

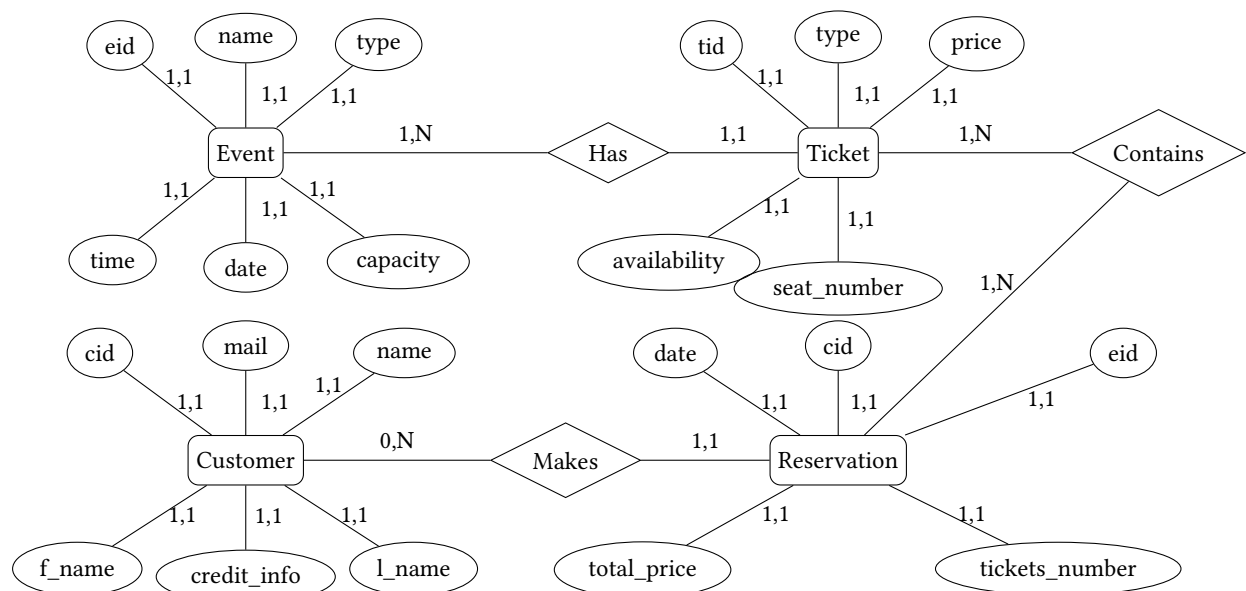
Project HY-360

Άγγελος-Τίτος Δήμογλης csd5078
Κωνσταντίνος Κουναλάκης csd5058
Δρακάκης Ραφαήλ csd5310

1η φάση

Η πρώτη φάση αφορά την δημιουργία ενός πλήρους εννοιολογικού μοντέλου.

E-R Διάγραμμα



Στο διάγραμμα φαίνονται τα γνωρίσματα όλων των οντοτήτων και σχέσεων και τα πρωτεύοντα κλειδιά

Σχετικά με τα γνωρίσματα και τις σχέσεις έχουμε:

Μια σχέση contains ανάμεσα στο ticket και το reservation, μια σχέση Makes ανάμεσα στον customer και το reservation. και μια σχέση Has ανάμεσα στο ticket και το Event.

Οι περιορισμοί για τις πληθικότητες φαίνονται στο σχήμα

Μετάφραση στο σχεσιακό μοντέλο

Παρακάτω φαίνονται οι πίνακες για το σχεσιακό μοντέλο

Customer				
cid	mail	credit_info	f_name	l_name

Event					
eid	name	type	time	date	capacity

Ticket				
tid	type	price	availability	seat_number

Reservation					
rid	eid	cid	date	total_price	tickets_number

Contains	
eid	tid

Makes	
cid	rid

Has	
tid	eid

Εντολές SQL για τις σχέσεις που προκύπτουν

```

CREATE TABLE Customer (
    cid INT PRIMARY KEY,
    mail VARCHAR(255) NOT NULL,
    credit_info VARCHAR(255),
    f_name VARCHAR(100),
    l_name VARCHAR(100)
);

CREATE TABLE Event (
    eid INT PRIMARY KEY,
    name VARCHAR(255) NOT NULL,
    type VARCHAR(100),
    time TIME,
    date DATE,
    capacity INT
);

CREATE TABLE Ticket (
    tid INT PRIMARY KEY,
    type VARCHAR(100),
    price DECIMAL(10, 2),
    availability BOOLEAN DEFAULT TRUE,
    seat_number INT
);

CREATE TABLE Reservation (
    rid INT PRIMARY KEY,
    eid INT,
    cid INT,
    date DATE,
    total_price DECIMAL(10, 2),

```

```

        tickets_number INT,
        FOREIGN KEY (eid) REFERENCES Event(eid),
        FOREIGN KEY (cid) REFERENCES Customer(cid)
    );

CREATE TABLE Contains (
    eid INT,
    tid INT,
    PRIMARY KEY (eid, tid),
    FOREIGN KEY (eid) REFERENCES Event(eid),
    FOREIGN KEY (tid) REFERENCES Ticket(tid)
);

CREATE TABLE Makes (
    cid INT,
    rid INT,
    PRIMARY KEY (cid, rid),
    FOREIGN KEY (cid) REFERENCES Customer(cid),
    FOREIGN KEY (rid) REFERENCES Reservation(rid)
);

CREATE TABLE Has (
    tid INT,
    eid INT,
    PRIMARY KEY (tid, eid),
    FOREIGN KEY (tid) REFERENCES Ticket(tid),
    FOREIGN KEY (eid) REFERENCES Event(eid)
);

```

Περιορισμοί Ακεραιότητας

Οι περιορισμοί ακεραιότητας για την βάση δεδομένων περιλαμβάνουν τα εξής:

- Πρωτεύοντα Κλειδιά
 - Customer(cid)
 - Event(eid)
 - Ticket(tid)
 - Reservation(rid)
- Ξένα Κλειδιά
 - Στον πίνακα Reservation, το πεδίο eid αναφέρεται στον πίνακα Event.
 - Στον πίνακα Reservation, το πεδίο cid αναφέρεται στον πίνακα Customer.
 - Στον πίνακα Contains, το πεδίο tid αναφέρεται στον πίνακα Ticket.
- Μοναδικότητα: Το πεδίο mail στον πίνακα Customer είναι μοναδικό για κάθε πελάτη.
- Υποχρεωτικά Πεδία: Τα πεδία cid στον πίνακα Customer, eid στον πίνακα Event, και tid στον πίνακα Ticket δεν επιτρέπουν κενές τιμές.
- Επιτρεπόμενες Τιμές: Οι τιμές του price στον πίνακα Ticket πρέπει να είναι θετικές.

Συναρτησιακές Εξαρτήσεις

Οι συναρτησιακές εξαρτήσεις μεταξύ των πεδίων των πινάκων είναι οι εξής:

- cid → mail, credit_info, f_name, l_name (από τον πίνακα Customer)
- eid → name, type, time, date, capacity (από τον πίνακα Event)
- tid → type, price, availability, seat_number (από τον πίνακα Ticket)
- rid → eid, cid, date, total_price, tickets_number (από τον πίνακα Reservation)
- eid, tid → tickets_number (από τη σχέση Contains)

Μετατροπή σε Τρίτη Κανονική Μορφή

1. Πίνακας Customer

- Κύριο Κλειδί: cid
- 1NF: Όλες οι τιμές είναι ατομικές.
- 2NF: Όλα τα χαρακτηριστικά εξαρτώνται πλήρως από το κύριο κλειδί cid.
- 3NF: Δεν υπάρχουν μεταβατικές εξαρτήσεις.

Συμπέρασμα: Ο πίνακας Customer βρίσκεται σε 3NF.

2. Πίνακας Event

- Κύριο Κλειδί: eid
- 1NF: Όλες οι τιμές είναι ατομικές.
- 2NF: Όλα τα χαρακτηριστικά εξαρτώνται πλήρως από το κύριο κλειδί eid.
- 3NF: Δεν υπάρχουν μεταβατικές εξαρτήσεις.

Συμπέρασμα: Ο πίνακας Event βρίσκεται σε 3NF.

3. Πίνακας Ticket

- Κύριο Κλειδί: tid
- 1NF: Όλες οι τιμές είναι ατομικές.
- 2NF: Όλα τα χαρακτηριστικά εξαρτώνται πλήρως από το κύριο κλειδί tid).
- 3NF: Δεν υπάρχουν μεταβατικές εξαρτήσεις.

Συμπέρασμα: Ο πίνακας Ticket βρίσκεται σε 3NF.

4. Πίνακας Reservation

- Κύριο Κλειδί: rid
- 1NF: Όλες οι τιμές είναι ατομικές.
- 2NF: Όλα τα χαρακτηριστικά εξαρτώνται πλήρως από το κύριο κλειδί rid.
- 3NF: Δεν υπάρχουν μεταβατικές εξαρτήσεις.

Συμπέρασμα: Ο πίνακας Reservation βρίσκεται σε 3NF.

5. Σχέση Contains

- **Κύριο Κλειδί:** Σύνθετο κλειδί (eid, tid)
- 1NF: Όλες οι τιμές είναι ατομικές.
- 2NF: Δεν υπάρχουν μερικές εξαρτήσεις.
- 3NF: Δεν υπάρχουν μεταβατικές εξαρτήσεις.

Συμπέρασμα: Η συσχέτιση Contains βρίσκεται σε 3NF.

6. Σχέση Makes

- **Κύριο Κλειδί:** Σύνθετο κλειδί (cid, rid)
- 1NF: Όλες οι τιμές είναι ατομικές.
- 2NF: Δεν υπάρχουν μερικές εξαρτήσεις.
- 3NF: Δεν υπάρχουν μεταβατικές εξαρτήσεις.

Συμπέρασμα: Η συσχέτιση Makes βρίσκεται σε 3NF.

7. Σχέση Has

- **Κύριο Κλειδί:** Σύνθετο κλειδί (tid, eid)
- 1NF: Όλες οι τιμές είναι ατομικές.
- 2NF: Δεν υπάρχουν μερικές εξαρτήσεις.
- 3NF: Δεν υπάρχουν μεταβατικές εξαρτήσεις.

Συμπέρασμα: Η συσχέτιση Has βρίσκεται σε 3NF

Ερωτήματα SQL

Ακολουθούν παραδείγματα SQL ερωτημάτων για την βάση δεδομένων:

- Ερώτημα για την εύρεση όλων των κρατήσεων ενός πελάτη:

```
SELECT * FROM Reservation  
WHERE cid = 164;
```

- Ερώτημα για την εύρεση όλων των διαθέσιμων εισιτηρίων για ένα γεγονός:

```
SELECT * FROM Ticket  
WHERE availability = TRUE AND eid = 163;
```

- Ερώτημα για την εύρεση του συνολικού κόστους μιας κράτησης:

```
SELECT SUM(price) FROM Ticket  
JOIN Reservation ON Ticket.eid = Reservation.eid  
WHERE Reservation.rid = 924;
```

Ψευδοκώδικας Διαδικασιών

```
PROCEDURE ADD_CUSTOMER(email, credit_info, first_name, last_name):
    IF email is not empty AND first_name is not empty
        AND last_name is not empty:
        INSERT INTO Customer (email, credit_info,
            first_name, last_name)
        RETURN success
    ELSE:
        RETURN "Missing required fields"

PROCEDURE CREATE_RESERVATION(customer_id, event_id,
                                ticket_count, total_price):
    IF ticket_count > 0 AND total_price > 0:
        SELECT capacity FROM Event
        WHERE eid = event_id
        IF capacity >= ticket_count:
            INSERT INTO Reservation (cid, eid,
                                    tickets_number, total_price)
            VALUES (customer_id, event_id, ticket_count, total_price)
            IF reservation was successfully created:
                FOR i = 1 TO ticket_count:
                    SELECT tid FROM Ticket
                    WHERE eid = event_id AND availability = TRUE
                    LIMIT 1
                    UPDATE Ticket
                    SET availability = FALSE
                    WHERE tid = selected_ticket_id AND eid = event_id
                RETURN reservation ID
            ELSE:
                RETURN "Reservation creation failed"
        ELSE:
            RETURN "Not enough available capacity
                for the requested tickets"
    ELSE:
        RETURN "Invalid ticket count or total price"

PROCEDURE UPDATE_TICKET_AVAILABILITY(ticket_id, availability):
    IF ticket_id exists AND availability is valid:
        UPDATE Ticket
        SET availability = availability
        WHERE ticket_id = ticket_id
        RETURN success
    ELSE:
        RETURN error "Invalid ticket ID or availability"

PROCEDURE CHECK_EVENT_CAPACITY(event_id):
    SELECT capacity
    FROM Event
    WHERE event_id = event_id
    RETURN capacity

PROCEDURE ASSIGN_TICKET_TO_EVENT(ticket_id, event_id):
    IF ticket_id exists AND event_id exists:
```

```

        INSERT INTO Has (ticket_id, event_id)
        RETURN success
ELSE:
    RETURN "Invalid ticket or event"

PROCEDURE CANCEL_RESERVATION(reservation_id):
    IF reservation_id exists:
        DELETE FROM Reservation
        WHERE reservation_id = reservation_id
        UPDATE ticket availability
        RETURN success
    ELSE:
        RETURN "Reservation not found"

PROCEDURE UPDATE_EVENT(event_id, new_capacity):
    IF event_id exists AND new_capacity > 0:
        UPDATE Event
        SET capacity = new_capacity
        WHERE event_id = event_id
        RETURN success
    ELSE:
        RETURN "Invalid event ID or capacity"

PROCEDURE GET_CUSTOMER_RESERVATIONS(customer_id):
    SELECT *
    FROM Reservation
    WHERE customer_id = customer_id
    RETURN reservation details

PROCEDURE GET_EVENT_TICKETS(event_id):
    SELECT *
    FROM Ticket
    WHERE event_id = event_id AND availability = TRUE
    RETURN available tickets


PROCEDURE GET_TOTAL_COST(reservation_id):
    SELECT SUM(price)
    FROM Ticket
    JOIN Reservation
        ON Ticket.event_id = Reservation.event_id
    WHERE Reservation.reservation_id = reservation_id
    RETURN total cost

```

2η φάση

Ενδεικτικά αποτελέσματα από την εκτέλεση των διαδικασιών



Add New Customer

Rafail	
Drakakis	
rafaildrakakis123@gmail.com	
345432	
<button>Add Customer</button>	

Customers Table

	cid	credit_info	f_name	l_name	mail
1	345432		Rafail	Drakakis	rafaildrakakis123@gmail.com

Add New Event

Scorpions last concert
Concert
10:33 PM 
12/18/2024 
12
6
4
2
10
5
3
Add Event

Events Table

capacity	date	eid	name	time	type
12	2024-12-18	1	Scorpions last concert	22:33	Concert

Book Tickets

Scorpions last concert ▼

rafaidrakakis123@gmail.com

5

Type for ticket #1: VIP(10€) ▼

Type for ticket #2: VIP(10€) ▼

Type for ticket #3: VIP(10€) ▼

Type for ticket #4: VIP(10€) ▼

Type for ticket #5: VIP(10€) ▼

Book Tickets

Tickets Table

availability	price	seat_number	tid	type
0	10	1	1	VIP
0	10	2	2	VIP
0	10	6	3	VIP
0	10	5	4	VIP
1	5	8	5	FrontRow
1	5	9	6	FrontRow
1	5	10	7	FrontRow
1	3	11	8	Normal
1	3	12	9	Normal
0	10	3	10	VIP
1	10	4	11	VIP
1	5	7	12	FrontRow

Search Available Seats

Scorpions last concert

Search

AvailableSeatsResult Table

seat_number	type
8	FrontRow
9	FrontRow
10	FrontRow
11	Normal
12	Normal
4	VIP
7	FrontRow

View profits

Scorpions last concert

All types

View

50

View Active Reservations

Starting date

12/16/2024

Final date

12/19/2024

View

ActiveReservationsResult Table

cid	date	eid	tickets_number	total_price
1	2024-12-18	1	5	50

View Event Statistics

Most Popular Event (most reservations)

Event with Highest Profit

Refresh Database

Events Table

capacity	date	eid	name	time	type
12	2024-12-18	1	Scorpions last concert	22:33	Concert

Most popular event is Scorpions last concert with 1 reservations.

View Event Statistics

Most Popular Event (most reservations)

Event with Highest Profit

Refresh Database

Events Table

capacity	date	eid	name	time	type
12	2024-12-18	1	Scorpions last concert	22:33	Concert

Highest profit event is Scorpions last concert with revenue of 50€.

Εγχειρίδιο χρήσης της εφαρμογής

Για να χρησιμοποιηθεί η εφαρμογή πρέπει να γίνουν τα ακόλουθα:

- Αρχικά, εάν θέλουμε μια καθαρή βάση, θα πρέπει να διαγράψουμε το αρχείο `eventManagement.db` το οποίο περιέχει τις προηγούμενες εγγραφές που έγιναν στην βάση.
- Τρέχουμε `python setupDB.py` ώστε να δημιουργήσουμε μια νέα βάση.
- Εκτελούμε `python app.py` για να τρέξουμε την εφαρμογή.
- Ανοίγουμε έναν browser και γράφουμε στη μπάρα διευθύνσεων `http://127.0.0.1:5000/`, αμέσως μετά θα δούμε το διαχειριστικό περιβάλλον της εφαρμογής μας.

Οι λειτουργίες που υποστηρίζονται είναι:

• Εγγραφή νέου πελάτη

Για να εγγραφείτε πρέπει να συμπληρωθούν τα παρακάτω πεδία:

- First name (μικρό όνομα, π.χ. John)
- Last name (επώνυμο, π.χ. Doe)
- Email (διεύθυνση ηλεκτρονικού ταχυδρομείου, π.χ. john.doe@gmail.com)
- Credit info (αριθμός πιστωτικής κάρτας, π.χ. 1234 5678 ...) (χωρίς κενά)

Πατήστε το μπλε κουμπί για εγγραφή.

• Δημιουργία νέας εκδήλωσης

Για να δημιουργήσετε μια νέα εκδήλωση απαιτούνται οι παρακάτω πληροφορίες:

- Όνομα εκδήλωσης
- Τύπος εκδήλωσης (π.χ. festival, comedy show, συναυλία)
- Ώρα διεξαγωγής σε 12 hour format (π.χ. 10:40 PM)
- Ημερομηνία διεξαγωγής σε format <month>/<day>/<year> (π.χ. 08/15/2025)
- Χωρητικότητα του χώρου διεξαγωγής σε θέσεις
- Πλήθος VIP θέσεων
- Πλήθος θέσεων πρώτης γραμμής
- Πλήθος υπολοίπων θέσεων
- Τιμή θέσης VIP
- Τιμή θέσης πρώτης γραμμής
- Τιμή κανονικής θέσης

Πατήστε το μπλε κουμπί για δημιουργία.

• Κράτηση εισιτηρίου

Για να κάνετε κράτηση εισιτηρίου:

1. Διαλέξτε την εκδήλωση από το dropdown menu.
2. Εισάγετε τη διεύθυνση e-mail.
3. Εισάγετε το πλήθος των εισιτηρίων που θέλετε.

Για επιβεβαίωση πατάμε το μπλε κουμπί.

- **Εύρεση θέσης**

Με την επιλογή του ονόματος της εκδήλωσης (μέσω του drop-down menu) θα εμφανίζονται οι θέσεις που είναι κενές.

- **Ακύρωση εκδήλωσης**

Για να ακυρωθεί η εκδήλωση, επιλέγουμε το όνομα από το drop-down menu.

- **Ακύρωση κράτησης**

Για να ακυρωθεί η κράτηση, εισάγουμε το mail με το οποίο έγινε η κράτηση. Όταν πατηθεί το μπλε κουμπί, η κράτηση ή οι κρατήσεις που έγιναν με αυτό το mail είναι άκυρες.

- **Εμφάνιση κέρδους**

1. Επιλέγουμε την εκδήλωση που θέλουμε.
2. Επιλέγουμε τύπο εκδήλωσης.

Πατώντας το View θα εμφανίζονται τα ανάλογα αποτελέσματα με βάση την είσοδο.

- **Εμφάνιση ενεργών κρατήσεων**

1. Επιλογή ημ/νίας έναρξης.
2. Επιλογή ημ/νίας τερματισμού.

Πατώντας το κουμπί θα εμφανιστούν οι κρατήσεις που βρίσκονται εντός αυτού του χρονικού διαστήματος.

- **Στατιστικά στοιχεία**

- Η πιο δημοφιλής εκδήλωση.
- Η εκδήλωση που έκανε τον περισσότερο τζίρο.

- **Εμφάνιση δεδομένων για πελάτες, εκδηλώσεις, εισιτήρια, κρατήσεις**

Οι πληροφορίες αυτές βρίσκονται στο τέλος της σελίδας, όπου και είναι εμφανείς οι πίνακες της βάσης δεδομένων.

Περιγραφή των περιορισμών της υλοποίησης

Δεδομένης της φύσης του project αλλά και της υλοποίησης, υπάρχουν οι εξής περιορισμοί:

- Τα δεδομένα έχουν προκαθορισμένα πεδία που ενδέχεται να μην καλύπτουν ειδικές περιπτώσεις (π.χ. πολλαπλά είδη πληρωμών, εκδηλώσεις πολλαπλών ημερών).
- Ελλιπής υποστήριξη πληρωμών: Η εφαρμογή διαχειρίζεται τη χρήση πιστωτικών καρτών, χωρίς να υπάρχουν μηχανισμοί ασφαλείας για τη διαχείριση ευαίσθητων δεδομένων.
- Δεν περιγράφεται πώς θα διασφαλιστεί η προστασία προσωπικών δεδομένων των πελατών (π.χ. για λόγους GDPR).