-- Create database and use it

CREATE DATABASE IF NOT EXISTS cv;

USE cv;

-- Drop all tables for a clean setup

DROP TABLE IF EXISTS BookedSeats, Bookings, Seats, TimeSlots, Screens, Theaters, Movies, Users;

-- Users table

CREATE TABLE Users (

user\_id INT PRIMARY KEY AUTO\_INCREMENT,

name VARCHAR(100) NOT NULL,

email VARCHAR(100) UNIQUE NOT NULL,

phone VARCHAR(15),

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

);

-- Movies table

CREATE TABLE Movies (

movie\_id INT PRIMARY KEY AUTO\_INCREMENT,

title VARCHAR(100) NOT NULL,

duration\_minutes INT CHECK (duration\_minutes > 0),

rating VARCHAR(10),

release\_date DATE

);

-- Theaters table

CREATE TABLE Theaters (

theater\_id INT PRIMARY KEY AUTO\_INCREMENT,

name VARCHAR(100) NOT NULL,

location VARCHAR(200) NOT NULL

);

-- Screens table

CREATE TABLE Screens (

screen\_id INT PRIMARY KEY AUTO\_INCREMENT,

theater\_id INT NOT NULL,

name VARCHAR(50) NOT NULL,

total\_seats INT CHECK (total\_seats > 0),

FOREIGN KEY (theater\_id) REFERENCES Theaters(theater\_id)

);

-- Seats table

CREATE TABLE Seats (

seat\_id INT PRIMARY KEY AUTO\_INCREMENT,

screen\_id INT NOT NULL,

seat\_number VARCHAR(10) NOT NULL,

seat\_type ENUM('Regular', 'Premium', 'VIP') DEFAULT 'Regular',

FOREIGN KEY (screen\_id) REFERENCES Screens(screen\_id),

UNIQUE (screen\_id, seat\_number)

);

-- Time Slots table

CREATE TABLE TimeSlots (

slot\_id INT PRIMARY KEY AUTO\_INCREMENT,

screen\_id INT NOT NULL,

movie\_id INT NOT NULL,

start\_time DATETIME NOT NULL,

end\_time DATETIME NOT NULL,

FOREIGN KEY (screen\_id) REFERENCES Screens(screen\_id),

FOREIGN KEY (movie\_id) REFERENCES Movies(movie\_id),

CHECK (start\_time < end\_time),

UNIQUE (screen\_id, start\_time, end\_time)

);

-- Bookings table

CREATE TABLE Bookings (

booking\_id INT PRIMARY KEY AUTO\_INCREMENT,

user\_id INT NOT NULL,

slot\_id INT NOT NULL,

booking\_time TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

FOREIGN KEY (user\_id) REFERENCES Users(user\_id),

FOREIGN KEY (slot\_id) REFERENCES TimeSlots(slot\_id)

);

-- Booked Seats table

CREATE TABLE BookedSeats (

booking\_id INT NOT NULL,

seat\_id INT NOT NULL,

slot\_id INT NOT NULL,

PRIMARY KEY (booking\_id, seat\_id),

FOREIGN KEY (booking\_id) REFERENCES Bookings(booking\_id) ON DELETE CASCADE,

FOREIGN KEY (seat\_id) REFERENCES Seats(seat\_id),

FOREIGN KEY (slot\_id) REFERENCES TimeSlots(slot\_id),

UNIQUE (seat\_id, slot\_id)

);

-- Insert Users

INSERT INTO Users (name, email, phone) VALUES

('Ali Khan', 'ali@example.com', '03001234567'),

('Sara Ahmed', 'sara@example.com', '03111234567'),

('Usman Malik', 'usman@example.com', '03211234567');

-- Insert Movies

INSERT INTO Movies (title, duration\_minutes, rating, release\_date) VALUES

('Interstellar', 169, 'PG-13', '2014-11-07'),

('The Dark Knight', 152, 'PG-13', '2008-07-18'),

('Inception', 148, 'PG-13', '2010-07-16');

-- Insert Theaters and Screens

INSERT INTO Theaters (name, location) VALUES

('Cineplex Lahore', 'MM Alam Road, Lahore'),

('Atrium Cinemas', 'Gulberg, Lahore'),

('Nueplex Cinemas', 'DHA, Karachi');

INSERT INTO Screens (theater\_id, name, total\_seats) VALUES

(1, 'Screen 1', 100),

(1, 'Screen 2', 80),

(2, 'Main Screen', 120),

(3, 'Gold Screen', 150);

-- Insert Seats for Screen 1 (100 seats)

SET @row := 0;

INSERT INTO Seats (screen\_id, seat\_number, seat\_type)

SELECT

1,

CONCAT('A', LPAD(@row := @row + 1, 2, '0')),

CASE

WHEN @row <= 10 THEN 'VIP'

WHEN @row <= 30 THEN 'Premium'

ELSE 'Regular'

END

FROM (

SELECT 0 FROM

(SELECT 1 UNION ALL SELECT 2 UNION ALL SELECT 3 UNION ALL SELECT 4

UNION ALL SELECT 5 UNION ALL SELECT 6 UNION ALL SELECT 7

UNION ALL SELECT 8 UNION ALL SELECT 9) t1,

(SELECT 1 UNION ALL SELECT 2 UNION ALL SELECT 3 UNION ALL SELECT 4

UNION ALL SELECT 5 UNION ALL SELECT 6 UNION ALL SELECT 7

UNION ALL SELECT 8 UNION ALL SELECT 9) t2

LIMIT 100

) AS numbers;

-- Insert Seats for Screen 2 (80 seats)

INSERT INTO Seats (screen\_id, seat\_number, seat\_type)

SELECT

2,

CONCAT('B', LPAD(seq, 2, '0')),

CASE

WHEN seq <= 8 THEN 'VIP'

WHEN seq <= 25 THEN 'Premium'

ELSE 'Regular'

END

FROM (

SELECT @num := @num + 1 AS seq

FROM (SELECT 1 UNION SELECT 2 UNION SELECT 3 UNION SELECT 4 UNION SELECT 5

UNION SELECT 6 UNION SELECT 7 UNION SELECT 8 UNION SELECT 9 UNION SELECT 10) t1,

(SELECT 1 UNION SELECT 2 UNION SELECT 3 UNION SELECT 4 UNION SELECT 5

UNION SELECT 6 UNION SELECT 7 UNION SELECT 8 UNION SELECT 9 UNION SELECT 10) t2,

(SELECT @num := 0) num\_init

LIMIT 80

) AS numbered;

-- Insert Time Slots

INSERT INTO TimeSlots (screen\_id, movie\_id, start\_time, end\_time) VALUES

(1, 1, '2024-01-01 15:00:00', '2024-01-01 17:49:00'),

(1, 2, '2024-01-01 18:00:00', '2024-01-01 20:32:00'),

(2, 3, '2026-05-28 15:00:00', '2026-05-28 17:28:00'),

(3, 1, '2026-05-29 19:00:00', '2026-05-29 21:49:00');

-- Drop old procedure if exists

DROP PROCEDURE IF EXISTS BookSeat;

-- Create procedure with screen-seat match validation

DELIMITER $$

CREATE PROCEDURE BookSeat (

IN p\_user\_id INT,

IN p\_slot\_id INT,

IN p\_seat\_id INT,

OUT booking\_status VARCHAR(100)

)

BEGIN

DECLARE booking\_id INT;

DECLARE slot\_screen\_id INT;

DECLARE seat\_screen\_id INT;

-- Fetch screen\_id from slot and seat

SELECT screen\_id INTO slot\_screen\_id FROM TimeSlots WHERE slot\_id = p\_slot\_id;

SELECT screen\_id INTO seat\_screen\_id FROM Seats WHERE seat\_id = p\_seat\_id;

-- Validation logic

IF slot\_screen\_id IS NULL OR seat\_screen\_id IS NULL THEN

SET booking\_status = '❌ Invalid seat or slot.';

ELSEIF slot\_screen\_id != seat\_screen\_id THEN

SET booking\_status = '❌ Seat does not belong to slot''s screen.';

ELSEIF EXISTS (

SELECT 1 FROM BookedSeats

WHERE seat\_id = p\_seat\_id AND slot\_id = p\_slot\_id

) THEN

SET booking\_status = '❌ Seat already booked.';

ELSE

-- Create booking

INSERT INTO Bookings (user\_id, slot\_id)

VALUES (p\_user\_id, p\_slot\_id);

SET booking\_id = LAST\_INSERT\_ID();

-- Book seat

INSERT INTO BookedSeats (booking\_id, seat\_id, slot\_id)

VALUES (booking\_id, p\_seat\_id, p\_slot\_id);

SET booking\_status = '✅ Seat booked successfully.';

END IF;

END$$

DELIMITER ;

-- Sample Bookings (replace seat IDs as needed)

CALL BookSeat(1, 1, 5, @status);

SELECT @status;

CALL BookSeat(2, 1, 6, @status);

SELECT @status;

CALL BookSeat(3, 2, 105, @status); -- Use correct seat\_id for screen\_id of slot\_id = 2

SELECT @status;