

Northeastern University
DAMG6210 Data Management and Database Design
Group Name: DATA INSIGHTS

TOPIC: EVENT MANAGEMENT SYSTEM

Name	NUID
Disha Patil	002768900
Anuj Kumar	002766036
Shubham Sable	002747045

PROBLEM STATEMENT

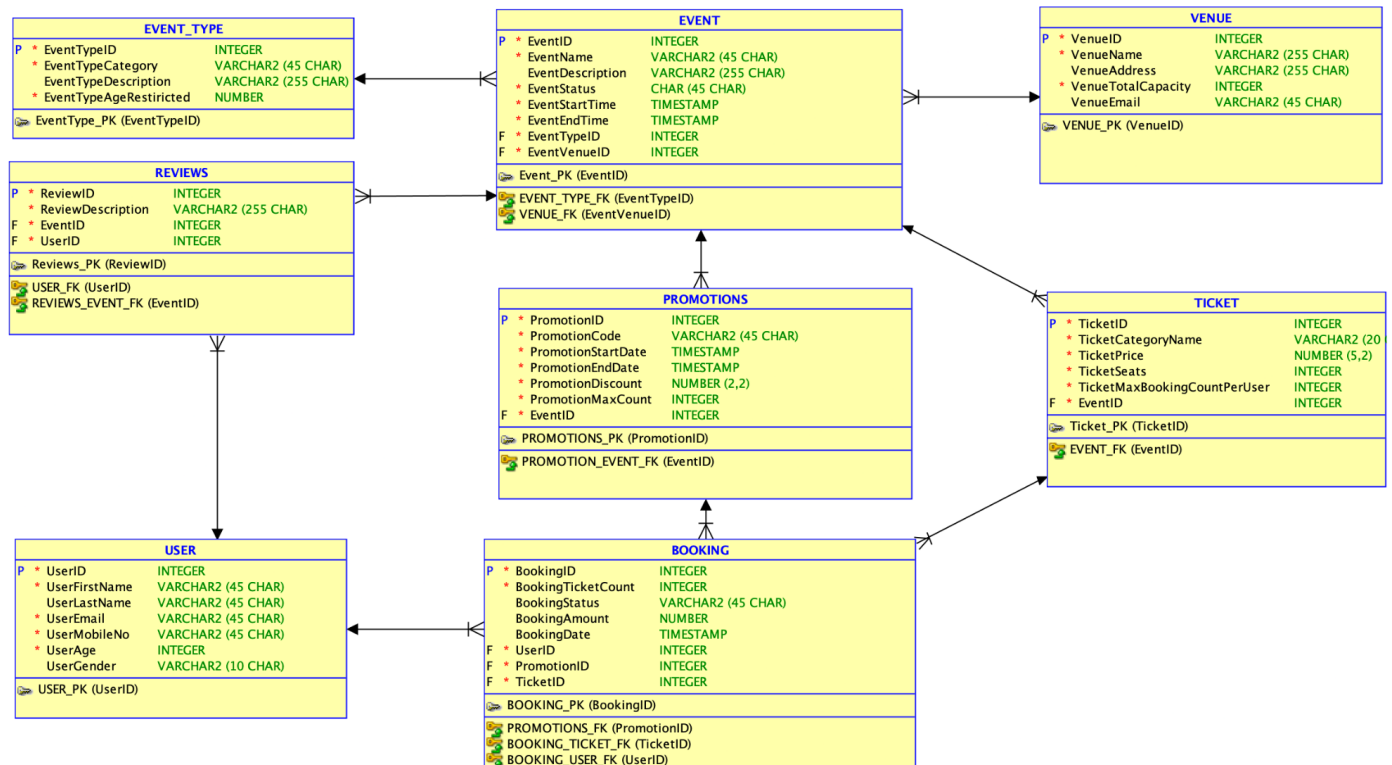
An event booking system is a critical component of the entertainment industry, facilitating the management of events, ticket sales, and user interactions. This system enables event organizers, companies, and customers to plan, promote, and attend various events seamlessly. The problem statement outlines the need for an Event Booking System to support the efficient and secure management of events, bookings, and user interactions. The current landscape of event management and booking is fragmented and lacks a centralized and userfriendly platform for all stakeholders involved. Organizers, companies, and customers face several challenges that need to be addressed by the proposed Event Booking System. With multiple users trying to book tickets simultaneously, ensuring data consistency and preventing issues like overbooking or double bookings is a challenge. Managing concurrent transactions and preventing conflicts is critical.

OBJECTIVES

- Develop a centralized Event Booking System for efficient event management.
- Create a user-friendly interface for customers to browse and book events seamlessly.

- Ensure data consistency and prevent overbooking issues during peak demand.
- Establish robust concurrency management mechanisms.
- Simplify the booking process with secure payment options.
- Implement stringent data security measures for user data protection.

Revised ER Diagram



CHANGES IMPLEMENTED

1. The 'Promotion Count Used' column has been removed, and it is now calculated based on the information in the Bookings table.
2. The 'Ticket Left Count' columns have been removed, and the ticket count is calculated based on the bookings data.
3. An 'EventTypeAgeRestricted' column has been added to restrict events based on age criteria.
4. An 'Event Status' field has been added to mark events as either 'To Be Done,' 'Completed,' or 'Canceled,' allowing for better event tracking and management."

5. Added "maxbookingcountPerUser," has been implemented to restrict users from booking a maximum number of tickets for a specific category. This ensures that each user can book only up to a specified limit of tickets for that category

BUSINESS RULES

1. User Registration: Users have the ability to register themselves on the platform.
2. Booking Restrictions: Only registered users are permitted to book tickets for events.
3. Event Type and Venue Association: Events can be associated only with available event types and venues that are listed within the system.
4. Seat Availability: An event booking can proceed if there are available seats in the chosen category.
5. Seat Type per Booking: Each booking is limited to one seat type for the chosen event.
6. The system should enforce a promotion code's validity period. Users can only apply promotion codes within the specified time frame. Expired promotion codes should be rejected.
7. Cancellation Policy: A booking can be canceled up to 30 minutes before the event's start time, and the cancellation will apply to all seats associated with that booking.
8. Seat Availability Check: Before a booking is finalized, the system verifies the availability of seats to ensure they are not already occupied.
9. Implement robust data protection measures to safeguard user information, including personal details and payment data. Ensure compliance with data privacy regulations to build user trust.
10. Only the users who have registered should have the option to provide reviews. This feedback system helps organizers and future attendees make informed decisions.

CONSTRAINTS:

1. Ticket Booking Age Restrictions:

Business Logic: Users must meet a minimum age requirement to book certain types of tickets (e.g., "18+" tickets).

Implementation: When a user selects a ticket category that has an age restriction, request the user's date of birth during the booking process. Calculate the user's age based on the date of birth provided and compare it to the minimum age requirement for the selected ticket category. If the user's age doesn't meet the criteria, display an error message and prevent the booking.

2. Promotion Start Date:

Business Logic: Promotions can only be applied to bookings if the current date is equal to or later than the promotion start date.

Implementation: When a user applies a promotion code during the booking process, the system should check the current date. If the current date is earlier than the promotion start date, display an error message indicating that the promotion is not yet active. Only allow the promotion to be applied if the current date meets the criteria.

Similarly, If the current date is on or after the promotion start date but later than the event start date, display a message indicating that the promotion can only be used before the event starts. Only allow the promotion to be applied if the current date meets both criteria: It's on or after the promotion start date, and it's before the event start date.

3. Number of Tickets:

Business Logic: Enforce a limit on the maximum number of tickets that a single user can book for an event.

Implementation: During the booking process, keep track of the number of tickets a user is attempting to book for an event. If the count exceeds the defined maximum limit, inform the user that they have reached the maximum booking limit for that event and restrict further bookings.

4. After Start Time of Event - No Bookings:

Business Logic: Once the event's start time has passed, no further bookings should be allowed for that event.

Implementation: As the event's start time approaches, the system should disable the booking option for that event. Users attempting to book tickets after the event has started should receive an error message indicating that bookings are no longer available for the event.

5. Venue Availability:

Business Logic: Ensure that a venue is available for the event's specified date and time.

Implementation: When an event is being created or when a user is attempting to book tickets for an event, the system should query the database to check the availability of the specified venue for the event's date and time. If the venue is already booked for that period, the system should provide alternative options or dates for the event or suggest selecting a different venue.

6 . Seat Availability:

Business logic - The number of seats booked does not exceed the available seat count, preventing overbooking. Here's how this logic can be implemented:

Implementation: When a user attempts to make a booking, the system should check the number of available seats for the event. If the number of available seats is greater than or equal to the ticket count requested by the user, the system allows the booking to proceed. If the number of available seats is less than the ticket count, display an error message indicating that there are insufficient seats available. Inform the user of the current seat availability and suggest booking a smaller number of tickets.

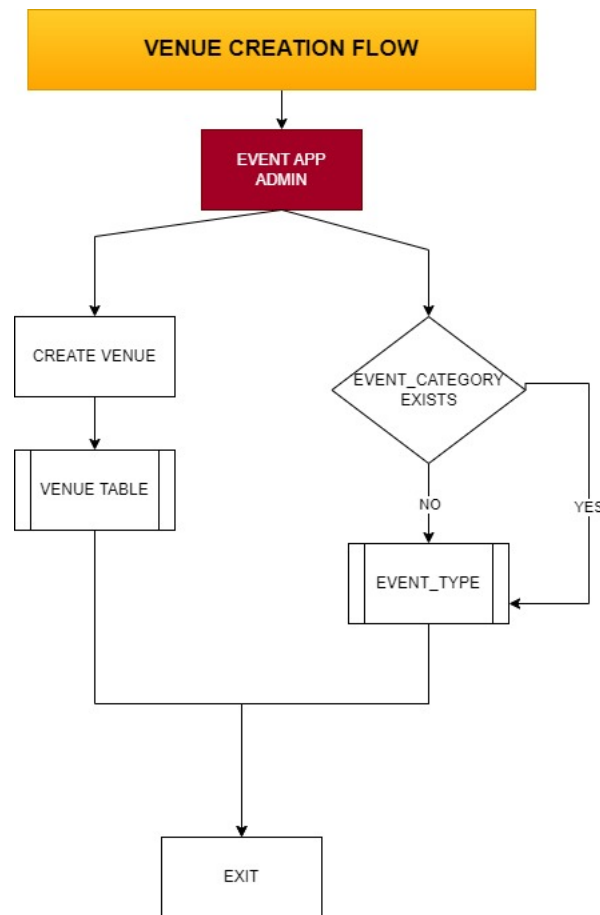
7. Ticket Creation

Business Logic: Checking the venue's capacity and then adding the desired number of tickets.

Implementation: When creating tickets for an event, the system should check the venue's capacity, which specifies the maximum number of attendees the venue can accommodate.

DATA FLOW DIAGRAMS

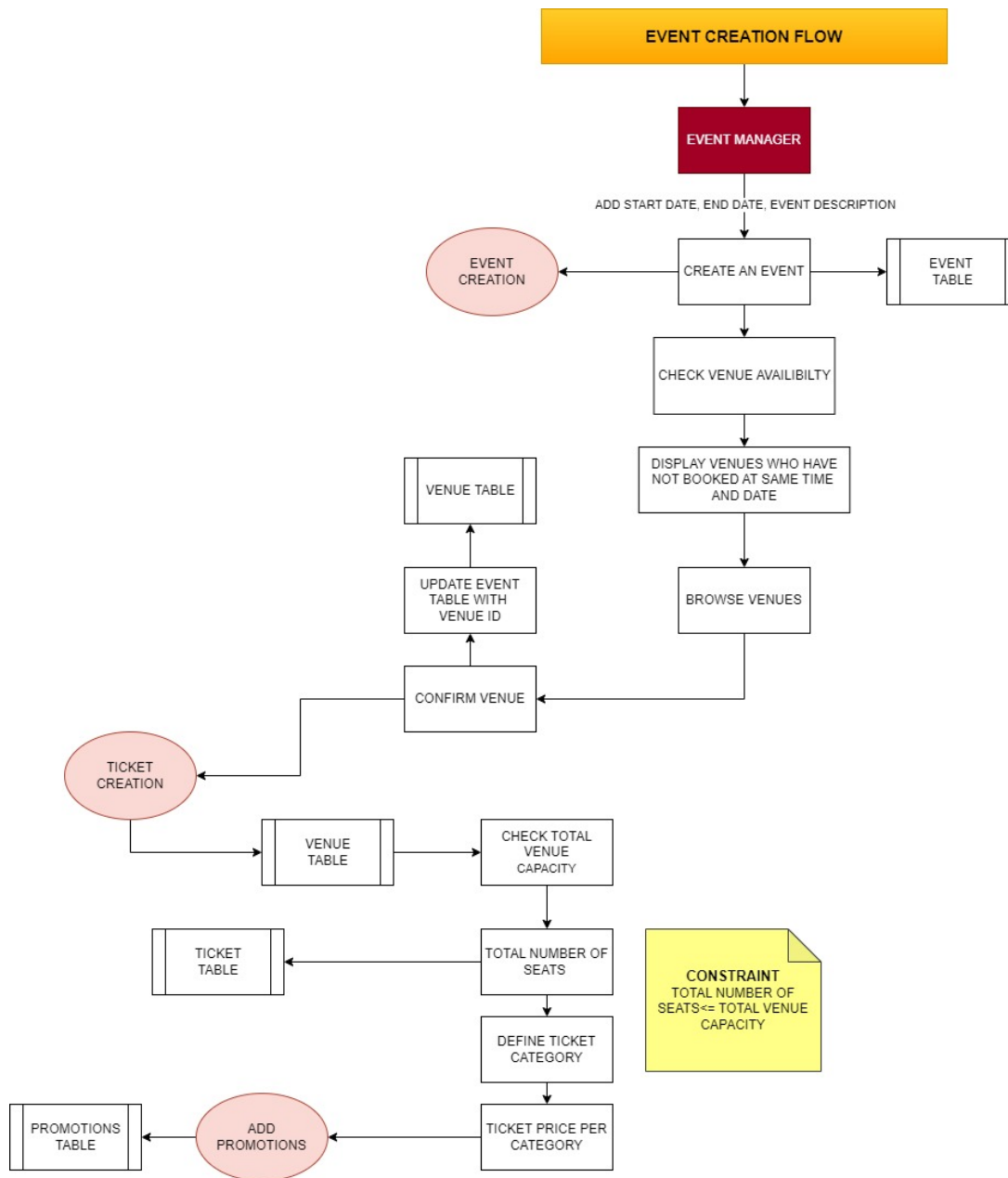
Venue Creation Flow



Venue Creation workflow:

1. The event app administrator initiates the process by creating a new venue entry within the venue table.
2. The administrator provides essential details for the venue, including the venue name, physical address, total capacity, and the venue's designated email address.
3. The administrator conducts a preliminary check to confirm the existence of the event category associated with the venue.
4. If the event category does not already exist, the administrator creates a new event category and assigns it a tag, indicating whether it's age-restricted or not.

EVENT CREATION FLOW

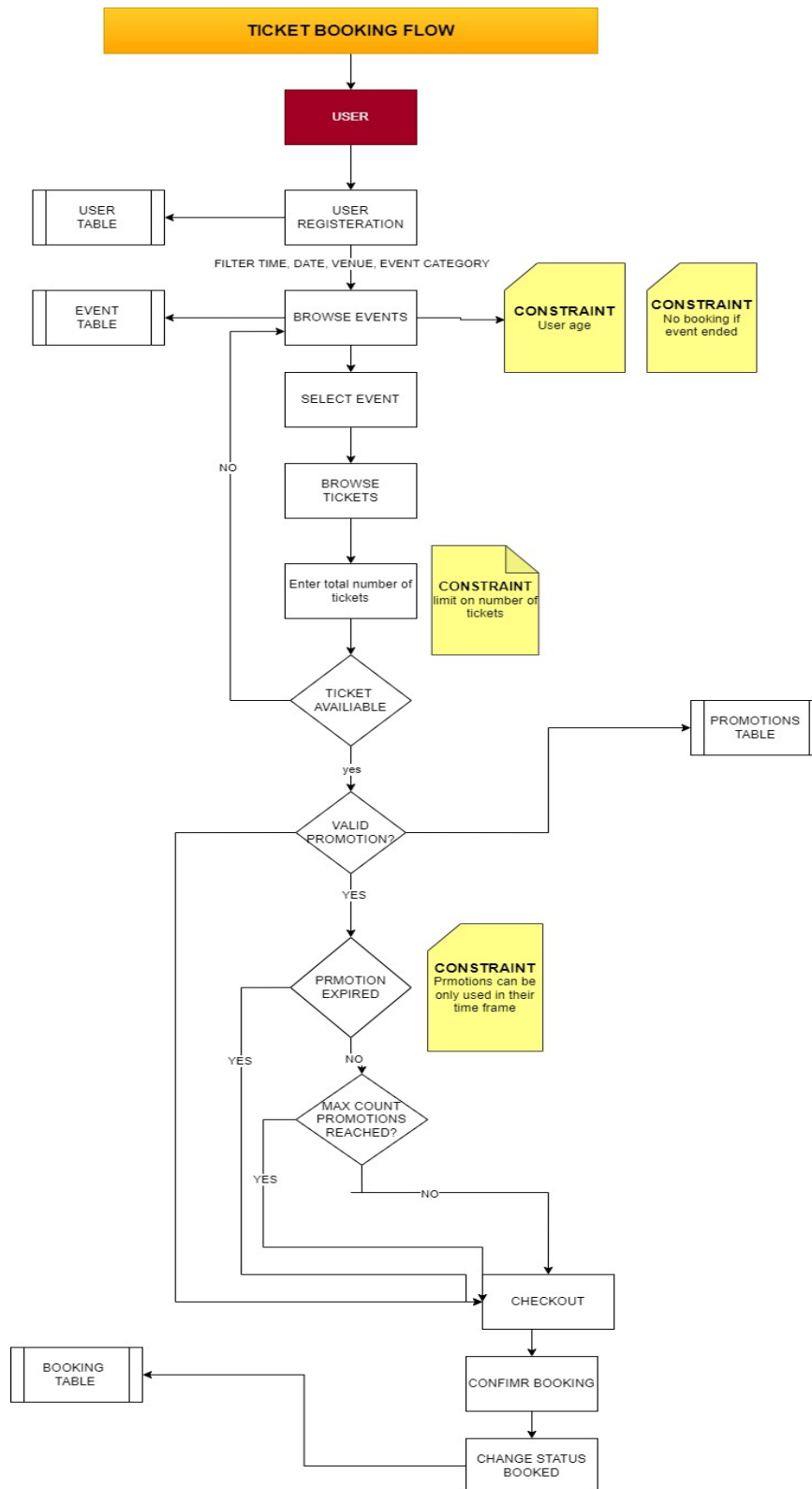


Event creating Flow:

1. The event manager initiates event creation, providing event name, description, and start/end date and time.
2. The system checks venue availability from a list of admin-created venues.

3. The event manager selects and confirms a venue, and the venue ID is linked to the event.
4. Ticket categories are created, specifying seats and prices, ensuring they don't exceed the venue's capacity.
5. Promotions are added, including the code, start/end dates, and usage limits.
6. The event is created, ready for attendees to book tickets and enjoy.

TICKET BOOKING FLOW

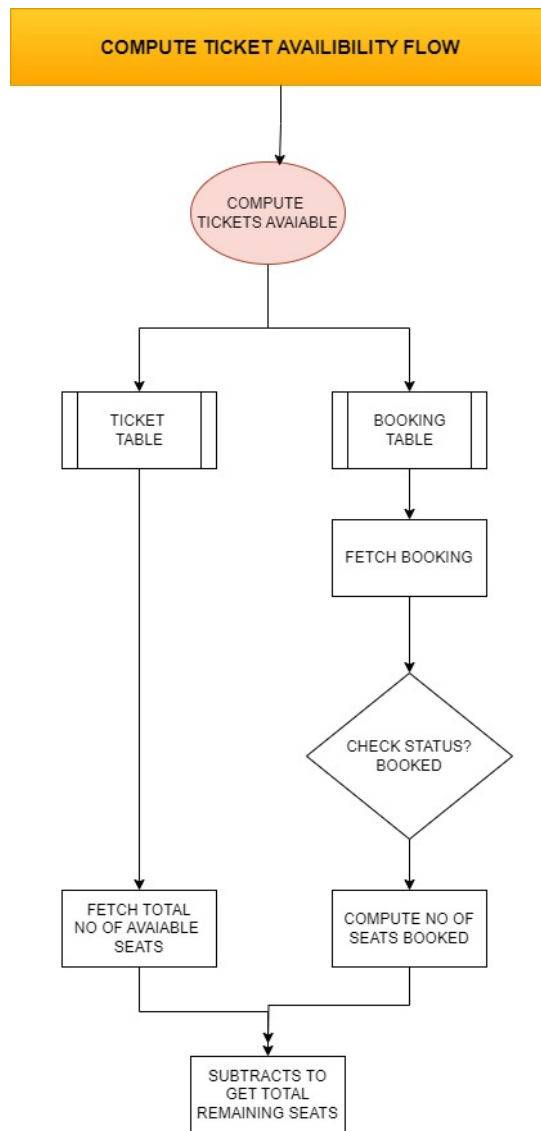


To book a ticket the following steps are followed:

1. Users can filter events based on criteria like time, venue, or category, then browse through the list of available events.

2. Age restrictions are in place to prevent underage users from viewing age-restricted events.
3. A specific event is selected from the list, and users proceed to view available tickets for that event.
4. Users specify the number of tickets they wish to purchase. The system enforces constraints to ensure that the selected quantity does not exceed the allowed booking limit per category.
5. The system checks the availability of seats to verify that the requested number of seats is available for the chosen event.
6. The system verifies the promotion code against a promotion table to ensure its validity.
7. If the promotion code is valid, the system checks if the limit on the number of times the code can be used has been reached.
8. If the promotion code is valid and its usage limit has not been reached, the user can proceed to checkout the tickets. The booking table is updated with the booking entry to confirm the ticket purchase.

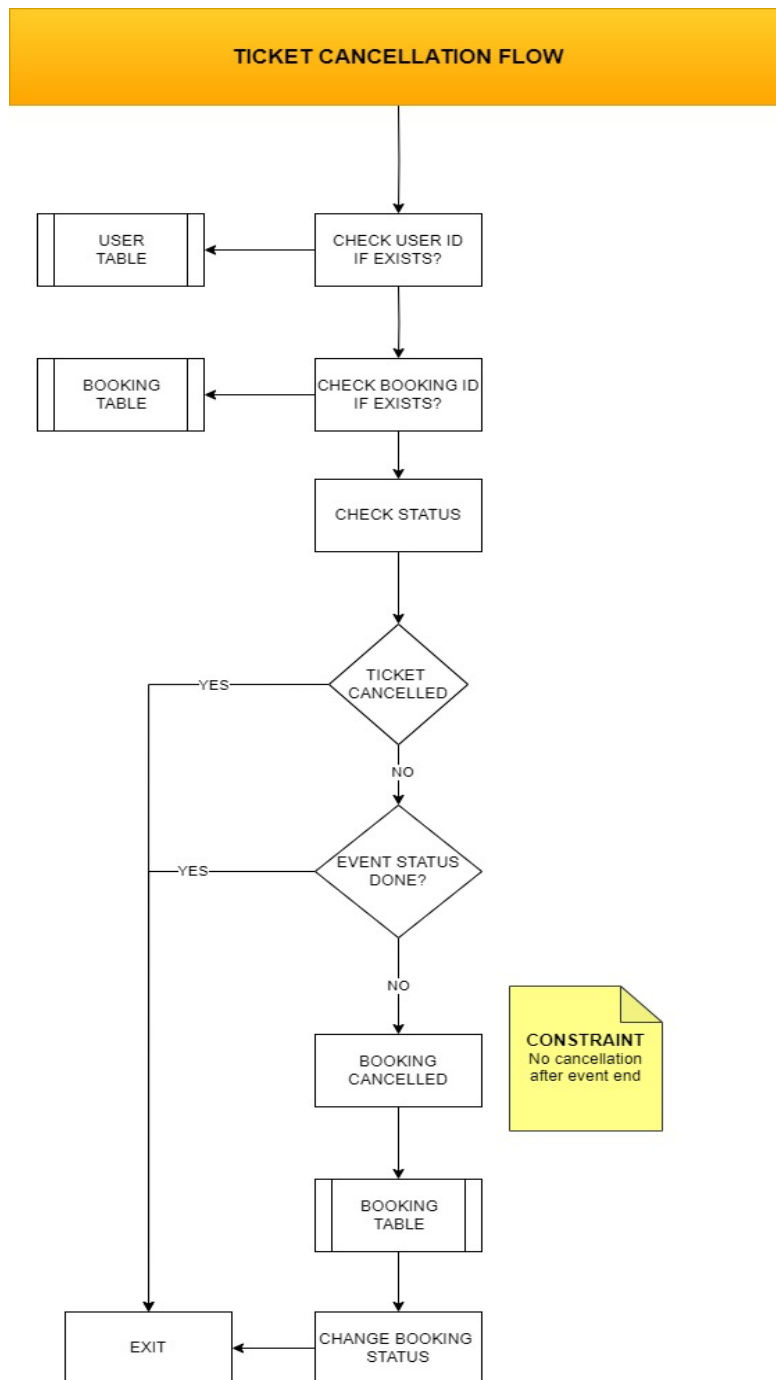
COMPUTE TICKET AVAILIBLTY FLOW



Following are the steps for computing the number of available tickets with improved grammar and clarity:

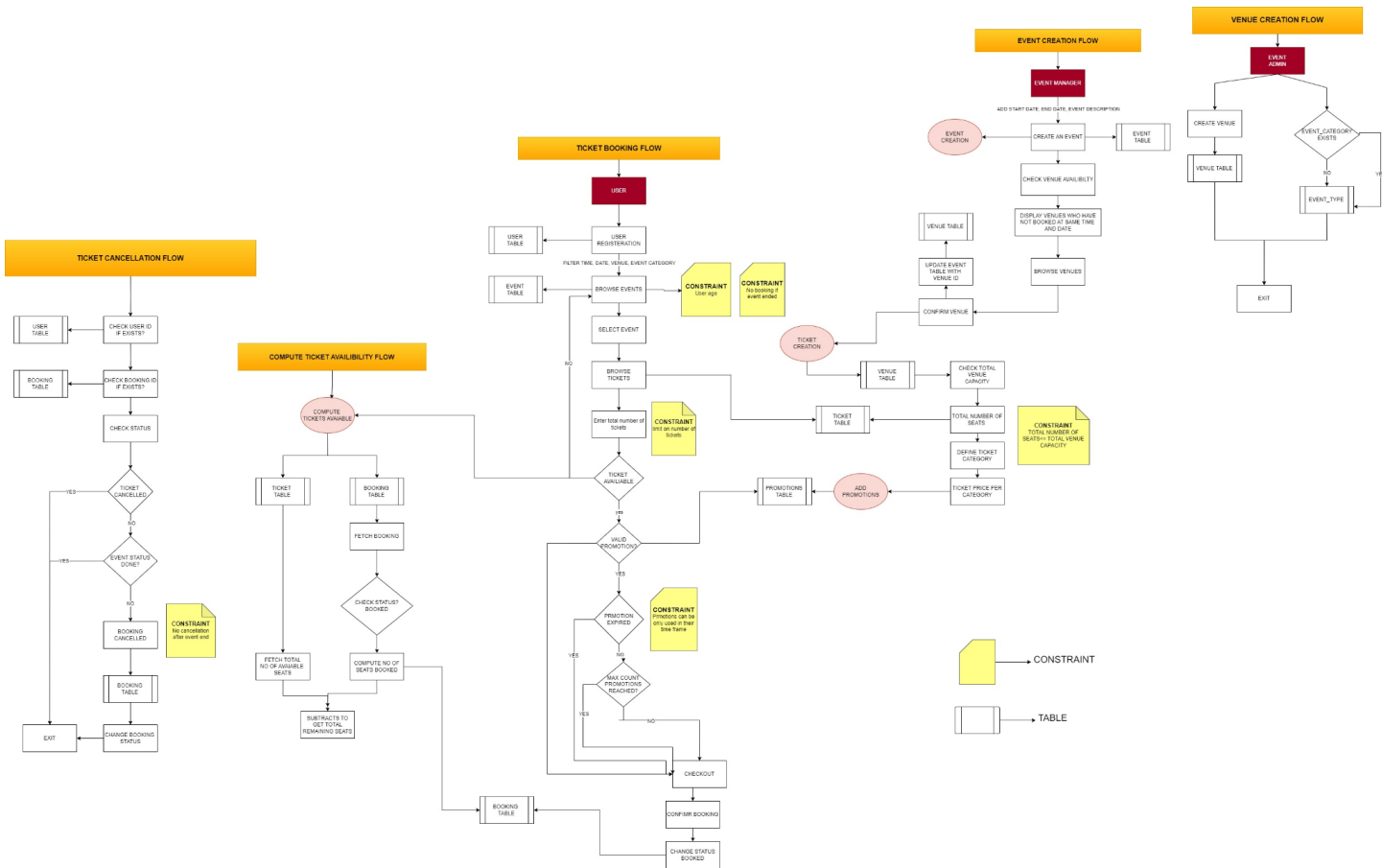
1. Begin by accessing the bookings table.
2. Sum all the bookings where the booking status is marked as "booked" for the specific ticket type.
3. Retrieve the total number of available seats for the event and seat category.
4. Subtract the sum of booked tickets from the total number of available seats.
5. The result represents the number of available seats or tickets for that particular ticket type.

TICKET CANCELLATION FLOW



1. Verify the existence of the user's ID and booking within the system.
2. Ensure that the booking is not already marked as canceled.
3. Verify that the event associated with the booking has not already started.
4. Restrict the ability to cancel a booking to within 30 minutes before the event's scheduled start time.
5. If the conditions are met and the cancellation is successful, update the booking status to "canceled" in the booking table.

COMPLETE DATA FLOW DIAGRAM



SECURITY CONSTRAINTS ARE:

In total, a user with database can access can play one of the below three roles , i.e:

- 1.Event app admin
- 2.Event Manager
- 3.User

TABLES INVOLVED IN THE PROCESS ARE:

- 1.VENUE
- 2.EVENT
- 3.EVENT_TYPE
- 4.USER
- 5.TICKET
- 6.BOOKING
- 7.PROMOTIONS
- 8.REVIEWS

USER ACCESIBILITY

ROLE - EVENT APP ADMIN

Role	Access Type	Comment
USER Table	Full access-Create, Read,Update, Delete	Can create, read, update and delete user
EVENT Table	Full access-Create, Read,Update, Delete	Can create ,read, update, event
BOOKING Table	Full access-Create, Read,Update, Delete	Can create booking, read, update and delete them
VENUE Table	Full access-Create, Read,Update, Delete	Can create ,read, update venue
TICKET Table	Full access-Create, Read,Update, Delete	Can create ,read, update ticket
EVENT_TYPE Table	Full access-Create, Read,Update, Delete	Can create ,read, update event_type
PROMOTIONS Table	Full access-Create, Read,Update, Delete	Can create ,read, update promotions
REVIEWS Table	Full access-Create,	Can create ,read, update

	Read,Update, Delete	reviews
--	---------------------	---------

ROLE - EVENT MANAGER

Role	Access Type	Columns	Comment
USER Table	Read	UserFirstName, UserLastName, UserEmail, UserAge,UserMobile UserGender	Check details of user who have booked the tickets
EVENT Table	Create, Read,Update	EventName, EventDescription,EventStartTime,EventEndTime	Creates, Reads and Updates the table
BOOKING Table	Read	BookingStatus, BookingTicketCount, BookingDate, BookingAmount	Check the bookings of user
VENUE Table	Read	VenueName,VenueAddress,VenueTotalCapacity,VenueEmail	Checks the venue availability for booking the venue
TICKET Table	Create, Read,Update	TicketCategoryName, TicketPrice,TicketSeats	Creates tickets for events and updates the price,category etc
EVENT_TYPE Table	Read	EventTypeName, EventTypeDescription, EventTypeAgeRestricted	Browse the event_type to select type for event
PROMOTIONS Table	Create, Read,Update,	PromotionCode, PromotionStartDate,PromotionEndDate, PromotionDiscount, PromotionMaxCount	Creates, checks and updates promotions on tickets
REVIEWS Table	Read	ReviewDescription	View all the reviews of users

ROLE - USER

Role	Access Type	Columns	Comment
USER Table	Read, Create	UserFirstName, UserLastName, UserEmail, UserAge,UserMobile UserGender	User can create and check their own data
EVENT Table	Read	EventName, EventDescription,Event StartTime,EventEndTime	Browse the events available
BOOKING Table	Read	BookingStatus, BookingTicketCount, BookingDate, BookingAmount	To view their bookings
VENUE Table	Read	VenueName,VenueAd dress, VenueEmail	To check the available venues for booking tickets
TICKET Table	Read	TicketCategoryName, TicketPrice	View the ticket status
EVENT_TYPE Table	Read	EventTypeName, EventTypeDescription, EventTypeAgeRestrict ed	Filter and browse all the event types to book event tickets
PROMOTIONS Table	Read	PromotionCode, PromotionDiscount	Check all the available promotions
REVIEWS Table	Read, Create	ReviewDescription	Create and read their views

Scripts:

Created users and granted permissions

These SQL statements are often used in database management to establish different user accounts with specific access privileges and authentication credentials for various individuals or applications interacting with the database.

```
CREATE USER Event_App_Admin IDENTIFIED BY AppOwner_123;  
CREATE USER Event_Manager IDENTIFIED BY Organizer_123;  
CREATE USER Attendee IDENTIFIED BY Event_User_123;
```

Worksheet Query Builder

```
CREATE USER Event_App_Admin IDENTIFIED BY AppOwner_123;  
CREATE USER Event_Manager IDENTIFIED BY Organizer_123;  
CREATE USER Attendee IDENTIFIED BY Event_User_123;
```

Script Output x



Task completed in 1.801 seconds

User EVENT_APP_ADMIN created.

User EVENT_MANAGER created.

User ATTENDEE created.

Permission to add data

```
GRANT CONNECT, RESOURCE TO Event_App_Admin;
```

- Grants connection and resource management privileges to the user or role named "Event_App_Admin."
- Enables database connection and extensive administrative control over schema objects.

```
GRANT CONNECT, RESOURCE TO Event_Manager;
```

- Similar to the first statement, it grants connection and resource management privileges to the "Event_Manager" user or role.

```
GRANT CONNECT TO Attendee;
```

- Grants only the connection privilege to the "Attendee" user or role.
- Allows the user to connect to the database but does not provide administrative control.

```
GRANT CONNECT,RESOURCE TO Event_App_Admin;
```

```
GRANT CONNECT,RESOURCE TO Event_Manager;
```

```
GRANT CONNECT TO Attendee;
```

```
GRANT CONNECT,RESOURCE TO Event_App_Admin;
```

```
GRANT CONNECT,RESOURCE TO Event_Manager;
```

```
GRANT CONNECT TO Attendee;
```

```
ALTER USER Event_App_Admin QUOTA 10m ON DATA;
```

```
ALTER USER Event_Manager QUOTA 10m ON DATA;
```

Script Output

Task completed in 0.117 seconds

Grant succeeded.

Grant succeeded.

Grant succeeded.


User EVENT_APP_ADMIN altered.

User EVENT_MANAGER altered.

These SQL statements are modifying two user accounts, "Event_App_Admin" and "Event_Manager," within a database. They are setting a storage quota of 10 megabytes (10m) for each of these users specifically on the "DATA" object within the database. This means that each user is restricted to using a maximum of 10MB of storage space within the "DATA" object. If they reach this limit, they may need to delete data to free up space before they can insert more data. These statements help manage and control the allocation of storage resources for these users within the database.

```
ALTER USER Event_App_Admin QUOTA 10m ON DATA;  
ALTER USER Event_Manager QUOTA 10m ON DATA;
```

```
ALTER USER Event_App_Admin QUOTA 10m ON DATA;  
ALTER USER Event_Manager QUOTA 10m ON DATA;
```

 Script Output x



Task completed in 0.117 seconds

Grant succeeded.

Grant succeeded.

Grant succeeded.

User EVENT_APP_ADMIN altered.

User EVENT_MANAGER altered.

