Movie Ticket Booking System

Ameen Safi

Student ID: 920689065

Github: asafi67

Checkpoint #	**Date Submitted**
Checkpoint 1	2/20/2024
Checkpoint 2	3/7/2024

Checkpoint 3	3/21/2024

Table of Contents

Project Description

The design of this database system is aimed at improving human-computer interaction (HCI) for both customers and theater management. The system will enhance user experience, as well as promote a boost in efficiency for those who are managing the sales. While modern ticket booking has come a long way from the days of having to physically wait in a long line on the day of a show, only to find out when one reaches the booth that the show is sold out, certain imperfections still exist. Limited visibility into available seats often leads to annoyed customers. Long physical waits have mostly just transferred to long virtual wait times. Furthermore, while reliance on technology has increased a substantial amount, theater admins still often face the challenge of optimizing the space of their theaters, and simultaneously managing bookings in the most profitable way.

This movie ticket booking database system is an adaptive platform which performs a variety of capabilities to manage the movie booking process. It will allow users to view showtimes, select their seats, search for available movies, and even purchase tickets. Similarly, those who manage movie viewing establishments can use the system to manage their showtime schedules, manage seating arrangements, ticket inventory, and manage customer data.

1. Seat Selection: A feature for allowing users to look at what seats are available for certain showtimes, get real-time updates on their availability, and actually choose their seats by selecting seat icons on an interactive diagram of the theater room.

2. Recommendations: A feature for movie recommendations that cater to the personal preferences of the user. They will also be based on previous movie ticket purchases, and what their viewing history is.
3. Loyalty Rewards Feature: A feature that rewards customers who frequently watch movies. Rewards include things like discounts, special offers, and exclusive benefits.
4. Predictive Analysis: A feature that can predict the demand for movies. This will help optimize marketing strategies, and also allocate seating within the theater better.

An existing product within the current market for movie ticket booking which would benefit from this database system is Fandango. Fandango is one of the most popular online ticketing platforms, which is also used for non-movie events like concerts or comedy shows. However, Fandango's seat selection is not as dynamic as could be with the ability to have live updates on seating changes. Personalized recommendations are still in an infancy state within Fandango. Such aspects would be improved with this db, and lead to more loyalty from customers and higher user satisfaction reports.

In addition, Cineplex Odeon is a theater chain very popular in North America. Our database will allow Cineplex to analyze the patterns in user booking frequencies and habits. This will allow Cineplex to better specify their marketing to target users with promotions, offers, special discounts that would lead to more sales and eventually increased loyalty.

Functional Database Requirements

1. Customer

1.1. Customer can register an account
1.2. Customer can perform logging in with a username and password
1.3. Customer shall select a profile picture to represent their account
1.4. Customer's username shall be unique
1.5. Customer can edit profile information
1.6. Customer can delete their account
1.7. Customer shall only be allowed a single active session per device at any given time
1.8. Customer can solicit help from the help center
1.9. Customer shall have access to accessibility info regarding their showings or events
1.10. Customer can login/register using their email or social media
1.11. Customer can share their reviews of movies on social media directly from the system
1.12. Customer can change their password in the event they forget
1.13. Customer can report issues in their booking experience
1.14. Customer can share movie bookings on social media directly from the system
1.15. Customer shall have access to view their ticket booking history
1.16. Customer can filter their search for shows based on categories
1.17. Customers can see what movies are coming soon and future showtimes.
1.18. Customer can use loyalty rewards points to get discounts or other reward items
1.19. Customers can view details about movies, such as plot, cast or even reviews.
1.20. Customer can select seats for desired showtimes
1.21. Customer can complete transactions online
1.22. Customer can cancel transactions
1.23. Customer can update notification preferences
1.24. Customer can opt in for promotional offers and newsletters
1.25. Customer can review movies after viewing them
1.26. Customer shall receive confirmation emails after their booking transactions have been completed
1.27. Customer shall receive confirmation emails after their booking transactions have been canceled
1.28. Customer can send ticket confirmation information to someone via SMS
1.29. Customer can split payment for their movie tickets into multiple forms of payment
1.30. Customer can gift movie tickets to other customers as part of the social feature

1.31. Customer can sign up for text alerts regarding promotions and offers

2. Theater Management
   2.1. Management can view system logs or error reports
   2.2. Management can perform database backups on routine
   2.3. Management can monitor performance of the system
   2.4. Management can apply software updates or patches
   2.5. Management can log in and have admin privileges
   2.6. Management can update movie categories
   2.7. Management can upload movie poster images
   2.8. Management shall have to authenticate when logging in using multi-factor authentication
   2.9. Management shall have their access be mitigated by their role, and depending on their administrative tasks
   2.10. Management can analyze trends or patterns in booking
   2.11. Management can monitor customer engagement
   2.12. And admin can add new information such as movies to the database
   2.13. Management can edit/update movie details
   2.14. Management can manage seating arrangements or different locations of theaters
   2.15. Management can manage the accounts of users: functions such as creation, deletion or updates.
   2.16. Management can troubleshoot technical issues
   2.17. Management can view or respond to customer with questions or grievances
   2.18. Management can monitor customer reviews
   2.19. Management can create reports for ticket revenue and sales
   2.20. Management can create, add, or delete admin accounts
   2.21. Management can edit configurations in the system settings and preferences
   2.22. Management can send system reports to other admins logged into the system
   2.23. Management can handle customer transactions for which the payment methods were fraudulent
   2.24. Management can rearrange the projections rooms to accommodate ticket demand
   2.25. Management can close ticket sales for certain showtimes based on emergencies
   2.26. Management can choose when sales open for different movies
   2.27. Management can assist customers in seat selection
   2.28. Management can schedule the dates for promotional offers and discounts
   2.29. Management can create promotional bundle packages
   2.30. Management can edit whether or not a movie shows up as “sold out”
   2.31. Management can verify movie attendance

Non-Functional Database Requirements

1. Performance
    1.1. DB shall host tools to control monitoring performance operations and optimization
    1.2. DB shall handle at least 1000 user connections at a given time
    1.3. DB shall have a response time below 2 seconds for common querying
    1.4. DB shall have an average of 100 transactions per second throughput
    1.5. DB shall optimize indexing for data retrieval
2. Security
    2.1. User authentication will be encrypted
    2.2. Crucial information will be restricted based on roles and permissions
    2.3. Logs will be kept to monitor user activity
    2.4. Security audits and vulnerability assessments will be routine
    2.5. Backed up data shall be encrypted
3. Scalability
    3.1. DB shall support automatic scaling
    3.2. DB shall have the ability to perform load balancing by distributing data across multiple servers
    3.3. DB shall host tools for data partitioning
    3.4. DB shall support upgrades which occur without downtime
    3.5. DB shall support horizontal scalability
4. Capability

4.1. DB shall host ways to perform data validation

4.2. DB shall support full-text search functionality

4.3. DB shall support location data and media files

4.4. DB shall support ACID properties

4.5. DB shall support transaction isolation

5. Environmental

5.1. DB shall be compatible with Linux, MacOS, Windows and other OS

5.2. DB shall seek to minimize consumption of energy

5.3. DB shall support cloud deployment functionality

5.4. DB shall be optimized for fault tolerance

5.5. DB shall comply with industry standards for data processing and storage

6. Coding Standards

6.1. DB shall comply with industry naming conventions

6.2. DB shall enforce standard coding

6.3. DB shall follow versioning and deployment process standard for schema changes

6.4. DB shall host tools for code analysis for potential bugs

6.5. DB shall optimize queries for improved performance and efficient resource consumption

7. Media Storage

7.1. DB shall support media such as images or videos

7.2. Media files shall be compressed

7.3. DB shall support the streaming of large media files

7.4. DB shall host tools for media organization

7.5. DB shall accommodate increasing amounts of media content

8. Privacy

8.1. DB shall follow data privacy regulations like GDPR and CCPA

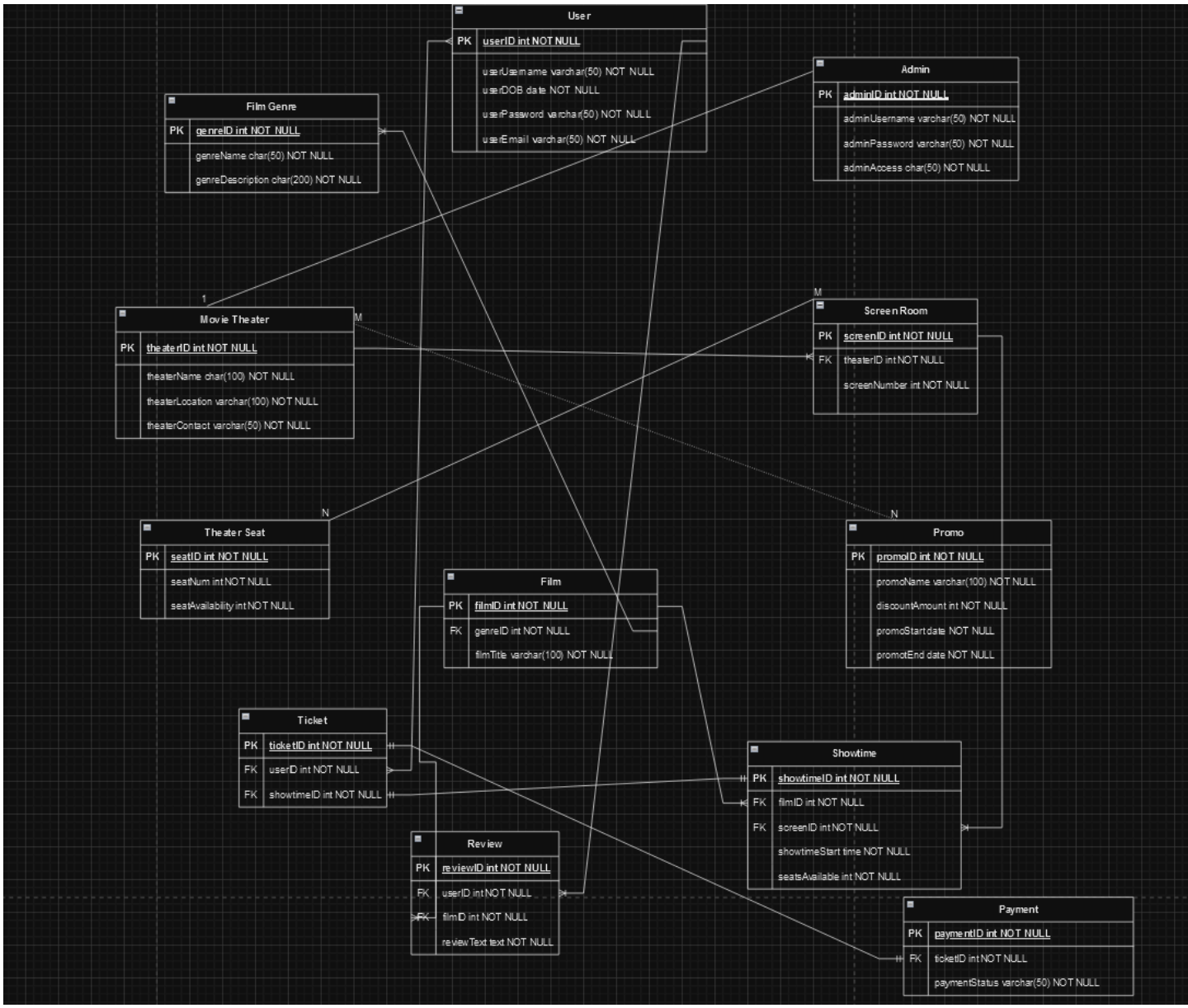
8.2. Information which is personally identifiable shall be encrypted when either in transit or even at rest

8.3. DB shall implement data anonymization

8.4. DB shall mask or redact sensitive information

8.5. DB shall audit access to sensitive information

Entity Relationship Diagram



Entity Descriptions

Note: comp = composite, alphanum = alphanumeric

1. Movie Theater (Strong)
    - theaterId: key, numeric
    - theaterName: comp, multi-value, alphanum
    - theaterLocation: comp, alphanum
    - theaterContact: comp, alphanum
2. User (Strong)
    - userId: key, numeric
    - userDOB: multi-value, timestamp
    - userPassword: comp, alphanum
    - userUsername: comp, alphanum
    - userEmail: comp, alphanum
    - userProfilePic: comp, alphanum
3. Screen Room (Strong)
    - screenId: key, numeric
    - theaterId: comp, numeric
    - screenNumber: comp, numeric
4. Showtime (Strong)
    - showtimeId: key, numeric
    - showtimeStart: comp, timestamp
    - filmId: comp, numeric
    - screenId: comp, numeric

- seatsAvailable: composite, numeric

5. Theater Seat (Strong)
    - seatId: key, numeric
    - seatNum: comp, alphanum
    - seatAvailability: comp, alphanum
6. Ticket (Strong)
    - ticketId: key, numeric
    - userId: comp, numeric
    - showtimeId: comp, numeric
7. Payment (Strong)
    - paymentId: key, numeric
    - ticketId: comp numeric
    - paymentStatus: comp, alphanum
8. Admin (Strong)
    - adminId: key, numeric
    - adminUsername: comp, alphanum
    - adminPassword: comp, alphanum
    - adminAccess: comp, alphanum
9. Film Genre (Strong)
    - genreId: key, numeric
    - genreName: comp, alphanum
    - genreDescription: comp, alphanum
10. Film (Strong)

- filmId: key, numeric
- filmTitle: comp, alphanum
- genreID: comp, alphanum

11. Review Entity (Strong)
    - reviewId: key, numeric
    - userId: comp, numeric
    - filmId: comp, numeric
    - reviewText: comp, alphanumeric
12. Promo Entity (Strong)
    - promoId: key, numeric
    - promoName: comp, multi-value, alphanum
    - discountAmount: comp, numeric
    - promoStart: comp, timestamp
    - promoEnd: comp, timestamp

EER

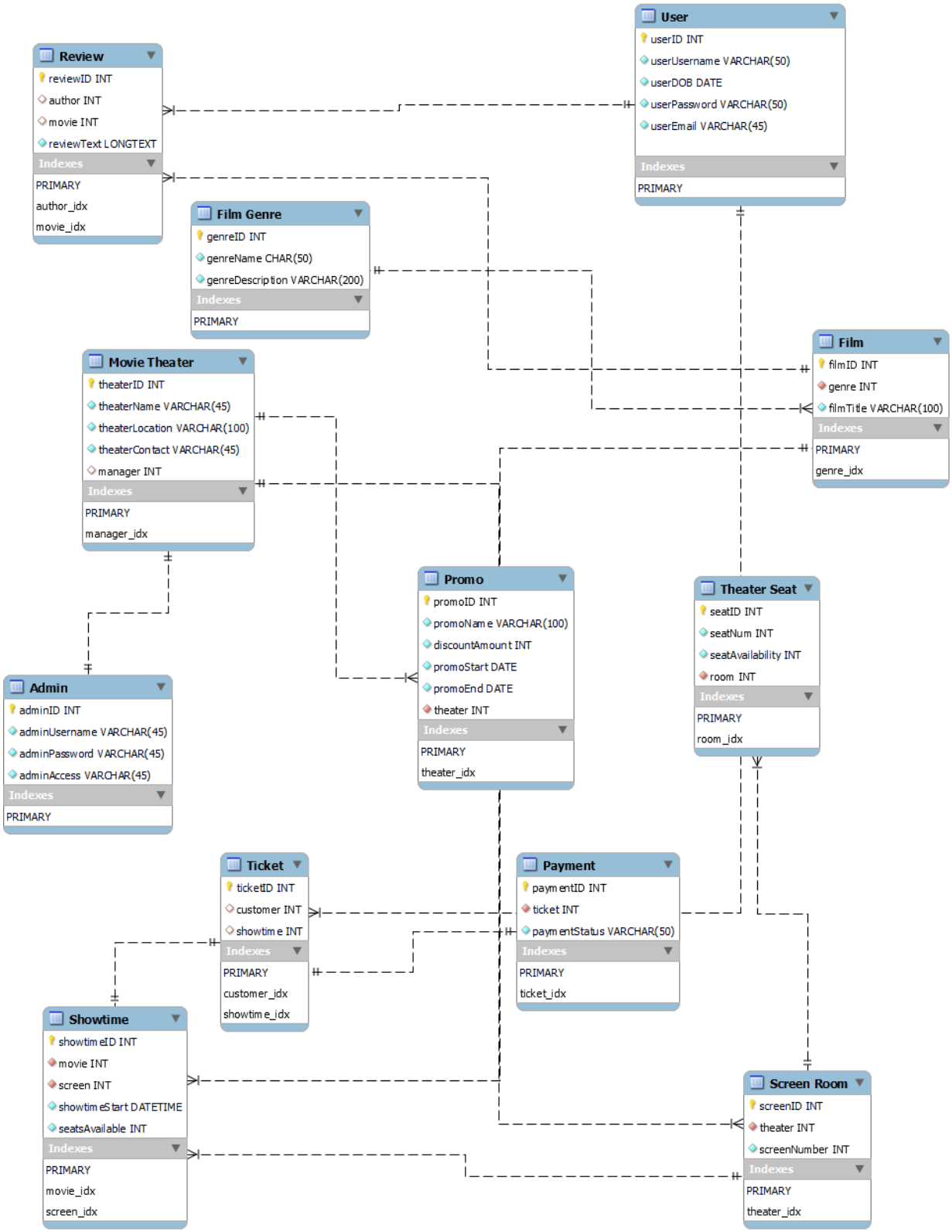

User
userID INT
userUsername VARCHAR(50)
userDOB DATE
userPassword VARCHAR(50)
userEmail VARCHAR(45)
Indexes
PRIMARY
Review
reviewID INT
author INT
movie INT
reviewText LONGTEXT
Indexes
PRIMARY
author_idx
movie_idx
Film Genre
genreID INT
genreName CHAR(50)
genreDescription VARCHAR(200)
Indexes
PRIMARY
Film
filmID INT
genre INT
filmTitle VARCHAR(100)
Indexes
PRIMARY
genre_idx
Movie Theater
theaterID INT
theaterName VARCHAR(45)
theaterLocation VARCHAR(100)
theaterContact VARCHAR(45)
manager INT
Indexes
PRIMARY
manager_idx
Promo
promoID INT
promoName VARCHAR(100)
discountAmount INT
promoStart DATE
promoEnd DATE
theater INT
Indexes
PRIMARY
theater_idx
Theater Seat
seatID INT
seatNum INT
seatAvailability INT
room INT
Indexes
PRIMARY
room_idx
Admin
adminID INT
adminUsername VARCHAR(45)
adminPassword VARCHAR(45)
adminAccess VARCHAR(45)
Indexes
PRIMARY
Ticket
ticketID INT
customer INT
showtime INT
Indexes
PRIMARY
customer_idx
showtime_idx
Payment
paymentID INT
ticket INT
paymentStatus VARCHAR(50)
Indexes
PRIMARY
ticket_idx
Showtime
showtimeID INT
movie INT
screen INT
showtimeStart DATETIME
seatsAvailable INT
Indexes
PRIMARY
movie_idx
screen_idx
Screen Room
screenID INT
theater INT
screenNumber INT
Indexes
PRIMARY
theater_idx

Table	FK	On Delete	On Update	Comment
Review	author	NO ACTION	CASCADE	The review referencing the deleted user can still exist just without a reference to a particular user. So we do NO ACTION.
Review	movie	SET NULL	CASCADE	A review may exist only if a movie that it is reviewing exists. So if the movie is deleted we set the review to null.
Movie Theater	manager	CASCADE	CASCADE	We want to ensure that the db correctly tracks the current manager of the theater at any given time.
Ticket	customer	CASCADE	CASCADE	Since customers own their tickets, we want the ticket's owner status to reflect what is known of that user in the database.
Ticket	showtime	CASCADE	CASCADE	Since a ticket has a one to one relationship with a showtime, if a showtime disappears the ticket must as well.
Payment	ticket	CASCADE	NO ACTION	Since a payment has a one to one relationship with a ticket, if a ticket is deleted (refund) then the payment must also be deleted.
Screen Room	theater	CASCADE	CASCADE	Screen rooms exist inside of theaters so if the theater is updated or deleted that must be reflected in the screen room table as well.
Promo	theater	CASCADE	CASCADE	Theaters have the promotions, so if the theaters are updated or deleted that must be reflected within the promotion.
Film	genre	CASCADE	CASCADE	Films have a film genre attribute so films must reflect changes made to the genres table.

Showtime	movie	CASCADE	CASCADE	If the movie is changed or deleted that can affect the showtime because of varying movie lengths or if the movie is deleted there will be nothing to show during the showtime.
Showtime	Screen	CASCADE	NO ACTION	If the screen room is updated the showtime may still occur in that room so the showtime is left with NO ACTION.