수험자 이름: 채점자 이름: 점수:

채점시 반드시 틀린부분을 체크해서 정답을 옆에 적어주세요

1~5번: 5점, 6~10번: 15점

1. "company"라는 데이터베이스를 생성하고, 이 데이터베이스 내에 "employees" 테이블을 생성하세요.

테이블은 다음과 같은 컬럼을 가지도록 해 주세요.

employee_id (INT, PRIMARY KEY, AUTO_INCREMENT)

name (VARCHAR(50))

position (VARCHAR(50))

salary (DECIMAL(10, 2))

department_id (INT)

답:

```
CREATE DATABASE company;
USE company;
CREATE TABLE employees (
    employee_id INT PRIMARY KEY AUTO_INCREMENT,
    name VARCHAR(50),
    position VARCHAR(50),
    salary DECIMAL(10, 2),
    department_id INT
);
```

2. "employees" 테이블에 다음 데이터를 삽입하세요.

name	position	salary	department_id
John Doe	Manager	8000.00	1
Jane Smith	Developer	6000.00	2
Mary Lee	Designer	5500.00	2
Peter Brown	Analyst	5000.00	3
Lucy White	HR Officer	4500.00	1

답:

```
INSERT INTO employees (name, position, salary, department_id)
VALUES
('John Doe', 'Manager', 8000.00, 1),
('Jane Smith', 'Developer', 6000.00, 2),
('Mary Lee', 'Designer', 5500.00, 2),
('Peter Brown', 'Analyst', 5000.00, 3),
('Lucy White', 'HR Officer', 4500.00, 1);
```

4. 연봉이 5000 이상인 직원만 조회하는 SQL 쿼리를 작성하세요.

답:

```
SELECT * FROM employees
WHERE salary >= 5000;
```

5. 연봉이 높은 순서로 직원 목록을 조회하는 SQL 쿼리를 작성하세요.

답:

```
SELECT * FROM employees
ORDER BY salary DESC;
```

6. 연봉이 전체 직원의 평균 연봉 이상인 직원의 목록을 조회하세요.
답:

```
SELECT
    name,
    position,
    salary
FROM
    employees
WHERE
    salary >= (SELECT AVG(salary) FROM employees);
```

7~10번 문제는 아래 테이블 구성들을 활용하여 해결하세요.

```
DROP DATABASE IF EXISTS ig_clone;
CREATE DATABASE ig_clone;
USE ig_clone;

CREATE TABLE users (
    id INTEGER AUTO_INCREMENT PRIMARY KEY,
    username VARCHAR(255) UNIQUE NOT NULL,
    created_at TIMESTAMP DEFAULT NOW()
);

CREATE TABLE photos (
    id INTEGER AUTO_INCREMENT PRIMARY KEY,
    image_url VARCHAR(255) NOT NULL,
    user_id INTEGER NOT NULL,
    created_at TIMESTAMP DEFAULT NOW(),
    FOREIGN KEY(user_id) REFERENCES users(id)
);
```

```
CREATE TABLE comments (
    id INTEGER AUTO_INCREMENT PRIMARY KEY,
    comment_text VARCHAR(255) NOT NULL,
    photo_id INTEGER NOT NULL,
    user_id INTEGER NOT NULL,
    created_at TIMESTAMP DEFAULT NOW(),
    FOREIGN KEY(photo_id) REFERENCES photos(id),
    FOREIGN KEY(user_id) REFERENCES users(id)
);

CREATE TABLE likes (
    user_id INTEGER NOT NULL,
    photo_id INTEGER NOT NULL,
    created_at TIMESTAMP DEFAULT NOW(),
    FOREIGN KEY(user_id) REFERENCES users(id),
    FOREIGN KEY(photo_id) REFERENCES photos(id),
    PRIMARY KEY(user_id, photo_id)
);

CREATE TABLE follows (
    follower_id INTEGER NOT NULL,
    followee_id INTEGER NOT NULL,
    created_at TIMESTAMP DEFAULT NOW(),
    FOREIGN KEY(follower_id) REFERENCES users(id),
    FOREIGN KEY(followee_id) REFERENCES users(id),
    PRIMARY KEY(follower_id, followee_id)
);

CREATE TABLE tags (
  id INTEGER AUTO_INCREMENT PRIMARY KEY,
  tag_name VARCHAR(255) UNIQUE,
  created_at TIMESTAMP DEFAULT NOW()
);

CREATE TABLE photo_tags (
    photo_id INTEGER NOT NULL,
    tag_id INTEGER NOT NULL,
    FOREIGN KEY(photo_id) REFERENCES photos(id),
    FOREIGN KEY(tag_id) REFERENCES tags(id),
    PRIMARY KEY(photo_id, tag_id)
);
```

7. 사진을 3장 이상 올린 사용자를 출력해주세요.

```
SELECT user_id, COUNT(*) AS photo_count
FROM photos
GROUP BY user_id
HAVING photo_count >= 3;
```

8. 좋아요를 한번도 누르지 않은 사용자를 출력해주세요

```
SELECT users.username
FROM users
LEFT JOIN likes ON users.id = likes.user_id
WHERE likes.user_id IS NULL;
```

9. 자기 사진에 스스로 좋아요 누른 경우를 출력해주세요

```
SELECT users.username, photos.id AS photo_id
FROM photos
JOIN likes ON photos.id = likes.photo_id
JOIN users ON photos.user_id = users.id
WHERE photos.user_id = likes.user_id;
```

10. 사용자별로 누른 좋아요 개수를 출력하세요.

```
SELECT users.username, COUNT(*) AS total_likes
FROM users
JOIN likes ON users.id = likes.user_id
GROUP BY users.id;
```