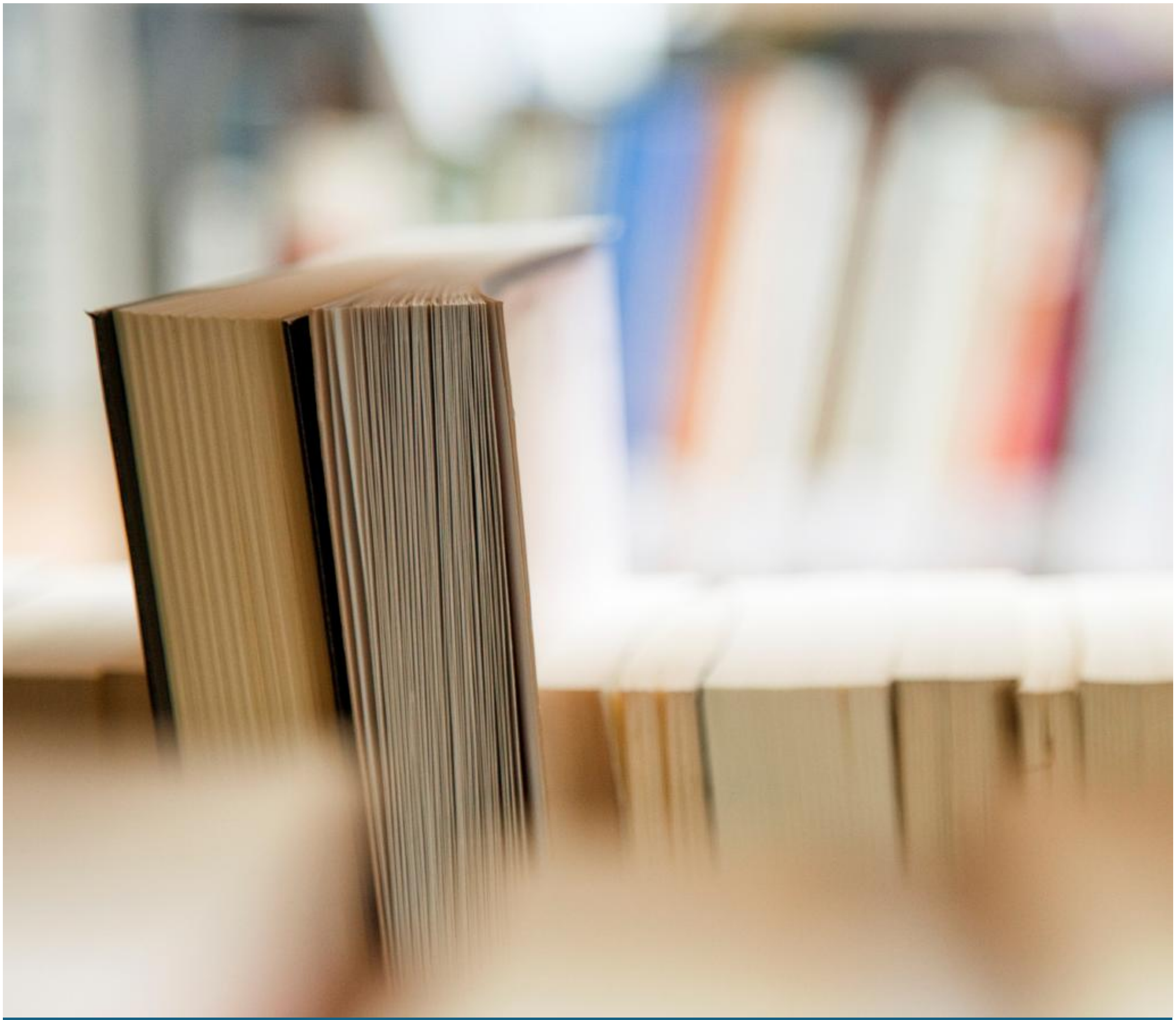


Project

Heli Patel, Halvin Silva Mayes, Yash Patel, Ayush Prajapati

11/30/25

Relational Databases

Phase 1 – Project Idea & Use Case

Project Title: Movie Theater Ticket Booking System

Scenario (Real-World Problem)
Our team chose a movie theater as the real-world scenario.
The theater needs a system to manage movies, halls, showtimes, seats, and ticket bookings.
A relational database will help keep everything organized and accurate

Use Case: Booking a Ticket

The main use case of our system is booking a ticket.

Steps of the booking process:

1. The customer chooses a movie.
2. The customer selects a showtime.
3. The system shows the hall seats and which seats are available.
4. The customer picks one or more seats.
5. The system calculates the total ticket price.
6. The customer confirms the booking.
7. The system creates a Booking record and saves the selected seats.
8. Those seats become reserved so no one else can book them.
9. The customer receives confirmation.

Data stored in the database:

- Customers
- Movies
- Halls
- Showtimes
- Seats
- Bookings
- Seat details for each booking

A relational database is required because the system has many connected parts, such as movies, showtimes, halls, seats, customers, and bookings.

Entities and Attributes

1) Hall

- HallID – INT, PK
- HallName – STRING
- Capacity – INT
- Description – STRING

2) Movie

- MovieID – INT, PK
- Title – STRING
- Genre – STRING
- DurationMinutes – INT
- Language – STRING

- Rating – STRING
- ReleaseDate – DATE
- Status – STRING

3) Showtime

- ShowtimeID – INT, PK
- MovieID – INT, FK → Movie.MovieID
- HallID – INT, FK → Hall.HallID
- StartDateTime – DATETIME
- BaseTicketPrice – DECIMAL
- Status – STRING

4) Customer

- CustomerID – INT, PK
- FirstName – STRING
- LastName – STRING
- Email – STRING
- Phone – STRING
- CreatedDate – DATETIME

5) Seat

- SeatID – INT, PK
- HallID – INT, FK → Hall.HallID
- RowLabel – STRING

- SeatNumber – INT
- SeatType – STRING
- IsActive – BOOL

6) Booking

- BookingID – INT, PK
- CustomerID – INT, FK → Customer.CustomerID
- ShowtimeID – INT, FK → Showtime.ShowtimeID
- BookingDateTime – DATETIME
- Status – STRING

7) BookingSeat

- BookingSeatID – INT, PK
- BookingID – INT, FK → Booking.BookingID
- SeatID – INT, FK → Seat.SeatID
- SeatPriceAtBooking – DECIMAL

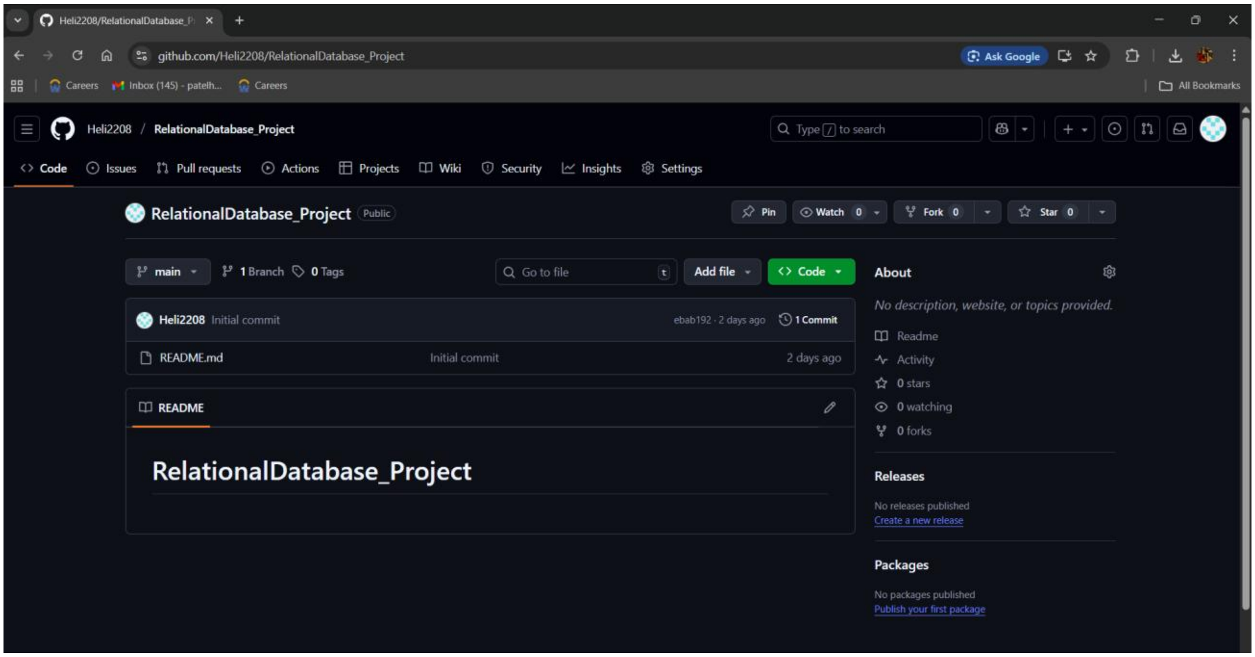
2. Relationships

1. **Hall – Showtime**
   - One Hall **hosts many** Showtimes.
   - Each Showtime is in **one** Hall.
2. **Movie – Showtime**
   - One Movie **has many** Showtimes.
   - Each Showtime is for **one** Movie.

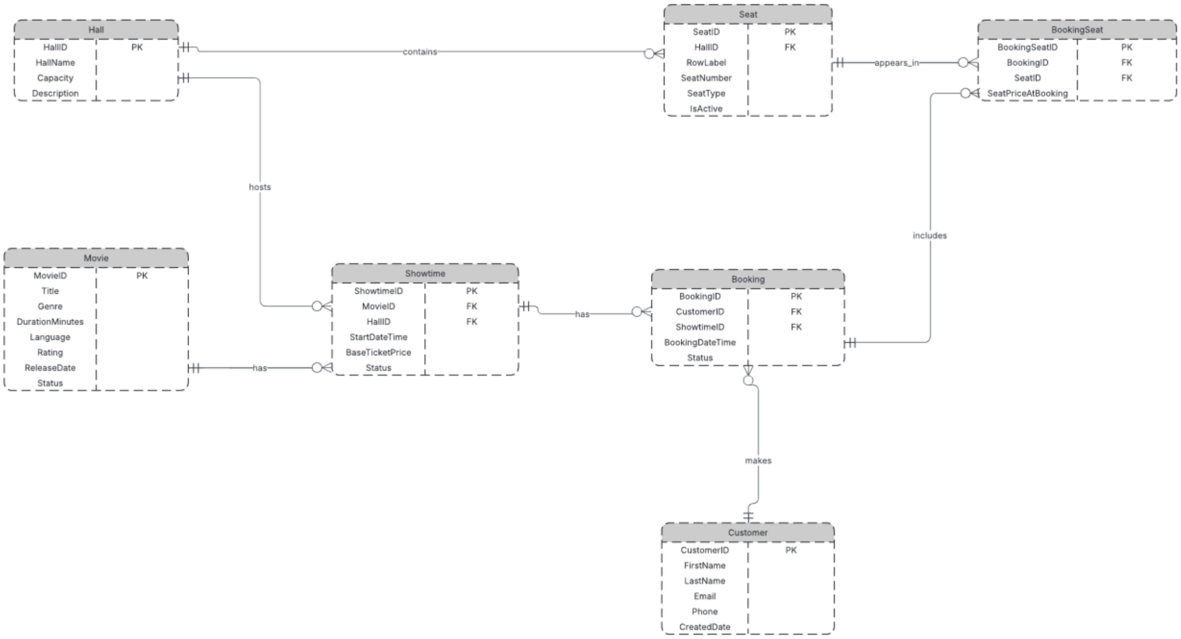
3. **Hall – Seat**
    - One Hall **contains many** Seats.
    - Each Seat belongs to **one** Hall.
4. **Customer – Booking**
    - One Customer can **make many** Bookings.
    - Each Booking is made by **one** Customer.
5. **Showtime – Booking**
    - One Showtime can **have many** Bookings.
    - Each Booking is for **one** Showtime.
6. **Booking – BookingSeat**
    - One Booking **includes many** BookingSeat records.
    - Each BookingSeat belongs to **one** Booking.
7. **Seat – BookingSeat**
    - One Seat can appear in **many** BookingSeat records (different bookings/showtimes).
    - Each BookingSeat is for **one** Seat.

GitHub Link:

https://github.com/Heli2208/RelationalDatabase_Project



Phase 2: ERD



Hall	
HallID	PK
HallName	
Capacity	
Description	

Seat	
SeatID	PK
HallID	FK
RowLabel	
SeatNumber	
SeatType	
IsActive	

BookingSeat	
BookingSeatID	PK
BookingID	FK
SeatID	FK
SeatPriceAtBooking	

Movie	
MovieID	PK
Title	
Genre	
DurationMinutes	
Language	
Rating	
ReleaseDate	
Status	

Showtime	
ShowtimeID	PK
MovieID	FK
HallID	FK
StartDateTime	
BaseTicketPrice	
Status	

Booking	
BookingID	PK
CustomerID	FK
ShowtimeID	FK
BookingDateTime	
Status	

Customer	
CustomerID	PK
FirstName	
LastName	
Email	
Phone	
CreatedDate	