Project phase A (team 1)

Μέλη Ομάδας

CSD4549 Ευθυμίου Γεράσιμος

CSD4676 Μακρογιάννης Δημήτριος

CSD4559 Βιδάλης Δημήτριος

Εννοιολογική Μοντελοποιήση της βάσης

Εισαγωγή

Το παρόν project αφορά τον σχεδιασμό και την υλοποίηση μιας βάσης δεδομένων για τη διαχείριση εκδηλώσεων. Το σύστημα καλύπτει την αποθήκευση βασικών πληροφοριών για εκδηλώσεις, πελάτες, εισιτήρια και κρατήσεις, με σκοπό την παροχή ολοκληρωμένων λειτουργιών όπως εγγραφή νέων πελατών, δημιουργία εκδηλώσεων, αναζήτηση διαθέσιμων θέσεων, κράτηση εισιτηρίων και διαχείριση ακυρώσεων. Επιπλέον, περιλαμβάνονται διαδικασίες που υποστηρίζουν την ανάλυση δεδομένων, όπως υπολογισμός εσόδων, δημοφιλέστερες εκδηλώσεις και προβολή κρατήσεων. Η ανάπτυξη περιλαμβάνει εννοιολογική και σχεσιακή μοντελοποίηση, καθώς και κανονικοποίηση σε τρίτη κανονική μορφή, ώστε να διασφαλιστεί η ακεραιότητα και η αποδοτική διαχείριση των δεδομένων.

Διάγραμμα Οντοτήτων-Σχέσεων (E-R)

Το ER διάγραμμα απεικονίζει τις οντότητες, τις σχέσεις μεταξύ τους και τα χαρακτηριστικά τους.

Οι βασικές οντότητες είναι:

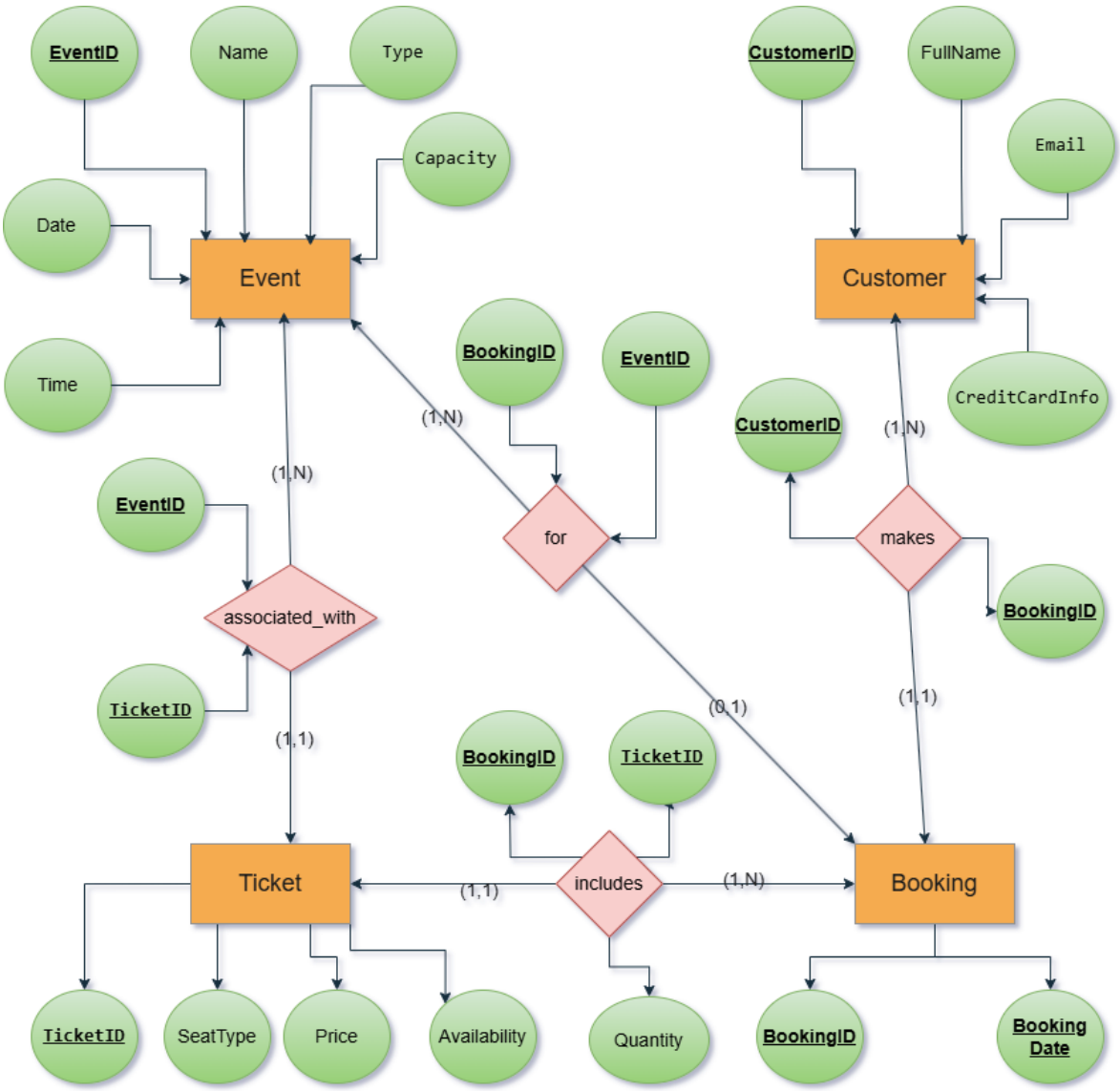
Event (Εκδήλωση)

Customer (Πελάτης)

Ticket (Εισιτήριο)

Booking (Κράτηση)

Διάγραμμα



Γνωρίσματα Οντοτήτων, Σχέσεων

Οντότητες:

Customer

CustomerID (Primary Key, INT),

FullName (VARCHAR),

Email (VARCHAR),

CreditCardInfo (VARCHAR)

Event

EventID (Primary Key, INT),

Name (VARCHAR),

Date (DATE),

Time (TIME),

Type (VARCHAR),

Capacity (INT)

Ticket

TicketID (Primary Key, INT),

SeatType (VARCHAR),

Price (DECIMAL),

Availability (INT)

Booking

BookingID (Primary Key, INT) ,

CustomerID (Foreign Key, INT),

EventID (Foreign Key, INT),

BookingDate (DATE)

Σχέσεις:

1) makes (Customer → Booking)

CustomerID → BookingID

Πληθικότητα: (1, N)

Ένας πελάτης μπορεί να κάνει πολλές κρατήσεις.

Πληθικότητα: (1, 1)

Κάθε κράτηση ανήκει σε έναν πελάτη.

2) for (Booking → Event)

EventID → BookingID

Πληθικότητα: (1,N)

Μια εκδήλωση μπορεί να έχει πολλές κρατήσεις.

Πληθικότητα: (0, 1)

Κάθε κράτηση αφορά μία μόνο εκδήλωση.

3) associated_with (Ticket → Event)

EventID → TicketID

Πληθικότητα: (1,N)

Κάθε εκδήλωση έχει πολλούς τύπους εισιτηρίων (π.χ., VIP, γενική είσοδος).

Πληθικότητα: (1, 1)

Κάθε τύπος εισιτηρίου σχετίζεται με μία συγκεκριμένη εκδήλωση.

4) includes (Booking → Ticket)

Για να συνδέσουμε τις κρατήσεις με τα εισιτήρια που αγοράστηκαν, δημιουργούμε τη σχέση includes, η οποία περιλαμβάνει τα παρακάτω:

Γνωρίσματα

BookingID (Foreign Key, από Booking)

TicketID (Foreign Key, από Ticket)

Quantity (INT, πλήθος εισιτηρίων που περιλαμβάνονται στην κράτηση)

Συνδέει κάθε κράτηση με τα εισιτήρια που έχουν αγοραστεί. Το γνώρισμα Quantity καταγράφει πόσα εισιτήρια στην κράτηση.

Πληθικότητα: (1,N)

Μια κράτηση μπορεί να περιλαμβάνει πολλούς τύπους εισιτηρίων (π.χ., VIP και γενική είσοδος).

Πληθικότητα: (1, 1)

Ένα εισιτήριο μπορεί να εμφανίζεται σε μία κράτηση.

Επεξηγήσεις για Μη Προφανή Γνωρίσματα και Σχέσεις

CreditCardInfo: Χρησιμοποιείται για την ολοκλήρωση πληρωμών και αποθηκεύεται σε ασφαλή μορφή.

Availability (Ticket): Δείχνει τον αριθμό διαθέσιμων εισιτηρίων.

AmountPaid (Booking): Το συνολικό ποσό που πληρώθηκε για την κράτηση.

NumTIckets(Booking): Το συνολικό πλήθος των εισιτηρίων που έχουν κρατηθεί. Αποθηκεύεται στο Booking

Μετάφραση στο Σχεσιακό Μοντέλο

Πίνακες:

Event [EventID, Name, Date, Time, Type, Capacity]

Customers [CustomerID, FullName, Email, CreditCard]

Tickets [TicketID, EventID, Type, Price, Availability]

Booking [BookingID, CustomerID, EventID, NumTickets, BookingDate, Payment]

Makes [BookingId, CustomerId]

Includes [BookingId, TicketID, Quantity]

Associated_with [EventID, TicketID]

Εντολές γλώσσας ορισμού για τις σχέσεις (DDL)

```
CREATE TABLE Event (
    EventID INT PRIMARY KEY,
    Name VARCHAR(100),
    Type VARCHAR(50),
    Capacity INT,
    Date DATE,
    Time TIME
);

CREATE TABLE Customer (
    CustomerID INT PRIMARY KEY,
    FullName VARCHAR(100),
    Email VARCHAR(100),
    CreditCardInfo VARCHAR(50)
```

```
);

CREATE TABLE Ticket (
    TicketID INT PRIMARY KEY,
    EventID INT,
    SeatType VARCHAR(50),
    Price DECIMAL(10, 2),
    Availability BOOLEAN,
    FOREIGN KEY (EventID) REFERENCES Event(EventID)
);




CREATE TABLE Booking (
    BookingID INT PRIMARY KEY,
    BookingDate DATE,
    Payment DECIMAL(10, 2),
    FOREIGN KEY (CustomerID) REFERENCES Customer(CustomerID),
    FOREIGN KEY (EventID) REFERENCES Event(EventID)
);

CREATE TABLE BookingTicket (
    BookingID INT,
    TicketID INT,
    Quantity INT,
    PRIMARY KEY (BookingID, TicketID),
    FOREIGN KEY (BookingID) REFERENCES Booking(BookingID),
    FOREIGN KEY (TicketID) REFERENCES Ticket(TicketID)
);
```

Οι πίνακες του συστήματος είναι οι εξής:

- **Event (Εκδήλωση)**: Περιέχει λεπτομέρειες για τις εκδηλώσεις, όπως το όνομα, τον τύπο (π.χ. συναυλία, θεατρική παράσταση), τη χωρητικότητα, την ημερομηνία και την ώρα.

- **Customer (Πελάτης)**: Περιέχει τα στοιχεία των πελατών, όπως το όνομα, το email και τις πληροφορίες πιστωτικής κάρτας.
- **Ticket (Εισιτήριο)**: Περιέχει τα δεδομένα για τα εισιτήρια, όπως τον τύπο θέσης (π.χ. VIP ή γενική είσοδος), την τιμή, τη διαθεσιμότητα και την ποσότητα για κάθε εκδήλωση.
- **Booking (Κράτηση)**: Περιέχει τις πληροφορίες για τις κρατήσεις, όπως την ημερομηνία της κράτησης, το ποσό πληρωμής και την εκδήλωση για την οποία έγινε η κράτηση.
- **BookingTicket (Κράτηση-Εισιτήριο)**: Είναι ένας πίνακας σύνδεσης που συνδέει τις κρατήσεις με τα εισιτήρια. Κρατάει την ποσότητα των εισιτηρίων για κάθε κράτηση.

Σχέσεις:

- Κάθε **Εκδήλωση** μπορεί να έχει πολλά **Εισιτήρια**.
- Κάθε **Πελάτης** μπορεί να κάνει πολλές **Κρατήσεις**.
- Κάθε **Κράτηση** μπορεί να περιλαμβάνει πολλά **Εισιτήρια**, και ένα **Εισιτήριο** μπορεί να ανήκει σε πολλές **Κρατήσεις** (αυτό γίνεται μέσω του πίνακα **BookingTicket**).

Για το συγκεκριμένο σύστημα που περιγράφουμε, οι περιορισμοί ακεραιότητας και οι συναρτησιακές εξαρτήσεις είναι κρίσιμες για την αποφυγή ανακολουθιών στα δεδομένα και τη διασφάλιση της σωστής λειτουργίας της βάσης δεδομένων. Παρακάτω ακολουθούν οι αντίστοιχοι περιορισμοί και οι εξαρτήσεις.

Περιορισμοί Ακεραιότητας

Περιορισμός Πρωτεύοντος Κλειδιού (Primary Key Constraint):

Event: Το EventID είναι το πρωτεύον κλειδί για τον πίνακα Event, που διασφαλίζει ότι κάθε εκδήλωση έχει μοναδικό αναγνωριστικό.

Customer: Το CustomerID είναι το πρωτεύον κλειδί για τον πίνακα Customer, διασφαλίζοντας ότι κάθε πελάτης είναι μοναδικός.

Ticket: Το TicketID είναι το πρωτεύον κλειδί για τον πίνακα Ticket, διασφαλίζοντας ότι κάθε εισιτήριο είναι μοναδικό.

Booking: Το BookingID είναι το πρωτεύον κλειδί για τον πίνακα Booking, διασφαλίζοντας ότι κάθε κράτηση έχει μοναδικό αναγνωριστικό.

BookingTicket: Το ζεύγος BookingID και TicketID είναι το σύνθετο πρωτεύον κλειδί για τον πίνακα BookingTicket, εξασφαλίζοντας ότι κάθε κράτηση για εισιτήριο είναι μοναδική.

Περιορισμός Ξένου Κλειδιού (Foreign Key Constraint):

Event: Το EventID στον πίνακα Ticket αναφέρεται στο πρωτεύον κλειδί του πίνακα Event, διασφαλίζοντας ότι κάθε εισιτήριο σχετίζεται με μια υπάρχουσα εκδήλωση.

Customer: Το CustomerID στον πίνακα Booking αναφέρεται στο πρωτεύον κλειδί του πίνακα Customer, διασφαλίζοντας ότι κάθε κράτηση ανήκει σε έναν πελάτη.

Event: Το EventID στον πίνακα Booking αναφέρεται στο πρωτεύον κλειδί του πίνακα Event, διασφαλίζοντας ότι κάθε κράτηση ανήκει σε μια εκδήλωση.

Booking: Το BookingID στον πίνακα BookingTicket αναφέρεται στο πρωτεύον κλειδί του πίνακα Booking, διασφαλίζοντας ότι κάθε κράτηση εισιτηρίου ανήκει σε μια υπάρχουσα κράτηση.

Ticket: Το TicketID στον πίνακα BookingTicket αναφέρεται στο πρωτεύον κλειδί του πίνακα Ticket, διασφαλίζοντας ότι κάθε κράτηση εισιτηρίου ανήκει σε ένα υπάρχον εισιτήριο.

Περιορισμός Μοναδικότητας (Unique Constraint):

Email: Στον πίνακα Customer, το πεδίο Email πρέπει να είναι μοναδικό για κάθε πελάτη. Δεν μπορεί να υπάρχουν δύο πελάτες με το ίδιο email.

Περιορισμός Μη Κενής Τιμής (NOT NULL Constraint):

Τα πεδία που είναι απαραίτητα για τη δημιουργία μιας εγγραφής (όπως EventID, CustomerID, TicketID, κ.λπ.) δεν πρέπει να είναι κενά.

Για παράδειγμα, το πεδίο Name στον πίνακα Event ή το πεδίο FullName στον πίνακα Customer δεν πρέπει να είναι κενά.

Περιορισμός Ελάχιστης και Μέγιστης Τιμής (Check Constraint):

Στον πίνακα Ticket, το πεδίο Price μπορεί να έχει περιορισμό ώστε η τιμή του εισιτηρίου να είναι μεγαλύτερη από το 0 (π.χ., CHECK (Price > 0)).

Στον πίνακα Ticket, το πεδίο Availability μπορεί να είναι boolean, που περιορίζει τις τιμές του σε TRUE ή FALSE.

Συναρτησιακές Εξαρτήσεις

Event:

Το EventID εξαρτάται πλήρως από το Event (πρωτεύον κλειδί).

Τα πεδία Name, Type, Capacity, Date, και Time εξαρτώνται πλήρως από το EventID. Δεν υπάρχει κάποια εξάρτηση μεταξύ τους.

Customer:

Το CustomerID εξαρτάται πλήρως από τον πελάτη.

Τα πεδία FullName, Email, και CreditCardInfo εξαρτώνται πλήρως από το CustomerID.

Ticket:

Το TicketID εξαρτάται πλήρως από το εισιτήριο.

Τα πεδία SeatType, Price, Availability, και Quantity εξαρτώνται πλήρως από το TicketID.

Booking:

Το BookingID εξαρτάται πλήρως από την κράτηση.

Τα πεδία CustomerID, EventID, BookingDate, και Payment εξαρτώνται πλήρως από το BookingID.

BookingTicket:

Το ζεύγος BookingID και TicketID εξαρτάται πλήρως από την εγγραφή στον πίνακα. Το πεδίο Quantity εξαρτάται από αυτό το ζεύγος.

Οι περιορισμοί ακεραιότητας αφορούν τους κανόνες που πρέπει να τηρούνται για να εξασφαλιστεί η ακεραιότητα των δεδομένων στη βάση, ενώ τα κλειδιά αφορούν την μοναδικότητα και τη σύνδεση των δεδομένων μεταξύ των πινάκων. Ας δούμε λοιπόν τους **περιορισμούς ακεραιότητας** για τη συγκεκριμένη βάση δεδομένων, σύμφωνα με τις συναρτησιακές εξαρτήσεις και το μοντέλο.

Περιορισμοί Ακεραιότητας

Ακεραιότητα Οντοτήτων (Entity Integrity):

Περιορισμός: Κάθε πίνακας πρέπει να έχει πρωτεύον κλειδί, και καμία εγγραφή δεν μπορεί να έχει τιμή NULL στο πρωτεύον κλειδί.

Εφαρμογή:

Στους πίνακες Event, Customer, Ticket, Booking και BookingTicket, το πρωτεύον κλειδί δεν μπορεί να είναι NULL.

Επομένως, πεδία όπως το EventID, CustomerID, TicketID, BookingID, και το σύνθετο κλειδί (BookingID, TicketID) δεν επιτρέπεται να είναι NULL.

Ακεραιότητα Σχέσεων (Referential Integrity):

Περιορισμός: Τα εξωτερικά κλειδιά (foreign keys) πρέπει να αντιστοιχούν σε εγγραφές που υπάρχουν στους αντίστοιχους πίνακες.

Εφαρμογή:

Στον πίνακα Booking, το πεδίο CustomerID πρέπει να αντιστοιχεί σε μια υπάρχουσα εγγραφή στον πίνακα Customer.

Στον πίνακα Booking, το πεδίο EventID πρέπει να αντιστοιχεί σε μια υπάρχουσα εγγραφή στον πίνακα Event.

Στον πίνακα BookingTicket, το πεδίο TicketID πρέπει να αντιστοιχεί σε μια υπάρχουσα εγγραφή στον πίνακα Ticket.

Οποιαδήποτε εγγραφή σε αυτούς τους πίνακες που παραβιάζει αυτήν την εξάρτηση (π.χ. εισαγωγή μιας κράτησης με CustomerID που δεν υπάρχει στον πίνακα Customer) θα απορρίπτεται.

Ακεραιότητα Περιορισμών (Domain Integrity):

Περιορισμός: Τα πεδία πρέπει να περιέχουν δεδομένα που ανήκουν σε συγκεκριμένο πεδίο τιμών (domain).

Εφαρμογή:

Τα πεδία όπως το Email στον πίνακα Customer πρέπει να ακολουθούν το σωστό μορφότυπο ενός email (π.χ. user@example.com).

Τα πεδία όπως το Price στον πίνακα Ticket πρέπει να περιέχουν θετικές τιμές (π.χ. Price > 0).

Το πεδίο Quantity στον πίνακα BookingTicket πρέπει να περιέχει μη αρνητικές ακέραιες τιμές (π.χ. Quantity >= 0).

Το πεδίο Availability στον πίνακα Ticket πρέπει να είναι είτε Available είτε Unavailable.

Ακεραιότητα Αξιοπιστίας (Consistency Integrity):

Περιορισμός: Οι κανόνες της επιχείρησης πρέπει να τηρούνται για να διασφαλιστεί ότι τα δεδομένα παραμένουν συνεπή.

Εφαρμογή:

Ο αριθμός των εισιτηρίων που ζητούνται στην κράτηση (στο πεδίο Quantity στον πίνακα BookingTicket) δεν μπορεί να υπερβαίνει τον αριθμό των διαθέσιμων εισιτηρίων (στο πεδίο Availability στον πίνακα Ticket).

Το ποσό πληρωμής για μια κράτηση πρέπει να είναι τουλάχιστον το συνολικό ποσό για τα εισιτήρια που έχουν αγοραστεί.

Εάν μια εκδήλωση ακυρωθεί, όλες οι σχετικές κρατήσεις πρέπει να ακυρωθούν και οι πληρωμές να επιστραφούν, διατηρώντας τη συνεπή κατάσταση των δεδομένων.

Ακεραιότητα Ενημέρωσης (Update Integrity):

Περιορισμός: Οι αλλαγές σε δεδομένα δεν πρέπει να προκαλούν ανεπιθύμητες συνέπειες ή διαταραχές στις σχέσεις.

Εφαρμογή:

Εάν κάποιος πελάτης αλλάξει τα στοιχεία του (π.χ. Email ή CreditCardInfo στον πίνακα Customer), οι σχετικές εγγραφές στις κρατήσεις ή στις πληρωμές θα πρέπει να ενημερώνονται αυτόματα χωρίς να χάνονται δεδομένα.

Αν μια εκδήλωση αλλάξει (π.χ. ημερομηνία ή ώρα), οι κρατήσεις για αυτή την εκδήλωση πρέπει να ενημερωθούν για να αντικατοπτρίζουν τη νέα ημερομηνία/ώρα.

Ακεραιότητα Διαγραφής (Deletion Integrity):

Περιορισμός: Η διαγραφή μιας εγγραφής δεν πρέπει να οδηγεί σε «ορφανές» εγγραφές που δεν συνδέονται με άλλες.

Εφαρμογή:

Αν διαγραφεί μια εκδήλωση από τον πίνακα Event, τότε όλες οι σχετικές κρατήσεις στον πίνακα Booking και τα εισιτήρια στον πίνακα BookingTicket πρέπει να διαγραφούν ή να ενημερωθούν με τρόπο ώστε να μην υπάρχουν «ορφανές» εγγραφές.

Αν διαγραφεί ένας πελάτης από τον πίνακα Customer, όλες οι κρατήσεις που σχετίζονται με αυτόν τον πελάτη στον πίνακα Booking θα πρέπει να διαγραφούν ή να ενημερωθούν αναλόγως.

Ακολουθεί μια περιγραφή των πιθανών ερωτήσεων που μπορούμε να θέσουμε στη βάση δεδομένων, μαζί με παραδείγματα εντολών SQL που τις υλοποιούν:

1. Κατάσταση διαθέσιμων και κρατημένων θέσεων ανά εκδήλωση

Θέλουμε να δούμε πόσες θέσεις είναι διαθέσιμες και κρατημένες για κάθε εκδήλωση.

```
SELECT
    e.Name AS EventName,
    t.SeatType,
    t.Availability AS TotalSeats,
    SUM(bt.Quantity) AS ReservedSeats,
    (t.Availability - SUM(bt.Quantity)) AS AvailableSeats
FROM
    Event e
JOIN
    Ticket t ON e.EventID = t.EventID
LEFT JOIN
    BookingTicket bt ON t.TicketID = bt.TicketID
GROUP BY
```

```
    e.Name, t.SeatType, t.Availability;
```

2. Έσοδα από πωλήσεις ανά εκδήλωση

Υπολογίζουμε τα συνολικά έσοδα από πωλήσεις εισιτηρίων για κάθε εκδήλωση.

```
SELECT
    e.Name AS EventName,
    SUM(bt.Quantity * t.Price) AS TotalRevenue
FROM
    BookingTicket bt
JOIN
    Ticket t ON bt.TicketID = t.TicketID
JOIN
    Event e ON t.EventID = e.EventID
GROUP BY
    e.Name;
```

3. Δημοφιλέστερη εκδήλωση βάσει κρατήσεων

Βρίσκουμε την εκδήλωση με τις περισσότερες κρατήσεις.

```
SELECT
    e.Name AS EventName,
    COUNT(b.BookingID) AS TotalBookings
FROM
```

```
    Booking b
JOIN
    Event e ON b.EventID = e.EventID
GROUP BY
    e.Name
ORDER BY
    TotalBookings DESC
LIMIT 1;
```

4. Εκδήλωση με τα περισσότερα έσοδα σε ένα χρονικό εύρος

Βρίσκουμε την εκδήλωση που έχει αποφέρει τα περισσότερα έσοδα σε ένα συγκεκριμένο χρονικό διάστημα.

```
SELECT
    e.Name AS EventName,
    SUM(bt.Quantity * t.Price) AS TotalRevenue
FROM
    BookingTicket bt
JOIN
    Ticket t ON bt.TicketID = t.TicketID
JOIN
    Event e ON t.EventID = e.EventID
JOIN
    Booking b ON bt.BookingID = b.BookingID
WHERE
    b.BookingDate BETWEEN '2024-01-01' AND '2024-12-31'
GROUP BY
    e.Name
ORDER BY
    TotalRevenue DESC
LIMIT 1;
```

5. Προβολή κρατήσεων ανά χρονική περίοδο

Βρίσκουμε όλες τις κρατήσεις που έγιναν σε συγκεκριμένο χρονικό διάστημα.

```
SELECT
    b.BookingID,
    c.FullName AS CustomerName,
    e.Name AS EventName,
    b.BookingDate,
    SUM(bt.Quantity * t.Price) AS TotalPayment
FROM
    Booking b
JOIN
    Customer c ON b.CustomerID = c.CustomerID
JOIN
    Event e ON b.EventID = e.EventID
JOIN
    BookingTicket bt ON b.BookingID = bt.BookingID
JOIN
    Ticket t ON bt.TicketID = t.TicketID
WHERE
    b.BookingDate BETWEEN '2024-01-01' AND '2024-12-31'
GROUP BY
    b.BookingID, c.FullName, e.Name, b.BookingDate;
```

```
SELECT
    b.BookingID,
    c.FullName AS CustomerName,
    e.Name AS EventName,
    b.BookingDate,
    SUM(bt.Quantity * t.Price) AS TotalPayment
FROM
    Booking b
JOIN
    Customer c ON b.CustomerID = c.CustomerID
JOIN
    Event e ON b.EventID = e.EventID
JOIN
    BookingTicket bt ON b.BookingID = bt.BookingID
JOIN
    Ticket t ON bt.TicketID = t.TicketID
WHERE
    b.BookingDate BETWEEN '2024-01-01' AND '2024-12-31'
GROUP BY
    b.BookingID, c.FullName, e.Name, b.BookingDate;
```

6. Συνολικά έσοδα από VIP ή γενικά εισιτήρια ανά εκδήλωση

Υπολογίζουμε τα έσοδα από συγκεκριμένο τύπο εισιτηρίων (π.χ. VIP) για κάθε εκδήλωση.

```
SELECT
    e.Name AS EventName,
    t.SeatType,
    SUM(bt.Quantity * t.Price) AS Revenue
FROM
    BookingTicket bt
JOIN
    Ticket t ON bt.TicketID = t.TicketID
JOIN
    Event e ON t.EventID = e.EventID
WHERE
    t.SeatType = 'VIP'
GROUP BY
    e.Name, t.SeatType;
```

7. Πελάτες που έχουν κάνει κρατήσεις για συγκεκριμένη εκδήλωση

Βρίσκουμε όλους τους πελάτες που έχουν κάνει κράτηση για μια συγκεκριμένη εκδήλωση.

```
SELECT
    c.FullName AS CustomerName,
    c.Email,
    b.BookingID,
    SUM(bt.Quantity * t.Price) AS TotalPayment
FROM
    Customer c
JOIN
    Booking b ON c.CustomerID = b.CustomerID
JOIN
    BookingTicket bt ON b.BookingID = bt.BookingID
JOIN
    Ticket t ON bt.TicketID = t.TicketID
```

```
JOIN
    Event e ON b.EventID = e.EventID
WHERE
    e.Name = 'Concert XYZ'
GROUP BY
    c.FullName, c.Email, b.BookingID;
```

8. Συνολικά έσοδα για όλες τις εκδηλώσεις

Βρίσκουμε το συνολικό ποσό που έχει συγκεντρωθεί από όλες τις πωλήσεις εισιτηρίων.

```
SELECT
    SUM(bt.Quantity * t.Price) AS TotalRevenue
FROM
    BookingTicket bt
JOIN
    Ticket t ON bt.TicketID = t.TicketID;
```

Με αυτές τις ερωτήσεις, μπορούμε να ανακτήσουμε κρίσιμα δεδομένα για την παρακολούθηση της απόδοσης της επιχείρησης και τη διαχείριση των εκδηλώσεων!

Όψη για Έσοδα ανά Εκδήλωση

Περιγραφή: Η όψη "EventRevenueView" εμφανίζει τα συνολικά έσοδα από τις κρατήσεις εισιτηρίων για κάθε εκδήλωση. Χρησιμοποιείται για την ταχύτερη εξαγωγή αναφορών σχετικά με τα έσοδα.

Χρήση: Απλοποιεί την εξαγωγή οικονομικών δεδομένων, χωρίς να χρειάζεται να γράφονται συνεχώς πολύπλοκα ερωτήματα.

Εντολή SQL:

```
CREATE VIEW EventRevenueView AS
SELECT
```

```
    e.Name AS EventName,
    SUM(bt.Quantity * t.Price) AS TotalRevenue
FROM
    BookingTicket bt
JOIN
    Ticket t ON bt.TicketID = t.TicketID
JOIN
    Event e ON t.EventID = e.EventID
GROUP BY
    e.Name;
```

Όψη για Πελάτες με Πολλαπλές Κρατήσεις

Περιγραφή: Η όψη "FrequentCustomersView" εμφανίζει τους πελάτες που έχουν κάνει περισσότερες από μία κρατήσεις.

Χρήση: Βοηθά στη δημιουργία αναφορών για τους πιο συχνούς πελάτες, ώστε να γίνουν προωθητικές ενέργειες.

Εντολή SQL:

```
CREATE VIEW FrequentCustomersView AS
SELECT
    c.CustomerID,
    c.FullName,
    COUNT(b.BookingID) AS TotalBookings
FROM
    Customer c
JOIN
    Booking b ON c.CustomerID = b.CustomerID
GROUP BY
    c.CustomerID, c.FullName
HAVING
    COUNT(b.BookingID) > 1;
```

3: Όψη για Διαθέσιμες Θέσεις Ανά Εκδήλωση

Περιγραφή: Η όψη "AvailableSeatsView" δείχνει τις διαθέσιμες θέσεις ανά εκδήλωση και τύπο εισιτηρίου.

Χρήση: Διευκολύνει την άμεση προβολή της διαθεσιμότητας χωρίς να γίνονται σύνθετες ενώσεις στους πίνακες.

Εντολή SQL:

```
CREATE VIEW AvailableSeatsView AS
SELECT
    e.Name AS EventName,
    t.SeatType,
    (t.Availability - COALESCE(SUM(bt.Quantity), 0)) AS AvailableSeats
FROM
    Ticket t
JOIN
    Event e ON t.EventID = e.EventID
LEFT JOIN
    BookingTicket bt ON t.TicketID = bt.TicketID
GROUP BY  e.Name, t.SeatType, t.Availability;
```

Αυτές είναι μερικες όψεις οι οποίες μπορούν να χρησιμοποιηθούν στην δικιά μας βάση δεδομένων αλλά δεν είναι σίγουρο πως θα χρησιμοποιηθούν

Η περιγραφή σε ψευδοκώδικα των διαδικασιών

1. Εγγραφή Νέου Πελάτη

```
Διαδικασία Εγγραφή_Νέου_Πελάτη
  Εμφάνισε "Εισαγάγετε τα στοιχεία του πελάτη"
  Διάβασε FullName, Email, CreditCardInfo
  Αν FullName ΚΕΝΟ Ή Email ΚΕΝΟ Ή CreditCardInfo ΚΕΝΟ Τότε
    Εμφάνισε "Η εγγραφή απέτυχε. Όλα τα πεδία είναι υποχρεωτικά."
    Επιστροφή
  Τέλος Αν
  Αποθήκευσε τα στοιχεία του πελάτη στον πίνακα Customer
  Εμφάνισε "Η εγγραφή ολοκληρώθηκε επιτυχώς"
```

Τέλος Διαδικασίας

2. Δημιουργία Νέας Εκδήλωσης

Διαδικασία Δημιουργία_Νέας_Εκδήλωσης
  Εμφάνισε "Εισαγάγετε τα στοιχεία της εκδήλωσης"
  Διάβασε EventName, EventDate, EventTime, EventType, Capacity
  Αν EventName ΚΕΝΟ Ή EventDate ΚΕΝΟ Ή Capacity <= 0 Τότε
    Εμφάνισε "Η δημιουργία της εκδήλωσης απέτυχε. Ελέγξτε τα στοιχεία."
    Επιστροφή
  Τέλος Αν
  Αποθήκευσε την εκδήλωση στον πίνακα Event
  Εμφάνισε "Η εκδήλωση δημιουργήθηκε επιτυχώς"
Τέλος Διαδικασίας

3. Αναζήτηση Διαθέσιμων Θέσεων

Διαδικασία Αναζήτηση_Διαθέσιμων_Θέσεων
  Εμφάνισε "Εισαγάγετε το όνομα της εκδήλωσης"
  Διάβασε EventName
  Αν EventName δεν υπάρχει στον πίνακα Event Τότε
    Εμφάνισε "Η εκδήλωση δεν βρέθηκε"
    Επιστροφή
  Τέλος Αν
  Αναζήτησε στον πίνακα Ticket τις διαθέσιμες θέσεις για την εκδήλωση EventName
  Εμφάνισε τις διαθέσιμες θέσεις και τους τύπους εισιτηρίων
Τέλος Διαδικασίας

4. Κράτηση Εισιτηρίων

Διαδικασία Κράτηση_Εισιτηρίων
  Εμφάνισε "Εισαγάγετε το EventName, τον αριθμό εισιτηρίων και τον τύπο θέσης"
  Διάβασε EventName, Quantity, SeatType
  Αν EventName δεν υπάρχει στον πίνακα Event Τότε
    Εμφάνισε "Η εκδήλωση δεν βρέθηκε"

Επιστροφή
Τέλος Αν
Αναζήτησε τη διαθεσιμότητα εισιτηρίων στον πίνακα Ticket για τον συγκεκριμένο τύπο θέσης
Αν Quantity > Διαθέσιμα_Εισιτήρια Τότε
Εμφάνισε "Ανεπαρκής διαθεσιμότητα εισιτηρίων"
Επιστροφή
Τέλος Αν
Μείωσε τα διαθέσιμα εισιτήρια στον πίνακα Ticket
Καταχώρισε την κράτηση στον πίνακα Booking
Εμφάνισε "Η κράτηση ολοκληρώθηκε επιτυχώς"
Τέλος Διαδικασίας

5. Ακύρωση Κράτησης

Διαδικασία Ακύρωση_Κράτησης
Εμφάνισε "Εισαγάγετε το BookingID για ακύρωση"
Διάβασε BookingID
Αν BookingID δεν υπάρχει στον πίνακα Booking Τότε
Εμφάνισε "Η κράτηση δεν βρέθηκε"
Επιστροφή
Τέλος Αν
Επίστρεψε τα χρηματα στον Customer
Ενημέρωσε τα διαθέσιμα εισιτήρια στον πίνακα Ticket
Διαγραφή της εγγραφής από τον πίνακα Booking
Εμφάνισε "Η κράτηση ακυρώθηκε επιτυχώς"
Τέλος Διαδικασίας

6. Ακύρωση Εκδήλωσης

Διαδικασία Ακύρωση_Εκδήλωσης
Εμφάνισε "Εισαγάγετε το EventID για ακύρωση"
Διάβασε EventID
Αν EventID δεν υπάρχει στον πίνακα Event Τότε
Εμφάνισε "Η εκδήλωση δεν βρέθηκε"

Επιστροφή

Τέλος Αν

Διαγραφή όλων των σχετικών κρατήσεων από τον πίνακα Booking

Ενημέρωσε τα διαθέσιμα εισιτήρια στον πίνακα Ticket

Διαγραφή της εκδήλωσης από τον πίνακα Event

Εμφάνισε "Η εκδήλωση ακυρώθηκε επιτυχώς"

Τέλος Διαδικασίας

Αυτή η αναφορά αφορά τη Φάση 1 του έργου, όπου παρουσιάζονται η εννοιολογική μοντελοποίηση και οι βασικές προδιαγραφές του συστήματος. Το πρότζεκτ μπορεί να υλοποιηθεί με διάφορους τρόπους, ανάλογα με τις ανάγκες και τις απαιτήσεις που θα προκύψουν. Υπάρχει μεγάλη πιθανότητα να αλλάξουν αρκετά πράγματα κατά τη διάρκεια της υλοποίησης, είτε λόγω νέων αναγκών είτε λόγω προσαρμογών που θα γίνουν για τη βελτίωση του τελικού αποτελέσματος.