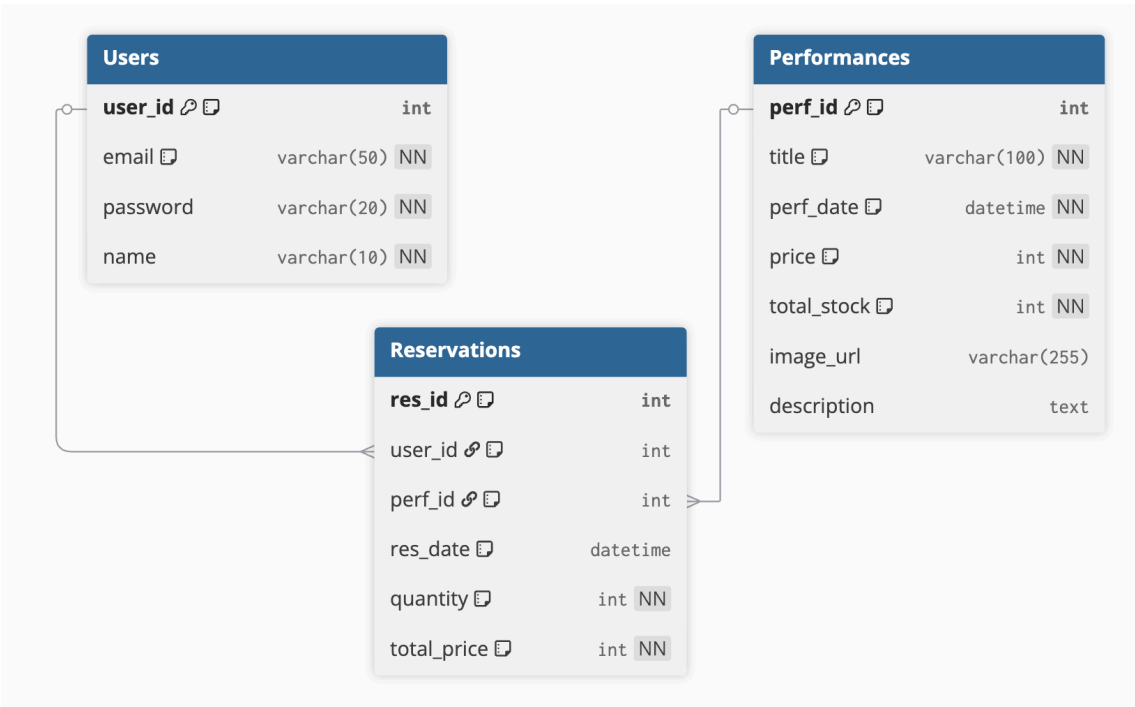


웹 풀스택 - 프로젝트 1

🕒 생성일	@2026년 1월 14일 오전 9:54
☰ 태그	

데이터베이스 모델링

▼ 1. ERD 설계 (DBDiagram)



관계	관계차수	이유
User : Reservation	1:N	한 명의 사용자는 여러 번의 공연을 예매할 수 있다
Performance : Reservation	1:N	하나의 공연은 여러 사용자에게 티켓을 판매할 수 있다.
Reservation	N:M (연결고리)	Reservation 테이블은 User, Performance 테이블 사이의 N:M (다대다) 관계를 풀어주는 연결 고리 역할을 한다

▼ 2. 테이블 생성

```
-- 1. 사용자 테이블
CREATE TABLE Users (
    user_id INT AUTO_INCREMENT PRIMARY KEY, -- 고유 식별자
    email VARCHAR(50) NOT NULL UNIQUE,      -- 로그인 ID (중복 불가)
    password VARCHAR(255) NOT NULL,         -- 비밀번호 (암호화 고려)
    name VARCHAR(50) NOT NULL               -- 이름
);

-- 2. 공연 테이블
CREATE TABLE Performances (
    perf_id INT AUTO_INCREMENT PRIMARY KEY, -- 공연 ID
    title VARCHAR(100) NOT NULL,            -- 공연명
    perf_date DATETIME NOT NULL,            -- 공연 일시
    price INT NOT NULL,                     -- 티켓 단가
    total_stock INT NOT NULL DEFAULT 100,   -- 전체 재고
    image_url VARCHAR(255),                 -- 이미지 URL
    description TEXT                        -- 공연 상세 설명
);

-- 3. 예매 테이블
CREATE TABLE Reservations (
    res_id INT AUTO_INCREMENT PRIMARY KEY, -- 예매 번호
    user_id INT NOT NULL,                  -- 예매자 ID (FK)
    perf_id INT NOT NULL,                  -- 예매 공연 ID (FK)
    res_date DATETIME DEFAULT NOW(),       -- 예매 시점
    quantity INT NOT NULL,                 -- 수량 (인당 100제한 로직용)
    total_price INT NOT NULL,              -- 총 결제 금액

    FOREIGN KEY (user_id) REFERENCES Users(user_id),
    FOREIGN KEY (perf_id) REFERENCES Performances(perf_id)
);
```

▼ 3. 데이터 삽입

-- 1. 사용자 테이블

```
INSERT INTO Users (name, email, password) VALUES  
( '이규현', 'leekh010502@naver.com', '1234'),  
( '테스트 사용자', 'test@example.com', 'password123');
```

-- 2. 공연 테이블

```
INSERT INTO Performances (title, perf_date, price, total_stock, image_url,  
description) VALUES  
( '뮤지컬 렌트', '2024-07-15 19:00:00', 80000, 150, 'http://example.com/rent.jpg', '뉴욕의 이스트 빌리지에서 펼쳐지는 청춘들의 이야기.'),  
( '연극 햄릿', '2024-08-01 20:00:00', 60000, 100, 'http://example.com/hamlet.jpg', '셰익스피어의 명작, 햄릿을 현대적으로 재해석한 공연.');
```

-- 3. 예매 테이블

```
INSERT INTO Reservations (user_id, perf_id, quantity, total_price) VALUES  
(1, 1, 2, 160000),  
(2, 2, 1, 60000);
```

▼ 4. 데이터 조회

-- 1. 사용자 테이블

```
SELECT * FROM Users;
```

-- 2. 공연 테이블

```
SELECT * FROM Performances;
```

-- 3. 예매 테이블

```
SELECT * FROM Reservations;
```

▼ 5. 데이터 수정

-- 1. 사용자 테이블

```
UPDATE Users SET password = 'newpassword456' WHERE user_id = 2;
```

-- 2. 공연 테이블

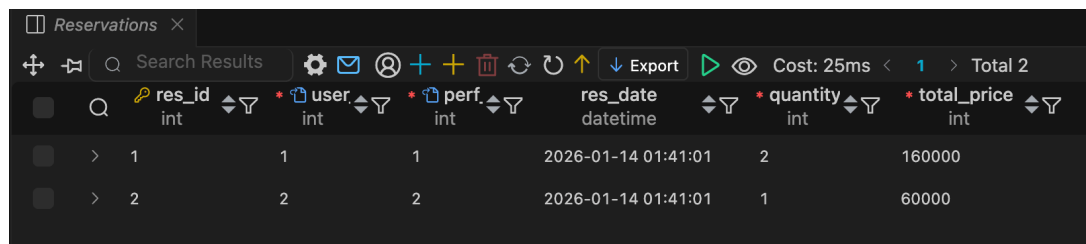
```
UPDATE Performances SET total_stock = total_stock + 2 WHERE perf_id = 1;
```

-- 3. 예매 테이블

```
UPDATE Reservations SET quantity = 4, total_price = 320000 WHERE res_id = 1;
```

▼ 결과 이미지

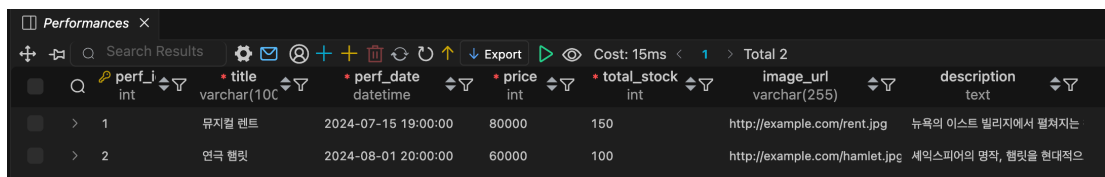
- 사용자 (User) 테이블 생성 후 정보 입력



The screenshot shows a database interface for the 'Reservations' table. The table has columns: res_id (int), user_id (int), perf_id (int), res_date (datetime), quantity (int), and total_price (int). There are 2 rows of data.

	res_id	user_id	perf_id	res_date	quantity	total_price
>	1	1	1	2026-01-14 01:41:01	2	160000
>	2	2	2	2026-01-14 01:41:01	1	60000

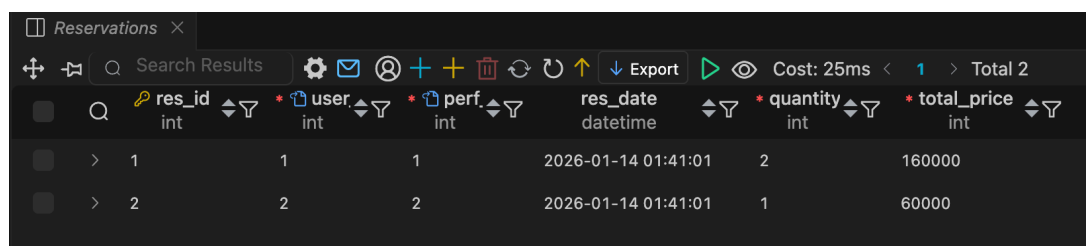
- 공연 (Performance) 테이블 생성 후 정보 입력



The screenshot shows a database interface for the 'Performances' table. The table has columns: perf_id (int), title (varchar(100)), perf_date (datetime), price (int), total_stock (int), image_url (varchar(255)), and description (text). There are 2 rows of data.

	perf_id	title	perf_date	price	total_stock	image_url	description
>	1	무지컬 렌트	2024-07-15 19:00:00	80000	150	http://example.com/rent.jpg	뉴욕의 이스트 빌리지에서 펼쳐지는
>	2	연극 햄릿	2024-08-01 20:00:00	60000	100	http://example.com/hamlet.jpg	셰익스피어의 명작, 햄릿을 현대적으로

- 예매 (Reservation) 테이블 생성 후 정보 입력



The screenshot shows a database interface for the 'Reservations' table. The table has columns: res_id (int), user_id (int), perf_id (int), res_date (datetime), quantity (int), and total_price (int). There are 2 rows of data.

	res_id	user_id	perf_id	res_date	quantity	total_price
>	1	1	1	2026-01-14 01:41:01	2	160000
>	2	2	2	2026-01-14 01:41:01	1	60000