SQL 기본 문법 & CRUD 실습

> *데이터 모델링으로 설계한 의료 데이터베이스를*
> *실제로 **조작(입력·조회·수정·삭제)** 해보자!*

목차 (Table of Contents)

GOAL

SQL 기본 문법을 익히고 MySQL 환경에서 직접 실습한다
환자·진료·검사 데이터를 CRUD로 조작해본다
의료 데이터에서 조건 검색과 정렬을 수행할 수 있다

> *“ERD로 설계한 병원 데이터베이스를*
> *이제 손으로 움직여볼 차례!”*

SQL 실습 시작하기

> *ERD로 설계한 의료 데이터베이스를 실제 MySQL에서 다뤄보는 실습!*

실습 환경 준비

항목	내용
DBMS	MySQL 8.x
GUI 툴	MySQL Workbench
실습 데이터셋	환자, 의사, 예약 테이블
목표	CRUD 명령어를 직접 수행하고 결과 확인

MySQL이란?

- 오픈소스 관계형 데이터베이스 관리 시스템 (RDBMS)
- SQL(Structured Query Language)을 사용
- 의료기관에서도 널리 사용되는 범용 DBMS

> *“데이터베이스 중 가장 대중적”*

MySQL Workbench란?

- MySQL 공식 GUI 도구
- **쿼리 작성, 실행, 결과 확인**을 직관적으로 지원
- **ERD 설계, 백업, 계정 관리**도 가능

> *SQL 초보자에게 가장 친절한 시각화 도구*

Workbench 주요 영역

영역	설명
Connections	DB 서버 접속 정보
Query Editor	SQL 문 작성 및 실행 창
Result Grid	SELECT 결과 출력 영역
Navigator	스키마, 테이블, 컬럼 구조 확인

MySQL 설치 & 접속 절차

1. https://dev.mysql.com/downloads/ 접속
2. MySQL Installer 다운로드
3. “Developer Default” 선택
4. root 비밀번호 설정
5. 설치 완료 후 Workbench 실행
6. root 계정으로 접속

SQL이란?

SQL (Structured Query Language)

: 데이터베이스에 **질문(Query)** 을 던지고, **요청(Request)** 을 수행하는 언어

SQL의 핵심 목적

SQL은 단순히 데이터를 “꺼내는” 언어가 아니라,

데이터를 정의(Define) → 조작(Manipulate) → 제어(Control) 하는 언어

역할	설명	예시
데이터 정의	어떤 데이터가, 어떤 구조로 저장될지 정의	환자 테이블 만들기
데이터 조작	데이터를 넣고, 수정하고, 삭제하고, 꺼내기	진료 기록 등록/조회
데이터 제어	누가 접근할 수 있는지 설정	연구자에게 조회만 허용
트랜잭션 관리	여러 명령을 하나의 작업 단위로 묶기	환자 정보 수정 중 에러 시 되돌리기

SQL의 4대 구성요소

분류	의미	대표 명령어
DDL	데이터 정의	`CREATE`, `ALTER`, `DROP`
DML	데이터 조작	`SELECT`, `INSERT`, `UPDATE`, `DELETE`
DCL	권한 제어	`GRANT`, `REVOKE`
TCL	트랜잭션 제어	`COMMIT`, `ROLLBACK`, `SAVEPOINT`

실습용 데이터베이스 생성

```
CREATE DATABASE hospital;
USE hospital;
```

> `hospital` *스키마 안에 환자·의사·예약 테이블을 만들어보기*

데이터 타입

- 데이터 타입은 문자, 숫자, 날짜 등 여러가지가 있음
- 제공되는 데이터 타입은 DBMS마다 차이가 있을 수 있음
- MySQL 데이터 타입: https://dev.mysql.com/doc/refman/8.0/en/data-types.html

숫자형

타입	설명	범위 / 예시
INT	정수 (기본형)	-2,147,483,648 ~ 2,147,483,647
TINYINT	아주 작은 정수	-128 ~ 127
SMALLINT	작은 정수	약 ±3만
MEDIUMINT	중간 크기 정수	약 ±800만
BIGINT	매우 큰 정수	±9경 정도
DECIMAL(p,s)	고정 소수점 (정확한 금액 등)	`DECIMAL(6,2)` → 9999.99
FLOAT	단정밀도 실수 (근사값)	약 7자리 정밀도
DOUBLE	배정밀도 실수	약 15자리 정밀도
BIT(n)	비트 단위 저장	`BIT(8)` → 8비트(1바이트)

날짜·시간형

타입	설명	예시
DATE	연·월·일	`2025-10-22`
TIME	시·분·초	`13:45:30`
DATETIME	날짜 + 시간	`2025-10-22 13:45:30`
TIMESTAMP	날짜+시간 (타임존 포함, 자동 갱신 가능)	`CURRENT_TIMESTAMP`
YEAR(4)	연도	`2025`

DATETIME vs TIMESTAMP

- DATETIME: 단순 시간 저장 (타임존 영향 X)
- TIMESTAMP: 자동으로 서버 타임존에 맞게 변환

문자형

타입	설명	특징 / 용도
CHAR(n)	고정 길이 문자열	항상 n바이트, 주민번호 등 고정형 데이터
VARCHAR(n)	가변 길이 문자열	이름, 메모 등
TEXT	긴 텍스트 (최대 65KB)	코멘트, 기사 내용 등
MEDIUMTEXT	더 긴 텍스트 (16MB)	긴 보고서
LONGTEXT	매우 긴 텍스트 (4GB)	로그, 기록 데이터
ENUM('a','b',...)	미리 정의된 값 중 하나	성별, 상태값
SET('a','b',...)	여러 값을 동시에 선택 가능	복수 태그 저장
BLOB	이진 데이터	이미지, 파일, 영상 등

테이블 생성 (환자)

```sql
CREATE TABLE patients (
    patient_id INT AUTO_INCREMENT PRIMARY KEY,   -- 기본키 (환자 고유번호)
    name VARCHAR(50) NOT NULL,                   -- 이름
    birth_date DATE,                             -- 생년월일
    gender ENUM('M', 'F') NOT NULL,              -- 성별
    phone VARCHAR(20) UNIQUE,                    -- 전화번호 (유일 값)
    email VARCHAR(100),                          -- 이메일
    address VARCHAR(200),                        -- 주소
    visit_count INT DEFAULT 0,                   -- 방문 횟수
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

```
CREATE TABLE 테이블명(속성1 데이터타입1 제약조건1, 속성2 데이터타입2 제약조건2, ...);
```

테이블 생성 (의사, 예약)

```sql
CREATE TABLE doctors (
    doctor_id INT AUTO_INCREMENT PRIMARY KEY,  -- 기본키 (의사 ID)
    name VARCHAR(50) NOT NULL,                 -- 이름
    specialty VARCHAR(100) NOT NULL,           -- 전문과목
    phone VARCHAR(20) UNIQUE,                  -- 휴대폰 (유일 값)
    email VARCHAR(100),
    hire_date DATE
);

CREATE TABLE appointments (
    appointment_id INT AUTO_INCREMENT PRIMARY KEY,  -- 진료 예약 ID
    patient_id INT NOT NULL,                        -- 환자 ID (FK)
    doctor_id INT NOT NULL,                         -- 의사 ID (FK)
    visit_date DATE NOT NULL,                       -- 진료 날짜
    status ENUM('Scheduled', 'Completed', 'Cancelled') DEFAULT 'Scheduled',  -- 상태
    note VARCHAR(255),                              -- 진료 메모
    patient_name VARCHAR(50),                       -- JOIN UPDATE 예제용
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (patient_id) REFERENCES patients(patient_id),
    FOREIGN KEY (doctor_id) REFERENCES doctors(doctor_id)
);
```

SQL 기본 구조

CRUD = Create / Read / Update / Delete

데이터베이스에서 데이터를 추가 → 조회 → 수정 → 삭제 하는 네 가지 핵심 동작

동작	의미	SQL 명령어
C (Create)	데이터 추가	`INSERT`
R (Read)	데이터 조회	`SELECT`
U (Update)	데이터 수정	`UPDATE`
D (Delete)	데이터 삭제	`DELETE`

데이터 입력 (INSERT)

```
INSERT INTO 테이블명 (컬럼1, 컬럼2, 컬럼3, ...)
VALUES (값1, 값2, 값3, ...);
```

- INTO : 어느 테이블에 넣을지 지정
- (컬럼1, 컬럼2, …) : 어떤 필드에 넣을지 명시
- VALUES : 실제 입력할 값

데이터 입력 (INSERT)

환자 테이블

```
INSERT INTO patients
(name, birth_date, gender, phone, email, address, visit_count, created_at)
VALUES
('이동원', '1993-01-01', 'M', '010-1234-5678', 'dongwon@naver.com', '서울 강남구 테헤란로 12', 0, NOW()),
('김지수', '1995-12-31', 'F', '010-9876-5432', 'jisu@hanmail.com', '서울 마포구 연남동 77', 0, NOW()),
('김은중', '2000-05-15', 'M', '010-5555-6666', 'eunjoong@.com', '경기 성남시 분당구 판교로 88', 0, NOW());
```

의사 테이블

```sql
INSERT INTO doctors
(name, specialty, phone, email, hire_date, created_at)
VALUES
('박지훈', '외과', '010-1111-2222', 'pjh@hospital.com', '2018-03-12', NOW()),
('최가은', '피부과', '010-2222-3333', 'gaen@hospital.com', '2020-07-01', NOW()),
('이현우', '정형외과', '010-3333-4444', 'hlee@hospital.com', '2019-11-05', NOW());
```

```sql
INSERT INTO appointments
(patient_id, doctor_id, visit_date, status, note, patient_name, created_at)
VALUES
(1, 1, '2025-10-22 09:00:00', 'Scheduled', '거북목 증상으로 내원', '이동원', NOW()),
(2, 2, '2025-10-22 10:30:00', 'Completed', '강아지털 알레르기 재진', '김지수', NOW()),
(3, 3, '2025-10-22 11:15:00', 'Scheduled', '고양이가 할퀴어서 검사 예정', '김은중', NOW());
```

데이터 조회 (SELECT)

```
SELECT 컬럼명1, 컬럼명2, ...
FROM 테이블명
[WHERE 조건]
[ORDER BY 정렬기준 ASC/DESC]
[LIMIT 개수];
```

- *: 모든 컬럼을 조회
- WHERE : 조건 지정
- ORDER BY : 정렬 기준
- LIMIT : 출력 개수 제한

데이터 조회 (SELECT)

```sql
SELECT * FROM patients;
SELECT name, gender FROM doctors;

SELECT p.name, d.name AS doctor, a.visit_date
FROM appointments a
JOIN patients p ON a.patient_id = p.patient_id
JOIN doctors d ON a.doctor_id = d.doctor_id;
```

조건 검색 (WHERE)

```sql
SELECT * FROM patients WHERE gender = 'F';
SELECT * FROM appointments WHERE visit_date > '2025-10-22 09:00';
```

데이터 수정 (UPDATE)

```
UPDATE 테이블명
SET 컬럼1 = 값1, 컬럼2 = 값2, ...
WHERE 조건;
```

- SET: 변경할 컬럼과 새 값을 지정
- WHERE: 조건을 명시 (없으면 모든 행이 수정됨 ⚠️)

단일 컬럼 수정

```sql
UPDATE patients
SET phone = '010-7777-8888'
WHERE patient_id = 1;
```

> *특정 환자의 전화번호만 수정*

여러 컬럼 동시 수정

```sql
UPDATE patients
SET address = '서울 강남구',
    email = 'patient01@hospital.com'
WHERE patient_id = 1;
```

방문 횟수 증가

```sql
UPDATE patients
SET visit_count = visit_count + 1
WHERE name = '김지수';
```

조건을 이용한 일괄 업데이트

```
UPDATE appointments
SET status = 'Completed'
WHERE visit_date < NOW();
```

진료 메모 수정

```sql
UPDATE appointments
SET note = '거북목 증상 완화 — 물리치료 예약 권장'
WHERE appointment_id = 1;
```

진료 일정 변경

```sql
UPDATE appointments
SET visit_date = '2025-10-23 09:30:00'
WHERE patient_id = 2;
```

정렬 (ORDER BY)

```
SELECT * FROM appointments ORDER BY visit_date DESC;
```

- 최근 예약 순으로 정렬
- DESC = 내림차순, ASC = 오름차순

결과 제한 (LIMIT)

```sql
SELECT * FROM patients LIMIT 3;
```

🗑️ 데이터 삭제 (DELETE)

```
DELETE FROM 테이블명
WHERE 조건;
```

> *WHERE 절이 없으면 테이블의 모든 데이터가 삭제!*

특정 환자 진료 기록 삭제

```sql
DELETE FROM appointments WHERE appointment_id = 2;
```

특정 환자 예약 삭제

```sql
DELETE FROM appointments WHERE patient_id = 2;
```

IS vs =

구분	`=` 연산자	`IS` 연산자
비교 대상	일반 값 (숫자, 문자 등)	`NULL`, `TRUE`, `FALSE` 같은 특수값
의미	“값이 같은가?”	“이 값이 그 상태인가?”
예시	`WHERE age = 30` → age가 30인 행	`WHERE phone IS NULL` → phone이 비어 있는 행
잘못된 예시	`WHERE phone = NULL` ❌ (항상 false)	—

SQL에서 NULL은 “값이 없음” 을 의미

범위 필터 (BETWEEN, 비교)

```sql
SELECT * FROM appointments
WHERE visit_date >= '2025-10-22 00:00:00'
  AND visit_date <  '2025-10-23 00:00:00';

SELECT * FROM appointments
WHERE visit_date BETWEEN '2025-10-20 00:00:00' AND '2025-10-25 23:59:59';
```

집합 필터 (IN / NOT IN)

```
SELECT * FROM appointments
WHERE status NOT IN ('Cancelled');
```

패턴 매칭 (LIKE)

```sql
-- 성이 '김'인 환자
SELECT * FROM patients
WHERE name LIKE '김%';

-- 휴대폰 국번이 010-1234로 시작
SELECT * FROM patients
WHERE phone LIKE '010-1234%';

-- 메모에 '알레르기' 포함된 예약
SELECT * FROM appointments
WHERE note LIKE '%알레르기%';
```

날짜/시간 편의 필터

```sql
-- 오늘 예약
SELECT * FROM appointments
WHERE DATE(visit_date) = CURDATE();

-- 이번 주(월~일) 예약 (월요일=1 가정)
SELECT * FROM appointments
WHERE YEARWEEK(visit_date, 1) = YEARWEEK(CURDATE(), 1);

-- 과거 예약(완료 처리 대상)
SELECT * FROM appointments
WHERE visit_date < NOW();
```

테이블 구조 변경

컬럼 수정

```
ALTER TABLE 테이블명
MODIFY COLUMN 컬럼명 데이터타입 [옵션];

ALTER TABLE 테이블명
CHANGE COLUMN 기존컬럼명 새컬럼명 데이터타입 [옵션];
```

MODIFY vs CHANGE

명령어	설명	이름 변경	타입 변경
MODIFY COLUMN	데이터 타입, 제약조건 등을 변경	❌ 불가능	✅ 가능
CHANGE COLUMN	이름과 타입을 모두 변경	✅ 가능	✅ 가능

예시1: 데이터 타입 변경

```sql
ALTER TABLE patients
MODIFY COLUMN phone VARCHAR(30);
```

> *컬럼의 길이를 20 -> 30으로 확장*

예시2: 컬럼 이름까지 바꾸기

```
ALTER TABLE patients
CHANGE COLUMN phone contact_number VARCHAR(30);
```

> *phone을 contact_number로 이름 변경 + 타입 변경 가능*

예시 3: 컬럼 기본값 변경

```
ALTER TABLE appointments
MODIFY COLUMN status ENUM('Scheduled','Completed','Cancelled') DEFAULT 'Scheduled';
```

> *예약 상태 기본값을 'Scheduled'로 설정*

예시 4: 컬럼 추가 / 삭제

```sql
ALTER TABLE patients
ADD COLUMN blood_type ENUM('A','B','O','AB');

ALTER TABLE patients
DROP COLUMN blood_type;
```

> *컬럼 추가와 삭제도 ALTER TABLE로 수행*

실습 1 — 병원 데이터베이스 생성

요구사항

1. hospital_lab이라는 데이터베이스를 새로 만든다.
2. 해당 DB를 사용하도록 설정한다.

실습 2 — 환자 테이블 만들기

요구사항

1. patients 테이블을 생성한다.
2. 기호에 맞게 환자 컬럼들을 생성한다.

실습 3 — 의사 테이블 만들기

요구사항

1. doctors 테이블 생성
2. 기호에 맞게 의사 컬럼들을 생성한다.

실습 4 —예약 테이블 만들기

요구사항

1. appointments 테이블 생성
2. patient_id, doctor_id는 각각 외래키(FK)로 설정
3. 예약 상태(status)는 'Scheduled', 'Completed', 'Cancelled' 중 하나만 가능

실습 5 — 데이터 삽입 (INSERT)

요구사항

1. 환자 3명, 의사 3명, 예약 3건을 등록하라.

실습 6 — 데이터 조회 (SELECT)

요구사항

1. 모든 환자 데이터를 조회
2. 여성 환자만 조회
3. 예약일 순서대로 내림차순 정렬

실습 7 — 데이터 수정 (UPDATE)

요구사항

1. 특정 환자의 전화번호를 010-9999-8888로 변경
2. 모든 과거 예약(visit_date < NOW())의 상태를 'Completed'로 변경

실습 8 — 데이터 삭제 (DELETE)

요구사항

1. 예약번호 3번 삭제
2. 특정 환자를 이름 기준으로 삭제

실습 9 — 조건 검색 (WHERE + LIKE + BETWEEN)

요구사항

1. 특정 성씨로 시작하는 환자
2. 특정 날짜사이 사이 예약
3. 특정 단어가 들어간 메모 조회