤봉길	python

- 다음과 같이 등급을 의미하는 테이블을 생성합니다.

```
create table grades(
    glevel varchar2(2),
    lowsai number,
    highsai number
);
insert into grades values('A', 0, 99) ;
insert into grades values('B', 100, 199) ;
insert into grades values('C', 200, 299) ;
insert into grades values('D', 300, 399) ;
insert into grades values('E', 400, 500) ;
commit ;

select * from grades;
```

	⚡ GLEVEL	⚡ LOWSAL	⚡ HIGHSAL
1 A		0	99
2 B		100	199
3 C		200	299
4 D		300	399
5 E		400	500

- 생성한 등급 테이블을 이용하여 각 고객들의 등급 정보를 출력해 보세요.

```
select e.name, e.salary, g.glevel
from customers e, grades g
where e.salary between g.losal and g.hisal
order by e.name;
```

	⚡ NAME	⚡ SALARY	⚡ GLEVEL
1	김유신	200	C
2	신사임당	300	D
3	안중근	400	E
4	윤봉길	100	B
5	홍범도	500	E

- 고객의 이름과 주문한 책의 아이디와 판매 금액을 조회해 보세요.  
단, 주문하지 않은 고객들의 이름 정보도 출력이 되어야 합니다.

```
select e.name, b.bid, b.saleprice
from customers e left outer join orders b
on e.id = b.cid ;
```

	⚡ NAME	⚡ BID	⚡ SALEPRICE
1	윤봉길	java	6000
2	윤봉길	python	21000
3	김유신	database	8000
4	신사임당	jsp	6000
5	안중근	spring	20000
6	윤봉길	jsp	12000
7	안중근	jsp	13000
8	신사임당	java	12000
9	김유신	java	7000
10	신사임당	python	13000
11	홍범도	(null)	(null)

- 고객과 매니저 정보를 같이 출력해 보세요.  
매니저에 대한 정보는 manager 컬럼을 참조하면 됩니다.

```
select emp.name || '의 관리자는 ' || mgr.name || '입니다.' as result
from customers emp inner join customers mgr
on emp.manager = mgr.id ;
```

RESULT
1 윤봉길의 관리자는 신사임당입니다.
2 김유신의 관리자는 신사임당입니다.
3 안중근의 관리자는 홍범도입니다.

- 고객과 매니저 정보를 같이 출력해 보세요.  
매니저도 하나의 행으로 출력이 되어야 합니다.

```
select emp.name || '의 관리자는 ' || nvl(mgr.name, '본인') || '입니다.' as result
from customers emp left outer join customers mgr
on emp.manager = mgr.id ;
```

RESULT
1 김유신의 관리자는 신사임당입니다.
2 윤봉길의 관리자는 신사임당입니다.
3 안중근의 관리자는 홍범도입니다.
4 홍범도의 관리자는 본인입니다.
5 신사임당의 관리자는 본인입니다.

- 각 고객들의 이름과 주문 건수를 출력해 보세요.

```
select e.name, count(*) as cnt
from customers e inner join orders b
on e.id=b.cid group by e.name order by e.name desc ;
```

NAME	CNT
1 윤봉길	3
2 안중근	2
3 신사임당	3
4 김유신	2

- 각 고객들의 이름과 주문 건수를 출력해 보세요.  
단, 주문하지 않은 고객들의 이름 정보도 출력이 되어야 합니다.

```
select e.name, count(cid) as cnt
from customers e left outer join orders b
on e.id=b.cid group by e.name order by cnt desc, e.name asc ;
```

NAME	CNT
1 신사임당	3
2 윤봉길	3
3 김유신	2
4 안중근	2
5 홍범도	0

## 13. 서브 쿼리

- 평균 급여보다 더 많은 급여를 받는 고객들의 이름과 급여를 출력해 보겠습니다.

```
select name, salary from customers
where salary >= (select avg(salary) from customers) ;
```

	NAME	SALARY
1	신사임당	300
2	안중근	400
3	홍범도	500

- 관리자의 이름인 '신사임당'인 고객들의 아이디, 이름, 관리자 아이디를 출력해 보세요.  
이 문제는 관리자의 이름 무엇인지를 확인하는 구문이 서브 쿼리에 놓여야 합니다.

```
select id, name, manager from customers
where manager = (select id from customers where name = '신사임당') ;
```

	ID	NAME	MANAGER
1	yoon	윤봉길	shin
2	yusin	김유신	shin

- 관리자의 이름인 '신사임당'인 고객들의 아이디, 이름, 관리자 아이디를 출력해 보세요. 단, '신사임당'도 같이 출력되어야 합니다. 추가 조건은 and나 or 연산자를 사용하여 처리하면 됩니다.

```
select id, name, manager from customers
where manager = (select id from customers where name = '신사임당') or name = '신사임당' ;
```

	ID	NAME	MANAGER
1	yoon	윤봉길	shin
2	yusin	김유신	shin
3	shin	신사임당	(null)

- '안중근'의 급여보다 적은 급여를 받는 고객들의 이름과 급여를 조회해 보세요.

```
select name, salary from customers
where salary < (select salary from customers where name = '안중근') ;
```

	NAME	SALARY
1	윤봉길	100
2	김유신	200
3	신사임당	300

- 관리자가 '신사임당'이면서, 전체 회원의 평균 급여 보다 적은 급여를 받는 회원의 아이디, 이름, 급여를 조회해 보세요.

```
select id, name, salary from customers
where manager = (select id from customers where name = '신사임당')
and salary < (select avg(salary) from customers );
```

ID	NAME	SALARY
1 yoon	윤봉길	100
2 yusin	김유신	200

- 관리자의 이름이 '신사임당'이거나 '홍범도'인 고객들의 아이디와 이름과 관리자 아이디를 조회해 보겠습니다.

두 관리자에 대한 서브 쿼리이므로 in 키워드를 사용하면 됩니다.

```
select id, name, manager from customers
where manager in (select id from customers where name in ('신사임당', '홍범도')) ;
```

ID	NAME	MANAGER
1 an	안중근	hong
2 yoon	윤봉길	shin
3 yusin	김유신	shin

- 주문을 한번이라도 한 고객 정보를 조회해 보겠습니다.

주문은 한 사람이 여러 번 작성할 수 있으므로 중복 데이터 배제를 위한 키워드 distinct를 사용하는 게 효율적입니다.

```
select distinct cid from orders where cid is not null ;
```

CID
1 yoon
2 an
3 yusin
4 shin

```
select name, address, phone from customers
where id in (select distinct cid from orders where cid is not null) ;
```

NAME	ADDRESS	PHONE
1 안중근	조선 강원도	000-4444-4444
2 신사임당	백제 공주	000-3333-3333
3 윤봉길	중국 상하이	000-1111-1111
4 김유신	신라 경주	000-2222-2222



- 매니저의 아이디가 'shin'이 아닌 모든 고객들의 정보를 출력해 보세요.  
관리자 아이디가 'shin'인 회원도 목록에서 배제되어야 합니다.

```
select name, address, phone, salary from customers
where id not in (select id from customers where manager in ('shin'))
and id <> 'shin'
order by name ;
```

	NAME	ADDRESS	PHONE	SALARY
1	안중근	조선 강원도	000-4444-4444	400
2	홍범도	일제 시대 간도	(null)	500

- 관리자가 '신사임당'인 회원들을 이름과 급여를 조회해 보겠습니다.

```
select name, salary from customers where manager = 'shin' ;
```

	NAME	SALARY
1	윤봉길	100
2	김유신	200

- 전체 고객 중에서 관리자가 '신사임당'인 고객들 중에서 최저 급여를 받는 자보다 더 많이 급여를 받는 고객들의 이름과 급여를 조회해 보세요.

```
select name, salary from customers
where salary > any (select salary from customers where manager = 'shin') ;
```

	NAME	SALARY
1	홍범도	500
2	안중근	400
3	신사임당	300
4	김유신	200

- 전체 고객 중에서 관리자가 '신사임당'인 고객들 중에서 최대 급여를 받는 고객보다 더 적게 받는 고객들의 이름과 급여를 조회해 보세요.

```
select name, salary from customers
where salary < any (select salary from customers where manager = 'shin') ;
```

	NAME	SALARY
1	윤봉길	100

- 전체 고객 중에서 관리자가 '신사임당'인 고객 중에서 최대 급여자 보다 많이 받는 고객들을 조회해 보겠습니다.

```
select name, salary from customers
where salary > all (select salary from customers where manager = 'shin') ;
```

	NAME	SALARY
1	신사임당	300
2	안중근	400
3	홍범도	500

- 전체 고객 중에서 관리자가 '신사임당'인 고객 중에서 최저 급여를 받은 자보다 더 적게 받는 고객들의 이름과 급여를 조회해 보세요. 조회되는 데이터는 0건입니다.

```
select name, salary from customers
where salary < all (select salary from customers where manager = 'shin') ;
```

	NAME	SALARY
--	------	--------

## 다중 컬럼 서브 쿼리

- 지금까지 사용한 서브 쿼리는 where 절에 비교 하기 위한 컬럼의 개수가 1개이었었습니다. 필요에 따라서, 2개 이상의 서브 쿼리를 사용할 수 있습니다. 이러한 서브 쿼리를 다중 컬럼 서브 쿼리라고 합니다.

- 출판사별로 최저 단가를 가진 서적의 단가를 조회해 보겠습니다. 우선 다음과 같이 출판사별로 최저 단가의 정보를 우선 조회해 봅니다.

```
select publisher, min(price) from books group by publisher ;
```

	PUBLISHER	MIN(PRICE)
1	소셜 미디어	14300
2	대한 Books	8800
3	민국 미디어	123.4567
4	대한 출판사	123.4567
5	지한 출판사	6600

- 방금 사용했던 문장을 서브 쿼리로 두고, 메인 쿼리의 where 절에 'where (gender, salary)'의 형식으로 작성을 하면 다중 컬럼 서브 쿼리가 됩니다.

```
select bname, price, publisher from books
where (publisher, price) in (select publisher, min(price) from books group by publisher) ;
```

	BNAME	PRICE	PUBLISHER
1	jsp 프로그래밍	8800	대한 Books
2	파이썬 프로그래밍	123.4567	민국 미디어
3	R 프로그래밍	123.4567	민국 미디어
4	데이터 베이스 개론	123.4567	대한 출판사
5	xml 프로그래밍	14300	소셜 미디어
6	프로그래밍 이야기	6600	지한 출판사

## 14. 제약 조건

- 곱돌이 사용자의 모든 테이블에 대하여 데이터 사전을 이용하여 제약 조건을 확인해 보도록 하세요.

```
select table_name, constraint_name, constraint_type, status, search_condition
from user_constraints ;
```

	TABLE_NAME	CONSTRAINT_NAME	CONSTRAINT_TYPE	STATUS	SEARCH_CONDITION
1	BOOKS	SYS_C008687	P	ENABLED	(null)
2	CUSTOMERS	SYS_C008680	P	ENABLED	(null)
3	ORDERS	SYS_C008681	P	ENABLED	(null)

- 데이터 사전 user\_cons\_columns와 조인한 결과를 출력해 보세요.

```
select t.constraint_name, t.table_name, c.column_name, t.constraint_type, t.status,
t.search_condition
from user_constraints t join user_cons_columns c
on t.table_name = c.table_name and t.constraint_name = c.constraint_name ;
```

	CONSTRAINT_NAME	TABLE_NAME	COLUMN_NAME	CONSTRAINT_TYPE	STATUS	SEARCH_CONDITION
1	SYS_C008680	CUSTOMERS	ID	P	ENABLED	(null)
2	SYS_C008687	BOOKS	BID	P	ENABLED	(null)
3	SYS_C008681	ORDERS	OID	P	ENABLED	(null)

- 각 테이블의 제약 조건의 이름을 그림과 같이 변경해 보세요.

```
alter table BOOKS rename constraint SYS_C008687 to BOOKS_BID_PK ;
alter table CUSTOMERS rename constraint SYS_C008680 to CUSTOMERS_ID_PK ;
alter table ORDERS rename constraint SYS_C008681 to ORDERS_OID_PK ;
```

TABLE_NAME	CONSTRAINT_NAME	CONSTRAINT_TYPE	STATUS	SEARCH_CONDITION
1 BOOKS	BOOKS_BID_PK	P	ENABLED	(null)
2 CUSTOMERS	CUSTOMERS_ID_PK	P	ENABLED	(null)
3 ORDERS	ORDERS_OID_PK	P	ENABLED	(null)

- 모든 테이블에 대한 제약 조건의 정보를 다시 확인해 보도록 합니다.

```
select t.constraint_name, t.table_name, c.column_name, t.constraint_type, t.status,
       t.search_condition
from user_constraints t join user_cons_columns c
on t.table_name = c.table_name and t.constraint_name = c.constraint_name ;
```

CONSTRAINT_NAME	TABLE_NAME	COLUMN_NAME	CONSTRAINT_TYPE	STATUS	SEARCH_CONDITION
1 BOOKS_BID_PK	BOOKS	BID	P	ENABLED	(null)
2 ORDERS_OID_PK	ORDERS	OID	P	ENABLED	(null)
3 CUSTOMERS_ID_PK	CUSTOMERS	ID	P	ENABLED	(null)

## 15. 참조 무결성 제약 조건

- 주문 테이블과 고객 테이블에 대하여 참조 무결성 제약 조건을 설정해 보세요.

고객의 id 컬럼과 주문 테이블의 cid 컬럼이 개념적으로 동일한 컬럼입니다.

```
alter table orders
add constraint orders_cid_fk
foreign key(cid) references customers(id) ;
```

- 다음 문장을 사용하여 고객 '윤봉길'의 탈퇴 테스트를 진행해 보세요. 탈퇴가 가능한지 파악해보고, 왜 이러한 이유가 발생했는지 설명해 보세요.

```
delete from customers where id = 'an' ;
```

삭제가 되지 않는 이유는, 해당 고객 '윤봉길'은 이미 주문을 몇 건 하였으므로, 참조 무결성 제약 조건에 의하여 삭제가 불가능합니다.

- 다음 문장이 왜 오류가 발생하는 지 설명해 보세요.

```
update orders set cid = 'choi' ;
```

고객의 아이디가 'choi'인 고객은 존재하지 않습니다.

따라서, 참조 무결성 제약 조건에 의하여 위반 사항이 발생하였습니다.

- 고객이 회원 탈퇴를 하는 경우, 해당 주문 매출 내역은 그대로 보존이 되어야 합니다.  
이를 위하여 제약 조건을 변경해 보세요.

```
alter table orders drop constraint orders_cid_fk ;

alter table orders add constraint orders_cid_fk
foreign key(cid) references customers(id) on delete set null ;
```

- 다음과 같이 고객 탈퇴 테스트를 수행해 보세요.

#### 고객 탈퇴 테스트

주문 테이블에 remark(히스토리를 저장할 컬럼) 컬럼을 추가합니다.  
주문 테이블에서 고객 'yusin'이 있는지 확인합니다.  
고객 테이블에서 고객 'yusin'을 삭제합니다.  
주문 테이블에서 고객 'yusin'의 컬럼 cid와 remark 컬럼 정보를 확인합니다.  
롤백을 합니다.

```
alter table orders add(remark varchar2(255)) ;
select * from orders ;
update orders set remark = '고객 yusin' || '이(가) ' || sysdate || ' 시점에 탈퇴함'
where cid = 'yusin' ;
delete from customers where id = 'yusin' ;
select * from orders ;
rollback ;
```

- 특정 서적이 삭제가 될 때, 해당 서적과 관련된 매출은 자동으로 사라진다고 가정하겠습니다.  
이를 위하여 제약 조건을 변경해 보세요.

```
alter table orders add constraint orders_bid_fk
foreign key(bid) references books(bid) on delete cascade ;
```

- 다음과 같이 서적 탈퇴 테스트를 수행해 보세요.

## 서적 삭제 테스트

주문 테이블에 서적 'jsp'가 존재하는 지 확인합니다.  
 서적 테이블에서 서적 'jsp'를 삭제합니다.  
 주문 테이블에 서적 'jsp'가 삭제 되었는 지 확인합니다.  
 롤백을 합니다.

```
select * from orders where bid = 'jsp' ;
delete from books where bid = 'jsp' ;
select * from orders where bid = 'jsp' ;
rollback ;
```

- 주문을 한 고객의 이름, 주소, 서적의 이름, 판매 금액, 단가를 조회하되 서적의 이름으로 내림차순, 사람 이름을 오름차순으로 정렬해보세요.

```
select name, address, bname, saleprice, price
from (customers c inner join orders o
on c.id=o.cid) inner join books b
on o.bid=b.bid
order by bname desc, name asc ;
```

	NAME	ADDRESS	BNAME	SALEPRICE	PRICE
1	신사임당	백제 공주	파이썬 프로그래밍	13000	123.4567
2	윤봉길	중국 상하이	파이썬 프로그래밍	21000	123.4567
3	김유신	신라 경주	자바 프로그래밍	7000	7700
4	신사임당	백제 공주	자바 프로그래밍	12000	7700
5	윤봉길	중국 상하이	자바 프로그래밍	6000	7700
6	안중근	조선 강원도	스프링 Framework	20000	22000
7	김유신	신라 경주	데이터 베이스 개론	8000	123.4567
8	신사임당	백제 공주	jsp 프로그래밍	6000	8800
9	안중근	조선 강원도	jsp 프로그래밍	13000	8800
10	윤봉길	중국 상하이	jsp 프로그래밍	12000	8800

## 16. 뷰(View)

- 출판사의 이름이 '민국 미디어'인 출판사의 아이디, 서적 이름, 출판사 이름을 보여주는 뷰 View01를 생성해 보도록 하세요.

```
create or replace view view01
as
select bid, bname, publisher from books
where publisher = '민국 미디어' ;
```

- 모든 출판사의 정보를 보여 주는 뷰 View02를 생성해 보세요.

```
create or replace view view02 as
select * from books ;
```

- 서적의 이름인 '프로그래밍'으로 끝이 나는 서적을 조회해 주는 뷰 View03를 생성해 보세요.

```
create or replace view view03 as
select * from books where bname like ' %프로그래밍 ' ;
```

- CTAS 기법을 사용하여 somebook이라는 임시 테이블을 생성합니다.

```
-- drop table somebook purge ;
create table somebook as
select * from books where bname like ' %프로그래밍 ' ;

alter table somebook add primary key(bid) ;
desc somebook ;
```

이름	널?	유형
BID	NOT NULL	VARCHAR2(30)
BNAME		VARCHAR2(40)
PUBLISHER		VARCHAR2(40)
PRICE		NUMBER
INPUTDATE		DATE

- somebook 테이블을 모두 조회하는 뷰 view04를 생성하도록 하세요.

```
create or replace view view04 as
select * from somebook ;
```

- somebook 테이블을 에서 서적의 이름, 출판사, 단가를 출력해주는 뷰 view05를 생성하도록 하세요.

```
create or replace view view05 as
select bname, publisher, price from somebook ;
```

- 다음과 같이 데이터를 추가해 봅니다.

```
insert into view04 values('nodejs', '노드 JS', '국민 출판사', 1000, sysdate) ;
select * from view04 ;
```

BID	BNAME	PUBLISHER	PRICE	INPUTDATE
1 java	자바 프로그래밍	대한 출판사	7700	70/08/28
2 jsp	jsp 프로그래밍	대한 Books	8800	80/08/15
3 python	파이썬 프로그래밍	민국 미디어	123.4567 (null)	
4 r	R 프로그래밍	민국 미디어	123.4567	75/08/11
5 xml	xml 프로그래밍	소셜 미디어	14300	70/06/06
6 nodejs	노드 JS	국민 출판사	1000	21/10/15

- 다음 구문은 왜 오류가 발생하는 지 설명해 보세요.

```
insert into view05 values('노드 JS', '국민 출판사', 1000) ;
```

기본 키인 bid 컬럼에 대한 정보를 입력할 수 없기 때문입니다.

- customers 테이블과 orders 테이블을 이용하여 각 고객들의 이름과 주문 건수를 출력해 주는 뷰 View06를 생성해 보세요.

```
create or replace view view06 as
select e.name, count(*) as cnt
from customers e inner join orders b
on e.id=b.cid group by e.name order by e.name desc ;
```

- 고객의 이름과 주문한 책의 아이디를 조회해 주는 뷰 View07를 생성해 보세요.

```
create or replace view view07 as
select customers.name, orders.bid
from customers join orders
on customers.id = orders.cid
order by customers.name, orders.bid ;

select * from view07 ;
```

	NAME	BID
1	김유신	database
2	김유신	java
3	신사임당	java
4	신사임당	jsp
5	신사임당	python
6	안중근	jsp
7	안중근	spring
8	윤봉길	java
9	윤봉길	jsp
10	윤봉길	python



- 모든 서적을 조회해 주는 뷰 View08을 생성하되, 읽기 전용으로 생성해 보세요.

```
create or replace view view08 as
select * from books with read only ;

select * from view08 ;
```

⚡ BID	⚡ BNAME	⚡ PUBLISHER	⚡ PRICE	⚡ INPUTDATE
1 java	자바 프로그래밍	대한 출판사	7700	70/08/28
2 jsp	jsp 프로그래밍	대한 Books	8800	80/08/15
3 python	파이썬 프로그래밍	민국 미디어	123.4567	(null)
4 r	R 프로그래밍	민국 미디어	123.4567	75/08/11
5 database	데이터 베이스 개론	대한 출판사	123.4567	77/12/01
6 mybatis	마이바티스	대한 출판사	(null)	45/08/15
7 spring	스프링 Framework	소셜 미디어	20000	(null)
8 xml	xml 프로그래밍	소셜 미디어	13000	70/06/06
9 story	프로그래밍 이야기	지한 출판사	6600	70/09/09
10 timeseries	시계열 예측	지한 출판사	13000	50/06/25

- 다음 구문을 사용하여 테스트를 수행해 보세요.

```
update view08 set price = 1000 ;
```

명령의 62 행에서 시작하는 중 오류 발생 -

```
update view08 set price = 1000
```

오류 발생 명령행: 62 열: 19

오류 보고 -

SQL 오류: ORA-42399: cannot perform a DML operation on a read-only view  
42399.0000 - "cannot perform a DML operation on a read-only view"

- 주문 테이블 조회시 1번에 3건씩 조회를 한다고 가정하겠습니다.

2페이지의 목록을 조회하는 topN 구문을 작성해 보세요.

```
select oid, cid, bid, saleprice, saledate, remark
from (
    select rank() over(order by oid desc) as ranking, oid, cid, bid, saleprice, saledate,
    remark
    from orders )
where ranking between 4 and 6 ;
```

⚡ OID	⚡ CID	⚡ BID	⚡ SALEPRICE	⚡ SALEDATE	⚡ REMARK
1	7 an	jsp	13000	21/08/15	(null)
2	6 yoon	jsp	12000	21/07/07	(null)
3	5 an	spring	20000	21/07/05	(null)

## 17. 데이터 제어어

- '곰돌이' 사용자와 '순돌이' 사용자에서 다음 질문 사항에 대한 sql 구문들을 작성해 보세요.

### 데이터 제어어 실습

- ① 관리자는 '곰돌이' 사용자에게 뷰 및 테이블 생성 권한을 부여하세요.
- ② 관리자는 '순돌이' 사용자에게 시퀀스, 뷰, 테이블 생성 권한을 부여하세요.
- ③ '곰돌이' 사용자는 '순돌이' 사용자에게 'books' 테이블 조회 권한을 부여 하도록 합니다.
- ④ '순돌이' 사용자는 '곰돌이' 사용자의 'books' 테이블 내용 중에서 서적의 이름이 '프로그래밍'으로 끝나는 서적을 위한 뷰 onlyprogramming을 생성하도록 합니다.

### 롤 실습하기

- ① 관리자가 사용자 bluesky 계정을 생성하고, 접속 및 resource 권한을 부여해 줍니다.
- ② 신규 롤 roletest를 생성하고, 테이블/뷰/시퀀스 생성 권한을 부여합니다.
- ③ 해당 롤을 사용자 bluesky 계정에게 부여합니다.
- ④ 사용자 bluesky는 테이블 humans을 다음과 같이 생성하도록 합니다.
- ⑤ viewtest 뷰(나이가 30이상인 사원들만 조회하는 구문)를 생성하도록 합니다.
- ⑥ 시퀀스 생성 권한을 사용하여 시퀀스 seqtest01을 생성하도록 합니다.
- ⑦ 관리자는 롤 roletest에서 뷰 생성 권한을 제거합니다.
- ⑧ 블루 스카이 사용자가 뷰 viewtest02를 생성할수 있는 지 확인해 보세요.
- ⑨ 관리자는 블루 스카이 사용자에게 부여했던 롤 roletest를 박탈하도록 합니다.
- ⑩ 관리자는 롤 roletest를 삭제하도록 합니다.

```
-- '관리자' 세션에서 실행
grant create view, create table to gomdori ;
grant create sequence, create view, create table to sundori ;

-- '곰돌이' 세션에서 실행
grant select on books to sundori ;

-- '순돌이' 세션에서 실행
select * from gomdori.books ;
```

- '순돌이' 세션에서 실행 gomdori.books 테이블을 조회한 결과입니다.

BID	BNAME	PUBLISHER	PRICE	INPUTDATE
1 java	자바 프로그래밍	대한 출판사	7700	70/08/28
2 jsp	jsp 프로그래밍	대한 Books	8800	80/08/15
3 python	파이썬 프로그래밍	민국 미디어	123.4567	(null)
4 r	R 프로그래밍	민국 미디어	123.4567	75/08/11
5 database	데이터 베이스 개론	대한 출판사	123.4567	77/12/01
6 mybatis	마이바티스	대한 출판사	(null)	45/08/15
7 spring	스프링 Framework	소셜 미디어	20000	(null)
8 xml	xml 프로그래밍	소셜 미디어	13000	70/06/06
9 story	프로그래밍 이야기	지한 출판사	6600	70/09/09
10 timeseries	시계열 예측	지한 출판사	13000	50/06/25

```
-- '순돌이' 세션에서 실행
create or replace view onlyprogramming as
select * from gomdori.books
where bname like '%프로그래밍' ;

select * from onlyprogramming ;
```

- '순돌이' 세션에서 onlyprogramming을 조회한 결과입니다.

BID	BNAME	PUBLISHER	PRICE	INPUTDATE
1 java	자바 프로그래밍	대한 출판사	7700	70/08/28
2 jsp	jsp 프로그래밍	대한 Books	8800	80/08/15
3 python	파이썬 프로그래밍	민국 미디어	123.4567	(null)
4 r	R 프로그래밍	민국 미디어	123.4567	75/08/11
5 xml	xml 프로그래밍	소셜 미디어	14300	70/06/06

## 롤 실습하기

- ① 관리자가 사용자 bluesky 계정을 생성하고, 접속 및 resource 권한을 부여해 줍니다.
- ② 신규 롤 roletest를 생성하고, 테이블/뷰/시퀀스 생성 권한을 부여합니다.
- ③ 해당 롤을 사용자 bluesky 계정에 부여합니다.
- ④ 사용자 bluesky는 테이블 humans을 다음과 같이 생성하도록 합니다.
- ⑤ viewtest 뷰(나이가 30이상인 사원들만 조회하는 구문)를 생성하도록 합니다.
- ⑥ 시퀀스 생성 권한을 사용하여 시퀀스 seqtest01을 생성하도록 합니다.
- ⑦ 관리자는 롤 roletest에서 뷰 생성 권한을 제거합니다.
- ⑧ 블루 스카이 사용자가 뷰 viewtest02를 생성할 수 있는 지 확인해 보세요.
- ⑨ 관리자는 블루 스카이 사용자에게 부여했던 롤 roletest를 박탈하도록 합니다.
- ⑩ 관리자는 롤 roletest를 삭제하도록 합니다.

- 관리자 세션에서 다음과 같이 진행합니다.

```
create user bluesky identified by oracle account unlock
default tablespace users quota unlimited on users ;
-- User BLUESKY이(가) 생성되었습니다.

grant connect, resource to bluesky ;
-- Grant을(를) 성공했습니다.

-- 해당 롤을 생성하고, 몇 가지 권한을 롤에 포함시킵니다.
-- 생성한 롤을 bluesky 계정에 부여하도록 합니다.
create role roletest ;
grant create table, create view, create sequence to roletest ;
grant roletest to bluesky ;
```

- bluesky 세션에서 확인한 롤의 정보입니다.

```
select * from role_sys_privs where role='ROLETEST' ;
```

ROLE	PRIVILEGE	ADMIN_OPTION
1 ROLETEST	CREATE TABLE	NO
2 ROLETEST	CREATE SEQUENCE	NO
3 ROLETEST	CREATE VIEW	NO

- 테이블 humans를 다음과 같이 생성해 보세요

```
create table humans(
    id varchar2(30) primary key,
    name varchar2(30),
    age number
);
insert into humans values('kang', '강감찬', 10);
insert into humans values('lee', '이순신', 50);
commit ;

select * from humans ;
```

ID	NAME	AGE
1 kang	강감찬	10
2 lee	이순신	50

- 부여 받은 뷰 생성 권한을 사용하여 viewtest를 생성해 보세요.  
viewtest 뷰는 나이가 30이상인 직원들만 조회하는 구문입니다.

```
create or replace view viewtest as
select * from humans where age >= 30 ;

select * from viewtest ;
```

ID	NAME	AGE
1 lee	이순신	50

- 부여 받은 시퀀스 생성 권한을 사용하여 시퀀스 seqtest01을 생성해 보세요.

```
create sequence seqtest01 ;
```

- 다시 관리자 세션에서 실행합니다.  
관리자는 roletest이라는 롤에서 뷰 생성 권한을 제거 시켜야 합니다.

```
revoke create view from roletest ;
-- Revoke을(를) 성공했습니다.
```

- 블루 스카이 사용자가 뷰 viewtest02를 생성할수 있는 지 확인해 보세요. 생성이 안 된다면 왜 안되나요?

```
create or replace view viewtest02
as
select * from humans ;
```

오류 보고 -

ORA-01031: insufficient privileges

01031. 00000 - "insufficient privileges"

\*Cause: An attempt was made to perform a database operation without the necessary privileges.

\*Action: Ask your database administrator or designated security administrator to grant you the necessary privileges

- 뷰 생성 권한이 불가능한 이유는 이전에 관리자가 roletest로부터 뷰 생성 권한을 박탈하였기 때문입니다. 롤에 대하여 어떠한 권한을 추가/삭제하게 되면 동적으로 사용자에게 반영이 되기 때문입니다.

- 관리자는 sundori 사원에게 부여한 roletest 롤을 제거하도록 하고, 롤을 삭제하도록 합니다.

```
revoke roletest from sundori ;
drop role roletest ;
```

## 18. 벌크 로딩

- 다음과 같은 우편 번호 데이터를 사용하여 벌크 로딩을 수행해 보세요.

파일/테이블 이름	설명
newPost.csv	우편 번호 정보를 담고 있는 엑셀 파일입니다.
post_ctl.ctl	엑셀 파일을 어떤 방식으로 테이블에 넣을 것인가를 명시해 둔 관리 장부 파일입니다.
postcodes 테이블	테러 정보들을 저장할 테이블 이름입니다.

### 벌크 로딩 실습하기

- ① 다음은 실습용 테이블을 생성합니다. sql developer나 sql 사용을 위한 command 창에서 다음과 같이 테이블을 생성하도록 합니다.
- ② 마지막으로 sqlldr 유틸리티를 사용하여 벌크 로딩을 수행하시면 됩니다. 내 컴퓨터를 사용하여 'temp'라는 임시용 폴더를 하나 생성합니다. 그리고, 오라클 설치 경로를 확인하고, cmd 창을 열고 벌크 로딩을 실행합니다.
- ③ 그러면, SQL\*Loader가 실행이 되고 있음을 알리며, 진행이 됩니다.
- ④ 마지막으로 데이터 베이스에 다음 문장을 실행하면, 57196개의 행이 조회 되어야 합니다.

## 이름) post\_ctl.ctl

```
load data
infile 'newPost.csv'
insert into table postcodes
FIELDS TERMINATED BY ','
TRAILING NULLCOLS(
    area_cd char, si_nm char, sgg_nm char, emd_nm char, ri_nm char, rd_nm char,
    udrgrnd_yn char, mt_yn char, bd_ma_sn char, bd_sb_sn char, lndn_ma_sn char,
    lndn_sb_sn char, search_word char, display_word char, display_word_dtail char
)
```

- 다음은 실습용 테이블을 생성합니다. sql developer나 sql 사용을 위한 command 창에서 다음과 같이 테이블을 생성하도록 합니다.

## 테이블 생성하기

```
create table postcodes(
    area_cd varchar2(50),
    si_nm varchar2(50),
    sgg_nm varchar2(50),
    emd_nm varchar2(50),
    ri_nm varchar2(50),
    rd_nm varchar2(50),
    udrgrnd_yn varchar2(50),
    mt_yn varchar2(50),
    bd_ma_sn varchar2(50),
    bd_sb_sn varchar2(50),
    lndn_ma_sn varchar2(50),
    lndn_sb_sn varchar2(50),
    search_word varchar2(255),
    display_word varchar2(255),
    display_word_dtail varchar2(255)
);
```

- 마지막으로 sqlldr 유틸리티를 사용하여 벌크 로딩을 수행하시면 됩니다. 내 컴퓨터를 사용하여 'temp'라는 임시용 폴더를 하나 생성합니다. 그리고, 오라클 설치 경로를 확인하고, cmd 창을 열고, 다음과 같이 실행합니다.

## 벌크 로딩 실행하기

```
c:
cd \temp
# 방법) 오라클설치경로\sqlldr.exe userid=사용자아이디/사용자비밀번호 control=컨트롤파일이름.ctl
C:\oracle\app\oracle\product\11.2.0\server\bin\sqlldr.exe userid=gomdori/omygod
control=post_ctl.ctl
```

- 그러면, 다음과 같이 SQL\*Loader가 실행이 되고 있음을 알리며, 진행이 됩니다.

```
SQL*Loader: Release 11.2.0.2.0 - Production on 화 6월 14 15:17:51 2021

Copyright (c) 1982, 2009, Oracle and/or its affiliates. All rights reserved.

Commit point reached - logical record count 64
Commit point reached - logical record count 128
... 이하 생략
Commit point reached - logical record count 797937
Commit point reached - logical record count 797941
```

- 마지막으로 데이터 베이스에 다음 문장을 실행하면, 57196개의 행이 조회 되어야 합니다.

#### 데이터 결과 확인하기

```
-- 인서트된 행수를 다음 문장으로 확인합니다.
select count(*) from postcodes ;

-- 결과 : 797941
```