Day(BE)118_DML, 서브쿼리

Date	@2022/12/26
작성일시	@December 26, 2022 9:16 AM
강의 번호	US101
Skilll	Data_SQL
Instructor	조성희
Sum up	☑
Review	☐

GitHub - Kray273/SQLLab

Day23 https://www.notion.so/Day23-2dac3079bfe044e6ac6a0e905d76b237 Day24 https://www.notion.so/Day24-690b518731314b0ba14207dfd37c4a58 Day25 ...

https://github.com/Kray273/SQLLab

Kray273/**SQLLab**

1 Contributor 0 Issues 0 Stars 0 Forks

https://s3-us-west-2.amazonaws.com/secure.notion-static.com/5574dcea-af9d-4082-aa33-312f051cb693/day2_(1).pdf

▼ Review

RDBMS

행-레코드-ROW-튜플
열-COLUMN
행과열 교차점 – FIELD(1개정보)

SQL

DDL	
DML - MANIPULATION	테이블 데이터 저장 수정 삭제 SQL
DQL	SELECT 조회
DCL	
TCL	

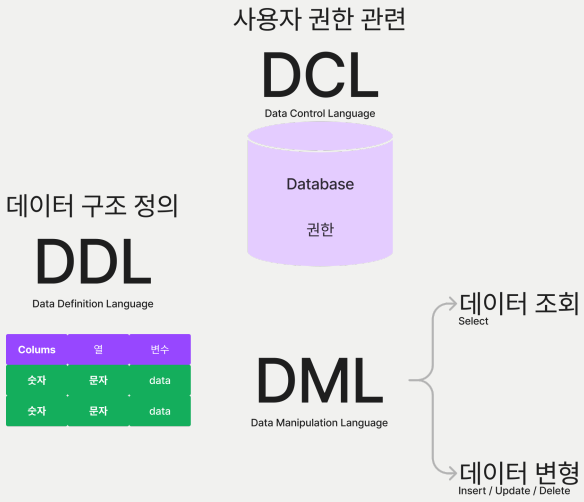
select 문법

```
SELECT 컬럼명, *, AS ALIAS, 연산식, DISTINCT, 함수 (그룹함수)
FROM 테이블명
WHERE 추출레코드조건식(레코드갯수 줄어든다)
GROUP BY 그루핑기준컬럼명
HAVING 그룹함수 조건식
ORDER BY 레코드출력순서 컬럼명1 ASC |DESC , 컬럼명2, 1, 2, ALIAS...
(LIMIT 0,10)-->MYSQL , MARIA DB-->DB 어플리케이션 데이터를 페이지 레코드수 구성

- 조건식
산술연산자 +-*/
비교연산자 = (자바 ==) > >= ....
논리연산자 AND OR NOT (자바 && ||)
목록 IN( , , , , , )
범위 BETWEEN ~ AND
```

```
패턴 LIKE ,
% - 자리수 제한없는 모든 문자
_  - 1자리 모든 문자
```

▼ DML_데이터 관리

DML : Data Manipulation Language

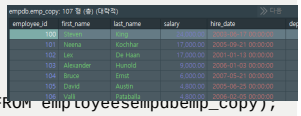


1. 명령 프롬프트	2. HEIDISQL + ...
mysql -u 사용자 -p엔터 암호 ; xxxx mysql -u 사용자 -p암호 mysql -u root -p1234 show databases; use mysql; show tables; select user, host from user; create database empdb; create user emp@'localhost' identified by '암호'; grant all p.... on empdb.* to emp@'localhost'; mysql -u emp -pemp;	1. 메뉴 – 툴 익숙 + sql 직접 입력 connection 생성 127.0.0.1,emp,emp

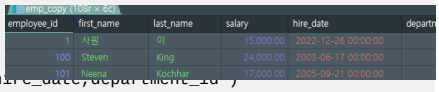
▼ 데이터 변형 문법

데이터 수정

```
-- 6장 데이터 관리 DML
-- employee 테이블 사번, 이름, 성, 급여, 입사일, 부서코드 칼럼 복사 테이블
CREATE TABLE emp_copy
(SELECT employee_id, first_name, last_name, salary, hire_date, department_id FROM employeesempdbemp_copy);
```

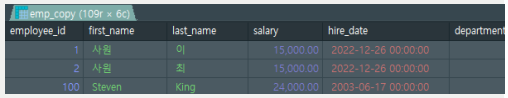


```
-- 1. 이사원 15000 2022-12-26 10 삽입
-- insert into 테이블명(칼럼명 리스트 ,,,,) values(data,,,,)
INSERT INTO emp_copy(employee_id,first_name,last_name, salary, hire_date,department_id )
  VALUES(1,'사원', "이", 15000, '2022-12-26', 10 );
SELECT * FROM emp_copy ORDER BY employee_id;
```



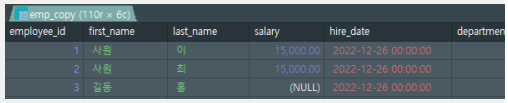
employee_id	first_name	last_name	salary	hire_date	departr
1	사원	이	15,000.00	2022-12-26 00:00:00	
100	Steven	King	24,000.00	2003-06-17 00:00:00	
101	Neena	Kochhar	17,000.00	2005-09-21 00:00:00	

```
-- 2 최사원  15000 2022-12-26 80 삽입
INSERT INTO emp_copy -- 모든 컬럼에 차례대로 들어갈 시 생략 가능
  VALUES(2,'사원', "최", 15000, '2022-12-26', 80 );
SELECT * FROM emp_copy ORDER BY employee_id;
```



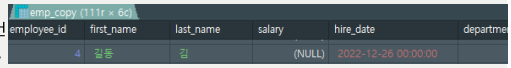
employee_id	first_name	last_name	salary	hire_date	department
1	사원	이	15,000.00	2022-12-26 00:00:00	
2	사원	최	15,000.00	2022-12-26 00:00:00	
100	Steven	King	24,000.00	2003-06-17 00:00:00	

```
-- 3 홍길동 null '2022-12-26' null  삽입
INSERT INTO emp_copy
  VALUE(3, '길동', '홍', NULL, '2022-12-26', NULL);
SELECT * FROM emp_copy ORDER BY employee_id;
```



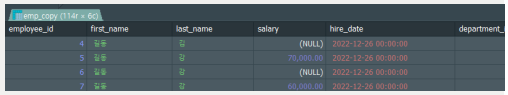
employee_id	first_name	last_name	salary	hire_date	departmen
1	사원	이	15,000.00	2022-12-26 00:00:00	
2	사원	최	15,000.00	2022-12-26 00:00:00	
3	길동	홍	(NULL)	2022-12-26 00:00:00	

```
-- 4 김길동 null null null( emp_copy 이미 입사일 not null 조건
INSERT INTO emp_copy (employee_id, first_name, last_name,
  VALUE(4 , '길동', '김', '2022-12-26');
```



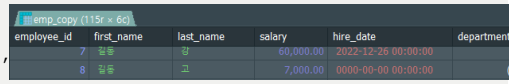
employee_id	first_name	last_name	salary	hire_date	departme
4	길동	김	(NULL)	2022-12-26 00:00:00	

```
-- 여러개 insert
INSERT INTO emp_copy VALUE
  (5 , '길동', '강', 70000, '2022-12-26', NULL),
  (6 , '길동', '강', null, '2022-12-26', NULL),
  (7 , '길동', '강', 60000, '2022-12-26', 50);
```



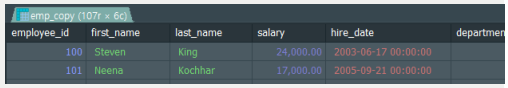
employee_id	first_name	last_name	salary	hire_date	department_
4	길동	김	(NULL)	2022-12-26 00:00:00	
5	길동	강	70,000.00	2022-12-26 00:00:00	
6	길동	강	(NULL)	2022-12-26 00:00:00	
7	길동	강	60,000.00	2022-12-26 00:00:00	

```
-- 오류
INSERT INTO emp_copy VALUES (8,'길동' , '고', 7000, '2012',
-- 년년년년 - 월월 - 일일 시시:분분:초초 mariaDB 기본형식
-- datetime(date, time)

-- 오류 무시하고 삽입
INSERT Ignore INTO emp_copy VALUES (8,'길동' , '고', 7000, '2012', NULL );
```



employee_id	first_name	last_name	salary	hire_date	department
7	길동	강	60,000.00	2022-12-26 00:00:00	
8	길동	고	7,000.00	0000-00-00 00:00:00	

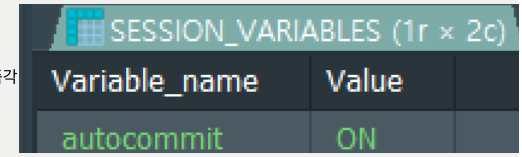
```
-- 데이터 전부 삭제
DELETE FROM emp_copy;
-- em_copy table 생성하지 말고 데이터 복사
```



employee_id	first_name	last_name	salary	hire_date	departmen
100	Steven	King	24,000.00	2003-06-17 00:00:00	
101	Neena	Kochhar	17,000.00	2005-09-21 00:00:00	

```
INSERT INTO emp_copy
SELECT employee_id, first_name, last_name, salary, hire_date, department_id FROM employees;
```

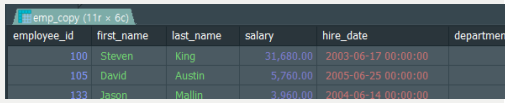
```
-- 커밋상태 확인
SHOW VARIABLES LIKE 'autocommit';
-- 마리아 db autocommit 자동실행으로 insert, update 등 DML 즉각
-- set autocommit = false; 로 변경가능
```




```
-- delete from 테이블명
-- select * From 테이블명하고 원하는 데이터가 나오면 삭제하자.
-- WHERE 조회 | 삭제 조건식 문법은 유사하다.

-- 데이터 전부 삭제
DELETE FROM emp_copy;--조건절을 꼭 쓰자!!

-- DELETE 문법은 AOTOCOMMIT 상태를 변경하면 복구 기회를 가진다.
-- WHERE 절 사용 가능
-- 하지만 TRUNCATE 문장은  AOTOCOMMINT 상태를 변경해도 무시, 복구 기회 없다.
-- WHERE 절 사용 불가
-- TRUNCATE emp_copy;
```

```
-- 100번 사원의 급여 인상, 10%증가
UPDATE emp_copy
SET salary = salary * 1.1
WHERE employee_id = 100;
```


```
-- 입사월이 6월인 사원의 부서를 20으로 배정하고 급여 20% 인상
UPDATE emp_copy
SET department_id = 20, salary = salary *1.2
WHERE hire_date LIKE '_____06%';

SELECT * FROM emp_copy WHERE hire_date LIKE '_____06%' ORDER BY employee_id;
```



▼ 제약조건

constraint = 현실세계의 모델링된 내용을 확인

현실세계 → 모델링 - product 테이블

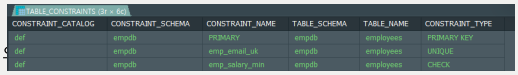
상품명 → not null, 가격 → int, 수량 → ≥0 : 데이터의 상태를 보존하기 위함 조건

- 데이터 상태 보존 = 데이터 무결성 보존
  - not null : 중복은 허용, null은 안됨.
  - unique : 중복 미허용(null은 하나만 허용)
  - primary key: 식별자의 조건으로 not null과 unique조건 모두 포함
  - check - 사용자 정의
  - foreign key

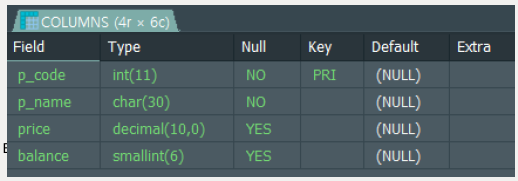

```
create table employees(
  employee_id xxx,
  .....
  department_id int,
  constraint fk_emp-dept foreign key(department_id)
  references departments(department_id)
);
```

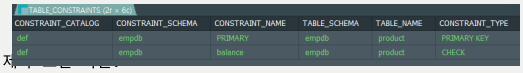
- 설정 효력은 DML문법 사용시 오류를 발생시킴.

```
-- 7장 제약조건
INSERT INTO VALUES(1, '이름', '성',NULL,NULL,NULL); -- 오
SELECT * FROM information_schema.TABLE_CONSTRAINTS
WHERE TABLE_NAME = 'employees'; -- 테이블 구조 확인
```

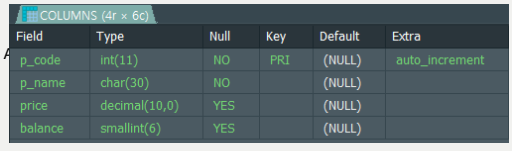


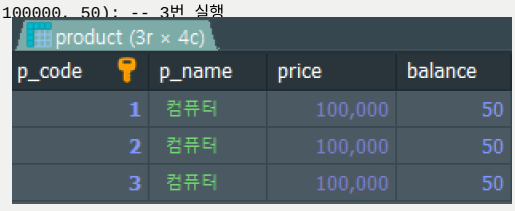
```
-- 상품코드 상품명 가격 수량
CREATE TABLE product(
  p_code INT PRIMARY KEY, -- 고유키 지정
  p_name CHAR(30) NOT NULL,  -- null 허용하지 않음
  price DECIMAL,  -- 실수
  balance SMALLINT CHECK(balance >= 0) -- 2byte의 정수 타
);

DESC product; -- primary, not null 정보
SELECT * FROM information_schema.TABLE_CONSTRAINTS
WHERE TABLE_NAME = 'product'; -- not null을 제외한 모든 제
```





```
-- product테이블 수정
ALTER TABLE product MODIFY COLUMN p_code INT NOT NULL A
DESC product;

-- 자동 증가 설정은 DB마다 다름.(제약 조건 아니다.)

INSERT product(p_name, price, balance) VALUE('컴퓨터', 100000, 50); -- 3번 실행
SELECT * FROM product;
```





```
-- 회원 정보 저장 테이블
/*  users 테이블
  user_id char(10) 중복 x, null 허용 x,
  user_pw 문자 5 자리 (한글 3byte so, 영문이나 숫자 기준 ) nu
  user_name 문자 30자리,
  user_email 문자 30 자리 중복 x,
  user_phone 문자 12자리 중복 x '010-'시작,
  address 문자 100자리
*/

CREATE TABLE users (
  user_id char(10) PRIMARY KEY,
  user_pw VARCHAR(5) NOT NULL,
  user_name VARCHAR(30),
  user_email VARCHAR(30) UNIQUE ,
  user_phone VARCHAR(12) UNIQUE CHECK (user_phone LIKE
  address VARCHAR(100)
);

ALTER TABLE users MODIFY COLUMN user_phone VARCHAR(14)

DESC users;
SELECT * FROM information_schema.TABLE_CONSTRAINTS
WHERE TABLE_NAME = 'users';
INSERT INTO users VALUES('id','pw','basic','test@a.com','010-1111-1111', 'address');
SELECT * FROM users;
-- check 설정 내용 확인
SELECT * FROM information_schema.CHECK_CONSTRAINTS WHERE TABLE_NAME = 'users';
```

COLUMNS (6r × 6c)

Field	Type	Null	Key	Default	Extra
user_id	char(10)	NO	PRI	(NULL)	
user_pw	varchar(5)	NO		(NULL)	
user_name	varchar(30)	YES		(NULL)	
user_email	varchar(30)	YES	UNI	(NULL)	
user_phone	varchar(14)	YES	UNI	(NULL)	
address	varchar(100)	YES		(NULL)	

TABLE_CONSTRAINTS (4r × 6c)

CONSTRAINT_CATALOG	CONSTRAINT_SCHEMA	CONSTRAINT_NAME	TABLE_SCHEMA	TABLE_NAME	CONSTRAINT_TYPE
def	empdb	PRIMARY	empdb	users	PRIMARY KEY
def	empdb	user_email	empdb	users	UNIQUE
def	empdb	user_phone	empdb	users	UNIQUE
def	empdb	user_phone	empdb	users	CHECK

users (1r × 6c)

user_id	user_pw	user_name	user_email	user_phone	address
id	pw	basic	test@a.com	010-1111-1111	address

CHECK_CONSTRAINTS (1r × 6c)

CONSTRAINT_CATALOG	CONSTRAINT_SCHEMA	TABLE_NAME	CONSTRAINT_NAME	LEVEL	CHECK_CLAUSE
def	empdb	users	user_phone	Column	`user_phone` like '010-%'

```
-- 컬럼 레벨로만 가능 _ not null
-- 테이블 레벨로만 가능 (maria DB 만 ) _ foreign key
-- 컬럼 + 테이블 레벨도 가능 _ PK, unique , check
CREATE TABLE board(
  seq INT NOT NULL AUTO_INCREMENT PRIMARY KEY, -- 컬럼레
  title VARCHAR(100) NOT NULL, -- 컬럼레벨 정의
  contents TEXT , -- text type은 영어기준 65536byte 까지
  viewcount INT DEFAULT 0, -- 컬럼레벨 정의
  writer CHAR(10), -- USERS의  user_id를 외래키로 사용_ Pk
  -- CONSTRAINT PK_board_seq PRIMARY KEY (seq),  PK 정
  CONSTRAINT fk_board_writer FOREIGN KEY(writer) REFERENCES users(user_id) -- 테이블 레벨 정의
  -- FK는 참조 테이블의 PK만 설정 가능, 타입과 크기가 동일
  -- check (컬럼 조건),
  -- Unique (컬럼명)
);

DESC board;
SELECT * FROM information_schema.KEY_COLUMN_USAGE
WHERE TABLE_NAME = 'board';

INSERT INTO board(title, contents, writer ) VALUES ('제목', '내용', 'none'); -- error
INSERT INTO board(title, contents, writer ) VALUES ('제목', '내용', 'id');

SELECT * FROM board; -- seq 가 2인 이유는 error를 한번 발생 시켜서!
```

COLUMNS (5r × 6c)

Field	Type	Null	Key	Default	Extra
seq	int(11)	NO	PRI	(NULL)	auto_increment
title	varchar(100)	NO		(NULL)	
contents	text	YES		(NULL)	
viewcount	int(11)	YES		0	
writer	char(10)	YES	MUL	(NULL)	

KEY_COLUMN_USAGE (2r × 12c)

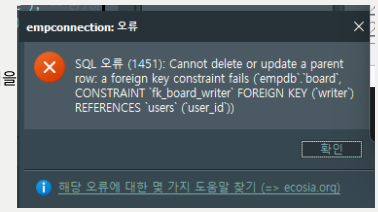
ION	POSITION_IN_UNIQUE_CONSTRAINT	REFERENCED_TABLE_SCHEMA	REFERENCED_TABLE_NAME	REFERENCED_COLUMN_NAME
1	1	empdb	users	user_id
1	(NULL)	(NULL)	(NULL)	(NULL)

board (2r × 5c)

seq	title	contents	viewcount	writer
2	제목	내용	0	id
3	제목2	내용	0	id2

```sql
-- CREATE TABLE - 2장(없는 테이블 정의) /|
-- ALTER TABLE (있는 테이블의 컬럼추가, 컬럼삭제, 컬럼타입수정, 컬럼제약조건 수정)-14장
ALTER TABLE 테이블명 ADD A INT ; --없던 컬럼 추가
ALTER TABLE 테이블명 MODIFY A CHAR(10); --있던 컬럼 타입이나 길이 수정
ALTER TABLE 테이블명 DROP XXXX; -- 있던 컬럼 삭제
ALTER TABLE 테이블명 ADD CONSTRAINT  ,,,; -- 존재하는 컬럼의 없던 제약조건 추가
ALTER TABLE 테이블명 MODIFY CONSTRAINT ....; --있던 컬럼 제약조건  수정
ALTER TABLE 테이블명 DROP CONSTRAINT  ; -- 제약조건  삭제
```

```sql
SELECT * FROM users WHERE user_id = 'id';
DELETE FROM users WHERE user_id = 'id'; -- error
-- board라는 자식 테이블에서 id를 참조중인 데이터 임으로 error 발생 자식을
  -- 자식데이터도 삭제
    -- 1. 외래키 제약 조건 삭제
    ALTER TABLE board DROP CONSTRAINT fk_board_writer;
    -- 2. 외래키 재정의
    ALTER TABLE board ADD CONSTRAINT fk_board_writer
    FOREIGN KEY(writer) REFERENCES users(user_id) ON DELETE CASCADE;
    -- 동시 업데이는 ON UPDATE CASCADE;
    FOREIGN KEY(writer) REFERENCES users(user_id) ON D

    DELETE FROM users WHERE user_id = 'id';
    SELECT * FROM users;
    SELECT * FROM board;


  -- null 처리
    -- 1. 외래키 제약 조건 삭제
    ALTER TABLE board DROP CONSTRAINT fk_board_writer;
    -- 2. 외래키 재정의
    ALTER TABLE board ADD CONSTRAINT fk_board_writer
    FOREIGN KEY(writer) REFERENCES users(user_id) ON D
    DELETE FROM users WHERE user_id = 'id2';
    SELECT * FROM users;
    SELECT * FROM board;
```



users (1r × 6c)

user_id	user_pw	user_name	user_email	user_phone	address
id2	pw	basic	test2@a.com	010-1211-1111	address

board (1r × 5c)

seq	title	contents	viewcount	writer
3	제목2	내용	0	id2

users (0r × 6c)

user_id	user_pw	user_name	user_email	user_phone	address

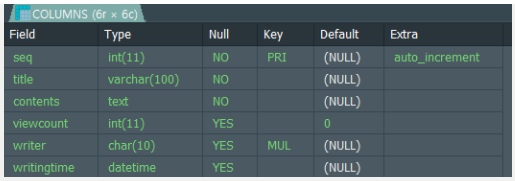
board (1r × 5c)

seq	title	contents	viewcount	writer
3	제목2	내용	0	(NULL)

ddl-data definition language	사용자 생성--> root 로 접속 가능/접속 허용 권한 create user ========== db 생성 create database 테이블 생성-구조변경-삭제 create table / alter table / drop table create table 테이블 (컬럼명 타입(길이) 제약조건 , , , ,); create table 테이블 (컬럼명 타입(길이) , , , , 제약조건 .....); alter table 테이블명 ADD/MODIFY/DROP --> 컬럼 추가 컬럼 수정 컬럼 삭제 alter table 테이블명 ADD CONSTRAINT /MODIFY ... /DROP ... --> 제약조건 추가 컬럼 수정 컬럼 삭제 DROP TABLE 테이블명;-->레코드+테이블삭제 DDL -->AUTO COMMIT(변경X)
dMl-data manipulation language	INSERT UPDATE DELETE... WHERE --> 레코드삭제(테이블존재)

	DML -->(기본 AUTO COMMIT(변경O)
dql-select	
제약조건설정	ddl실행시 NOT NULL UNIQUE PRIMARY KEY CHECK FOREIGN KEY
제약조건효력발생	dml실행시 ON DELETE Cascade on update cascade
	auto_increment – 정수 자동증가 default – 모든 타입 기본입력값 지정

```
-- contents 칼럼  null 허용
ALTER TABLE board MODIFY contents TEXT NOT NULL;
-- 작송 시간 칼럼 추가
ALTER TABLE board ADD writingtime DATETIME;
DESC board;
```


COLUMNS (6r × 6c)
Field Type Null Key Default Extra
seq int(11) NO PRI (NULL) auto_increment
title varchar(100) NO (NULL)
contents text NO (NULL)
viewcount int(11) YES 0
writer char(10) YES MUL (NULL)
writingtime datetime YES (NULL)

▼ 서브쿼리

다중적인 질문을 사용하는 쿼리문 = 단계적 질의 기능

```
SELECT * FROM employees WHERE FIRST_NAME LIKE 'K%';
-- 이름이 kelly인 사원과 동일 부서 근무자의 부서코드, 급여, 이름 조회
DESC employees;
SELECT department_id, salary, first_name
FROM employees
WHERE department_id = (SELECT department_id FROM employees
SELECT department_id FROM employees WHERE FIRST_NAME ='kel
```

employees (45r × 3c)

department_id	salary	first_name
50	8,000.00	Matthew
50	8,200.00	Adam
50	7,900.00	Payam
50	6,500.00	Shanta
50	5,800.00	Kevin

```
SELECT department_id, salary, first_name
FROM employees
WHERE department_id =
  (SELECT department_id FROM employees WHERE FIRST_NAME
-- peter 동명이인 3명 Subquery returns more than 1row

-- 다중행 서브쿼리 => IN
SELECT department_id, salary, first_name
FROM employees
WHERE department_id IN
  (SELECT department_id FROM employees WHERE FIRST_NAME
```

employees (45r × 3c)

department_id	salary	first_name
50	8,000.00	Matthew
50	8,200.00	Adam
50	7,900.00	Payam

employees (79r × 3c)

department_id	salary	first_name
50	8,000.00	Matthew
50	8,200.00	Adam
50	7,900.00	Payam

```
-- 최저 급여를 받는 사람의 이름
SELECT first_name, salary FROM employees
WHERE salary = ( SELECT Min(salary) FROM employees);
```

employees (1r × 2c)

first_name	salary
TJ	2,100.00

```
-- KELLY와 같은 부서이고 같은 일을 하는 사원의 부서코드 급여 이름
SELECT department_id, first_name, salary, job_id FROM e
WHERE department_id
  IN ( SELECT department_id FROM employees WHERE first_
 AND job_id
  IN ( SELECT job_id FROM employees WHERE first_name =

-- 다중열 쿼리문
SELECT department_id, first_name, salary, job_id FROM
WHERE (department_id, job_id) IN
   ( SELECT department_id, job_id FROM employees WHERE
```

employees (20r × 4c)

department_id	first_name	salary	job_id
50	Winston	3,200.00	SH_CLERK
50	Jean	3,100.00	SH_CLERK
50	Martha	2,500.00	SH_CLERK
50	Girard	2,800.00	SH_CLERK

employees (20r × 4c)

department_id	first_name	salary	job_id
50	Winston	3,200.00	SH_CLERK
50	Jean	3,100.00	SH_CLERK
50	Martha	2,500.00	SH_CLERK
50	Girard	2,800.00	SH_CLERK

▼ 문제

주문실습

https://s3-us-west-2.amazonaws.com/secure.notion-static.com/a9e235bb-17e8-4a46-8c8f-6f496c0925f5/%EC%A3%BC%EB%AC%B8%EC%8B%A4%EC%8A%B5.txt

```
- menu 테이블
INT(5)/CHAR(100)/DATETIME
제품코드 : 숫자 5자리 primary key
제품명 : 문자열 100자리 unique
가격 : 숫자 5자리 100 이상 - 10000 이하
재고량 : 숫자 3자리 0 이상
상세설명 : 문자열 4000자리
이미지파일명 : 문자열 50자리

- customer 테이블
고객아이디 : 문자열 30자리 primary key
고객명 : 문자열 30자리 not null
이메일 : 문자열 30자리 unique
핸드폰 : 문자열 11자리 010 시작하고 11자리까지(-제외)
가입일 : 날짜시간
잔액   : 숫자 7자리 0 이상

- orders 테이블
주문코드 : 숫자 5자리 primary key
고객코드 : 문자열 30자리 customer 테이블의 고객코드 foreign key
제품코드 : 숫자 5자리 menu 테이블의 제품코드 foreign key
구입수량 : 숫자 3자리 최대 100개까지 주문 가능
주문시간 : 날짜

<메뉴:menu테이블레코드들>
1, '아메리카노', 2000, 100,'핫,아이스 선택가능:추가요금없음', 'americano.jpg'

2, '카페라떼', 3000, 100 , '두유 변경가능:추가요금없음','latte.jpg'

3, '딸기바나나쥬스', 3000, 50 ,'딸기싱싱','ddalba.jpg'

4, '치즈케익', 5000, 10, '사이즈10*10','cheesecake.jpg'

5, '클럽샌드위치', 4500, 10, '치킨,베이컨선택가능:4조각','sandwich.jpg'

<고객:customer테이블레코드들>
'jung1', '유정은', 'jung1@kitri.com', '0102223333', '2022-12-26', 30000

'inchul1', '신인철', 'in1@bit.com' ,'0103335677', '2022-11-26' ,40000

'hee1', '황희정', 'heejung1@multi.com', '0102224444', '2021-12-26', 50000

== 메뉴와 고객 레코드 저장 완료 ==


<주문>
1. 메뉴 이름과 가격, 상세설명을 조회하여 출력한다.

2. 황희정 클럽샌드위치 2개 주문의 경우
   2-1. customer 테이블에서 황희정의 아이디와 잔액을 조회하여 출력한다.

   2-2. menu 테이블에서 클럽샌드위치의 제품코드를 조회한다.

   2-3. order 테이블에 주문번호는 1, 황희정아이디,클럽샌드위치의 제품코드, 주문시간은 현재시각, 구입수량은 2 의 레코드를 저장한다.
   2-4. menu 테이블의 클럽샌드위치 재고량 컬럼은 order 테이블의 구입수량만큼 뺀다.
   2-5. customer 테이블의 잔액 컬럼은 menu 테이블의 가격 * order 테이블의 구입수량만큼 뺀다.
   2-6. 최종적으로 다음과 같이 조회하여 결과를 확인한다.
이 때 결제액은
메뉴 테이블의 가격 * 주문 테이블의 구입수량
으로 계산한다.
고객아이디 고객이름 제품이름 구입수량 결제액 잔액
```

▼ Sol

실습

```
CREATE TABLE menu(
  product_code INT(5) primary KEY,
  product_name CHAR(100)  UNIQUE,
  price INT(5) CHECK(price BETWEEN 100 AND 10000),
  balance INT(3) CHECK(balance >=  0 ),
  product_desc TEXT,
  img_name CHAR(50)
);

DESC menu;

CREATE TABLE customer(
  id char(30) primary KEY,
  customer_name char(30) not NULL,
  email CHAR(30) UNIQUE,
  phone CHAR(11) CHECK(phone LIKE '010%'),
  reg_date datetime,
  balance INT(7) CHECK(balance >= 0)
);

DESC customer;

create table orders(
  order_code INT(5) primary KEY,
  id char(30),
  product_code INT(5),
  quantity INT(3) CHECK(quantity <= 100),
  order_date DATE,
  CONSTRAINT fk_customer_orders FOREIGN KEY(id) REFERENCES customer(id) ON DELETE SET null,
  CONSTRAINT fk_menu_orders FOREIGN KEY(product_code) REFERENCES menu(product_code) ON UPDATE CASCADE
);

DESC orders;
```

```
INSERT INTO menu VALUE (1, '아메리카노', 2000, 100,'핫,아이스
INSERT INTO menu VALUE (2, '카페라떼', 3000, 100 , '두유 변경
INSERT INTO menu VALUE (3, '딸기바나나쥬스', 3000, 50 ,'딸기싱
INSERT INTO menu VALUE (4, '치즈케익', 5000, 10, '사이즈10*10
INSERT INTO menu VALUE (5, '클럽샌드위치', 4500, 10, '치킨,베
SELECT * FROM menu;

INSERT INTO customer VALUE ('jung1', '유정은', 'jung1@kitri
INSERT INTO customer VALUE ('inchul1', '신인철', 'in1@bit.c
INSERT INTO customer VALUE ('hee1', '황희정', 'heejung1@mul
SELECT * FROM customer;
```

menu (5r × 6c)

product_code	product_name	price	balance	product_desc	img_name
1	아메리카노	2,000	100	핫,아이스 선택가능:추가요금없음	americano.
2	카페라떼	3,000	100	두유 변경가능:추가요금없음	latte.jpg
3	딸기바나나쥬스	3,000	50	딸기싱싱	ddalba.jpg
4	치즈케익	5,000	10	사이즈10*10	cheesecake
5	클럽샌드위치	4,500	10	치킨,베이컨선택가능:4조각	sandwich.j

customer (3r × 6c)

id	customer_name	email	phone	reg_date	bala
hee1	황희정	heejung1@multi.com	0102224444	2021-12-26 00:00:00	
inchul1	신인철	in1@bit.com	0103335677	2022-11-26 00:00:00	
jung1	유정은	jung1@kitri.com	0102223333	2022-12-26 00:00:00	

```
-- 1. 메뉴 이름과 가격, 상세설명을 조회하여 출력한다.
SELECT product_name,price, product_desc FROM menu;
```

menu (5r × 3c)

product_name	price	product_desc
아메리카노	2,000	핫,아이스 선택가능:추가요금없음
카페라떼	3,000	두유 변경가능:추가요금없음
딸기바나나쥬스	3,000	딸기싱싱
치즈케익	5,000	사이즈10*10
클럽샌드위치	4,500	치킨,베이컨선택가능:4조각

```
-- 2. 황희정 클럽샌드위치 2개 주문의 경우
  -- 2-1. customer 테이블에서 황희정의 아이디와 잔액을 조회하여 출력
  SELECT customer_name, id, balance  FROM customer
  WHERE customer_name = '황희정';
```

customer (1r × 3c)

customer_name	id	balance
황희정	hee1	50,0

```
-- 2-2. menu 테이블에서 클럽샌드위치의 제품코드를 조회한다.
SELECT product_name, product_code FROM menu
WHERE product_name = '클럽샌드위치';

-- 2-3. order 테이블에 주문번호는 1, 황희정아이디,클럽샌드위치의 저
INSERT INTO orders VALUE(1, 'hee1', '5',2,'2022-12-26'
SELECT * FROM orders;

-- 2-4. menu 테이블의 클럽샌드위치 재고량 컬럼은 order 테이블의 구
UPDATE menu SET balance = balance -
(SELECT quantity FROM orders WHERE id = 'hee1')
WHERE product_code = (SELECT product_code FROM orders WHERE id = 'hee1');
SELECT product_name, product_code, balance
FROM menu WHERE product_code =
  (SELECT product_code FROM orders WHERE id = 'hee1');

-- 2-5. customer 테이블의 잔액 컬럼은 menu 테이블의 가격 * orde
UPDATE customer SET balance =
  balance - (SELECT price * 2 FROM menu
    WHERE product_code = (SELECT product_code FROM orde
WHERE id = (SELECT id FROM orders WHERE id = 'hee1');
SELECT customer_name, id, balance  FROM customer
WHERE customer_name = '황희정';

-- 2-6. 최종적으로 다음과 같이 조회하여 결과를 확인한다.
  -- 고객아이디 고객이름 제품이름 구입수량 결제액 잔액
SELECT id AS '고객아이디',
  customer_name AS '고객이름',
  (SELECT product_name FROM menu WHERE product_code =
    (SELECT product_code FROM orders WHERE id = 'hee1')
  (SELECT quantity FROM orders WHERE id = 'hee1') AS '구입수량',
  -- 이 때 결제액은 메뉴 테이블의 가격 * 주문 테이블의 구입수량 으로 계산한다.
  (SELECT price * (SELECT quantity FROM orders WHERE id = 'hee1')
  FROM menu
  WHERE product_code =
    (SELECT product_code FROM orders WHERE id = 'hee1') ) AS '결제액',
  balance AS '잔액'
FROM customer
WHERE id = (SELECT id FROM orders WHERE id = 'hee1');
```