ΠΤΥΧΙΑΚΉ ΕΡΓΑΣΙΑ

“ ΣΧΕΔΙΑΣΜΟΣ ΚΑΙ ΥΛΟΠΟΙΗΣΗ ΒΑΣΗΣ ΔΕΔΟΜΕΝΩΝ ΣΥΣΤΗΜΑΤΟΣ ΕΠΙΧΕΙΡΗΜΑΤΙΚΟΥ ΕΝΔΙΑΦΕΡΟΝΤΟΣ“

ΔΑΓΑΛΑΚΗΣ ΚΩΝΣΤΑΝΤΙΝΟΣ

ΕΠΙΒΛΕΠΩΝ: ΤΑΜΠΑΚΑΣ ΒΑΣΙΛΕΙΟΣ, ΚΑΘΗΓΗΤΗΣ

ΑΝΤΙΡΡΙΟ, ΜΑΡΤΙΟΣ 2018

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή

Αντίρριο, Ημερομηνία

ΕΠΙΤΡΟΠΗ ΑΞΙΟΛΟΓΗΣΗΣ

1.Ονοματεπώνυμο, Υπογραφή
2.Ονοματεπώνυμο, Υπογραφή
3.Ονοματεπώνυμο, Υπογραφή

ΠΕΡΙΛΗΨΗ

Στα πλαίσια της πτυχιακής μου εργασίας μού ανατέθηκε ο σχεδιασμός και η υλοποίηση ενός συστήματος το οποίο θα μπορούσε να αυτοματοποιήσει τη λειτουργία ηλεκτρονικών κρατήσεων από ξενοδοχεία. Το παραδοτέο της εργασίας αυτής θα είναι μια λεπτομερής αναφορά η οποία θα εκθέτει και θα τεκμηριώνει το σχεδιασμό αυτού του συστήματος, ο κώδικας της υλοποίησής της καθώς και ένα βασικό γραφικό περιβάλλον για τον έλεγχο των υποστηριζόμενων λειτουργιών. Η υλοποίηση έγινε με χρήση του συστήματος διαχείρισης βάσεων δεδομένων MySQL.

ABSTRACT

In the context of my thesis, I was tasked with designing and implementing a system that could automate the online booking of hotels rooms by web customers. The deliverable of this work will be composed of a detailed report that will outline and document the design of this system, the implementation code, and a basic graphical environment for assuring the correct functionality of the supported operations. This implementation relied on using the MySQL database management system.

ΚΙΝΗΤΡΟ

Ο τουρισμός αποτελεί ένα από τα βασικότερα μέσα ανάπτυξης της οικονομίας της Ελλάδας όχι μόνο λόγο της αύξησης των εσόδων και της δημιουργίας νέων θέσεων εργασίας αλλά και γιατί με αυτό το τρόπο δημιουργείται κίνητρο ανάπτυξης της χώρας και σε άλλους τομείς. Όπως φαίνεται και στην εικόνα 1, ο τουρισμός συνεισφέρει σημαντικά στο Ακαθάριστο Εγχώριο Προϊόν (Α.Ε.Π.) της Ελλάδας και πιο συγκεκριμένα παρατηρείται αύξηση στην επί της εκατό συμβολή του σε βάθος χρόνου[1].
Έτσι, η παρούσα πτυχιακή εργασία βασίζεται σε ένα από τα πολλά θέματα που αφορούν στον τουρισμό όπως η δημιουργία μιας βάσης δεδομένων και έπειτα η επεξεργασία της από ένα site/μια ιστοσελίδα.

Εικόνα 1: Η συμβολή του τουρισμού στο Α.Ε.Π. από το 2010 έως το 2018.

ΣΥΝΤΟΜΗ ΠΕΡΙΓΡΑΦΗ

Η πτυχιακή με τίτλο "Σχεδιασμός και υλοποίηση βάσης δεδομένων συστήματος επιχειρηματικού ενδιαφέροντος" αναπτύχθηκε λόγο της ανάγκης διευκόλυνσης και γρήγορης εξυπηρέτησης μεγάλου αριθμού πελατών διαφόρων επιχειρήσεων, όπως ξενοδοχείων, κατά την τουριστική περίοδο. Αρχικά, σκοπός ήταν η δημιουργία μιας βάσης δεδομένων μέσω της οποίας θα εκτελούνται τα καθήκοντα ενός διαχειριστή ξενοδοχείου. Η βάση δεδομένων που δημιουργήθηκε περιλαμβάνει πεδία για την διαχείριση διαφόρων στοιχειών που αφορούν σε χρήστες της ιστοσελίδας, πελάτες, δωμάτια, πληρωμές, ταξιδιωτικά πρακτορεία, κρατήσεις και προσφορές. Στη συνέχεια, για να γίνει ακόμα πιο εύκολη και απλή η διαχείριση του συστήματος κρατήσεων ενός ξενοδοχείου από τον διαχειριστή, δημιουργήθηκε ένα site το οποίο συνδέθηκε με την βάση δεδομένων που αναφέρεται παραπάνω.
Για την δημιουργία της βάσης δεδομένων χρησιμοποιήθηκε το πρόγραμμα phpMyAdmin ενώ για το site χρησιμοποιήθηκαν οι εξής γλώσσες : php,html,css και JavaScript.

ΠΕΡΙΕΧΟΜΕΝΑ

ΚΕΦΑΛΑΙΟ 1^{ο}

Εργαλεία

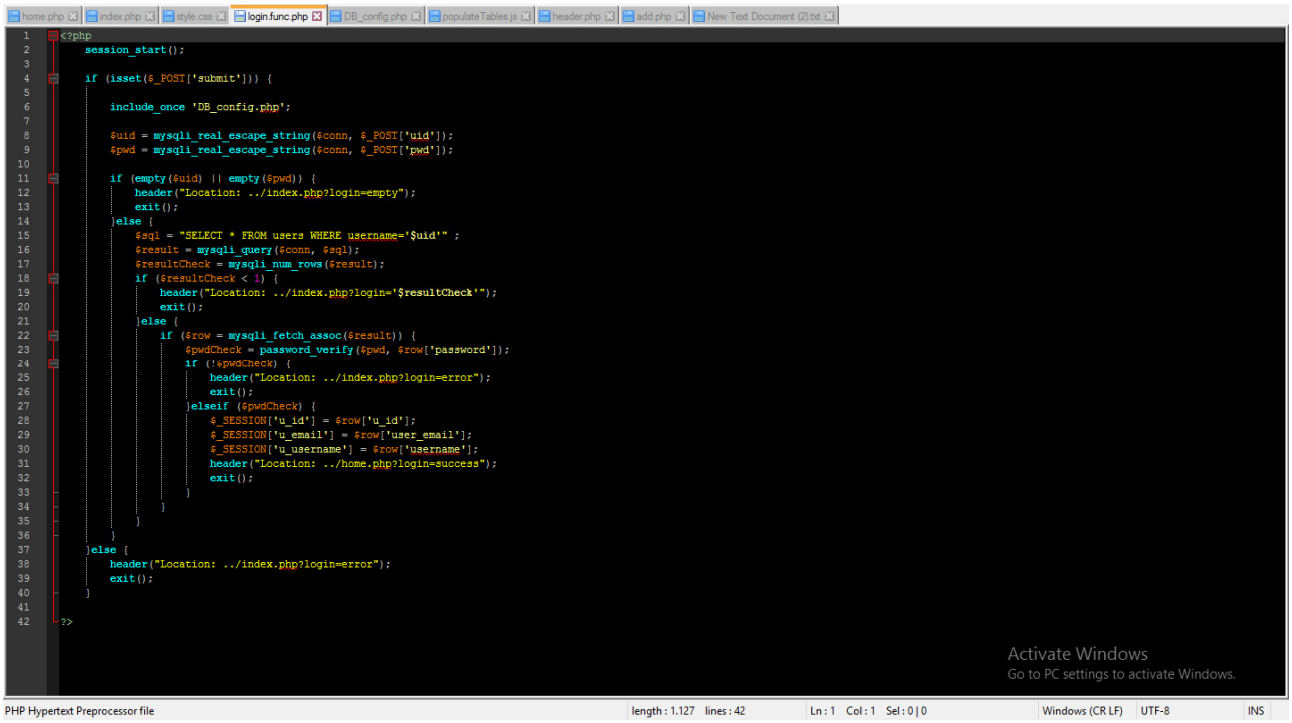
Παρακάτω θα αναλύσουμε τους τρόπους και τα εργαλεία που χρησιμοποιήθηκαν για την σχεδίαση, υλοποίηση και κατασκευή της διαδικτυακής μας εφαρμογής. Ως επί το πλείστων, αυτά είναι open source εργαλεία, τα οποία συνδέονται μεταξύ τους και μας δίνουν ένα πολύ αξιόλογο αποτέλεσμα. Παραδείγματος χάρη, το Xampp μπορεί να διαχειριστεί πολύ οργανωμένα και με εύκολο τρόπο τις open source δυνατότητες που μας παρέχουν η MySql και η PHP. Βασικός παράγοντας για να υλοποιηθεί η εφαρμογή, είναι η εκμάθηση βάσεων δεδομένων και συγκεκριμένα η SQL.
Αναλυτικότερα:

1.1 NOTEPAD ++

Το Notepad++ είναι ένα πρόγραμμα ανοιχτού λογισμικού και πιο συγκεκριμένα ένας text editor, δηλαδή ένα πρόγραμμα σύνταξης κειμένου. Το συγκεκριμένο πρόγραμμα χρησιμοποιείται ευρέως για το λόγο του ότι, είναι αρκετά ελαφρύ και ένας προγραμματιστής μπορεί να αποθηκεύσει πολλά μεγάλα κομμάτια κώδικα σε αυτό, χωρίς να χρειάζεται να καταναλώνει πολλούς πόρους. Χρησιμοποιεί και παρέχει βιβλιοθήκες από πληθώρα γλωσσών προγραμματισμού και αυτό το καθιστά αρκετά χρήσιμο και ευέλικτο.

Το βασικό του μειονέκτημα είναι, ότι όπως περιγράφει και η ονομασία του, είναι ένα απλό notepad, δηλαδή ένα απλό σημειωματάριο. Αυτό σημαίνει ότι δε δίνει την ευκαιρία στο χρήστη, όπως κάνουν άλλοι editors, να "τρέξει" και να ελέγξει τον κώδικα ή το πρόγραμμα του, για αυτό και χρησιμοποιείται κυρίως ως πρόχειρο ή αποθηκευτικός χώρος (ή back up) για το αρχικό τμήμα κώδικα που χρησιμοποιεί ο προγραμματιστής.

Στο παρακάτω παράδειγμα, υπάρχει κώδικας ανεπτυγμένος στη γλώσσα προγραμματισμού java, στο περιβάλλον του notepad++



Εικόνα 1.1.1 Βασικό περιβάλλον του Notepad++.

1.2.1 XAMP

Το Xampp είναι ένα ελεύθερο εργαλείο ανάπτυξης και δοκιμής ιστοσελίδων τοπικά στον υπολογιστή χωρίς να είναι απαραίτητη η σύνδεση στο διαδίκτυο. Περιλαμβάνει μία ελαφριά διανομή Apache, που το καθιστά εξαιρετικά εύκολο για τους προγραμματιστές να δημιουργήσουν έναν τοπικό server για τους σκοπούς της δοκιμής τους. Τοακρωνύμιότουείναι Cross-Platform (X), Apache (A), MySql (M), PHP (P), Perl (P). Τα παραπάνω λογισμικά είναι ότι χρειάζεται κάποιος για να δημιουργήσει ένας web server. Δεδομένου ότι οι περισσότερες πραγματικές web server υλοποιήσεις, χρησιμοποιούν τα ίδια components με το XAMPP, κάνει τη μετάβαση από έναν τοπικό server δοκιμών σε ένα πραγματικό server να είναι εξαιρετικά εύκολη. Η ανάπτυξη μίας διαδικτυακής εφαρμογής, χρησιμοποιώντας το Xampp είναι εξαιρετικά εύκολη ακόμα και για αρχάριους χρήστες καθώς περιλαμβάνει τις PHP και MySql, οι οποίες είναι εξίσου εύκολες να διδαχθούν.

Εικόνα 1.2.1: Λογότυπο του XAMPP

1.2.2 Τα 4 κύρια components του Xampp

1.Apache: Είναι μία πραγματική web server εφαρμογή που επεξεργάζεται και παρέχει περιεχόμενο web σε ένα υπολογιστή. Είναι ο πλέον δημοφιλής web server στο Internet και περιλαμβάνεται σχεδόν στο 50% όλων των ιστοσελίδων.

2.MySQL: Κάθε διαδικτυακή εφαρμογή, είτε απλή είτε πολύπλοκη, απαιτεί μια βάση δεδομένων για την αποθήκευση των δεδομένων. Η MySql, η οποία είναι ανοικτού κώδικα, είναι το πιο δημοφιλής σύστημα διαχείρισης βάσεων δεδομένων. Κινεί τα πάντα, από προσωπικές ιστοσελίδες απλής χρήσης, μέχρι επαγγελματικές πλατφόρμες όπως η WordPress.

3.PHP: Είναι μια server-side scripting γλώσσα, η οποία χρησιμοποιείται σε κάποιες από τις πιο δημοφιλείς ιστοσελίδες στον κόσμο, συμπεριλαμβανομένων τη WordPress και το Facebook. Είναι open source, εύκολη στην εκμάθηση και λειτουργεί άψογα με τη MySql, καθιστώντας την δημοφιλή επιλογή για τους web developers.

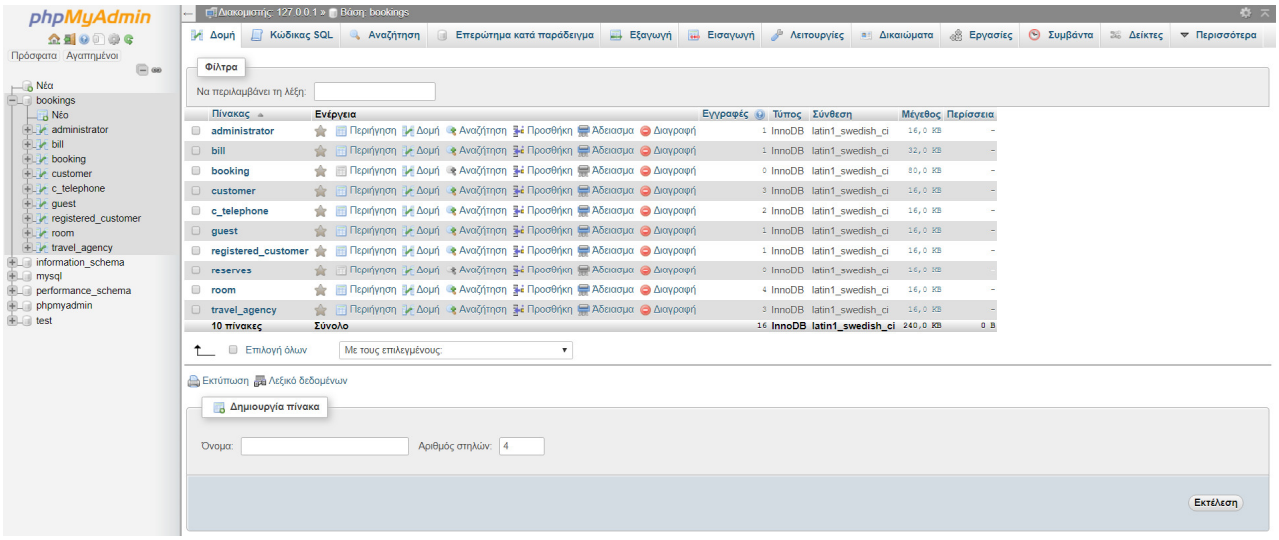
4.Perl: Perl είναι μία υψηλού επιπέδου, δυναμική γλώσσα προγραμματισμού που χρησιμοποιείται ευρέως στον προγραμματισμό διαδικτύου, system admin κ.α. Αν και λιγότερο δημοφιλή για ανάπτυξη web εφαρμογών, η Perl έχει εισέλθει σε ένα κομμάτι της αγοράς για δημιουργία εφαρμογών.

Διάφορες εκδόσεις του XAMPP μπορεί να έχουν πρόσθετα components για τη δημιουργία ενός πλήρη διακομιστή web, όπως το OpenSSL και το phpMyAdmin, το οποίο χρησιμοποιούμε για την υλοποίηση της πτυχιακής. Επίσης, μπορούμε να χειριστούμε τον localhost σαν έναν απομακρυσμένο υπολογιστή, χρησιμοποιώντας σύνδεση μέσω ενός FTP client. Χρησιμοποιώντας ένα πρόγραμμα όπως το Filezilla, έχουμε πολλά πλεονεκτήματα όταν εγκαθιστάτε ένα σύστημα διαχείρισης περιεχομένου(CMS) όπως το Joomla ή Wordpress. Είναι επίσης δυνατή η σύνδεση του localhost μέσω FTP με έναν HTML editor. Η προεπιλογή χρήστη FTP είναι "newuser", ο προεπιλεγμένος κωδικός πρόσβασης FTP είναι "wampp". Η προεπιλογή χρήστη MySql είναι "root" ενώ δεν έχει password.

Βασικές διαφορές που το καθιστούν λειτουργικότερο από τους ανταγωνιστές του είναι το γεγονός ότι μπορεί να δουλέψει εξίσου καλά σε όλα τα λειτουργικά συστήματα, συμπεριλαμβανομένων των Windows, Linux, Unix, Mac. Το interface του είναι πιο απλό προς το χρήστη και κάνει εύκολη τη λειτουργία του.

1.3.1 Phpmyadmin

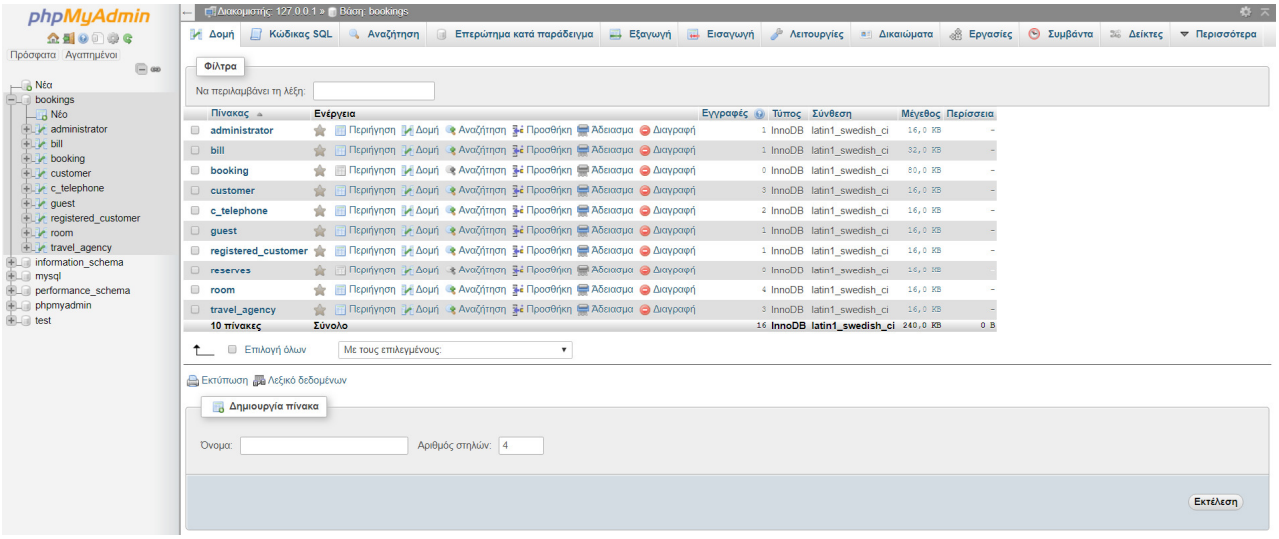
Το phpMyAdmin είναι ένα δωρεάν πρόγραμμα του WorldWideWeb. Το phpMyAdmin υποστηρίζει ένα ευρύ φάσμα δραστηριοτήτων με την MySQL.Οι πιο συχνά χρησιμοποιούμενες λειτουργίες του υποστηρίζονται από το περιβάλλον εργασίας χρήστη (διαχείριση βάσεων δεδομένων, πίνακες, πεδία, σχέσεις, ευρετήρια, οι χρήστες, άδειες, κλπ), ενώ εξακολουθείτε να έχετε τη δυνατότητα να εκτελέσετε άμεσα οποιαδήποτε δήλωση SQL. Σημείωση: Μπορούμε να έχουμε πρόσβαση μέσω του phpMyAdmin στο controlPanel. Μέσω του controlPanel έχουμε τη δυνατότητα για δημιουργίας και διαχείρισης των χρηστών και των βάσεων δεδομένων



Εικόνα 1.3.1: Βασικό περιβάλλον PHPMYADMIN.

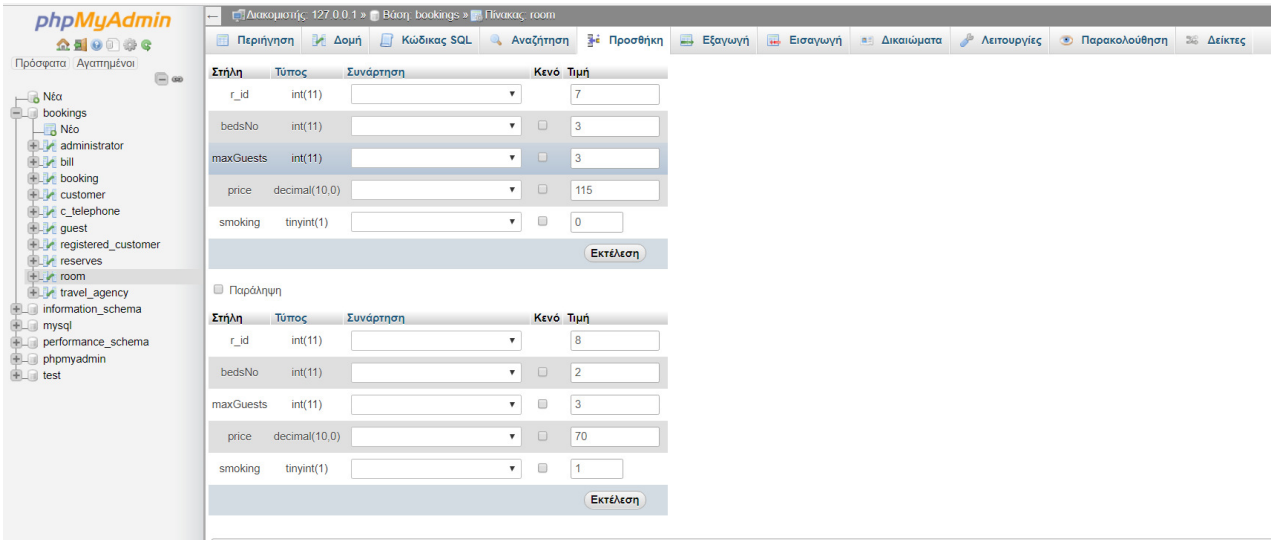
1.3.2 Δημιουργία βάσης δεδομένων στο phpMyAdmin

Η διαδικασία για να δημιουργήσουμε μία νέα βάση δεδομένων είναι πολύ εύκολη διαδικασία. Καθώς βρισκόμαστε στην κεντρική σελίδα του phpMyAdmin, πατάμε το κουμπί Databases. Ορίζουμε το νέο όνομα της βάσης και στο listbox επιλέγουμε την επιλογή greek_general_ci και πατάμε Create.



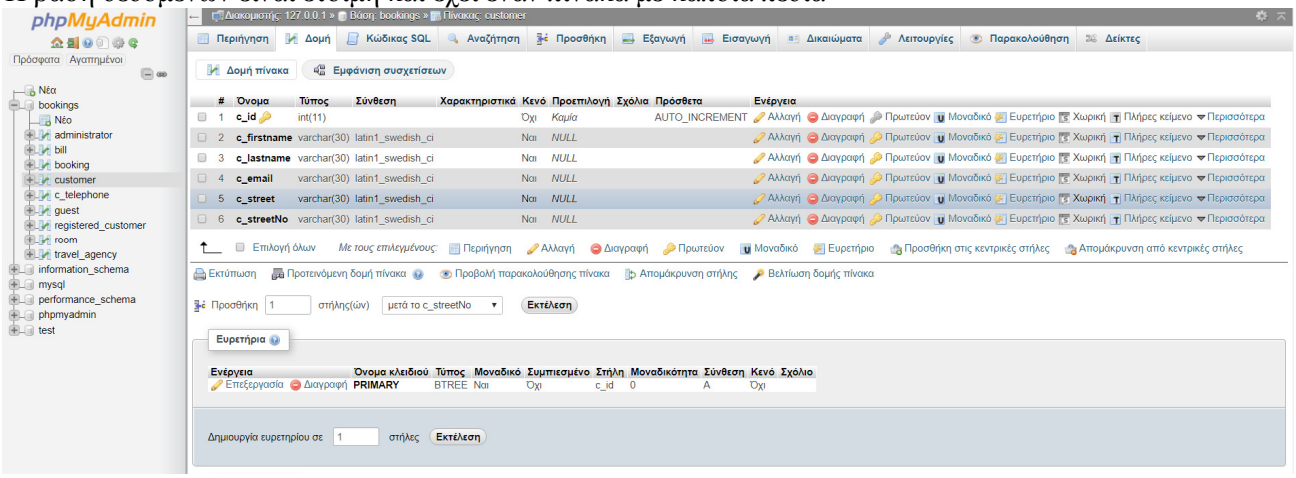
Εικόνα 1.3.2: Δημιουργία νέας βάσης δεδομένων

Για να φτιάξουμε μία εγγραφή, πατάμε πάνω στη βάση μας, δίνουμε το όνομα της εγγραφής (πίνακα) και επιλέγουμε τον αριθμό των χαρακτηριστικών που θα έχει ο πίνακάς μας. Δηλώνουμε τα χαρακτηριστικά και τον τύπο τους. Με αυτό τον τρόπο μπορούμε να δημιουργήσουμε πολλές εγγραφές με πολλά χαρακτηριστικά.



Εικόνα 1.3.3: ολοκλήρωση εγγραφής

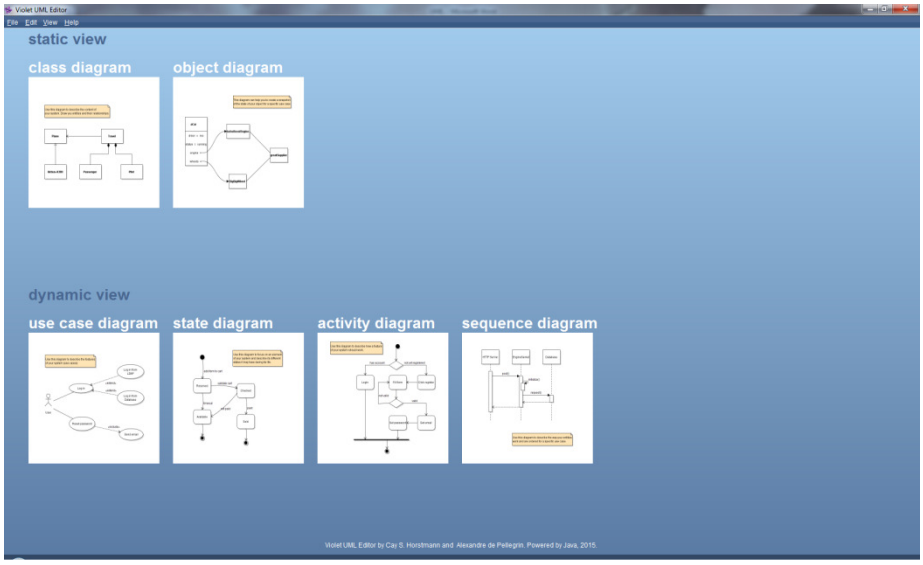
Η βάση δεδομένων είναι έτοιμη και έχει έναν πίνακα με κάποια πεδία



Εικόνα 1.3.4: ολοκλήρωση του πίνακα

1.4.1 VIOLETUML

Το violet UML editor παρέχει όλα τα γνωστά, όσον αφορά τη UML, για το χρήστη και τα διαγράμματα , όπως use-case diagram, class diagram, activity diagram κ.α. Αυτά βρίσκονται στο αρχικό μενού του προγράμματος, δηλαδή ο χρήστης μπορεί να επιλέξει τι θέλει να δημιουργήσει με την εκκίνηση του προγράμματος:



Εικόνα 1.4.1: Μενού επιλογής διαγράμματος του Violet UML.

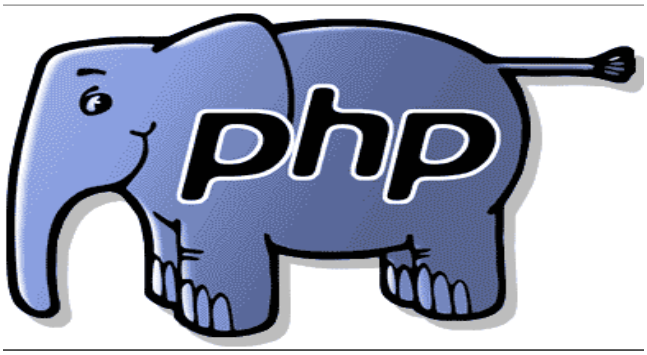
Επιλέγοντας το διάγραμμα που επιθυμεί να αναπτύξει, ο χρήστης, έχει στον εργασιακό χώρο που εν συνεχεία του παρέχεται κάθε λογής εργαλείο που επιθυμεί να χρησιμοποιήσει, προκειμένου να καταλήξει στο επιθυμητό αποτέλεσμα, δηλαδή τη σωστή απεικόνιση της δομής του προγράμματός του.

Εικόνα 1.4.2:Παράδειγμα ένωσης μεταξύ αντικειμένων.

ΚΕΦΑΛΑΙΟ 2ο

Γλώσσες Διαδικτυακού Προγραμματισμού

2.1.1 PHP



Εικονα 2.1.1 λογοτυπο php

Η PHP είναι πιθανός η πιο δημοφιλής scripting γλώσσα προγραμματισμού για το διαδίκτυο. Είναι αντικειμενοστραφής και χρησιμοποιείται για να ενισχύει τις ιστοσελίδες. Με την PHP μπορούμε να κάνουμε διάφορα πράματα, ‘όπως να δημιουργήσουμε σελίδες με login που χρειάζονται username και password, να ελέγξουμε πληροφορίες για μια φόρμα, δημιουργία forums, γκαλερί φωτογραφιών κ.α. Μια ιστοσελίδα γραμμένη σε PHP περνά από επεξεργασία από ένα διακομιστή π.χ. Apache, ώστε να παραχθεί σε πραγματικό χρόνο το τελικό περιεχόμενο που θα σταλεί στον τελικό χρήστη. Ένα αρχείο αποτελούμενο από κώδικα PHP θα πρέπει να έχει κατάληξη .php, .php4, .phtml κ.α. Ανταγωνιστικές προς αυτή γλώσσες προγραμματισμού είναι η ASP της Microsoft, η CFML της Allaire και η JSP της Sun. Το ακρωνύμιο της προέρχεται από το Hypertext Pre-processor. Αλλά με βάση αυτό θα έπρεπε να γράφεται HPP. Μία εναλλακτική εξήγηση είναι ότι τα αρχικά της προέρχονται από την παλαιότερη έκδοση του προγράμματος, η οποία ονομαζόταν Personal Home Page Tools.

2.1.2 PHP Ιστορική αναδρομή

Η ιδέα για τη δημιουργία της PHP ήρθε από τον Rasmus Lerdorf. Οι πρώτες versions της χρησιμοποιήθηκαν για να μπορεί κάποιος να ελέγχει αυτούς που μπαίνουν σε μία σελίδα. Η πρώτη έκδοση είχε το όνομα Personal Home Page Tools και δόθηκε για χρήση στο κοινό στις αρχές του 1995. Αποτελούνταν από μία απλοϊκή μηχανή αναζήτησης, η οποία καταλάβαινε ειδικές μακροεντολές και utilities, ένα guestbook, ένα μετρητή και κάποιο ακόμα υλικό. Ο αναλυτής ξαναγράφτηκε το 1995 και πήρε την ονομασία PHP/F1 version 2. Το όνομα F1 το πήρε από ένα άλλο πακέτο που έγραψε ο Rasmus, το οποίο διερμήνευε τα δεδομένα από της φόρμες της HTML. Στη συνέχεια συνδύασε τα εργαλεία scripts της PHP με τον File Interpreter και πρόσθεσε υποστήριξη για mSql. Η PHP/F1 αναπτύχθηκε ραγδαία, καθώς πολλοί χρήστες άρχισαν να προσφέρουν κώδικα σε αυτή. Έως τα τέλη του 1996 , η PHP/F1 χρησιμοποιούταν από 15.000 ιστοσελίδες παγκοσμίως, ενώ μέσα στο 1997 ο αριθμός αυτός είχε φτάσει τις 50.000. Την ίδια χρονιά,

οι Andi Gutmans και Zeev Suraski ξαναέγραψαν τη γλώσσα από την αρχή, βασιζόμενοι στην C . Έτσι έφτασε η δημιουργία της PHP 3.0. Κατόπιν, οι δύο αυτοί άντρες έφτιαξαν την εταιρία Zend (από τα αρχικά των ονομάτων τους), η οποία συνεχίζει μέχρι σήμερα την ανάπτυξη και εξέλιξη της γλώσσας αυτής. Το 1998 ήρθε η ώρα της PHP 4.0 και τον Ιούλιο του 2004 διατέθηκε η PHP 5.0 ενώ αυτή τη στιγμή έχουν διατεθεί τα πρώτα snapshots της επερχόμενης PHP 6.0 .

2.2 HTML

Εικονα 2.1.2: λογοτυπο HTML

Η γλώσσα προγραμματισμού html αναπτύχθηκε το 1990, βασισμένη στην τεχνική της SGML, η οποία έχει ως στόχο τη σωστή διαχείριση διαδικτυακών εγγράφων. Η html δημιουργήθηκε με αρχικό σκοπό, να παρέχει στους χρήστες, που δεν είχαν τις απαιτούμενες γνώσεις για να χρησιμοποιήσουν την τεχνική SGML, την ευκαιρία να μπορούν να διαχειριστούν και να δημοσιεύσουν διαδικτυακά έγγραφα που επιθυμούν ή θέλουν να αναπτύξουν.

Λόγω του ότι η γλώσσα ήταν αρκετά εύκολη στο να διδαχθεί και να χρησιμοποιηθεί, αλλά και λόγω της ικανότητας της, να μπορεί να αυτοσυντηρείται, γρήγορα άρχισε να χρησιμοποιείται για την ανάπτυξη πολλών εφαρμογών.

Η html, είναι μία γλώσσα προγραμματισμού με κύριο τομέα, αυτόν του διαδικτύου. Τα αρχικά της λέξης αυτής προέρχονται από τα αγγλικά "hyper text markup language" και στα ελληνικά "γλώσσα σήμανσης υπερκειμένου". Είναι η πλέον βασική και κύρια γλώσσα προγραμματισμού, για την ανάπτυξη περιβάλλοντος μίας ιστοσελίδας.

Η μορφή σύμφωνα με την οποία η html αναπτύσσεται και γράφεται, αποτελείται από πολλά στοιχεία όπως: ετικέτες, οι οποίες αντιπροσωπεύουν διάφορους τύπους στοιχείων (components), τα οποία προσδιορίζουν και αποτελούν κομμάτια του σκελετού της ιστοσελίδας. Οι ετικέτες απαρτίζονται από τα σύμβολα (μεγαλύτερο και μικρότερο από (<>)) όπως για παράδειγμα
. Συνήθως οι ετικέτες αυτές λειτουργούν με την εξής λογική: Ξεκινώντας ένα τύπο στοιχείου που θέλουμε να κάνει κάτι, ανοίγουμε την πρώτη ετικέτα η οποία αντιπροσωπεύει την έναρξη, γράφουμε το κομμάτι του κώδικα που επιθυμούμε και κλείνουμε την ετικέτα. Η έναρξη και η λήξη δια χωρίζονται ως εξής : <> και </ > αντίστοιχα.

Τα έγγραφα, τα οποία παράγονται, μέσω της HTML, διαβάζονται από τους περιηγητές διαδικτύου (web browsers). Η βασική δομή που αποτελεί το σκελετό από την αρχή έως το τέλος ενός html εγγράφου γίνεται με τον εξής τρόπο: Ως αρχή του εγγράφου δηλώνουμε ένα tag ως <html> για να συμβολίσουμε τον τύπο του εγγράφου, στη συνέχεια το δεύτερο tag που δηλώνουμε είναι το <body> , το οποίο αντιπροσωπεύει ότι και η λέξη, δηλαδή το σώμα του εγγράφου. Μέσα στο "body" δηλώνουμε και αναπτύσσουμε όλο το περιεχόμενο του εγγράφου html, δηλαδή κομμάτια text παραγράφων, τίτλων, γραφικού περιβάλλοντος και πολλά άλλα. Όταν θέλουμε να " κλείσουμε " το "body" και να φτάσουμε το έγγραφο στο τέλος, κλείνουμε το body ως εξής </body> . Ένα παράδειγμα κώδικα html εγγράφου, το οποίο περιέχει ένα τίτλο και μία παράγραφο, είναι το εξής:

```
<html>
<body>
<h1> Title </h1>
<p>
Paragraph Info
</p>
</body>
</html>
```

Επίσης, μέσα στο έγγραφο της html, μπορούν να δηλωθούν και πολλά άλλα στοιχεία, προέρχομενα από άλλους τύπους εγγράφων και κατα συνέπεια γραμμένα και ανεπτυγμένα από άλλες γλώσσες προγραμματισμού. Σύνηθες παράδειγμα αυτού, είναι η δήλωση javascript εγγράφων μέσα στο έγγραφο html που αναπτύσσουμε, αλλά και εγγράφων css. Η javascript βοηθάει στην υλοποίηση της λειτουργικότητας της Html και η Css στη διαμόρφωση του γραφικού περιβάλλοντος, γνωστό ως τεχνική styling, της html.

Υπάρχουν τρεις τρόποι, σύμφωνα με τους οποίους η html μπορεί να χρησιμοποιήσει έγγραφα άλλων γλωσσών. Ο ένας είναι, να δημιουργηθεί τοπικά το έγγραφο της άλλης γλώσσας που θέλουμε να δανειστούμε, μέσα στο ίδιο το έγγραφο της html. Ο άλλος τρόπος είναι, να δημιουργήσουμε ξεχωριστά το έγγραφο της άλλης γλώσσας και απλώς να το δηλώσουμε μέσα στο έγγραφο της html, προκειμένου να το αναγνωρίσει. Στον τρίτο τρόπο δεν δημιουργούμε κάποια ξεχωριστό έγγραφο όπως στις δύο προηγούμενες περιπτώσει, αλλά δηλώνουμε για το κάθε στοιχείο ξεχωριστά, τοπικά μέσα σε αυτό, αυτό που θέλουμε να προσδιορίσουμε. Για παράδειγμα θέλουμε να δώσουμε χρώμα σε έναν τίτλο στο έγγραφο html, μέσα στην αρχή του tag, γράφουμε style = "" και μέσα στα αυτάκια αυτό που θέλουμε να δώσουμε, στην προκειμένη χρώμα, έστω άσπρο. Άρα θα δηλωθεί ως εξής: <h1 style="color: white;"> Title </h1>.

Στην πρώτη περίπτωση, δημιουργούμε το έγγραφο μέσα στην html, ακριβως πριν το body. Δημιουργούμε πρώτα ένα νέο tag σε αυτη την περίπτωση, το <head> , το οποίο αντιπροσωπεύει ότι και η λέξη, δηλαδή το κεφάλι του εγγράφου της html. Είναι δηλαδή το τι θα δηλώσουμε αρχικά, προκειμένου να χρησιμοποιήσουμε για το χτίσιμο του σώματο "body" και στο τέλος (πριν αρχίσει το body) κλείνουμε το head με το tag </head>. Στο επόμενο παράδειγμα θα χρησιμοποιήσουμε τη γλώσσα προγραμματισμού CSS για να εφαρμόσουμε "styling" στον τίτλο του εγγράφου. Αυτό γίνεται ως εξής. Ακριβώς μετα το tag που ανοίγει το κεφάλι (<head>) δηλώνουμε το tag <style>, το οποίο αντιπροσωπεύει την έναρξη του εγγράφου CSS. Μέσα σε αυτό το πεδίο δηλώνουμε τα στοιχεία που θέλουμε κάνουμε styling, στην προκειμένη τον τίτλο που συμβολίζεται με το tag <h1>. Μέσα στο css το δηλώνουμε με το ίδιο το όνομα του tag, δηλαδή h1 (προκειμένου το body της Html να αναγνωρίσει το αντίστοιχο στοιχείο) ως εξής: h1 {..} , όπου μέσα στις αγκύλες δηλώνουμε από τι αυτό θα εμπεριέχεται (στην προκειμένη, το χρώμα, έστω άσπρο) και τέλος κλείνουμε το styling με το tag </style>. Όλη η δομή του html εγγράφου λοιπόν αυτού, θα έχει ως εξής:

```
<html>
<head>
<style> h1 { Color: white; } </style>
</head>
```

Εικόνα 2.2.2

Στη δεύτερη περίπτωση, δημιουργούμε ξεχωριστά το έγγραφο που θέλουμε να δανείσουμε σε αυτό της html, έστω ενα CSS και δηλώνουμε την ύπαρξη του μέσα στην html προκειμένου να αναγνωρίσει τα στοιχεία τα οποία θέλουμε να της δανείσουμε. Για να το επιτύχουμε αυτό, όπως προαναφέρεται και στην παραπάνω περίπτωση, δηλώνουμε μέσα στο script της CSS το όνομα του στοιχείου που θέλουμε να επεξεργαστούμε, ακριβώς με το ίδιο όνομα του tag της html που το χαρακτηρίζει. Δηλαδή όπως θα δούμε στο επόμενο παράδειγμα, θέλουμε ο τίτλος του html εγγράφου να είναι λευκός και χρησιμοποιούμε για αυτόν, το tag <h1>, άρα στο css, όπως και στο παραπάνω παράδειγμα θα το δηλώσουμε ως h1 { color : white; }. Για να γίνει η δήλωση του εγγράφου μέσα στην html, δηλώνουμε μέσα στο head την εξής γραμμη <link rel="stylesheet" type = "text/css" href="teststyle.css">. Αναλυτικά για αυτή την εντολή: Δηλώνουμε ότι το έγγραφο που δανειζόμαστε είναι τύπου "stylesheet" με το rel="stylesheet", δηλώνουμε ότι είναι τύπου κειμένου (text) css με το type = "text/css" και τέλος δηλώνουμε ποιο είναι αυτό το έγγραφο που θα δανειστούμε δίνοντας τον τύπο και το όνομα του, που σε αυτή την περίπτωση είναι το teststyle τύπου css, άρα href = " teststyle.css ". Αναλυτικά ο κώδικας στο παράδειγμα:

```
<html>
<head>
<linkrel="stylesheet"type="text/css"href="teststyle.css">
</head>
```

Στην τρίτη περίπτωση, δε δημιουργούμε κανένα ξεχωριστό έγγραφο, ούτε τοπικά μέσα στην html αλλά ούτε ως ξεχωριστό, όπως στις 2 προηγούμενες περιπτώσεις. Εδώ δηλώνουμε ξεχωριστά για το κάθε στοιχείο αυτά που θέλουμε να του προσδιορίσουμε, μέσα στο ίδιο το tag του. Δηλαδή, στο παράδειγμα το οποίο αναλύουμε, θέλουμε να δώσουμε ένα τίτλο του εγγράφου και να του δώσουμε λευκό χρώμα. Σε αυτή την περίπτωση, το επιτυγχάνουμε αυτό δηλώνοντας μέσα στο tag το ίδιο το style, δηλαδή ως εξής : <h1 style="color: white;"> Title </h1>.

Μέσα στον κώδικα το βλέπουμε αυτό στην παρακάτω εικόνα, ολοκληρωμένο:

```
<body>
<h1style="color:white;"> Title </h1>
<p>
Paragraph Info
</p>
</body>
```

Η τελευταία αυτή περίπτωση όμως, δεν ενθαρρύνεται γενικά, λόγω του ότι χρησιμοποιεί και καταναλώνει πολλούς πόρους χωρίς λόγο, για το λόγω της συχνής αναφοράς του ίδιου στοιχείου. Άρα, εν κατακλείδι, η γλώσσα html μπορεί να χρησιμοποιηθεί πολύ εύκολα για τη δημιουργία της όλης εικόνας και του γραφικού περιβάλλοντος μιας εφαρμογής, αλλά παράλληλα να βοηθηθεί από έγγραφα άλλων γλωσσών για περαιτέρω λειτουργίες. Ένα από τα αρκετά εύκολα και "γρήγορα", ανοιχτού λογισμικού, περιβάλλοντα, που μπορεί η γλώσσα (αλλά και πολλές άλλες γλώσσες) να αναπτυχθεί, είναι το notepad++ .

Ένα ολοκληρωμένο παράδειγμα μίας απλής σελίδας αναπτυσσόμενης σε html, με τη βοήθεια των στοιχείων απο άλλες γλώσσες, είναι το παρακάτω:

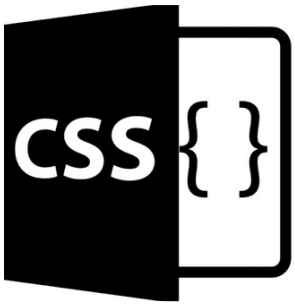
```
<html>
<head>
<style>
```

```
h1{
color: black;
font-style: oblique;
font-size: 185%;
}

button1 {
background-color: green; /*Green*/
color: white;
text-align: center;
font-size: 16px;
}
</style>
</head>
<body>

<h1> EDW EXOUME TON TITLO</h1>
<p> Edw </br> vazoume mia aplh </br> paragrafo, me allages </br> grammhs </p>

<buttontype="button1">Button</button>
</body>
</html>
```

Εικόνα 2.2.5

Αναλυτικότερα για το παραπάνω παράδειγμα, έχουμε δηλώσει ένα header μέσα στην html, το οποίο και χρησιμοποιούμε ως τίτλο του εγγράφου μας, μία απλή παράγραφο με αλλαγές γραμμής και ένα απλό κουμπί. Επίσης έχουμε δηλώσει τοπικά το stylesheet μας, όπου καθορίζουμε το πως θα φαίνεται το κουμπί και το header που χρησιμοποιούμε στο body της html.

2.3 CSS



Εικόνα 2.1.1**:** λογότυπο CSS

Η γλώσσα προγραμματισμού CSS, είναι τα αρχικά της αγγλικής πρότασης "Cascading Style Sheets", στα ελληνικά "Αλληλουχία φύλλων στυλ". Αναπτύχθηκε το 1994, με σκοπό την υποστήριξη άλλων προγραμματιστικών γλωσσών, στο θέμα του γραφικού τους περιβάλλοντος και

στη βελτιστοποίηση της μορφοποίησης του τελικού αποτελέσματος της εφαρμογής, που οι γλώσσες αυτές αναπτύσσουν.

Δημιουργήθηκε πιο συγκεκριμένα για να υποστηρίξει τη γλώσσα προγραμματισμού HTML στο γραφικό της περιβάλλον, αλλά γενικά για να βελτιώσει και να κάνει ευκολότερο, για τους προγραμματιστές, το περιβάλλον ανάπτυξης λογισμικού για το διαδίκτυο, που αφορά το γραφικό του κομμάτι.

Η γλώσσα προγραμματισμού CSS, είναι μία " βοηθητική", προς άλλες γλώσσες προγραμματισμού, γλώσσα, η οποία περιγράφεται κυρίως με μικρά έγγραφα κειμένου. Χρησιμοποιείται κυρίως για να καθορίσει την περιγραφή εγγράφων κειμένου άλλων, όσον αφορά το γραφικό τους περιβάλλον.

Είναι μία γλώσσα, που δε μπορεί από μόνη της να περιγράφει ή να χρησιμοποιηθεί κάπου, για το λόγο του ότι αποτελείται από μικρά κομμάτια κώδικα, τα οποία περιγράφουν αντικείμενα άλλων γλωσσών προγραμματισμού και πιο συγκεκριμένα τη μορφοποίηση αυτών. Έτσι λοιπόν, οι γλώσσες αυτές υιοθετούν το αντίστοιχο CSS έγγραφο κειμένου, που θέλουν να χρησιμοποιήσουν, με αποτέλεσμα να βελτιστοποιούν τη μορφοποίηση των αντικειμένων που έχουν δημιουργηθεί και η CSS περιγράφει.

Η σύνταξη της γλώσσας είναι αρκετά απλή και κοντά στο χρήστη (user-friendly). Χρησιμοποιεί ως αντικείμενα και συναρτήσεις περιγραφής της, αντίστοιχες αγγλικές λέξεις, οι οποίες αντιπροσωπεύουν τις ιδιότητες που περιγράφουν μέσα στη γλώσσα.

Πιο συγκεκριμένα, η δομή ενός εγγράφου CSS, αποτελείται απο πολλά μικρά τμήματα κώδικα, όπου το καθένα ξεχωριστά αποτελεί το σύνολο των ιδιοτήτων, που καθορίζουν το γραφικό τμήμα του αντικειμένου, που το κομμάτι κώδικα αυτού περιγράφει. Η σύνταξης της περιγραφής αυτής, έχει ως εξής: Δίνουμε πρώτα το όνομα του αντικειμένου που θέλουμε να περιγράψουμε, συγκεκριμένα όμως, το όνομα αυτό που θα δηλώσουμε μεσα στο CSS έγγραφο, πρέπει να είναι ακριβώς το ίδιο με αυτό του αντικειμένου, της γλώσσας που θα το χρησιμοποιήσει. Αν για παράδειγμα θέλουμε να περιγράψουμε ένα header στη γλώσσα προγραμματισμού html, το οποίο περιγράφεται με το tag <h1>, τότε και στο CSS το όνομα του αντικειμένου που θα δηλώσουμε, θα είναι το h1. Στη συνέχεια, αφού δηλώσουμε το όνομα, ανοίγουμε και κλείνουμε αγκύλες. Αν δηλαδή το όνομα είναι το h1, τότε ο κώδικας μετά το δεύτερο βήμα, θα έχει ως εξης : h1 {...}. Το επόμενο μας βήμα είναι να " γεμίσουμε " τις αγκύλες, δηλαδή μέσα σε αυτές τοποθετούμε ένα προς ένα, όλες τις ιδιότητες του αντικειμένου. Μερικές από αυτές μπορούν να είναι, το χρώμα του αντικειμένου, το χρώμα του κειμένου του, η στοίχιση του κειμένου, η στοίχιση του μέσα στη σελίδα και πολλά άλλα. Η σύνταξη των ιδιοτήτων των αντικειμένων στη CSS έχει ως εξής: Πρώτα δηλώνουμε τον τύπο μεταβλητής που θέλουμε να του δώσουμε ιδιότητα, παράδειγμα χρώμα (color), στη συνέχεια πληκτρολογούμε το σύμβολο της άνω κάτω τελείας ":", το οποίο αντιπροσωπεύει ότι μετά από αυτό ακολουθεί η ιδιότητα της μεταβλητής. Στο επόμενο βήμα γράφουμε την ιδιότητα που θα δώσουμε στη μεταβλητή μας και τέλος κλείνουμε την πρόταση με το σύμβολο του ελληνικού ερωτηματικού ";". Για παράδειγμα, θέλουμε να δώσουμε χρώμα κίτρινο σε ένα header ενός html εγγράφου, με όνομα (του header) h1. Θα γράψουμε μέσα στο έγγραφο CSS το εξής: h1 { color: yellow; }

Στο παρακάτω παράδειγμα που ακολουθεί, αναπτύσσουμε κώδικα ενός εγγράφου css (stylesheet), το οποίο μπορεί να χρησιμοποιηθεί από ένα έγγραφο html για τη μορφοποίηση των αντικειμένων του.

```
h1 {
color: yellow;
font-style: oblique;
font-size: 185%;
text-align: center;
}
```

```
button1 {
background-color: green;
color: white;
text-align: center;
font-size: 16px;
}

p1 {
background-color:red;
font-size: 14px;
}
```

Εικόνα 2.3.1

Πιο αναλυτικά, στο παραπάνω παράδειγμα, έχουμε δημιουργήσει 3 αντικείμενα. Ένα για το header h1, ένα για το κουμπί button1 κι ένα για τις παραγράφους p1, ενός εγγράφου html. Ως ιδιότητες των αντικειμένων αυτών, δίνουμε το χρώμα τους "color, background-color", το στυλ της "οικογένειας" (font-style), το μέγεθος (font-size) και τη στοίχιση (text-align) του κειμένου τους.

Το έγγραφο html, για του οποίου τα αντικείμενα δημιουργήθηκε το έγγραφο CSS αυτό, θα υιοθετήσει τις ιδιότητες αυτές και θα τις αποδόσει με τη σειρά του στα δικά του αντικείμενα, προκειμένου να αποδοθεί το επιθυμητό αποτέλεσμα, στην τελική μορφοποίησή τους.

ΚΕΦΑΛΑΙΟ 3ο

ΒΑΣΕΙΣ ΔΕΔΟΜΕΝΩΝ

Εικόνα 3.1.1 λογότυπο βάσεων δεδομένων

3.1 Τι είναι Βάση Δεδομένων και Ιστορικά Στοιχεία

Βάση Δεδομένων είναι ένα σύνολο αρχείων με υψηλό βαθμό οργάνωσης τα οποία είναι συνδεδεμένα μεταξύ τους με λογικές σχέσεις, ώστε να μπορούν να τις χρησιμοποιούν πολλοί χρήστες και πολλές εφαρμογές την ίδια στιγμή.
Η ιστορία των Βάσεων Δεδομένων ξεκίνησε στα τέλη της δεκαετίας του 1960 και αφορμή για αυτό ήταν τα τραπεζικά δεδομένα. Ονομαζόντουσαν βάσεις δεδομένων CODASYL, από το όνομα της επιτροπής που της τυποποίησε. Εκείνη την εποχή πρωτοεμφανίστηκε το IMS που ήταν το πιο γνωστό σύστημα διαχείρισης ιεραρχικών βάσεων δεδομένων (hierarchicalDBMS). Το IMS είναι προϊόν της εταιρίας IBM, το οποίο ακόμα και σήμερα έχει τις περισσότερες εγκαταστάσεις. Αυτά τα δύο μοντέλα χαρακτηρίζονται από το γεγονός ότι η διαχείριση των δεδομένων γίνεται σε χαμηλό επίπεδο με αποτέλεσμα οι εφαρμογές να σχετίζονται άμεσα από την τοποθεσία που έχουν αποθηκευτεί τα δεδομένα και ότι τα δεδομένα αυτά τα διαχειριζόμαστε ατομικά. Το 1970, ο E. F. Codd δημοσιεύει ένα άρθρο με τίτλο" A*RelationalModelofDataforLargeSharedDataBanks*", το οποίο έκανε την επανάσταση στο χώρο των βάσεων δεδομένων. Στα μέσα της δεκαετίας του 1980, όπου η Oracle είχε τα πρωτεία στις βάσεις δεδομένων, δημιουργήθηκε το πρώτο standard για μία γλώσσα διαχείρισης και επερώτησης σχεσιακών βάσεων δεδομένων, την StructuredQueryLanguage-SQL.

Εικονα 2.1.1**:** λογοτυπο SQL

3.2 Ιστορία της SQL

Η δημιουργία της SQL (1969) βασίστηκε στην ιδέα του ερευνητή Ted Codd των εργαστηρίων της
IBM, με αφορμή το σχεσιακό μοντέλο που εφαρμόστηκε από τον ίδιο. Αρχικά δημιουργήθηκε με σκοπό να παρέχει μία ημί-φυσική γλώσσα για το IBM System Relational database System. Το 1970, δύο επιστήμονες της IBM Almaden Donald D. Chamberlin και Raymond F. Boyce δημιούργησαν την πρώτη έκδοση της SQL η οποία ονομάστηκε SEQUEL. Το όνομα αυτό αλλάζει σε SQL λόγω του ότι το προηγούμενο ήταν κατοχυρωμένο από εταιρία κατασκευής αεροπλάνων. Το 1985, η IBM κάνει πατέντα την SQL. Οι κυριότερες διάλεκτοι είναι οι τυποποιημένες ANSI/ISO SQL. Πρόκειται για την σταθερή εκδοχή της γλώσσας όπως έχει οριστεί από δύο αναγνωρισμένους οργανισμούς. Υπήρξαν δυο πρότυπα: το SQL1 και το SQL2. Οι περισσότερες εφαρμογές προσπαθούν να μείνουν πιστές στην SQL2.

3.3 Τι είναι η SQL

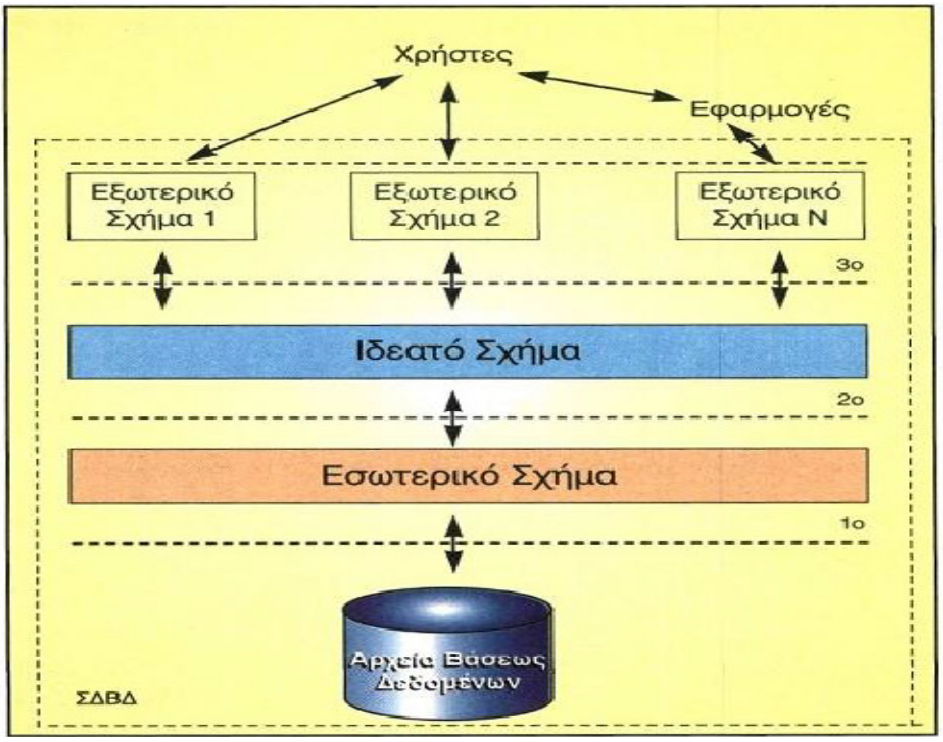
Είναι μία γλώσσα για τις βάσεις δεδομένων και σχεδιάστηκε για τη διαχείριση δεδομένων σε συστήματα διαχείρισης σχεσιακών βάσεων δεδομένων. Είναι μία σχετικά απλή γλώσσα και είναι υψηλού επιπέδου. Η Sql χρησιμοποιεί ειδικές γλώσσες προγραμματισμού, τις λεγόμενες γλώσσες ερωταπαντήσεων (query languages). Είναι γλώσσες μη διαδικαστικές, 4ης γενιάς. Εμείς κάνουμε την ερώτηση και περιμένουμε την απάντηση.
Κάποια από τα πλεονεκτήματά της είναι:

- Η SQL είναι πολύ εύκολη στην εκμάθηση
- Η SQL αποτελεί μια στάνταρτ γλώσσα του ANSI (ANSI standard language)
- Η SQL μάς δίνει τη δυνατότητα να έχουμε πρόσβαση σε μια βάση δεδομένων
- Η SQL μπορεί να εκτελέσει ερωτήματα (queries) και να αναζητήσει πληροφορίες σε μια βάση δεδομένων
- Η SQL μπορεί να ανακτήσει δεδομένα από μια βάση δεδομένων
- Η SQL μπορεί να εισαγάγει νέες εγγραφές σε μια βάση δεδομένων
- Η SQL μπορεί να διαγράψει εγγραφές από μια βάση δεδομένων
- Η SQL μπορεί να ενημερώσει εγγραφές σε μια βάση δεδομένων

Η SQL αποτελείται από τις εντολές και τα ορίσματά τους, τις οποίες μπορούμε να χρησιμοποιήσουμε με συγκεκριμένους κανόνες σύνταξης για να πάρουμε τα αποτελέσματα που θέλουμε. Με την SQL μπορούμε να δημιουργήσουμε μια βάση δεδομένων και τους πίνακές της με τα αντίστοιχα πεδία, να καταχωρήσουμε, να τροποποιήσουμε και να διαγράψουμε τα δεδομένα αυτά, να αλλάξουμε τη δομή των πινάκων με προσθήκη και διαγραφή πεδίων και να εμφανίσουμε.
Η SQL έχει διάφορα τμήματα, τα πιο βασικά είναι τα παρακάτω:
Τη Γλώσσα Ορισμού Δεδομένων (DDL, Data Definition Language), η οποία περιέχει τις απαραίτητες εντολές για τον ορισμό και την τροποποίηση του σχεσιακού σχήματος, τη δημιουργία, την τροποποίηση και τη διαγραφή σχέσεων. Περιέχει ακόμη εντολές δημιουργίας και επεξεργασίας όψεων και ορισμού περιορισμών ακεραιότητας.
Τη Γλώσσα Χειρισμού Δεδομένων (DML, Data Manipulation Language), η οποία περιέχει τις απαραίτητες εντολές για την εμφάνιση δεδομένων καθώς και για την καταχώρηση, τροποποίηση και διαγραφή των εγγραφών (πλειάδων) μιας σχέσης.

Τέλος, περιέχει εντολές για τον ορισμό και την επεξεργασία συναλλαγών και εντολές για την ασφάλεια.

3.4 Συστήματα Διαχείρισης Βάσεων Δεδομένων (ΣΔΒΔ-DBMS)



Εικονα 3.4.1Η αρχιτεκτονική ANSI ενός ΣΔΒΔ

Είναι ένα λογισμικό που μεσολαβεί ανάμεσα στα αρχεία δεδομένων και στις εφαρμογές που χρησιμοποιούνται από τους χρήστες. Είναι δηλαδή ένα πρόγραμμα που διαχειρίζεται βάσεις δεδομένων και με αυτό μπορούμε να τροποποιήσουμε, αποθηκεύσουμε, προσθέσουμε, αφαιρέσουμε ή διαγράψουμε τα δεδομένα. Τα δεδομένα αυτά πρέπει να είναι:

- Ολοκληρωμένα (Integrated), δηλαδή να είναι αποθηκευμένα και οργανωμένα σε αρχεία όπου δεν υπάρχει κάποια επανάληψη ή πλεονασμός ίδιων στοιχείων
- Καταμερισμένα (Shared), δηλαδή να μπορούν πολλοί χρήστες μαζί να έχουν τα ίδια δεδομένα, την ίδια χρονική στιγμή.

Από τη μεριά του ηλεκτρονικού υπολογιστή έχουμε:

- Το λογισμικό (software), που είναι τα προγράμματα και τα εργαλεία με τα οποία επεξεργαζόμαστε τα δεδομένα
- Το υλικό (hardware), που είναι τα περιφερειακά μέρη του υπολογιστή όπως σκληροί δίσκοι, dvd, cloud και εκεί αποθηκεύονται τα δεδομένα μας
- Προγραμματιστές, είναι αυτοί που αναπτύσσουν τις εφαρμογές του ΣΒΔ σε κάποια από τις γνωστές γλώσσες προγραμματισμού

- Διαχειριστής δεδομένων , είναι αυτός που έχει τη διοικητική αρμοδιότητα και ευθύνη για την οργάνωση της βάσης δεδομένων και την απόδοση δικαιωμάτων πρόσβασης στους χρήστες
- Διαχειριστής βάσης δεδομένων, είναι αυτός που λαμβάνει οδηγίες από τον διαχειριστή δεδομένων και διαθέτει τις τεχνικές γνώσεις και αρμοδιότητες για τη σωστή και αποδοτική λειτουργία του ΣΔΒΔ
  Τελικοί χρήστες, είναι αυτοί που χρησιμοποιούν κάποια εφαρμογή για να παίρνουν στοιχεία από μια βάση δεδομένων, έχουν τις λιγότερες δυνατότητες επέμβασης στα στοιχεία της βάσης δεδομένων, χρησιμοποιούν ειδικούς κωδικούς πρόσβασης και το σύστημα τούς επιτρέπει ανάλογα πρόσβαση σε συγκεκριμένο κομμάτι της βάσης δεδομένων

3.5 Εννοιολογικό, Λογικό και Φυσικό Σχήμα (Μοντέλο)

Για να είναι χρήσιμη η βάση δεδομένων μας πρέπει να μπορεί να ανακαλεί τα δεδομένα της αποτελεσματικά. Οι σχεδιαστές των βάσεων για την αποτελεσματική ανάκληση δεδομένων χρησιμοποιούν πολύπλοκες δομές δεδομένων, οι οποίες χρησιμοποιούνται για την αναπαράσταση των δεδομένων της βάσης δεδομένων. Όμως, οι χρήστες μιας Βάσης Δεδομένων δεν είναι ανάγκη να είναι έμπειροι προγραμματιστές, για αυτό το λόγο οι σχεδιαστές κρύβουν την πολυπλοκότητα από τους χρήστες μέσω διαφόρων επιπέδων αφαιρετικότητας. Για να επιτευχθεί η απόκρυψη της πολυπλοκότητας από τους χρήστες χρησιμοποιούνται τρία σχήματα όπως βλέπουμε στην εικόνα 3.5.1 το εννοιολογικό σχήμα, το λογικό σχήμα και το φυσικό σχήμα. Στη βιβλιογραφία το εννοιολογικό και το λογικό σχήμα αναφέρονται πολλές φορές ως ένα σχήμα για αυτό το λόγο στην παρούσα ενότητα το λογικό σχήμα αναφέρεται και ως λογικό (εννοιολογικό) .

Εικόνα 3.5.1 Εννοιολογικό, Λογικό και Φυσικό σχήμα

ΚΕΦΑΛΑΙΟ 4ο

ΚΥΡΙΟ ΜΕΡΟΣ ΤΗΣ ΒΑΣΗΣ BOOKINGS

Επισκόπηση βάσης δεδομένων Bookings

Υποθέτουμε ότι υπάρχουν δύο είδη πελατών (customers): *guestcustomers*, οι οποίοι μπορούν να κάνουν κάποια κράτηση χωρίς να έχουν προηγουμένως εγγραφεί στο σύστημα και οι *registeredcustomers*, οι οποίοι έχουν κάνει εγγραφή στο σύστημα και υπάρχουν αποθηκευμένα τα στοιχεία τους στη βάση. Όλοι οι πελάτες (ανεξάρτητα αν διαθέτουν ή όχι λογαριασμό) μπορούν να κάνουν κάποια κράτηση στο ηλεκτρονικό κατάστημα και να εξοφλούν το υπόλοιπο τους με τη χρήση πιστωτικής κάρτας. Οι πελάτες μπορούν επίσης να ακυρώσουν κάποια κράτηση, λαμβάνοντας ταυτόχρονα πίσω το αντίτιμο που πλήρωσαν για την συγκεκριμένη κράτηση (τα χρήματα επιστρέφονται στον τραπεζικό λογαριασμό που έχουν δηλώσει κατά τη διάρκεια της κράτησης). Θεωρούμε ότι οι δύο τύποι πελατών είναι διακεκριμένοι (δηλ., ένας guest δεν μπορεί να είναι συγχρόνως και registered). Οι παράγραφοι που ακολουθούν περιγράφουν τα δεδομένα τα οποία η βάση δεδομένων πρέπει να διαχειρίζεται για κάθε τύπο πελάτη, καθώς και τις διαδικασίες τις οποίες το σύστημά μας εκτελεί.

4.1. Απαιτήσεις

4.1.1 Δεδομένα

- Δεδομένα Πελατών: περιλαμβάνουν το αναγνωριστικό, το όνομα, το τηλέφωνο, την διεύθυνση ηλεκτρονικού ταχυδρομείου καθώς και την ταχυδρομική διεύθυνση καθώς και τον αριθμό λογαριασμού του
  .
- Δεδομένα Εγγεγραμμένων Πελατών : επιπλέον από τα στοιχεία που διατηρούνται για όλους τους πελάτες, κρατάμε το όνομα χρήστη καθώς και τον κωδικό του που χρησιμοποιεί για την είσοδο του στο σύστημα.

- Δεδομένα Διαχειριστή: περιλαμβάνουν το αναγνωριστικό, το όνομα χρήστη, τον κωδικό καθώς και τη διεύθυνση ηλεκτρονικού ταχυδρομείου.

- Δεδομένα Κρατήσεων: περιλαμβάνουν το αναγνωριστικό της κράτησης, την κατάσταση της κράτησης (ενεργή ή ακυρωμένη), την ημερομηνία άφιξης, την ημερομηνία αναχώρησης, τον αριθμό των επισκεπτών, τον αριθμό των δωματίων. Επιπλέον, κρατάμε το αναγνωριστικό του διαχειριστή που διαχειρίζεται την κράτηση, το αναγνωριστικό του πελάτη που κάνει την κράτηση και το αναγνωριστικό του ταξιδιωτικού γραφείου που προσφέρει την κράτηση.

- Δεδομένα Πληρωμής: περιλαμβάνουν το αναγνωριστικό της πληρωμής, το ποσό πληρωμής, τον αριθμό της πιστωτικής/χρεωστικής κάρτας, τον κάτοχο της και την ημερομηνία λήξης της. Επιπλέον, κρατάμε το αναγνωριστικό του πελάτη ο οποίος πληρώνει τη συγκεκριμένη συναλλαγή, καθώς και το αναγνωριστικό της κράτησης στην οποία αντιστοιχεί η πληρωμή.

- Δεδομένα Ταξιδιωτικών Γραφείων: περιλαμβάνουν το αναγνωριστικό, το όνομα του ταξιδιωτικού πρακτορείου καθώς και το τηλέφωνο επικοινωνίας.

- Δεδομένα Δωματίων: περιλαμβάνουν το αναγνωριστικό του δωματίου, τον αριθμό των κρεβατιών που υπάρχουν στο δωμάτιο, την τιμή του δωματίου, τον μέγιστο αριθμό επισκεπτών που μπορούν να καταλύσουν στο δωμάτιο καθώς και αν επιτρέπεται το κάπνισμα ή όχι στο χώρο του δωματίου.

4.1.2 Διαδικασίες

- Εγγραφή νέου πελάτη: πελάτης ανοίγει λογαριασμό στο ηλεκτρονικό κατάστημα. Αποθηκεύεται όλη η σχετική πληροφορία.

- Κλείσιμο λογαριασμού: πελάτης αποφασίζει να διαγράψει το λογαριασμό του. Κατά τη διαγραφή ενός λογαριασμού θα πρέπει να διαγράφονται και όλες οι πληροφορίες π.χ. κρατήσεις, πληρωμές κτλ που σχετίζονται με το συγκεκριμένο πελάτη.

- Κράτηση: πελάτης πραγματοποιεί μια κράτηση στο ηλεκτρονικό κατάστημα. Όσο διαρκεί η διαδικασία της αγοράς, η παραγγελία κρατείται σε καλάθι αγορών. Με την επιβεβαίωση της παραγγελίας, το καλάθι αδειάζει. Με την αγορά εκδίδεται τιμολόγιο το οποίο ο πελάτης οφείλει να εξοφλήσει με τη χρήση πιστωτικής ή χρεωστικής κάρτας.

- Ακύρωση: πελάτης ακυρώνει μια κράτηση την οποία έχει πραγματοποιήσει. Να σημειώσουμε ότι για τη διαδικασία αυτή θα μπορούσαμε απλά να διαγράψουμε την κράτηση, ωστόσο προτιμήθηκε η λύση της αλλαγής της κατάστασης της ώστε να είναι ευκολότερη η μετάβαση σε πιθανές βελτιώσεις (βλέπετε παρακάτω στην ενότητα «Βελτιώσεις».

- Πληρωμή: πελάτης πληρώνει το ποσό οφειλής του με βάση την κράτηση που έχει κάνει χρησιμοποιώντας την κάρτα του. Υποθέτουμε πως τα στοιχεία της κάρτα που εισάγει ο πελάτης είναι έγκυρα και δεν χρειάζονται περαιτέρω έλεγχο.

- Ερωτήσεις: οι ερωτήσεις περιλαμβάνουν την αναζήτηση διαθέσιμων κρατήσεων βάσει κάποιων τύπων δεδομένων που αποθηκεύονται. Συγκεκριμένα, οι πελάτες μπορούν να αναζητήσουν κρατήσεις για συγκεκριμένο εύρος τιμών. Επίσης, ένας πελάτης μπορεί να ζητήσει μια κατάσταση που δείχνει όλες τις κρατήσεις που είναι διαθέσιμες για μια συγκεκριμένη χρονική περίοδο.

- Καλύτερος πελάτης για ένα συγκεκριμένο έτος: το ηλεκτρονικό κατάστημα μπορεί να ενημερώνει για τον καλύτερο πελάτη ενός συγκεκριμένου έτους (με βάση τις ημερομηνίες checkin/checkout). Ο καλύτερος πελάτης προκύπτει από το μεγαλύτερο άθροισμα των πληρωμών των ενεργών κρατήσεων (μη ακυρωμένων) που έχουν γίνει για το συγκεκριμένο έτος.

- Κατάσταση δημοφιλών δωματίων: περιλαμβάνει τα 5 δωμάτια για τα οποία έχουν γίνει οι περισσότερες ενεργές κρατήσεις ταξινομημένα με βάση των αριθμό των κρατήσεων.

4.2. Μοντέλο Οντοτήτων – Συσχετίσεων (EntityRelationshipModel)

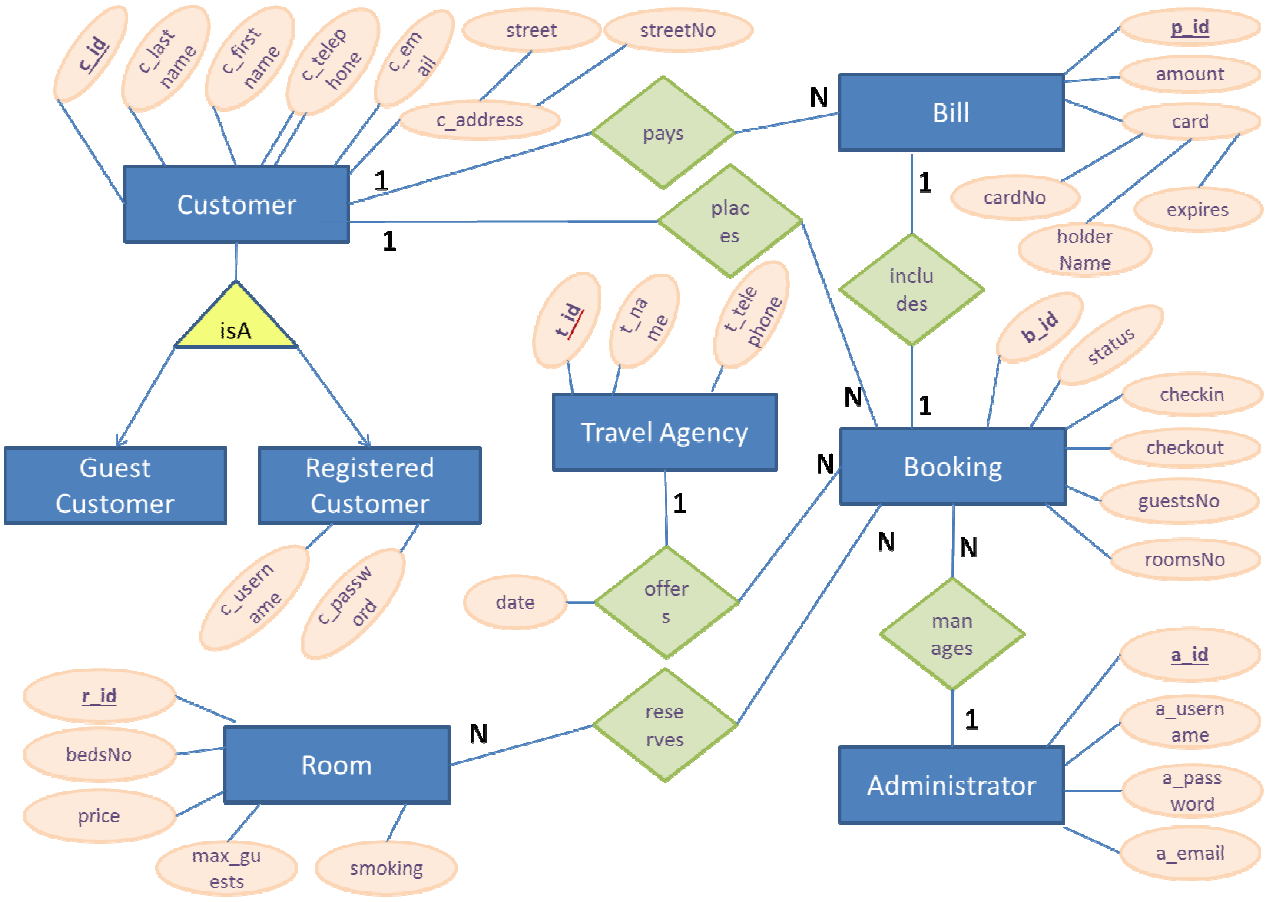
Το Μοντέλο Οντοτήτων Συσχετίσεων (Entity Relationship Model, ER Model)είναι μια διαγραμματική αναπαράσταση της δομής μιας βάσης δεδομένων και χρησιμοποιείται κατά τη φάση του λογικού σχεδιασμού της βάσης. Δηλαδή, δεν ασχολείται με τον τρόπο που αποθηκεύονται τα δεδομένα της βάσης, αλλά με την ταυτοποίηση των δεδομένων και με τον τρόπο με τον οποίο αυτά συσχετίζονται μεταξύ τους. Οι βασικές (θεμελιώδεις) έννοιες του μοντέλου αυτού είναι οι εξής :

- Οντότητες
  Αναπαριστούν ένα διακριτό αντικείμενο του πραγματικού κόσμου με συγκεκριμένες ιδιότητες - χαρακτηριστικά
- Ιδιότητες ή Χαρακτηριστικά
  Διακρίνονται σε *απλά* ή *σύνθετα* γνωρίσματα. Τα σύνθετα γνωρίσματα αποτελούν συνένωση των απλών γνωρισμάτων που τα συνθέτουν. Επίσης, διακρίνονται σε μονότιμα ή πλειότιμα ανάλογα με το αν επιτρέπεται η χρήση πολλαπλών τιμών (π.χ. διαφορετικά τηλέφωνα)
- Συσχετίσεις
  Ορίζουν μια σχέση μεταξύ διαφόρων οντοτήτων. Διακρίνουμε τρεις τύπους συσχετίσεων:
  1. **1-1**
     Όταν ένα στιγμιότυπο της μιας οντότητας συνδέεται μόνο με ένα στιγμιότυπο της άλλης οντότητας και αντιστρόφως
  2. **1-N**
     Όταν ένα στιγμιότυπο της μιας οντότητας μπορεί να συνδέεται με πολλά στιγμιότυπα της άλλης οντότητας, αλλά κάθε στιγμιότυπο της δεύτερης οντότητας συνδέεται μόνο με ένα στιγμιότυπο της πρώτης οντότητας.
  3. **N-N**
     Όταν ένα στιγμιότυπο της μιας οντότητας μπορεί να συνδέεται με πολλά στιγμιότυπα της άλλης οντότητας και επίσης κάθε στιγμιότυπο της δεύτερης οντότητας μπορεί να συνδέεται με πολλά στιγμιότυπα της πρώτης οντότητας.
- Περιορισμοί (κλειδιά, συμμετοχές, πληθικότητες κτλ)

Για να αναπαραστήσουμε ένα Μοντέλο Οντοτήτων – Συσχετίσεων χρησιμοποιούμε ειδικά διαγράμματα με τα εξής χαρακτηριστικά:

- τα **ορθογώνια** συμβολίζουν τις οντότητες
- οι **ρόμβοι** περιγράφουντις σχέσεις μεταξύ των διαφόρων οντοτήτων
- οι **ελλείψεις** αναπαριστούν τις ιδιότητες/χαρακτηριστικά των διαφόρων οντοτήτων
- το **τρίγωνο** συμβολίζει μια σχέση εξειδίκευσης (IsA). Μια σχέση IsA ορίζει μια σχέση υπερκλάσης/υποκλάσης
- η **διπλή γραμμή** σε ένα γνώρισμα περιγράφει ότι το γνώρισμα αυτό είναι πλειότιμο.

Τα παραπάνω αποτελούν τη λογική δομή μιας βάσης δεδομένων, μια εργασία που είναι απαραίτητο να γίνει πριν από την καταχώρηση και την επεξεργασία των στοιχείων (πληροφοριών) της βάσης δεδομένων. Γενικά το μοντέλο οντοτήτων – συσχετίσεων αποτελεί μια γενική περιγραφή των στοιχείων που απαρτίζουν μια βάση δεδομένων και απεικονίζει την αντίληψη που έχουμε για τα δεδομένα (εννοιολογικό), χωρίς να υπεισέρχεται σε λεπτομέρειες υλοποίησης. Με βάση τα παραπάνω προέκυψε το Διάγραμμα Οντοτήτων - Συσχετίσεων της παρούσας εφαρμογής το οποίο παρουσιάζεται στην εικόνα που ακολουθεί.



4.3. Μετάφραση Μοντέλου Οντοτήτων – Συσχετίσεων σε Σχεσιακό Μοντέλο

Το σχεσιακό μοντέλο αποτελεί το πιο συχνά χρησιμοποιούμενο μοντέλο για τη διαχείριση των δεδομένων. Η βασική αρχή μια σχεσιακής βάσης δεδομένων είναι η οργάνωση των δεδομένων της σε πίνακες οι οποίοι συνδέονται μεταξύ τους μέσω κοινών χαρακτηριστικών. Κάθε πίνακας περιλαμβάνει στοιχεία που αφορούν μια οντότητα (π.χ. τα στοιχεία ενός πελάτη). Σε κάθε πίνακα ορίζουμε ένα πεδίο ως πρωτεύων κλειδί (φαίνεται υπογραμμισμένο και με έντονη γραμματοσειρά), οι τιμές του οποίου πρέπει να είναι μοναδικές για κάθε εγγραφή έτσι ώστε να εξασφαλίζουμε ότι όλες οι εγγραφές του πίνακα θα διαφέρουν μεταξύ τους τουλάχιστον ως προς το πεδίο αυτό. Εφόσον η πληροφορία της βάσης δεδομένων, κατανέμεται σε πολλούς πίνακες, έναν για κάθε οντότητα, πρέπει να υπάρχει κι ένας τρόπος συσχετισμού των πινάκων μεταξύ τους. Η βασική αρχή που εφαρμόζεται είναι ότι ένα κοινό πεδίο συνδέει δύο διαφορετικούς πίνακες. Το κοινό πεδίο που χρησιμοποιούμε για την παραπάνω σύνδεση είναι το πρωτεύων κλειδί του ενός πίνακα, το οποίο προστίθεται στον άλλο πίνακα ως ξένο κλειδί (φαίνεται με διακεκομμένη υπογράμμιση). Επιπλέον, ισχύουν τα παρακάτω για τη μετατροπή των συσχετίσεων μεταξύ των διαφόρων οντοτήτων:

- Μια N-N συσχέτιση r μεταξύ οντοτήτων E και F απεικονίζεται σε μια σχέση R το σχήμα της οποίας περιέχει όλα τα γνωρίσματα που ανήκουν στα πρωτεύοντα κλειδιά των σχέσεων που αντιστοιχούν στις οντότητες E και F. Ο συνδυασμός αυτός σχηματίζει το πρωτεύον

κλειδί της R. Επίσης, το σχήμα της R περιέχει όλα τα γνωρίσματα της συσχέτισης r. Οι πλειάδες της R αντιστοιχούν στα στιγμιότυπα της r.

- Μια N-1 συσχέτιση r μεταξύ οντοτήτων E και F δεν απαιτεί τη δημιουργία νέας σχέσης για την αναπαράστασή της. Αν υποθέσουμε ότι ισχύει max - cardinality(F,R) = 1, τότε η σχέση της F πρέπει να περιέχει γνωρίσματα που αντιστοιχούν στο πρωτεύον κλειδί της E (ξένο κλειδί). Αν η F έχει υποχρεωτική συμμετοχή στην r, τότε το ξένο κλειδί δε δέχεται κενές τιμές.
- Μια 1-1 συσχέτιση r μεταξύ οντοτήτων E και F, υποδηλώνει ότι η συμμετοχή των οντοτήτων E και F μπορεί να είναι υποχρεωτική ή προαιρετική. Αν η συμμετοχή των οντοτήτων είναι προαιρετική, δημιουργούμε σχέσεις για τις E και F και προσθέτομε στη μία από αυτές ένα γνώρισμα για το πρωτεύον κλειδί της άλλης. Αν η συμμετοχή τους είναι υποχρεωτική, τότε οι δύο σχέσεις μπορούν να συνδυαστούν σε μία.

Με βάση τα παραπάνω το Μοντέλο Οντοτήτων-Συσχετίσεων της προηγούμενης ενότητας μπορεί να μεταφραστεί στο Σχεσιακό Μοντέλο ως εξής:

Customer

c_id	c_firstname	c_lastname	c_email	street	streetNo

Registered_Customer

c_id	c_username	c_password

Guest_Customer

c_id

c_telephone

c_id	telephone

Bill

p_id	amount	cardNo	holdername	expires	c_id

Booking

b_id	status	checkin	checkout	guestsNo	roomsNo	date_o	c_id	a_id	t_id	p_id

Room

r_id	bedsNo	maxGuests	price	smoking

Reserves

r_id	**b_id**

Travel Agency

t_id	t_name	t_telephone

Administrator

a_id	a_username	a_password	a_email

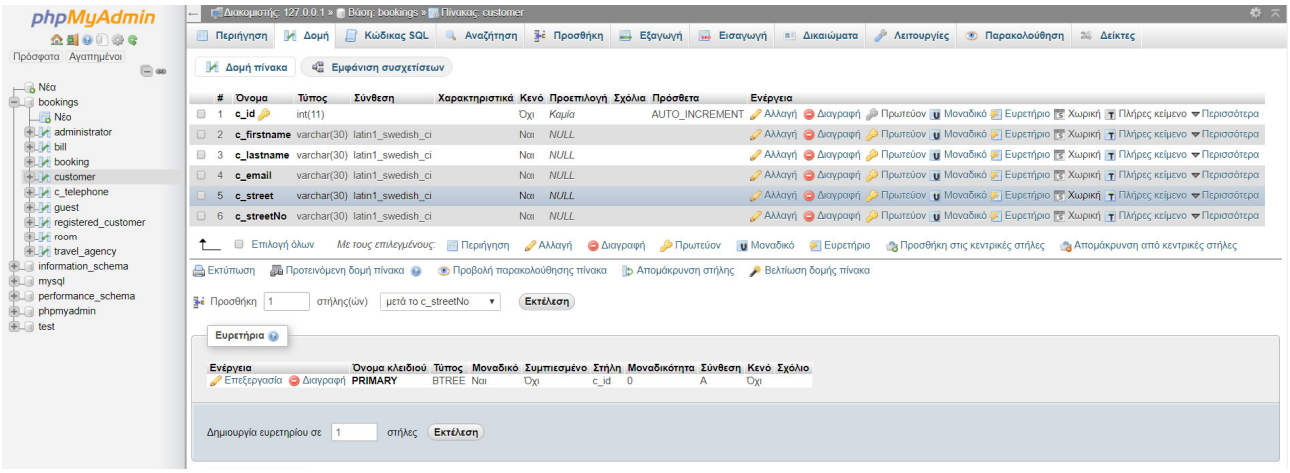
4.4. Πρωτεύοντα Κλειδιά

- Ο πίνακας Customer έχει πρωτεύον κλειδί το c_id, το οποίο αντιπροσωπεύει τον αριθμό του πελάτη που δίνεται από το σύστημα κατά την εγγραφή
- Ο πίνακας GuestCustomer όπως και ο RegisteredCustomer έχουν ως πρωτεύον κλειδί το c_id το οποίο το «κληρονομούν» από την οντότητα Customer
- Ο πίνακας c_telephone, ο οποίος δημιουργείται επειδή θεωρούμε το τηλέφωνο του πελάτη σαν πλειότιμο γνώρισμα, έχει ως πρωτεύον κλειδί το c_idCustomer
- Ο πίνακας Bill έχει ως πρωτεύον κλειδί το p_id το οποίο αντιστοιχεί στον αριθμό της πληρωμής
- Ο πίνακας Booking έχει ως πρωτεύον κλειδί το b_id, το οποίο δημιουργείται κατά την υποβολή μιας κράτησης στο σύστημα από έναν πελάτη
- Ο πίνακας Room έχει ως πρωτεύον κλειδί το r_id, το οποίο αντιστοιχεί στον αναγνωριστικό αριθμό ενός δωματίου
- Ο πίνακας Travel_Agency έχει ως πρωτεύον κλειδί το t_id το οποίο είναι ο αναγνωριστικό αριθμός του ταξιδιωτικού γραφείου
- Ο πίνακας Reserves έχει ως πρωτεύον κλειδί τα r_id και b_id, τα οποία δείχνουν για κάθε κράτηση δωματίου που γίνεται τον αριθμό δωματίου ή τους αριθμούς δωματίων που σχετίζονται με την συγκεκριμένη κράτηση
- Ο πίνακας Administrator έχει ως πρωτεύον κλειδί το a_id , το οποίο δείχνει τον μοναδικό αναγνωριστικό αριθμό κάθε διαχειριστή του συστήματος

4.5. Εντολές της Γλώσσας Επερωτήσεων SQL

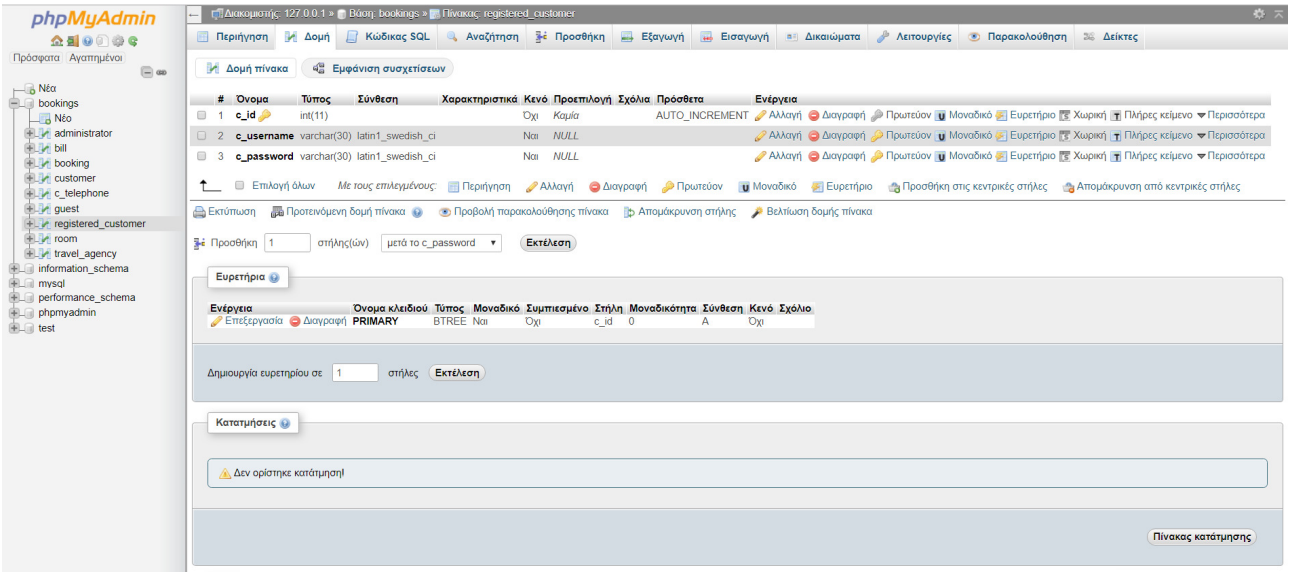
4.5.1. Δημιουργία Πινάκων

```
CREATE TABLE customer(
   c_id int NOT NULL AUTO_INCREMENT,
   c_firstname  varchar(30),
   c_lastname  varchar(30),
   c_email  varchar(30),
   c_street  varchar(30),
   c_streetNo  varchar(30),
   PRIMARY KEY (c_id)
);
```



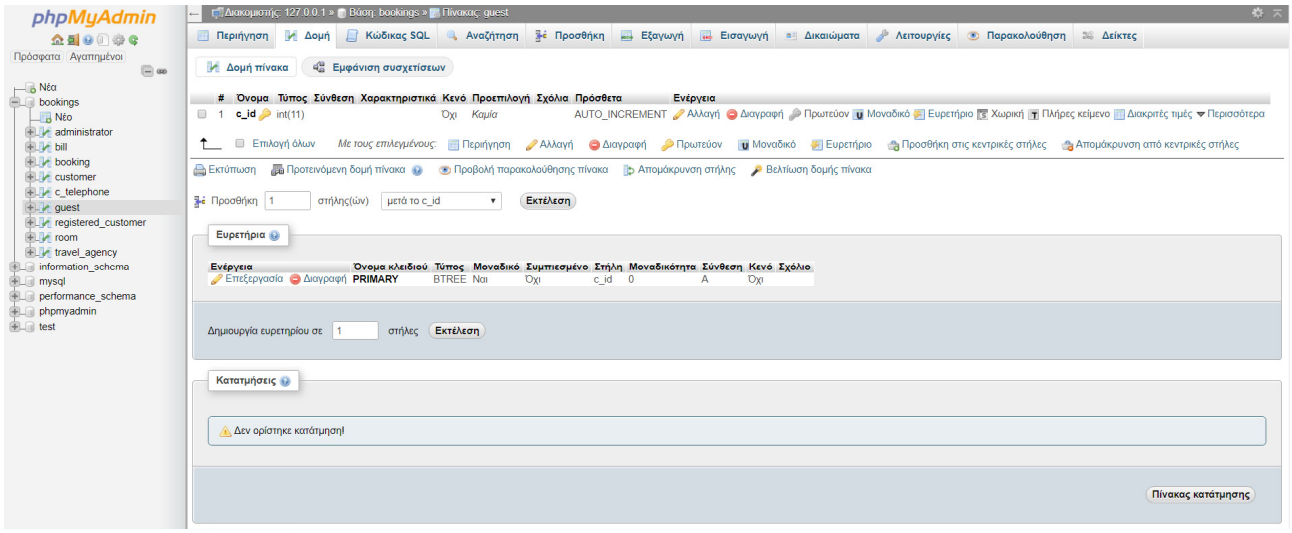
Πίνακας Customer στο PhpMyAdmin

```
CREATE TABLE registered_customer(
   c_id int  NOT NULL  REFERENCES customer(c_id),
   c_username  varchar(30),
   c_password  varchar(30),
   PRIMARY KEY (c_id)
);
```

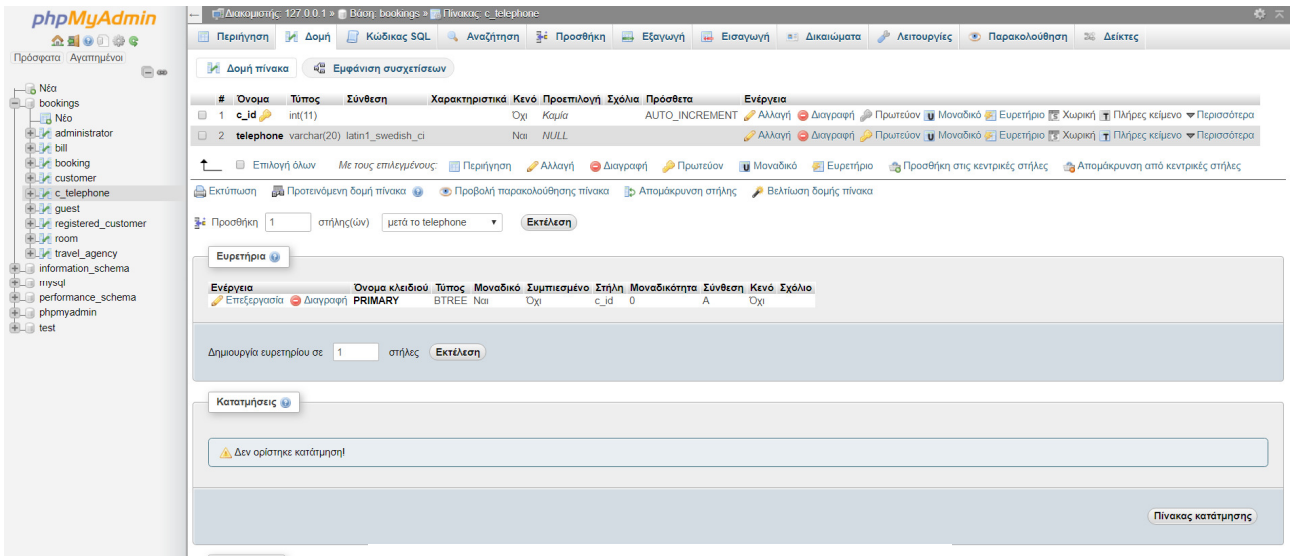


Πίνακας registered_customer στο PhpMyAdmin

```
CREATE TABLE guest(
   c_id int  NOT NULL  REFERENCES customer(c_id),
    PRIMARY KEY (c_id)
);
```

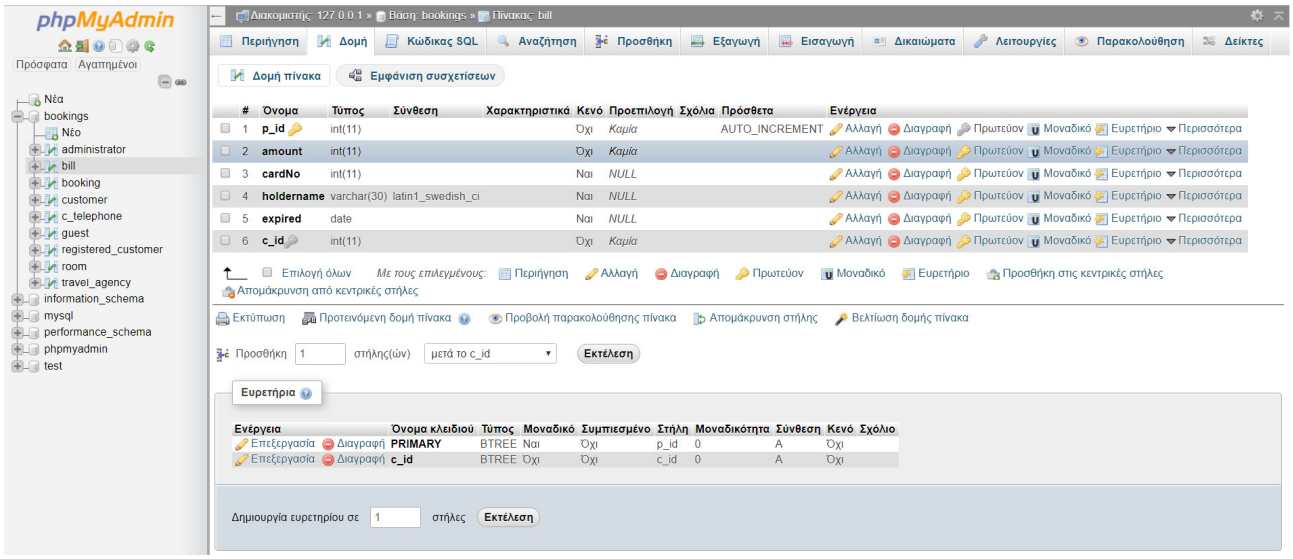



```
CREATE TABLE c_telephone(
   c_id int NOT NULL AUTO_INCREMENT,
   telephone varchar(20),
   PRIMARY KEY (c_id)
);
```



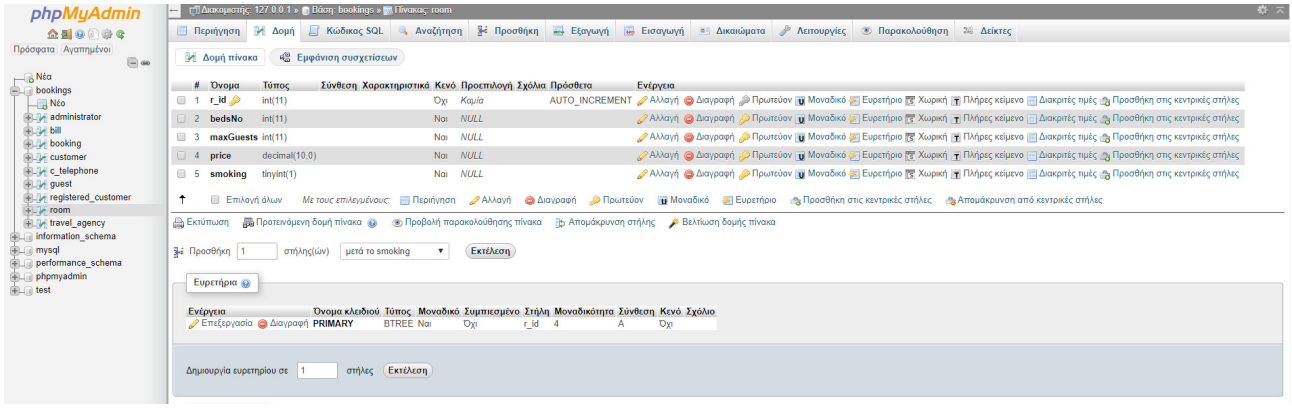
Πίνακας c_telephone στο PhpMyAdmin

```
CREATE TABLE bill (
    p_id int NOT NULL AUTO_INCREMENT,
    amount int NOT NULL,
    cardNo varchar(30),
    holdername varchar(30),
    expired date,
    c_id int NOT NULL,
    PRIMARY KEY (p_id),
    FOREIGN KEY (c_id) REFERENCES customer(c_id));
```



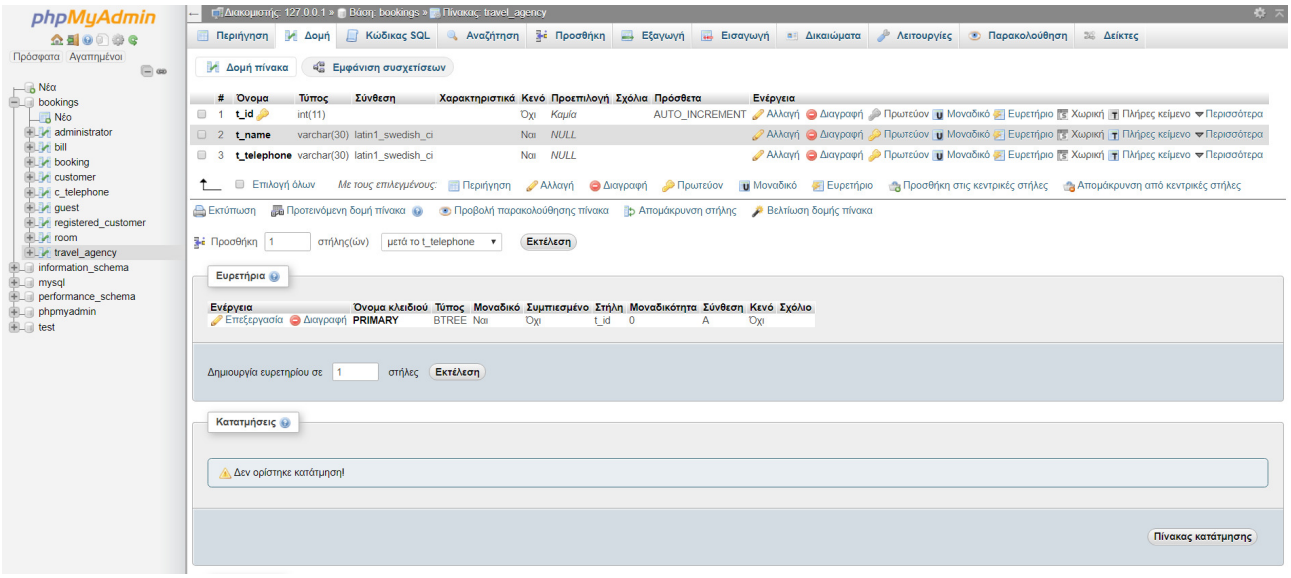
Πίνακας bill στο PhpMyAdmin

```
CREATE TABLE room(
    r_id int  NOT NULL AUTO_INCREMENT,
    bedsNo int,
    maxGuests int,
    price decimal,
    smoking  boolean,
    PRIMARY KEY (r_id)
);
```



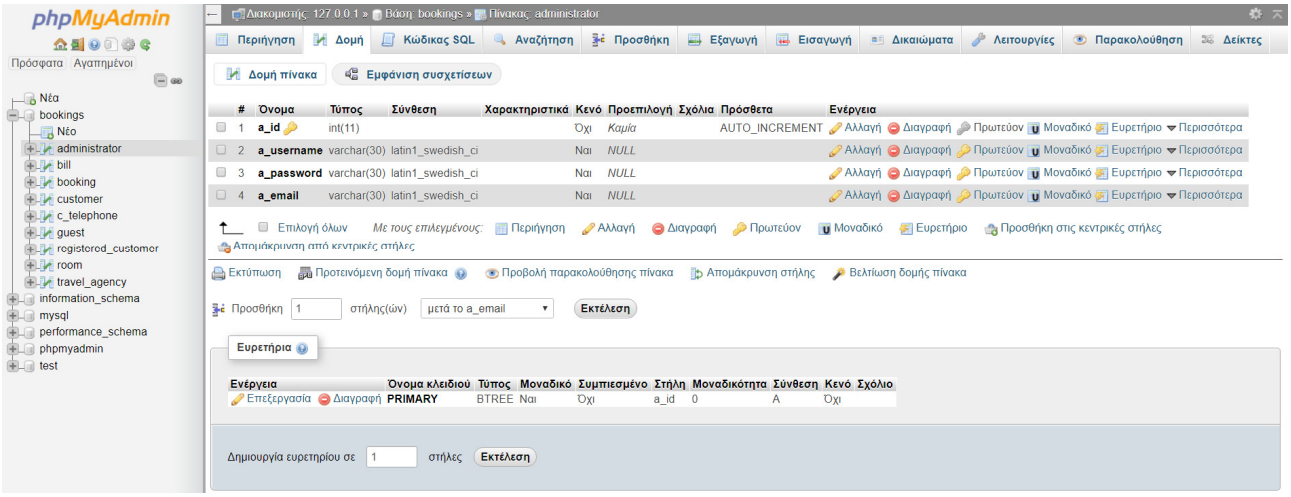
Πίνακας room στο PhpMyAdmin

```
CREATE TABLE travel_agency(
    t_id int NOT NULL AUTO_INCREMENT,
    t_name varchar(30),
    t_telephone varchar(30),
    PRIMARY KEY (t_id)
);
```



Πίνακας travel_agency στο PhpMyAdmin

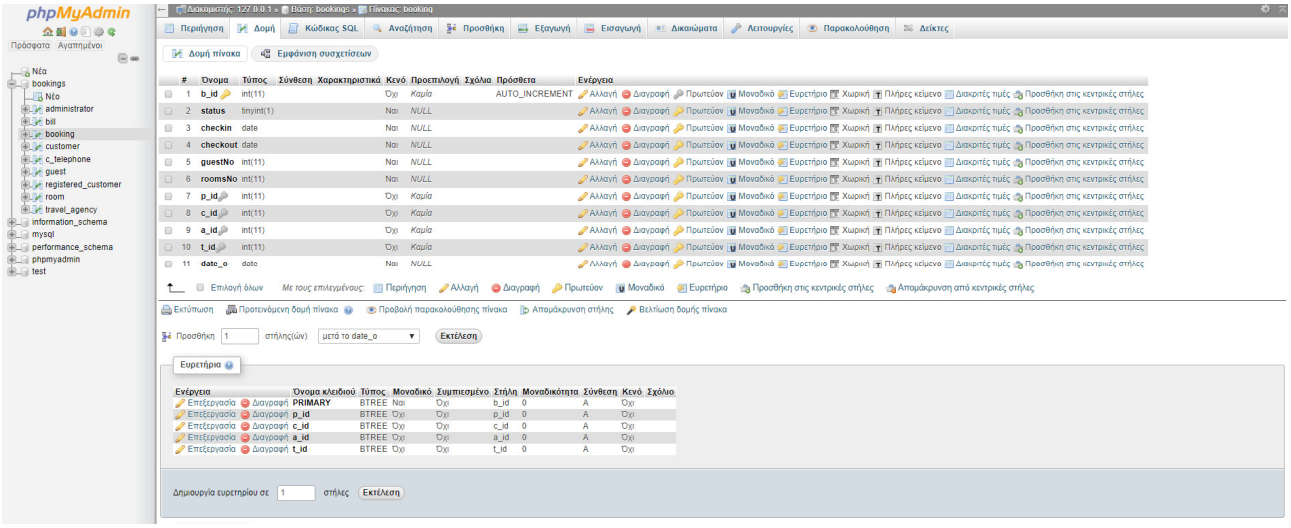
```
CREATE TABLE administrator(
    a_id int NOT NULL AUTO_INCREMENT,
    a_username varchar(30),
    a_password varchar(30),
    a_email varchar(30),
    PRIMARY KEY (a_id)
);
```



Πίνακας administrator στο PhpMyAdmin

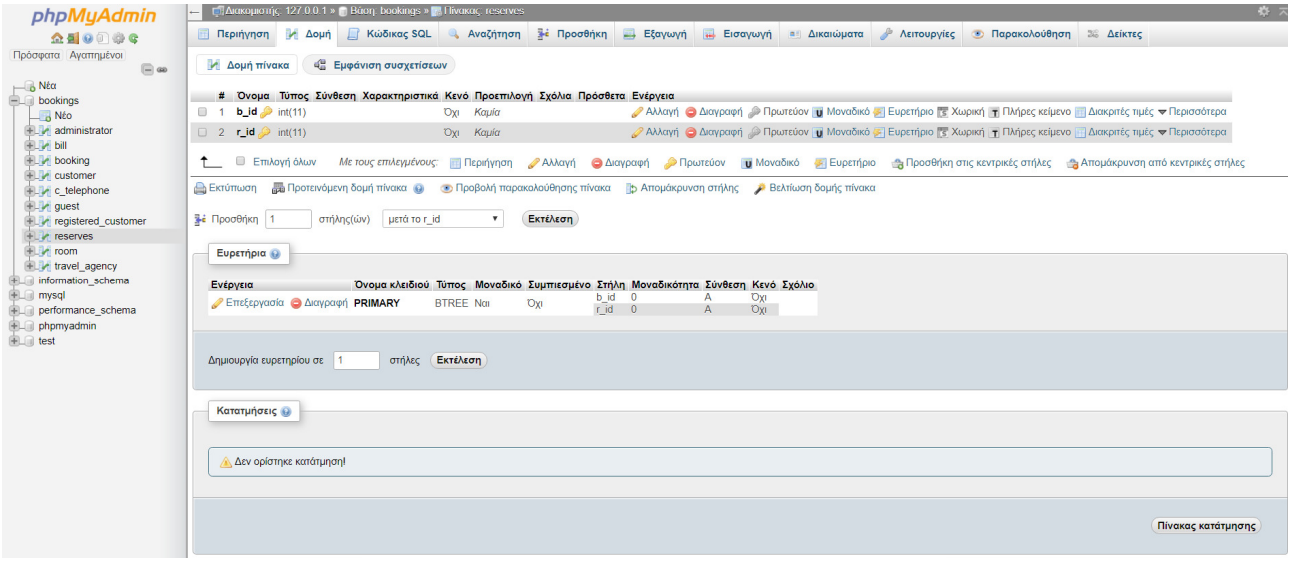
```
CREATE TABLE booking (
    b_id int NOT NULL AUTO_INCREMENT,
    status boolean,
    checkin date,
    checkout date,
    guestNo int,
    roomsNo int,
    p_id int NOT NULL,
    c_id int NOT NULL,
```

```
    a_id int NOT NULL,
    t_id int NOT NULL,
    date_o date,
    PRIMARY KEY (b_id),
    FOREIGN KEY (p_id) REFERENCES bill(p_id),
    FOREIGN KEY (c_id) REFERENCES customer(c_id),
    FOREIGN KEY (a_id) REFERENCES administrator(a_id),
    FOREIGN KEY (t_id) REFERENCES travel_agency(t_id)
);
```



Πίνακας booking στο PhpMyAdmin

```
CREATE TABLE reserves (
b_id int NOT NULL ,
r_id int NOT NULL ,
PRIMARY KEY (b_id, r_id)
);
```



Πίνακας reserves στο PhpMyAdmin

4.5.2. Ερωτήσεις προς τη βάση δεδομένων και Ενδεικτικά αποτελέσματα λειτουργίας

1. *Άνοιγμα Λογαριασμού Πελάτη*

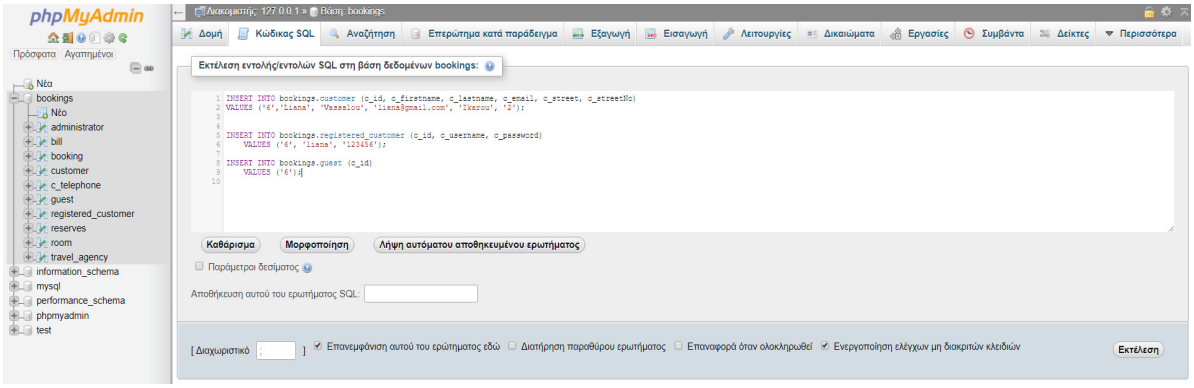
```
INSERT INTO bookings.customer (c_id, c_firstname, c_lastname, c_email,
c_street, c_streetNo)  VALUES ('6','Liana', 'Vassalou',
'liana@gmail.com', 'Ikarou', '2');
```

Άνοιγμα Λογαριασμού Εγγεγραμμένου Πελάτη (Εγγραφή ενός πελάτη στο σύστημα)

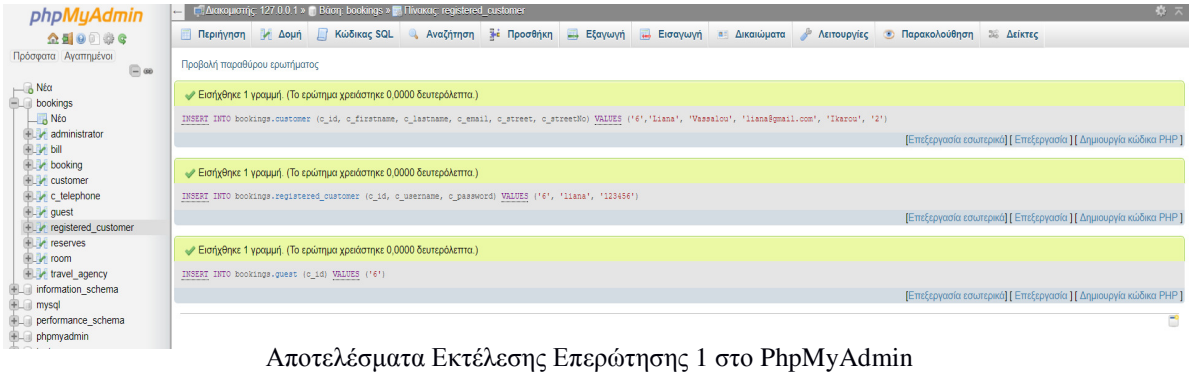
```
INSERT INTO bookings.registered_customer (c_id, c_username, c_password)
VALUES ('6', 'liana', '123456');
```

Άνοιγμα Λογαριασμού για Guest Πελάτη

```
INSERT INTO bookings.guest (c_id) VALUES ('6');
```



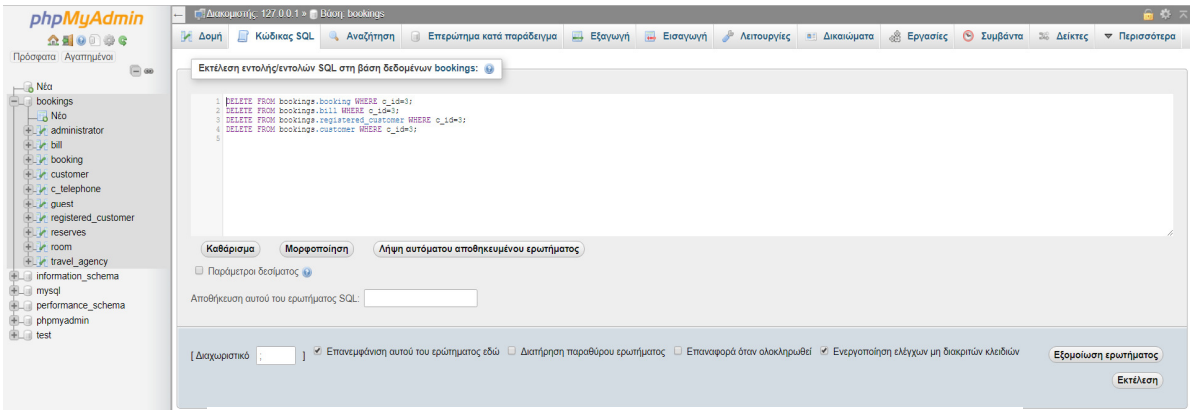
Στιγμιότυπο Εκτέλεσης Επερώτησης 1 στο PhpMyAdmin



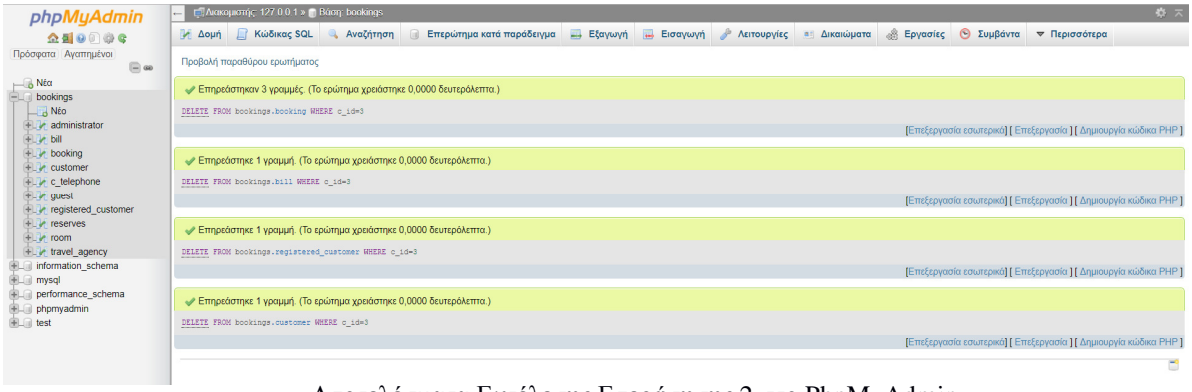
Αποτελέσματα Εκτέλεσης Επερώτησης 1 στο PhpMyAdmin

2. *Κλείσιμο Λογαριασμού Εγγεγραμμένου Πελάτη*

```
DELETE FROM bookings.booking WHERE c_id=3;
DELETE FROM bookings.bill WHERE c_id=3;
DELETE FROM bookings.registered_customer WHEREc_id=3;
DELETE FROM bookings.customer WHERE c_id=3;
```



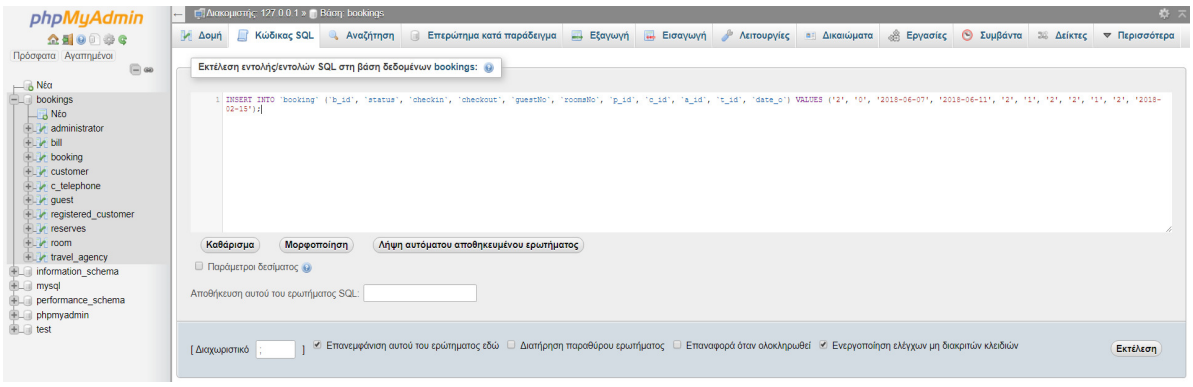
Στιγμιότυπο Εκτέλεσης Επερώτησης 2 στο PhpMyAdmin



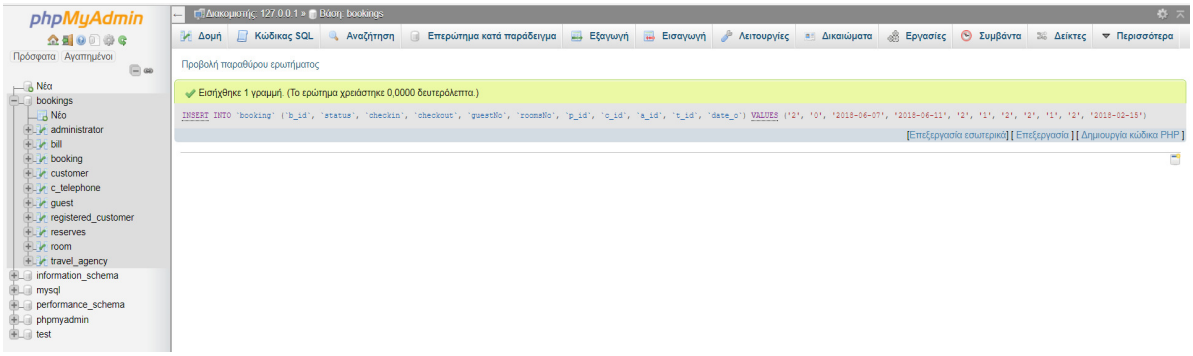
Αποτελέσματα Εκτέλεσης Επερώτησης 2 στο PhpMyAdmin

3. *Κράτηση δωματίου, ένας εγγεγραμμένος ή μη πελάτης να κάνει μία κράτηση δωματίου*

```
INSERT INTO `booking` (`b_id`, `status`, `checkin`, `checkout`,
`guestNo`, `roomsNo`, `p_id`, `c_id`, `a_id`, `t_id`, `date_o`)
VALUES ('2', '0', '2018-06-07', '2018-06-11', '2', '1', '2', '2',
'1', '2', '2018-02-15');
```



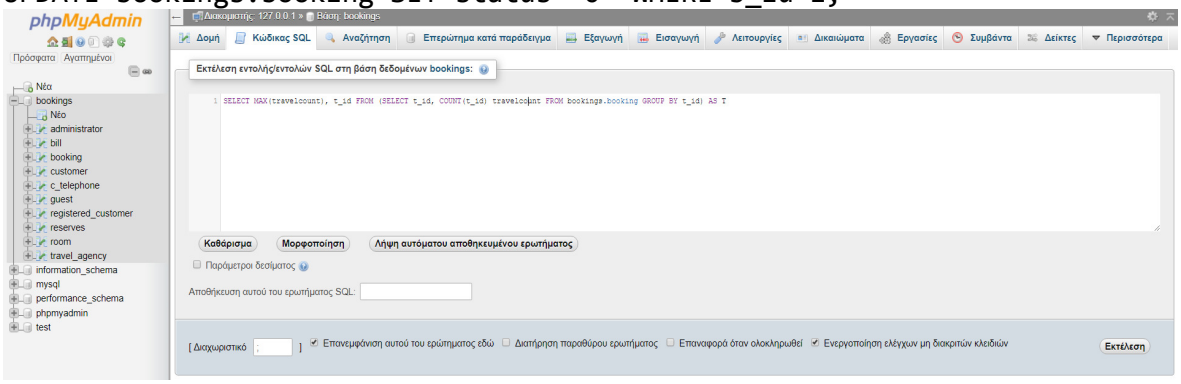
Στιγμιότυπο Εκτέλεσης Επερώτησης 3 στο PhpMyAdmin



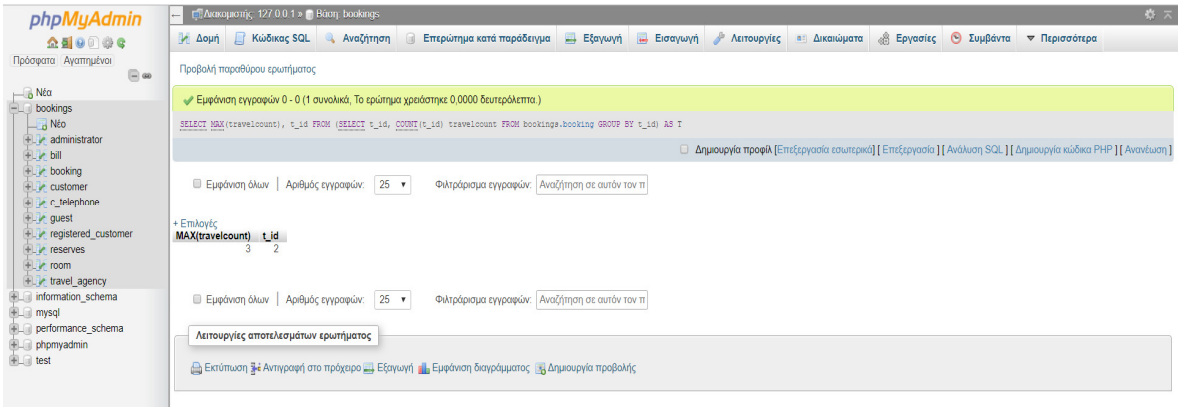
Αποτελέσματα Εκτέλεσης Επερώτησης 3 στο PhpMyAdmin

4. *Ακύρωση Κράτησης από πελάτη, ένας εγγεγραμμένος ή μη πελάτης αποφασίζει να ακυρώσει κάποια από τις υπάρχουσες ενεργές κρατήσεις του*

```
UPDATE bookings.booking SET status='0' WHERE b_id=1;
```



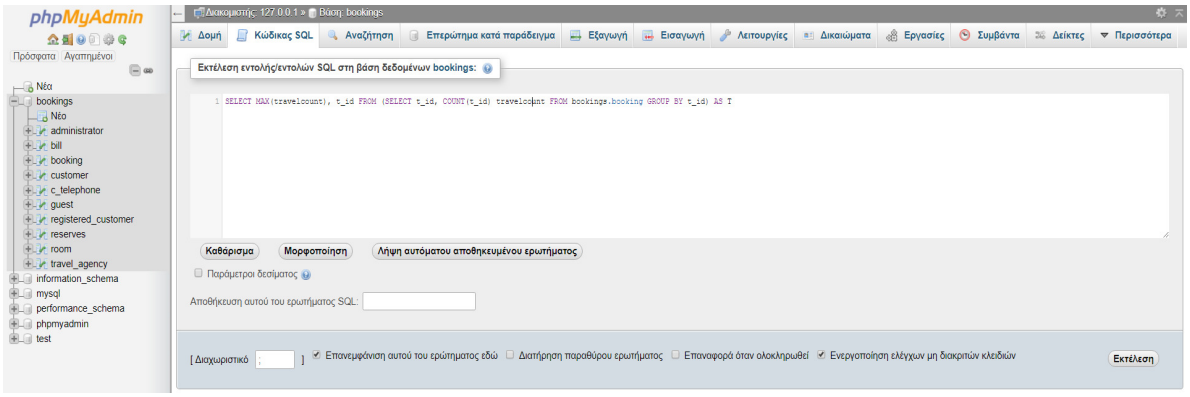
Στιγμιότυπο Εκτέλεσης Επερώτησης 4 στο PhpMyAdmin



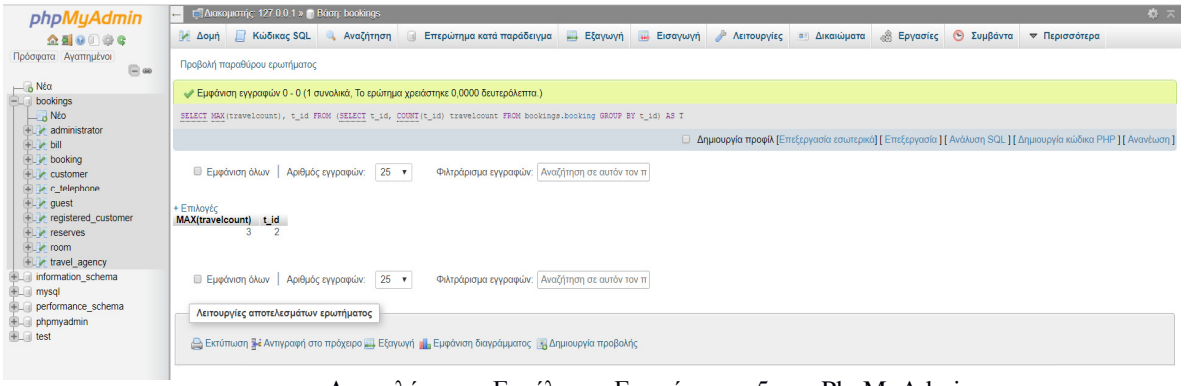
Αποτελέσματα Εκτέλεσης Επερώτησης 4 στο PhpMyAdmin

5. *Βρείτε το ταξιδιωτικό πρακτορείο με τις περισσότερες κρατήσεις*

```
SELECT MAX(travelcount), t_id FROM (SELECT t_id, COUNT(t_id)
travelcount FROM bookings.booking GROUP BY t_id) AS T
```



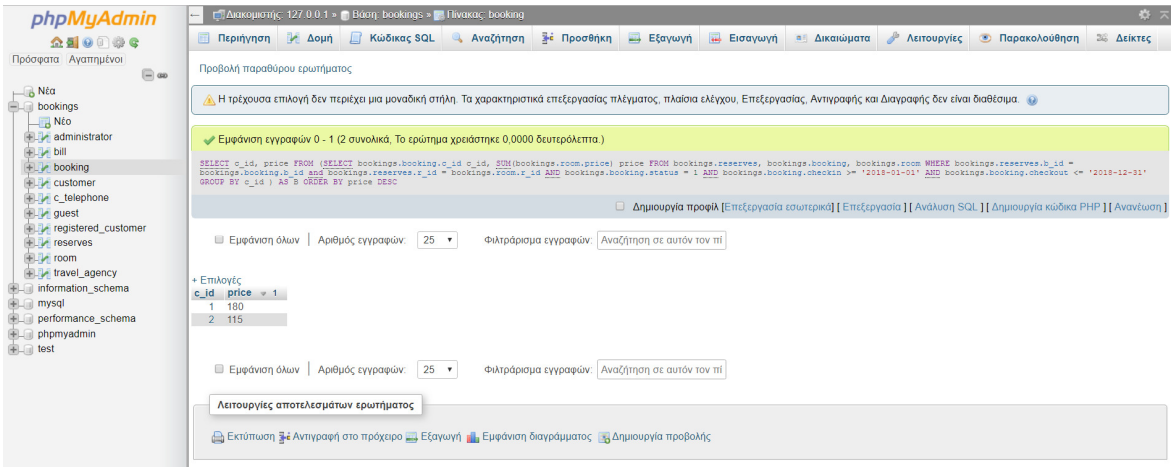
Στιγμιότυπο Εκτέλεσης Επερώτησης 5 στο PhpMyAdmin



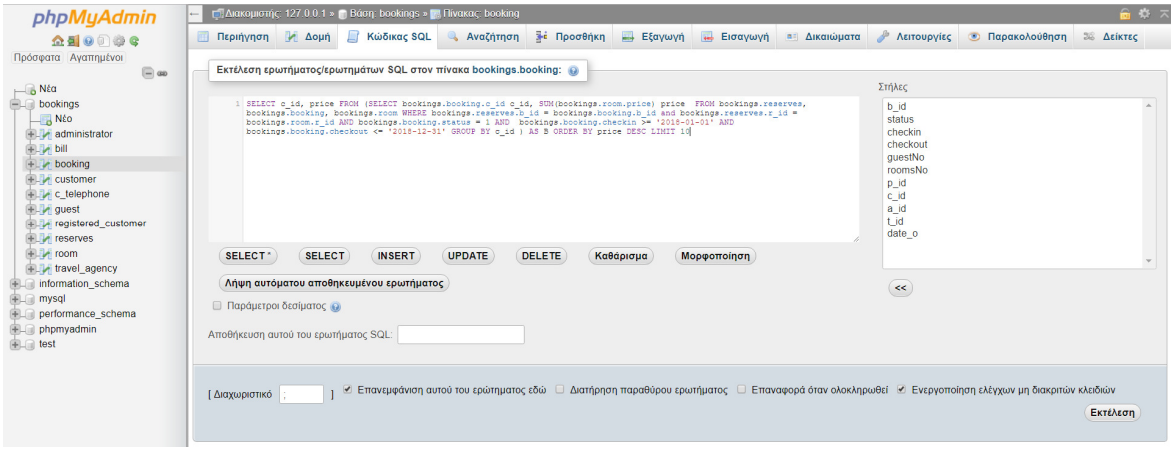
Αποτελέσματα Εκτέλεσης Επερώτησης 5 στο PhpMyAdmin

6. *Να βρεθεί η λίστα με τους καλύτερους πελάτες (οι δέκα καλύτεροι) για ένα συγκεκριμένο έτος. Ως καλύτεροι πελάτες θεωρούνται αυτοί που έχει κάνει συνολικά το μέγιστο «τζίρο» από ενεργές κρατήσεις*

```
SELECT c_id, price FROM (SELECT bookings.booking.c_id c_id,
SUM(bookings.room.price) price FROM bookings.reserves, bookings.booking,
bookings.room WHERE bookings.reserves.b_id = bookings.booking.b_id and
bookings.reserves.r_id = bookings.room.r_id AND bookings.booking.status =
1 AND bookings.booking.checkin >= '2018-01-01' AND
bookings.booking.checkout <= '2018-12-31' GROUP BY c_id ) AS B ORDER BY
price DESC LIMIT 10
```



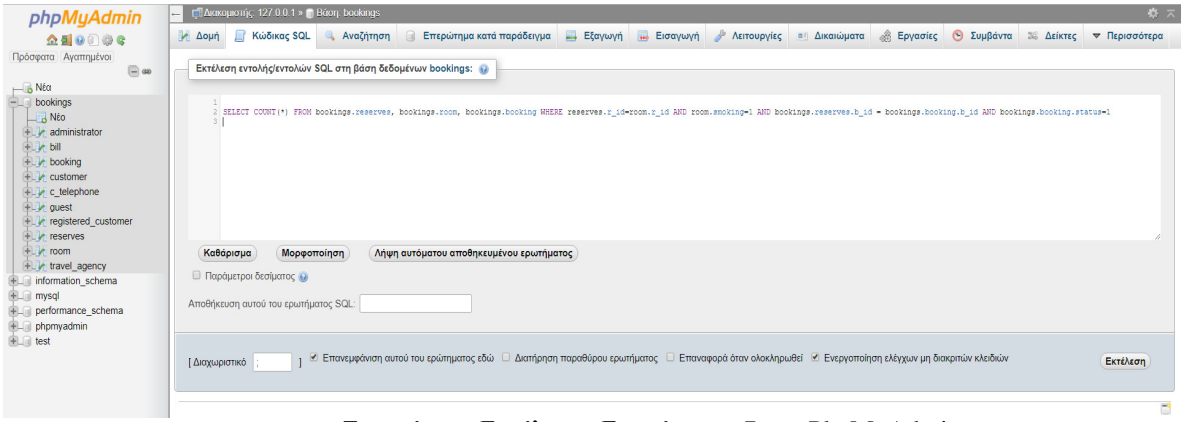
Στιγμιότυπο Εκτέλεσης Επερώτησης 6 στο PhpMyAdmin



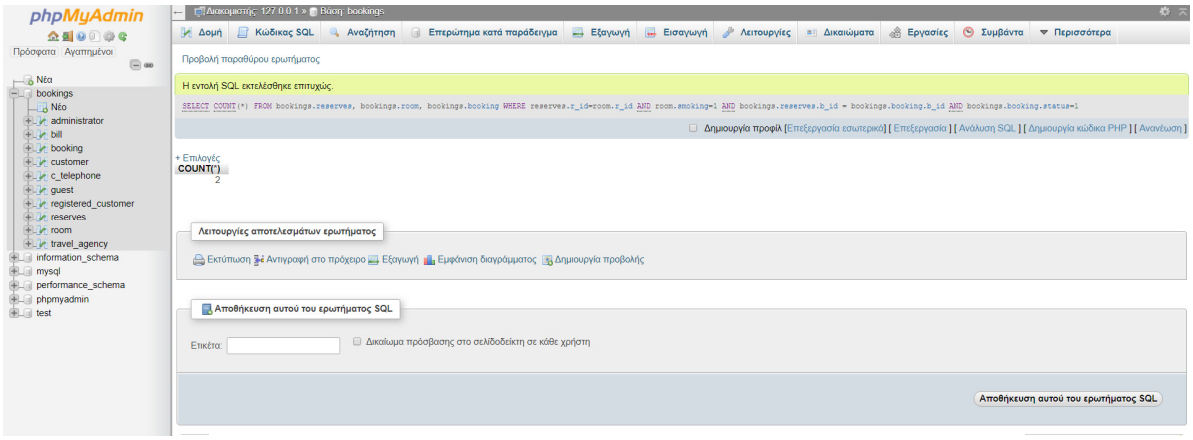
Αποτελέσματα Εκτέλεσης Επερώτησης 6 στο PhpMyAdmin

7. *Κρατήσεις σε δωμάτια που επιτρέπεται το κάπνισμα*

```
SELECT COUNT(*) FROM bookings.reserves, bookings.room, bookings.booking
WHERE reserves.r_id=room.r_id AND room.smoking=1 AND
bookings.reserves.b_id = bookings.booking.b_id AND
bookings.booking.status=1
```



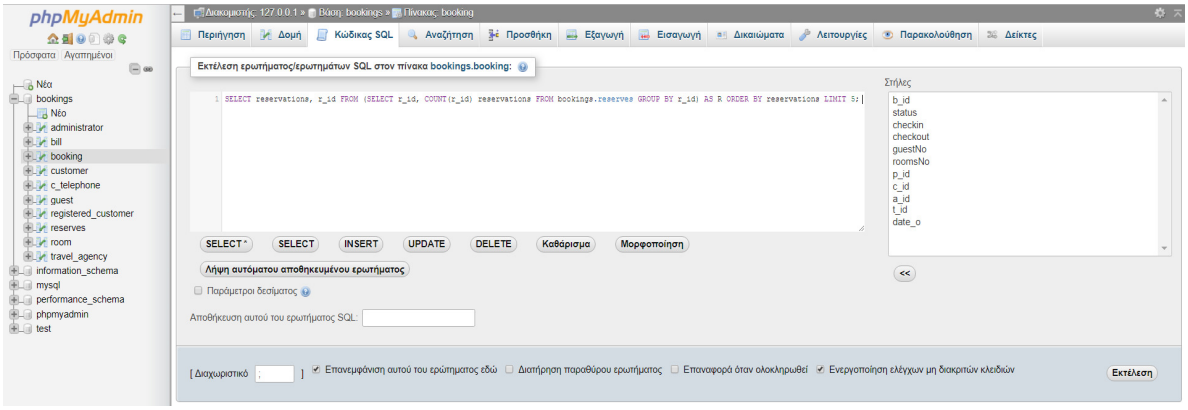
Στιγμιότυπο Εκτέλεσης Επερώτησης 7 στο PhpMyAdmin



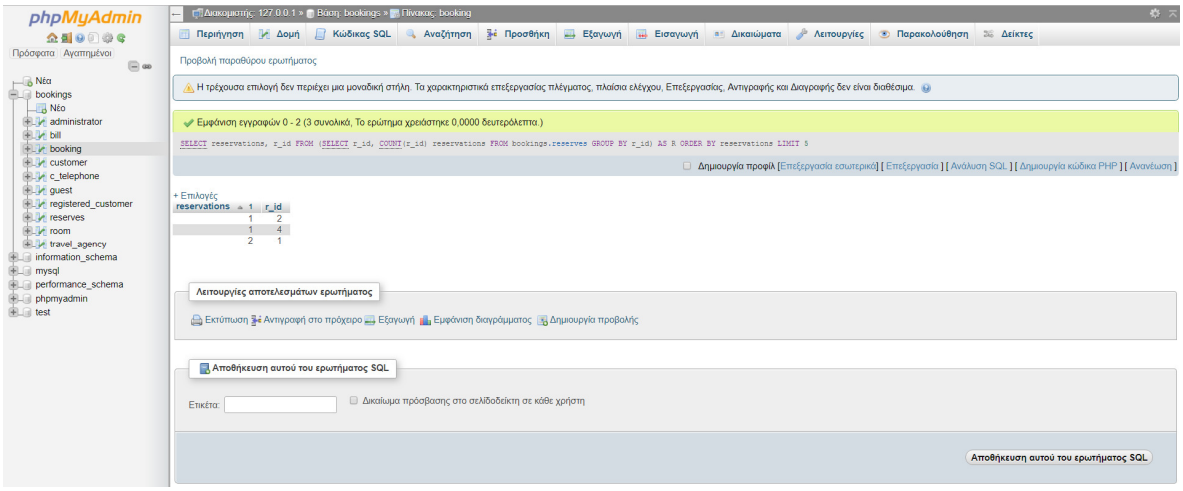
Αποτελέσματα Εκτέλεσης Επερώτησης 7 στο PhpMyAdmin

8. Κατάσταση δημοφιλών δωματίων (τα 5 πιο δημοφιλή), ποια δωμάτια κρατούνται πιο συχνά ταξινομημένα με βάση τον αριθμό των κρατήσεων που έχουν γίνει σε αυτά

```
SELECT reservations, r_id FROM (SELECT r_id, COUNT(r_id) reservations
FROM bookings.reserves GROUP BY r_id) AS R ORDER BY reservations LIMIT 5;
```



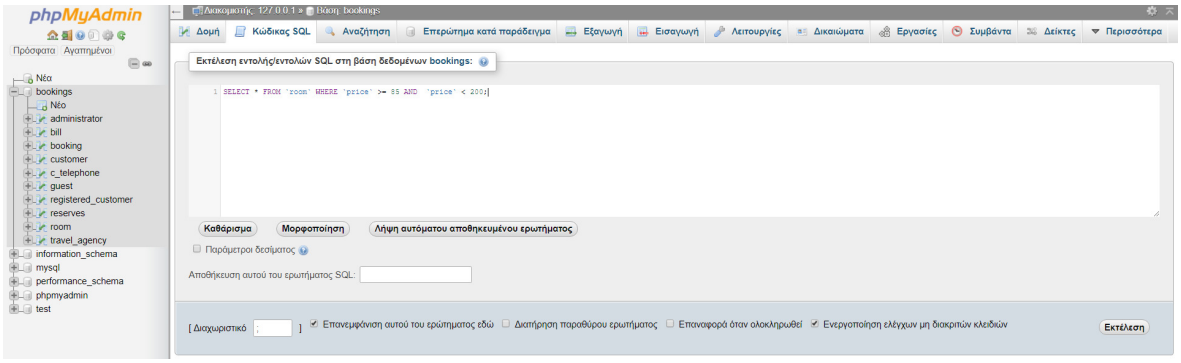
Στιγμιότυπο Εκτέλεσης Επερώτησης 8 στο PhpMyAdmin



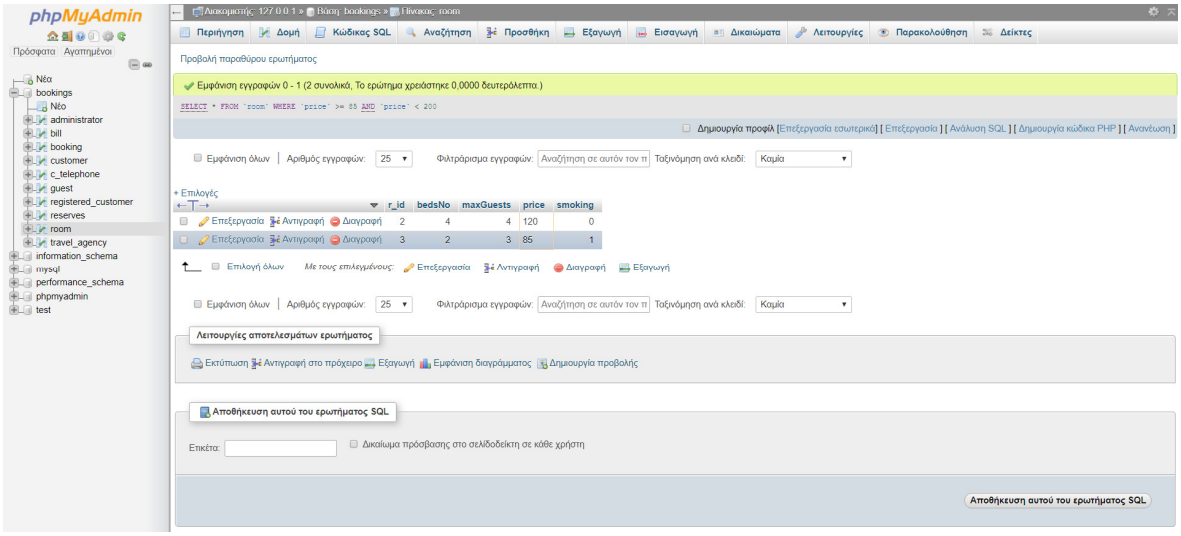
Αποτελέσματα Εκτέλεσης Επερώτησης 8 στο PhpMyAdmin

9. *Αναζήτηση δωματίων με βάση ένα συγκεκριμένο εύρος τιμών*

```
SELECT * FROM `room` WHERE `price` >= 85 AND `price` < 200
```



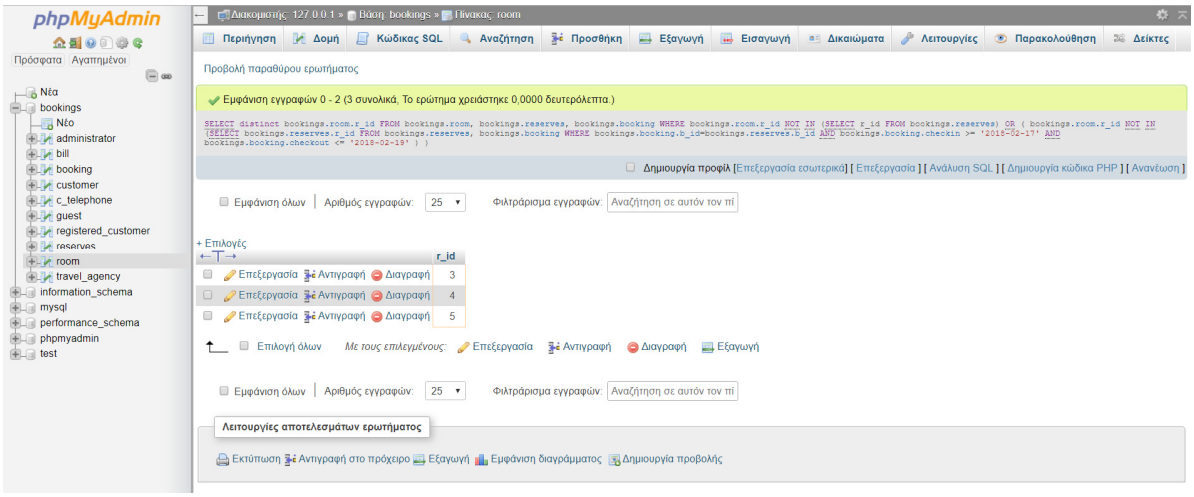
Στιγμιότυπο Εκτέλεσης Επερώτησης 9 στο PhpMyAdmin



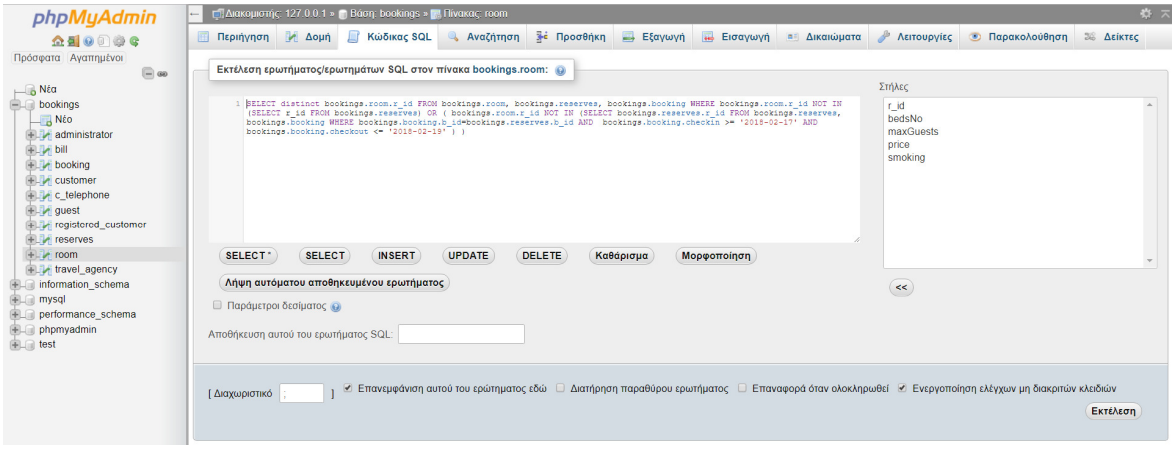
Αποτελέσματα Εκτέλεσης Επερώτησης 9 στο PhpMyAdmin

10. *Διαθεσιμότητα δωματίων για ένα συγκεκριμένο διάστημα*

```
SELECT distinct bookings.room.r_id FROM bookings.room, bookings.reserves,
bookings.booking WHERE bookings.room.r_id NOT IN (SELECT r_id FROM
bookings.reserves) OR ( bookings.room.r_id NOT IN (SELECT
bookings.reserves.r_id FROM bookings.reserves, bookings.booking WHERE
bookings.booking.b_id=bookings.reserves.b_id
AND  bookings.booking.checkin >= '2018-02-17' AND
bookings.booking.checkout <= '2018-02-19') )
```



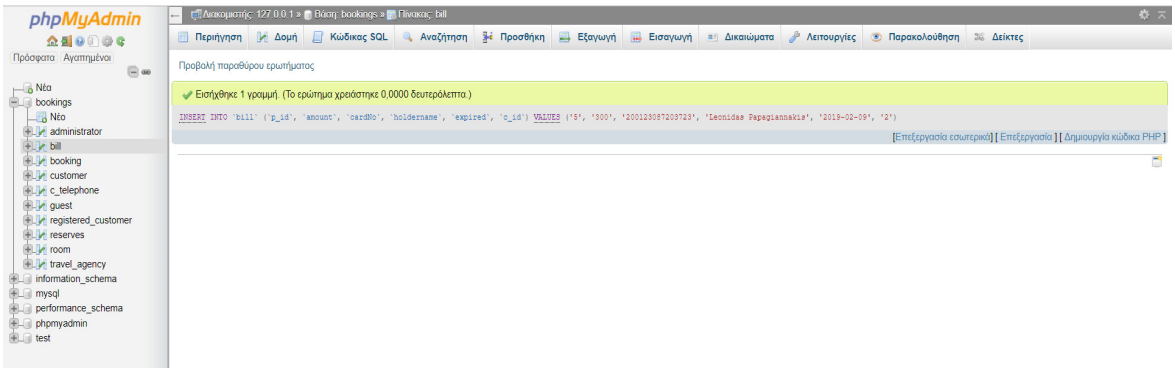
Στιγμιότυπο Εκτέλεσης Επερώτησης 10 στο PhpMyAdmin



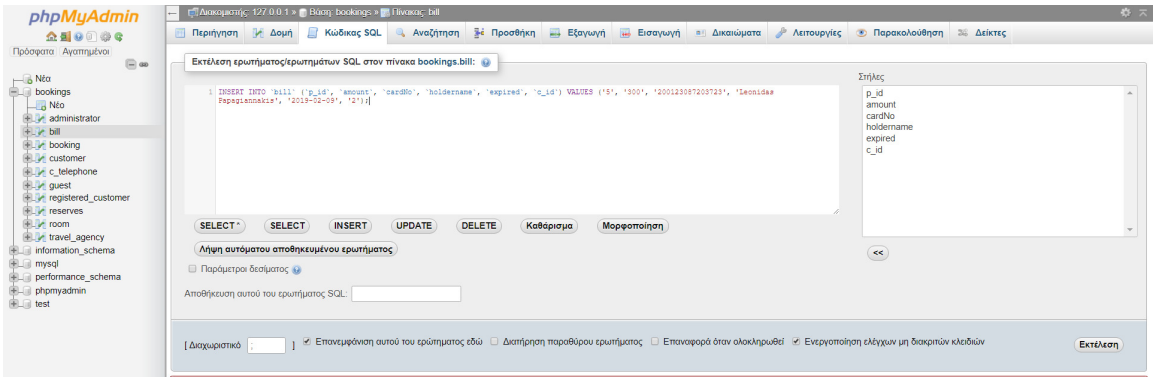
Αποτελέσματα Εκτέλεσης Επερώτησης 10 στο PhpMyAdmin

11. Πληρωμή κράτησης, ο πελάτης πληρώνει για την κράτηση την οποία έχει πραγματοποιήσει

```
INSERT INTO `bill` (`p_id`, `amount`, `cardNo`, `holdername`, `expired`,
`c_id`) VALUES ('5', '300', '200123087203723', 'Leonidas Papagiannakis',
'2019-02-09', '2')
```



Στιγμιότυπο Εκτέλεσης Επερώτησης 11 στο PhpMyAdmin



Αποτελέσματα Εκτέλεσης Επερώτησης 11 στο PhpMyAdmin

4.6. Βελτιώσεις

Μια δυνατότητα βελτίωσης, θα ήταν να μπορεί το σύστημα μας να υποστηρίζει την καταγραφή του ιστορικού των πελατών. Δηλαδή για κάθε δοσοληψία που πραγματοποιεί ένας εγγεγραμμένος χρήστης του συστήματος, όπως κράτηση, ακύρωση, πληρωμή, αλλαγή στοιχείων, να κρατιέται μια εγγραφή στη βάση μας. Όπως αναφέρθηκε και προηγουμένως, μια άλλη δυνατότητα θα ήταν η παραμετροποίηση των στοιχείων του λογαριασμού ενός πελάτη π.χ. η αλλαγή των προσωπικών του στοιχείων όπως email, διεύθυνση, τηλέφωνο, αριθμός πιστωτικής κάρτας κλπ.

Τέλος, ένας περιορισμός που έχει η υλοποίηση μας είναι ότι κατά την ακύρωση μιας κράτησης δεν υπάρχει επιστροφή χρημάτων στον πελάτη. Θα μπορούσαμε να ζητάμε από τους πελάτες να εισάγουν έναν αριθμό τραπεζικού λογαριασμού όπου σε περίπτωση ακύρωσης θα πιστωνόταν το αντίστοιχο ποσό.

5ο ΚΕΦΑΛΑΙΟ

ΙΣΤΟΣΕΛΙΔΑ BOOKINGS

5.1 Σύνδεση βάσης δεδομένων με την ιστοσελίδα και πληροφορίες για το πως θα εισέλθουμε η θα εξέλθουμε από την ιστοσελίδα

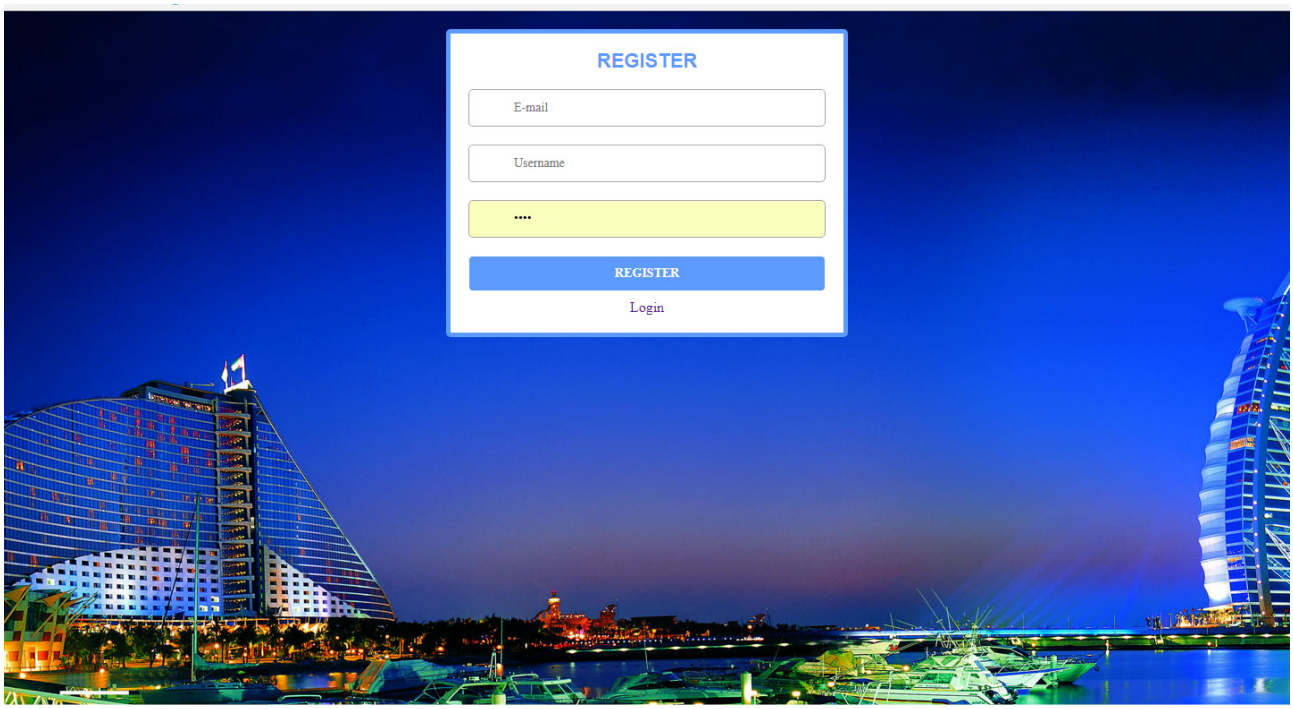
Προκειμένου να χρησιμοποιήσουμε την ιστοσελίδα μας θα πρέπει πρώτα να ενώσουμε την ιστοσελίδα μας με την βάση δεδομένων. Στην συγκεκριμένη περίπτωση έχουμε το αρχείο DB_config.php.

DB.config.php

```php
<?php

$dbServername="localhost";
$dbUsername="root";
$dbPassword="";
$dbName="bookings";

$conn=new mysqli($dbServername,$dbUsername,$dbPassword,$dbName);

if($conn->connect_error){
die("Connection failed: ".$conn->connect_error);
}
?>
```

Το Db_config.php ουσιαστικά παίρνει όλα τα στοιχεία από την βάση μας όπως είναι το όνομα του server , το username , το password και το όνομα της βάσης μας. Η επέκταση mysqli μας επιτρέπει να έχουμε πρόσβαση στη λειτουργικότητα που παρέχεται από την MySQL και μπορούμε κάνουμε την ένωση της βάσης με την ιστοσελίδα. Αν η ένωση αποτύχει μας πετάει μήνυμα “connection failed.”

Έχουμε δημιουργήσει μια πλατφόρμα για να μπορέσουμε να κάνουμε register. Σε αυτήν την πλατφόρμα έχουμε όλα τα πεδία για να κάνουμε register τα οποία είναι το mail,username και το password. Όταν συμπληρώσουμε τα πεδία τότε έχουμε κάνει register και πατάμε το κουμπί register. Αν έχουμε κάνει ποιο παλιά το register μας δεν χρειάζεται να το ξανακάνουμε απλά πατάμε στο πεδίο που λέγεται login και μας πάει κατευθείαν στην πλατφόρμα του login. Στην εικόνα που ακολουθεί μπορούμε να δούμε την πλατφόρμα του register.



Μπορούμε να δούμε και τον κώδικα που χρησιμοποιήθηκε για να δημιουργήσουμε το register.

register.func.php

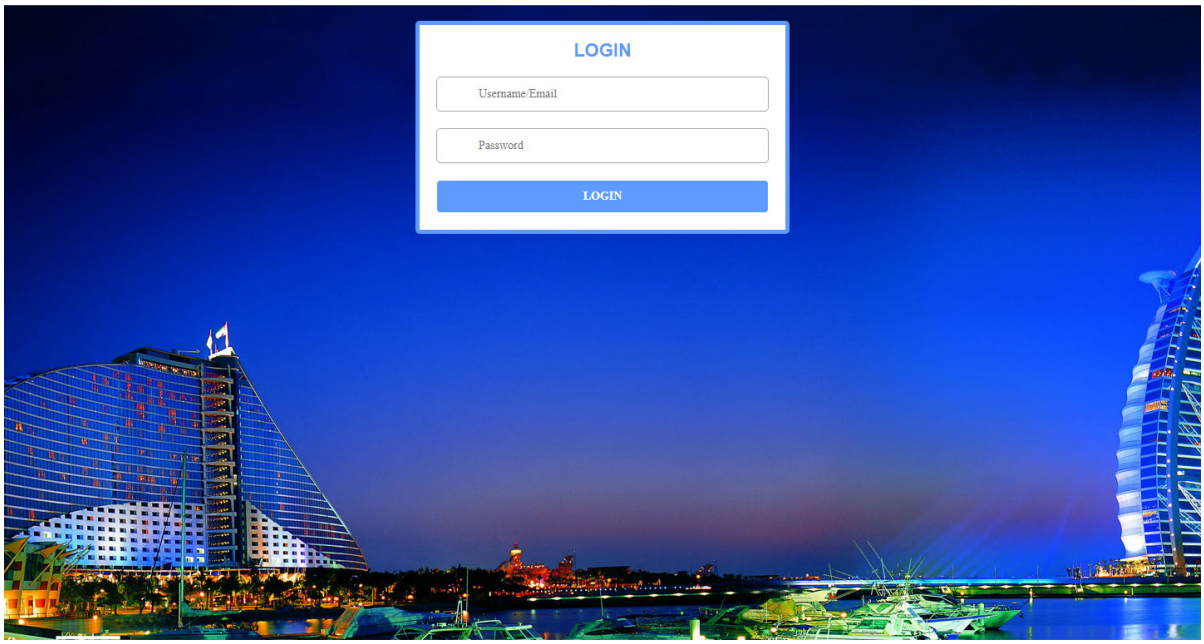
```php
<?php

if(isset($_POST['submit'])){

include_once'DB_config.php';

$email=mysqli_real_escape_string($conn,$_POST['email']);
$uid=mysqli_real_escape_string($conn,$_POST['uid']);
$pwd=mysqli_real_escape_string($conn,$_POST['pwd']);

//Empty Fields
if(empty($email)||empty($uid)||empty($pwd)){
header("Location: ../register.php?register=empty");
exit();
}else{

//Valid Email
if(!filter_var($email,FILTER_VALIDATE_EMAIL)){
header("Location: ../register.php?register=email");
exit();
}else{
$sql="SELECT * FROM administrator WHERE a_username='$uid';";
$result=mysqli_query($conn,$sql);
$resultCheck=mysqli_num_rows($result);

//Username Taken
if($resultCheck>0){
header("Location: ../register.php?register=usernametaken");
exit();
}else{
```

```
$sql="INSERT INTO administrator (a_email, a_username, a_password) VALUES ('$email',
'$uid', '$pwd');";
mysqli_query($conn,$sql);
header("Location: ../index.php?register=success");
exit();
}
}

}

}else{
header("Location: ../register.php");
exit();
}

?>
```

Αρχικά ενώνεται στην βάση μας μέσω του db_config στη συνέχεια παίρνουμε τα πεδια που θέλουμε για να μπορούμε να κάνουμε register που στην δικιά μας περίπτωση είναι το email,pwd και uid . τα αρχεία «καθαρίζονται» από κακόβουλους χαρακτήρες » με την βοήθεια της εντολής mysql_real_escape_string. Έπειτα Επαληθεύει εάν η τιμή είναι έγκυρη η διεύθυνση ηλεκτρονικού ταχυδρομείου μέσω του FILTER_VALIDATE_EMAIL αλλιώς παίρνει από την βάση δεδομένων μας τον πίνακα από τον οποίο θέλουμε να κάνουμε register στην ιστοσελίδας μας που στην προκιμενη περιπτωση είναι ο administrator.

```
<!doctype html>
<html>
<metaname="viewport"content="width=device-width, initial-scale=1">
<head>
<title></title>
<linkrel="stylesheet"type="text/css"href="style.css"></link>
</head>
<body>
<divclass="form-block">
<h1>Register</h1>
<formclass="register-form"action="func/register.func.php"method="POST">
<inputtype="text"name="email"placeholder="E-mail">
<inputtype="text"name="uid"placeholder="Username">
<inputtype="password"name="pwd"placeholder="Password">
<buttontype="submit"name="submit">Register</button>
<ahref="index.php">Login</a>
</form>
</div>
</body>
</html>
```

Έχουμε δημιουργήσει μια πλατφόρμα για να μπορούμε να συνδεθούμε στην ιστοσελίδα μας στην οποία μπορούμε να δούμε 2 πεδία τα οποία είναι: το όνομα και ο κωδικός πρόσβασης . αυτή η πλατφόρμα παίρνει τα στοιχεία από την βάση δεδομένων μας. Αν τα στοιχεία που θα βάλουμε είναι σωστά τότε θα εισέλθουμε κανονικά πατώντας το κουμπί “login” στην ιστοσελίδα μας αν όχι δεν θα μπορέσουμε να εισέλθουμε . Στην παρακάτω εικόνα μπορούμε να δούμε την πλατφόρμα που έχουμε δημιουργήσει.



Index.php

```
<!doctype html>
<html>
<metaname="viewport"content="width=device-width, initial-scale=1">
<head>
<title></title>
<linkrel="stylesheet"type="text/css"href="style.css"></link>
</head>
<body>
<divclass="form-block">
<h1>Login</h1>
<formaction="func/login.func.php"method="POST">
<inputtype="text"name="uid"placeholder="Username/Email">
<inputtype="password"name="pwd"placeholder="Password">
<buttontype="submit"name="submit">Login</button>
</form>

</div>
</body>
</html>
```

Για να μπορέσουμε να έχουμε γραφικό περιβάλλον για πεδίο του login μας ευθύνεται το index.php. Ενώνεται με το style.css και στην συνέχεια ενώνεται με το login.func.php για να μπορέσει να υλοποιηθεί η λειτουργία του login.

login.func.php

Για να μπορέσουμε να εισέλθουμε στο κύριο μέρος της ιστοσελίδας μας θα πρέπει να φτιάξουμε μια πλατφόρμα στην οποία θα πρέπει να τοποθετούμε το όνομα μας και τον κωδικό πρόσβασης μας . ο παρακάτω κώδικας παίρνει τα στοιχεία από την βάση δεδομένων και τα προσαρμόζει στα στοιχεία που ζητάει δηλαδή το όνομα και τον κωδικό πρόσβασης

```
<?php
session_start();

if(isset($_POST['submit'])){

include_once'DB_config.php';

$uid=mysqli_real_escape_string($conn,$_POST['uid']);
$pwd=mysqli_real_escape_string($conn,$_POST['pwd']);

if(empty($uid)||empty($pwd)){
```

```
header("Location: ../index.php?login=empty");
exit();
}else{
$sql="SELECT * FROM administrator WHERE a_username='$uid'";
$result=mysqli_query($conn,$sql);
$resultCheck=mysqli_num_rows($result);
if($resultCheck<1){
header("Location: ../index.php?login='$resultCheck'");
exit();
}else{
if($row=mysqli_fetch_assoc($result)){
if($pwd!=$row['a_password']){
header("Location: ../index.php?login=error");
exit();
}elseif($pwd==$row['a_password']){
$_SESSION['a_id']=$row['a_id'];
$_SESSION['a_email']=$row['user_email'];
$_SESSION['a_username']=$row['username'];
header("Location: ../home.php?login=success");
exit();
}
}
}
}
}else{
header("Location: ../index.php?login=error");
exit();
}

?>
```

Για να μπορούμε να αποσυνδεθούμε από την ιστοσελίδα μας έχουμε φτιάξει ένα κουμπί το οποίο μας αποσυνδέει . στην παρακάτω εικόνα μπορούμε να δούμε τον κώδικα που χρησιμοποιήσαμε για το συγκεκριμένο κουμπί.

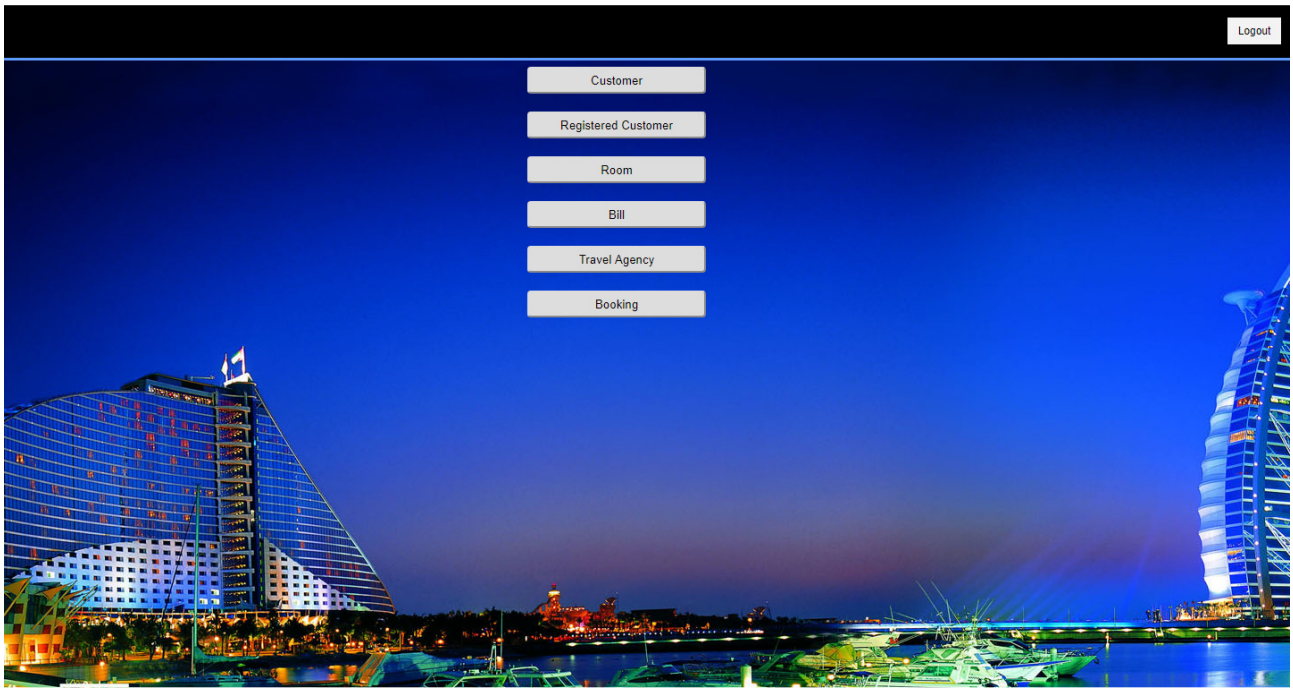
Logout.func.php

```
<?php

if(isset($_POST['submit'])){
session_start();
session_unset();
session_destroy();
header("Location: ../index.php");
exit();
}

?>
```

Αυτές είναι οι 4 μας απλές λειτουργίες ώστε να μπορούμε να συνδέσουμε την βάση δεδομένων μας με την ιστοσελίδα και έπειτα να μπορούμε να εισέλθουμε ή να αποσυνδεθούμε από την ιστοσελίδα μας

5.2 Το κύριο μέρος της ιστοσελίδα μας

Το κύριο μέρος της ιστοσελίδας αποτελείται από 7 κουμπιά όπως μπορούμε να δούμε στην εικόνα που ακολουθεί.

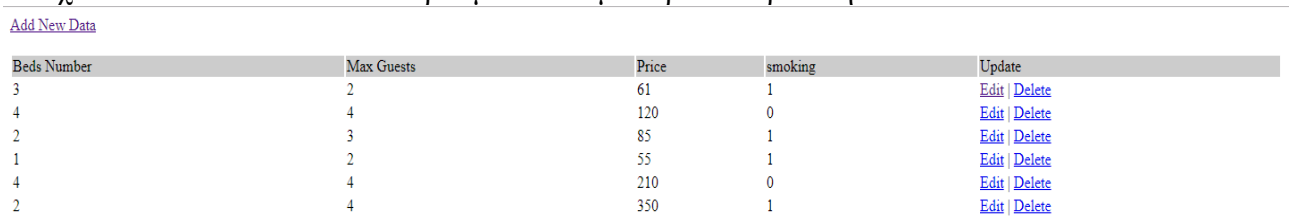


Στον κώδικα που ακολουθεί μπορούμε το πως δημιουργήσαμε τα κουμπιά

```
<?php
session_start();
if(!isset($_SESSION['a_id'])){
header("Location: ./index.php");
}
include_once"header.php"

?>

<scripttype="text/javascript"src="scripts/populateTables.js"></script>
<sectionclass="main-container">
<divclass="main-container button-grid">



<buttonstyle="margin:10px;padding:5px;"type="button"onclick="location.href='customer.php'
;">Customer</button>

<buttonstyle="margin:10px;padding:5px;"type="button"onclick="location.href='registeredcus
tomer.php';">Registered Customer</button>

<buttonstyle="margin:10px;padding:5px;"type="button"onclick="location.href='room.php';">R
oom</button>

<buttonstyle="margin:10px;padding:5px;"type="button"onclick="location.href='bill.php';">B
ill</button>

<buttonstyle="margin:10px;padding:5px;"type="button"onclick="location.href='travelagency.
php';">Travel Agency</button>

<buttonstyle="margin:10px;padding:5px;"type="button"onclick="location.href='booking.php';
">Booking</button>



</div>
</div>
```

```
</section>



<sectionclass="main-container">

</section>

<?php
include_once"footer.php"
?>
```

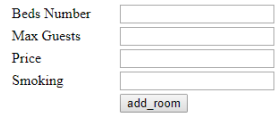
Αν πατήσουμε για παράδειγμα το κουμπί rooms τότε μπορούμε να δούμε αναλυτικά όλα τα στοιχεία του πίνακα rooms. Μπορούμε να δούμε περισσότερα στην εικόνα που ακολουθεί.



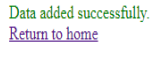
Add New Data

Beds Number	Max Guests	Price	smoking	Update
3	2	61	1	Edit | Delete
4	4	120	0	Edit | Delete
2	3	85	1	Edit | Delete
1	2	55	1	Edit | Delete
4	4	210	0	Edit | Delete
2	4	350	1	Edit | Delete

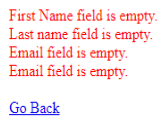
Αν πατήσουμε το "AddNewData" τότε μπορούμε να προσθέσουμε νέα δωμάτια στην βάση δεδομένων μας αφού υπάρχουν όλα τα πεδία του πίνακα room . όταν τα συμπληρώσουμε πατάμε το κουμπί "add_room"και προσθέτουμε το δωμάτιο . όταν πατήσουμε το κουμπί add_rooms με συμπληρωμένα όλα τα πεδία το θα δούμε ότι μας γράφει "Dataaddedsuccessfully"με πράσινα γράμματα ,αν δεν τα συμπληρώσουμε τα πεδία τότε θα μας γράψει ποια πεδία θα πρέπει να συμπληρωθούν με κόκκινα γράμματα. περισσότερα μπορούμε να δούμε στην εικόνες που ακολουθούν.



Beds Number
Max Guests
Price
Smoking
add_room



Data added successfully.
Return to home



First Name field is empty.
Last name field is empty.
Email field is empty.
Email field is empty.

Go Back

Αν θέλουμε απλά να τροποποιήσουμε το δωμάτιο τότε επιλέγουμε ποιο δωμάτιο θέλουμε και πατάμε το κουμπί edit.

Για να χρησιμοποιήσουμε όλες τις λειτουργίες του κουμπιού rooms δημιουργήσαμε 5 αρχεία τα οποία είναι τα εξής:

1. Room.php
2. Edit_room.php
3. Add_room.php
4. Add_room.html
5. Delete_room.php

Ας δούμε αναλυτικά τα αρχεία.

Room.php

```
<?php


$connection=mysqli_connect('localhost','root','');
mysqli_select_db($connection,'bookings');

$sql="select * from room";
$result=mysqli_query($connection,$sql);



?>

<html>
<head>
<title>Homepage</title>
</head>

<body>
<ahref="add_room.html">Add New Data</a><br/><br/>
<tablewidth='100%'border=0>
<trbgcolor='#CCCCCC'>
<td>Beds Number</td>
<td>Max Guests</td>
<td>Price</td>
<td>smoking</td>
<td>Update</td>
</tr>
<?php
//while($res = mysql_fetch_array($result)) { // mysql_fetch_array is deprecated, we need
to use mysqli_fetch_array
while($res=mysqli_fetch_array($result)){
echo"<tr>";
echo"<td>".$res['bedsNo']."</td>";
echo"<td>".$res['maxGuests']."</td>";
echo"<td>".$res['price']."</td>";
echo"<td>".$res['smoking']."</td>";
echo"<td><a href=\"edit_room.php?id=$res[r_id]\">Edit</a> |
<a href=\"delete_room.php?id=$res[r_id]\" onClick=\"return confirm('Are you sure you want
to delete?')\">Delete</a></td>";

}
?>
</table>
</body>
</
```

Edit_room.php

```
<?php
// including the database connection file
include_once("func/DB_config.php");
$connection=mysqli_connect('localhost','root','');
mysqli_select_db($connection,'bookings');

$sql="select * from bill ";
$result=mysqli_query($connection,$sql);

if(isset($_POST['update']))
{
$amount=$_POST['amount'];
$cardNo=$_POST['cardNo'];
$holdername=$_POST['holdername'];
$expired=$_POST['expired'];
$p_id=$_POST['p_id'];

// checking empty fields
if(empty($amount)||empty($cardNo)||empty($holdername)||empty($expired)){
if(empty($amount)){
echo"<font color='red'>amount field is empty.</font><br/>";
}

if(empty($cardNo)){
echo"<font color='red'>cardNo field is empty.</font><br/>";
}

if(empty($holdername)){
echo"<font color='red'>holdername field is empty.</font><br/>";
}

if(empty($expired)){
echo"<font color='red'>expired field is empty.</font><br/>";
}
}else{
//updating the table
$sqlq="UPDATE bill SET
amount='".$amount."',cardNo='".$cardNo."',holdername='".$holdername."',expired='".$expire
d."' WHERE p_id=".$p_id;
$result=mysqli_query($connection,$sqlq);

//redirectig to the display page. In our case, it is index.php
header("Location: home.php");
}
}
?>
<?php
//getting id from url
$p_id=$_GET['id'];

//selecting data associated with this particular id
$result=mysqli_query($connection,"SELECT * FROM bill WHERE p_id=".$p_id);

while($res=mysqli_fetch_array($result))
{
$amount=$res['amount'];
$cardNo=$res['cardNo'];
$holdername=$res['holdername'];
}$expired=$res['expired'];
?>
<html>
<head>
<title>Edit Data</title>
```

```
</head>

<body>
<ahref="index.php">Home</a>
<br/><br/>

<formname="form1"method="post"action="edit_bill.php">
<tableborder="0">
<tr>
<td>Amount</td>
<td><inputtype="text"name="amount"value="<?phpecho$amount;?>"></td>
</tr>
<tr>
<td>Card Number</td>
<td><inputtype="text"name="cardNo"value="<?phpecho$cardNo;?>"></td>
</tr>
<tr>
<td>Holder Name</td>
<td><inputtype="text"name="holdername"value="<?phpecho$holdername;?>"></td>
</tr>
<tr>
            <td>Expired</td>
      <td><inputtype="text"name="expired"value="<?phpecho$expired;?>"></td>
</tr>
<tr>
<td><inputtype="hidden"name="p_id"value=<?phpecho$_GET['id'];?>></td>
<td><inputtype="submit"name="update"value="Update"></td>
</tr>
</table>
</form>
</body>
</html>
```

Το edit_room ουσιαστικά κοιταει ολες τις παραμετρους από την βαση δεδομενων μας ετσι ώστε να μπορεσουμε να τροποποιησουμε το δωματιο που θελουμε.

Add_room.php

```
<html>
<head>
<title>Add Data</title>
</head>

<body>
<?php
$connection=mysqli_connect('localhost','root','');
mysqli_select_db($connection,'bookings');
//including the database connection file
include_once("func/DB_config.php");

if(isset($_POST['Submit'])){
$bedsNo=$_POST['bedsNo'];
$maxGuests=$_POST['maxGuests'];
$price=$_POST['price'];
$smoking=$_POST['smoking'];

// checking empty fields
if(empty($bedsNo)||empty($maxGuests)||empty($price)||empty($smoking)){
if(empty($bedsNo)){
echo"<font color='red'>First Name field is empty.</font><br/>";
}

if(empty($maxGuests)){
echo"<font color='red'>Last name field is empty.</font><br/>";
}
```

```
if(empty($price)){
echo"<font color='red'>Email field is empty.</font><br/>";
}
if(empty($$smoking)){
echo"<font color='red'>Email field is empty.</font><br/>";
}

//link to the previous page
echo"<br/><a href='javascript:self.history.back();'>Go Back</a>";
}else{
// if all the fields are filled (not empty)
//insert data to database
$result=mysqli_query($connection,"INSERT INTO room(bedsNo,maxGuests,price,smoking)
VALUES('$bedsNo','$maxGuests','$price','$smoking')");

//display success message
echo"<font color='green'>Data added successfully.";
echo"<br/><a href='home.php'>Return to home</a>";
}
}
?>
</body>
</html>
```

Add_room.html

```
<html>
<head>
<title>Add Data</title>
</head>

<body>
<ahref="room.php">Home</a>
<br/><br/>

<formaction="add_room.php"method="post"name="form1">
<tablewidth="25%"border="0">
<tr>
<td>Beds Number</td>
<td><inputtype="text"name="bedsNo"></td>
</tr>
<tr>
<td>Max Guests </td>
<td><inputtype="text"name="maxGuests"></td>
</tr>
<tr>
<td>Price</td>
<td><inputtype="text"name="price"></td>
</tr>
<tr>
<td>Smoking</td>
<td><inputtype="text"name="smoking"></td>
</tr>
<tr>
<td></td>
<td><inputtype="submit"name="Submit"value="add_room"></td>
</tr>
</table>
</form>
</body>
</html>
```

Delete_room.html

```
<html>
<body>


<?php

//including the database connection file
$connection=mysqli_connect('localhost','root','');
mysqli_select_db($connection,'bookings');
include("func/DB_config.php");

//getting id of the data from url
$id=$_GET['id'];

//deleting the row from table
$result=mysqli_query($conn,"DELETE FROM room WHERE r_id=".$id);

//redirecting to the display page (index.php in our case)
header("Location:home.php");
?>
</body>
</html>
```

Μέσω του delete_room.html μπορούμε να διαγράψουμε ότι δεν χριζόμαστε από τον πίνακα room. Αρχικά από ότι βλέπουμε ότι συνδέεται στην βάση μας μέσω του include στην συνέχεια παίρνει το id από το url και έπειτα διαγραφεί την γραμμή που δεν θέλουμε .

Style.css

Με το css ουσιαστικά μπορούμε να τροποποιήσουμε το γραφικό μας περιβάλλον. Έχουμε μιλήσει αναλυτικά στο δεύτερο κεφάλαιο (2.3)

```
/* RESET CODE */
html,body,div,span,object,iframe,
h1,h2,h3,h4,h5,h6,p,blockquote,pre,
abbr,address,cite,code,
del,dfn,em,img,ins,kbd,q,samp,
small,strong,sub,sup,var,
b,i,
dl,dt,dd,ol,ul,li,
fieldset,form,label,legend,
table,caption,tbody,tfoot,thead,tr,th,td,
article,aside,canvas,details,figcaption,figure,
footer,header,hgroup,menu,nav,section,summary,
time,mark,audio,video {
margin:0;
padding:0;
border:0;
outline:0;
font-size:100%;
vertical-align:baseline;
background:transparent;
}

body {
line-height:1;
```

```
}

article,aside,details,figcaption,figure,
footer,header,hgroup,menu,nav,section {
display:block;
}

nav ul {
list-style:none;
}

blockquote,q {
quotes:none;
}

blockquote:before,blockquote:after,
q:before,q:after{
content:'';
content:none;
}

a {
margin:0;
padding:0;
font-size:100%;
vertical-align:baseline;
background:transparent;
}

/* change colours to suit your needs */
ins {
background-color:#ff9;
color:#000;
text-decoration:none;
}

/* change colours to suit your needs */
mark {
background-color:#ff9;
color:#000;
font-style:italic;
font-weight:bold;
}

del {
text-decoration: line-through;
}

abbr[title],dfn[title]{
border-bottom:1px dotted;
cursor:help;
}

table {
border-collapse:collapse;
border-spacing:0;
}

/* change border colour to suit your needs */
hr {
display:block;
height:1px;
border:0;
border-top:1px solid #cccccc;
```

```
margin:1em 0;
padding:0;
}

input,select {
vertical-align:middle;
}
/* END OF RESET CODE */

/*-------------------------------------------------------------*/


a {
margin-top: 10px;
display: block;
text-align: center;
text-decoration: none;
}

body {
background-color: #ffffff;
background-image: url("./images/1.jpg");
background-repeat: no-repeat;
background-position: center center;
background-attachment: fixed;
}

.topnav{
overflow: hidden;
position: relative;
margin: 0 auto;
width: 100%;
font-family: helvetica, sans-serif;
}

header {
vertical-align: center;
z-index: 2;
position: fixed;
width: 100%;
height: 60px;
line-height: 60px;
background: #000;
color: white;
}

header nav {
width: 100%;
height: 60px;
background-color: #000;
border-bottom: #5e9bff 3px solid;
}

header nav ul {
display: block;
text-align: center;
float: left;
}

header nav ul li {
margin: 0px 9px;
float: left;
list-style: none;
}
```

```
header nav ul li a {
display: block;
font-family: arial;
font-size: 16px;
color: #fff;
line-height: 60px;
}

header nav ul li a:hover{
color: #5e9bff;
}

header .nav-login{
float: right;
}

header .nav-login form {
float: left;
padding-top: 15px;
}

header .nav-login form input {
float: left;
width: 100px;
height: 30px;
padding: 0px 10px;
margin-right: 10px;
border: none;
background-color: #ccc;
font-family: arial;
font-size: 12px;
color: #111;
}

header .nav-login form button {
float: left;
width: 60px;
height: 30px;
margin-right: 10px;
border: none;
background-color: #f3f3f3;
font-size: 12px;
cursor: pointer;
}

header .nav-login form button:hover{
background-color: #ccc;
}

header .nav-login a {
margin: 0px 10px;
float: left;
display: block;
width: 60px;
height: 60px;
border: none;
background-color: #f4f4f4;
cursor: pointer;
font-family: arial;
font-size: 16px;
color: #111;
line-height: 60px;
}

.main-container{
```

```
padding-top: 30px;
}

.main-container h2 {
line-height: 50px;
padding-top: 30px;
font-family: arial;
font-size: 40px;
color: #111;
text-align: center;
}

/* Login form */

.form-block{
max-width: 400px;
padding: 20px;
background: #fff;
border-radius: 5px;
border: 5px solid #5e9bff;
margin: 20px auto;
}

.form-block h1 {
font-family: helvetica;
text-align: center;
color: #5e9bff;
font-size: 22px;
text-transform: uppercase;
margin-top: 0;
margin-bottom: 20px;
}

.form-block input {
width: 100%;
height: 42px;
box-sizing: border-box;
border-radius: 5px;
border: 1px solid #a7a7a7;
margin-bottom: 20px;
font-size: 14px;
font-family: Montserrat;
padding: 0 20px 0 50px;
outline: none;
}

.form-block input:active, .form-block input:focus{
border: 3px solid #5e9bff;
}

.form-block button {
width: 100%;
height: 40px;
background: #5e9bff;
box-sizing: border-box;
border-radius: 5px;
border: 1px solid;
color: #fff;
font-weight: bold;
text-transform: uppercase;
font-size: 14px;
font-family: Montserrat;
outline: none;
cursor: pointer;
}
```

```
.form-block button:hover{
background: #ff7b81;
}

header nav {
width: 100%;
float: none;
}

header nav ul li {
font-size: 4px;
}

.button-grid{
width:20%;
margin: 0 auto;
}

.button-grid button {
float: left;
width: 200px;
height: 30px;
margin: 0 1px;
border-radius:4px;
}
```

Όλα τα υπόλοιπα αρχεία php δημιουργήθηκαν με βάση το παράδειγμα rooms.

Bill.php

```
<?php


$connection=mysqli_connect('localhost','root','');
mysqli_select_db($connection,'bookings');

$sql="select * from bill";
$result=mysqli_query($connection,$sql);



?>

<html>
<head>
<title>Homepage</title>
</head>

<body>
<ahref="add_bill.html">Add New Data</a><br/><br/>
<tablewidth='100%'border=0>
<trbgcolor='#CCCCCC'>
<td>Amount</td>
<td>cardNo</td>
<td>Holder Name</td>
<td>Expired</td>
<td>Update</td>
</tr>
<?php
//while($res = mysql_fetch_array($result)) { // mysql_fetch_array is deprecated, we need
to use mysqli_fetch_array
while($res=mysqli_fetch_array($result)){
echo"<tr>";
echo"<td>".$res['amount']."</td>";
```

```
echo"<td>".$res['cardNo']."</td>";
echo"<td>".$res['holdername']."</td>";
echo"<td>".$res['expired']."</td>";
echo"<td><a href=\"edit_bill.php?id=$res[p_id]\">Edit</a> |
<a href=\"delete_bill.php?id=$res[p_id]\" onClick=\"return confirm('Are you sure you want
to delete?')\">Delete</a></td>";

}
?>
</table>
</body>
</html>
```

Add_bill.html

```
<html>
<head>
<title>Add Data</title>
</head>

<body>
<ahref="bill.php">Home</a>
<br/><br/>

<formaction="add_bill.php"method="post"name="form1">
<tablewidth="25%"border="0">
<tr>
<td>Amount</td>
<td><inputtype="text"name="amount"></td>
</tr>
<tr>
<td>Card Number</td>
<td><inputtype="text"name="cardNo"></td>
</tr>
<tr>
<td>Holder Name</td>
<td><inputtype="text"name="holdername"></td>
</tr>
<tr>
<td>Expired</td>
<td><inputtype="text"name="expired"></td>
</tr>
<tr>
<td></td>
<td><inputtype="submit"name="Submit"value="add_bill.php"></td>
</tr>
</table>
</form>
</body>
</html>
```

Add_bill.php

```
<html>
<head>
<title>Add Data</title>
</head>

<body>
<?php
$connection=mysqli_connect('localhost','root','');
mysqli_select_db($connection,'bookings');
//including the database connection file
include_once("func/DB_config.php");
```

```
if(isset($_POST['Submit'])){
$amount=$_POST['amount'];
$cardNo=$_POST['cardNo'];
$holdername=$_POST['holdername'];
$expired=$_POST['expired'];


// checking empty fields
if(empty($amount)||empty($cardNo)||empty($holdername)||empty($expired)){
if(empty($amount)){
echo"<font color='red'>Name field is empty.</font><br/>";
}

if(empty($cardNo)){
echo"<font color='red'>type field is empty.</font><br/>";
}

if(empty($holdername)){
echo"<font color='red'>Email field is empty.</font><br/>";
}
if(empty($expired)){
echo"<font color='red'>Name field is empty.</font><br/>";
}
//link to the previous page
echo"<br/><a href='javascript:self.history.back();'>Go Back</a>";
}else{
// if all the fields are filled (not empty)
//insert data to database
$result=mysqli_query($connection,"INSERT INTO bill(amount,cardNo,holdername,expired)
VALUES('$amount','$cardNo','$holdername','$expired')");

//display success message
echo"<font color='green'>Data added successfully.";
echo"<br/><a href='home.php'>Return to home</a>";
}
}
?>
</body>
</html>
```

Edit_bill.php

```
<?php
// including the database connection file
include_once("func/DB_config.php");
$connection=mysqli_connect('localhost','root','');
mysqli_select_db($connection,'bookings');

$sql="select * from bill ";
$result=mysqli_query($connection,$sql);

if(isset($_POST['update']))
{
$amount=$_POST['amount'];
$cardNo=$_POST['cardNo'];
$holdername=$_POST['holdername'];
$expired=$_POST['expired'];
$p_id=$_POST['p_id'];

// checking empty fields
if(empty($amount)||empty($cardNo)||empty($holdername)||empty($expired)){
if(empty($amount)){
echo"<font color='red'>amount field is empty.</font><br/>";
}
```

```
if(empty($cardNo)){
echo"<font color='red'>cardNo field is empty.</font><br/>";
}

if(empty($holdername)){
echo"<font color='red'>holdername field is empty.</font><br/>";
}

if(empty($expired)){
echo"<font color='red'>expired field is empty.</font><br/>";
}
}else{
//updating the table
$sqlq="UPDATE bill SET
amount='".$amount."',cardNo='".$cardNo."',holdername='".$holdername."',expired='".$expire
d."' WHERE p_id=".$p_id;
$result=mysqli_query($connection,$sqlq);

//redirectig to the display page. In our case, it is index.php
header("Location: home.php");
}
}
?>
<?php
//getting id from url
$p_id=$_GET['id'];

//selecting data associated with this particular id
$result=mysqli_query($connection,"SELECT * FROM bill WHERE p_id=".$p_id);

while($res=mysqli_fetch_array($result))
{
$amount=$res['amount'];
$cardNo=$res['cardNo'];
$holdername=$res['holdername'];
}$expired=$res['expired'];
?>
<html>
<head>
<title>Edit Data</title>
</head>

<body>
<ahref="index.php">Home</a>
<br/><br/>

<formname="form1"method="post"action="edit_bill.php">
<tableborder="0">
<tr>
<td>Amount</td>
<td><inputtype="text"name="amount"value="<?phpecho$amount;?>"></td>
</tr>
<tr>
<td>Card Number</td>
<td><inputtype="text"name="cardNo"value="<?phpecho$cardNo;?>"></td>
</tr>
<tr>
<td>Holder Name</td>
<td><inputtype="text"name="holdername"value="<?phpecho$holdername;?>"></td>
</tr>
<tr>
<td>Expired</td>
<td><inputtype="text"name="expired"value="<?phpecho$expired;?>"></td>
</tr>
<tr>
<td><inputtype="hidden"name="p_id"value=<?phpecho$_GET['id'];?>></td>
```

```
<td><inputtype="submit"name="update"value="Update"></td>
</tr>
</table>
</form>
</body>
</html>
```

Delete_bill.php

```
<html>
<body>


<?php

//including the database connection file
$connection=mysqli_connect('localhost','root','');
mysqli_select_db($connection,'bookings');
include("DB_config.php");

//getting id of the data from url
$id=$_GET['id'];

//deleting the row from table
$result=mysqli_query($conn,"DELETE FROM bill WHERE p_id=".$id);

//redirecting to the display page (index.php in our case)
header("Location:home.php");
?>
</body>
</html>
```

Customer.php

```
<?php


$connection=mysqli_connect('localhost','root','');
mysqli_select_db($connection,'bookings');

$sql="select * from customer";
$result=mysqli_query($connection,$sql);



?>

<html>
<head>
<title>Homepage</title>
</head>

<body>
<ahref="add_customer.html">Add New Data</a><br/><br/>

<tablewidth='100%'border=0>
<trbgcolor='#CCCCCC'>
<td>Firs Name</td>
<td>Last Name</td>
<td>Email</td>
<td>Street</td>
<td>street Number</td>
<td>Update</td>
</tr>
<?php
```

```
//while($res = mysql_fetch_array($result)) { // mysql_fetch_array is deprecated, we need
to use mysqli_fetch_array
while($res=mysqli_fetch_array($result)){
echo"<tr>";
echo"<td>".$res['c_lastname']."</td>";
echo"<td>".$res['c_firstname']."</td>";
echo"<td>".$res['c_email']."</td>";
echo"<td>".$res['c_street']."</td>";
echo"<td>".$res['c_streetNo']."</td>";
echo"<td><a href=\"edit_customer.php?id=$res[c_id]\">Edit</a> |
<a href=\"delete_customer.php?id=$res[c_id]\" onClick=\"return confirm('Are you sure you
want to delete?')\">Delete</a></td>";

}
?>
</table>
</body>
</html>
```

Add_customer.html

```
<html>
<head>
<title>Add Data</title>
</head>

<body>
<ahref="customer.php">Home</a>
<br/><br/>

<formaction="add_customer.php"method="post"name="form1">
<tablewidth="25%"border="0">
<tr>
<td>First Name</td>
<td><inputtype="text"name="c_firstname"></td>
</tr>
<tr>
<td>Last Name</td>
<td><inputtype="text"name="c_lastname"></td>
</tr>
<tr>
<td>Email</td>
<td><inputtype="text"name="c_email"></td>
</tr>
<tr>
<td>Street</td>
<td><inputtype="text"name="c_street"></td>
</tr>
<tr>
<td>Street Number</td>
<td><inputtype="text"name="c_streetNo"></td>
</tr>
<tr>
<td></td>
<td><inputtype="submit"name="Submit"value="add_customer"></td>
</tr>
</table>
</form>
</body>
</html>
```

Add_customer.php

```
<html>
<head>
<title>Add Data</title>
```

```
</head>

<body>
<?php
$connection=mysqli_connect('localhost','root','');
mysqli_select_db($connection,'bookings');
//including the database connection file
include_once("func/DB_config.php");

if(isset($_POST['Submit'])){
$c_firstname=$_POST['c_firstname'];
$c_lastname=$_POST['c_lastname'];
$c_email=$_POST['c_email'];
$c_street=$_POST['c_street'];
$c_streetNo=$_POST['c_streetNo']



// checking empty fields
if(empty($c_firstname)||empty($c_lastname)||empty($c_email)||empty($c_street)||empty($c_s
treetNo)){
if(empty($c_firstname)){
echo"<font color='red'>First Name field is empty.</font><br/>";
}

if(empty($c_lastname)){
echo"<font color='red'>Last name field is empty.</font><br/>";
}

if(empty($c_email)){
echo"<font color='red'>Email field is empty.</font><br/>";
}
if(empty($$c_street)){
echo"<font color='red'>Email field is empty.</font><br/>";
}
if(empty($$c_streetNo)){
echo"<font color='red'>Email field is empty.</font><br/>";
}
//link to the previous page
echo"<br/><a href='javascript:self.history.back();'>Go Back</a>";
}else{
// if all the fields are filled (not empty)
//insert data to database
$result=mysqli_query($connection,"INSERT INTO
customer(c_firstname,c_lastname,c_email,c_street,c_streetNo)
VALUES('$f_name','$l_name','$c_email','$c_street','$c_streetNo')");

//display success message
echo"<font color='green'>Data added successfully.";
echo"<br/><a href='home.php'>Return to home</a>";
}
}
?>
</body>
</html>
```

Edit_customer.php

```
<html>
<head>
<title>Add Data</title>
</head>

<body>
<?php
$connection=mysqli_connect('localhost','root','');
```

```
mysqli_select_db($connection,'bookings');
//including the database connection file
include_once("func/DB_config.php");

if(isset($_POST['Submit'])){
$c_firstname=$_POST['c_firstname'];
$c_lastname=$_POST['c_lastname'];
$c_email=$_POST['c_email'];
$c_street=$_POST['c_street'];
$c_streetNo=$_POST['c_streetNo']



// checking empty fields
if(empty($c_firstname)||empty($c_lastname)||empty($c_email)||empty($c_street)||empty($c_s
treetNo)){
if(empty($c_firstname)){
echo"<font color='red'>First Name field is empty.</font><br/>";
}

if(empty($c_lastname)){
echo"<font color='red'>Last name field is empty.</font><br/>";
}

if(empty($c_email)){
echo"<font color='red'>Email field is empty.</font><br/>";
}
if(empty($$c_street)){
echo"<font color='red'>Email field is empty.</font><br/>";
}
if(empty($$c_streetNo)){
echo"<font color='red'>Email field is empty.</font><br/>";
}
//link to the previous page
echo"<br/><a href='javascript:self.history.back();'>Go Back</a>";
}else{
// if all the fields are filled (not empty)
//insert data to database
$result=mysqli_query($connection,"INSERT INTO
customer(c_firstname,c_lastname,c_email,c_street,c_streetNo)
VALUES('$f_name','$l_name','$c_email','$c_street','$c_streetNo')");

//display success message
echo"<font color='green'>Data added successfully.";
echo"<br/><a href='home.php'>Return to home</a>";
}
}
?>
</body>
</html>
```

Delete_customer.php

```
<html>
<body>


<?php

//including the database connection file
$connection=mysqli_connect('localhost','root','');
mysqli_select_db($connection,'bookings');
include("DB_config.php");

//getting id of the data from url
$id=$_GET['id'];
```

```
//deleting the row from table
$result=mysqli_query($conn,"DELETE FROM customer WHERE c_id=".$id);

//redirecting to the display page (index.php in our case)
header("Location:home.php");
?>
</body>
</html>
```

Booking.php

```
<?php


$connection=mysqli_connect('localhost','root','');
mysqli_select_db($connection,'bookings');

$sql="select * from booking";
$result=mysqli_query($connection,$sql);



?>

<html>
<head>
<title>Homepage</title>
</head>

<body>
<ahref="add_booking.html">Add New Data</a><br/><br/>
<tablewidth='100%'border=0>
<trbgcolor='#CCCCCC'>
<td>Status</td>
<td>Checkin</td>
<td>Checkout</td>
<td>Guest Number</td>
<td>Rooms Number</td>
</tr>
<?php
//while($res = mysql_fetch_array($result)) { // mysql_fetch_array is deprecated, we need
to use mysqli_fetch_array
while($res=mysqli_fetch_array($result)){
echo"<tr>";
echo"<td>".$res['status']."</td>";
echo"<td>".$res['checkin']."</td>";
echo"<td>".$res['checkout']."</td>";
echo"<td>".$res['guestNo']."</td>";
echo"<td>".$res['roomsNo']."</td>";
echo"<td><a href=\"edit_booking.php?id=$res[b_id]\">Edit</a> |
<a href=\"delete_booking.php?id=$res[b_id]\" onClick=\"return confirm('Are you sure you
want to delete?')\">Delete</a></td>";

}
?>
</table>
</body>
</html>
```

Add_booking.html

```
<html>
```

```
<head>
<title>Add Data</title>
</head>

<body>
<ahref="booking.php">Home</a>
<br/><br/>

<formaction="add_booking.php"method="post"name="form1">
<tablewidth="25%"border="0">
<tr>
<td>Status</td>
<td><inputtype="text"name="status"></td>
</tr>
<tr>
<td>Checkin</td>
<td><inputtype="text"name="checkin"></td>
</tr>
<tr>
<td>Checkout</td>
<td><inputtype="text"name="checkout"></td>
</tr>
<tr>
<td>Guest Number</td>
<td><inputtype="text"name="guestNo"></td>
</tr>
<tr>
<td>Room Number</td>
<td><inputtype="text"name="roomsNo"></td>
</tr>
<tr>
<td></td>
<td><inputtype="submit"name="Submit"value="add_booking"></td>
</tr>
</table>
</form>
</body>
</html>
```

Add_booking.php

```
<html>
<head>
<title>Add Data</title>
</head>

<body>
<?php
$connection=mysqli_connect('localhost','root','');
mysqli_select_db($connection,'bookings');
//including the database connection file
include_once("func/DB_config.php");

if(isset($_POST['Submit'])){
$status=$_POST['status'];
$checkin=$_POST['checkin'];
$checkout=$_POST['checkout'];
$guestNo=$_POST['guestNo'];
$roomsNo=$_POST['roomsNo'];


// checking empty fields
if(empty($status)||empty($checkin)||empty($checkout)||empty($guestNo)||empty($roomsNo)){
if(empty($status)){
echo"<font color='red'>status field is empty.</font><br/>";
}
```

```
if(empty($checkin)){
echo"<font color='red'>checkin field is empty.</font><br/>";
}

if(empty($checkout)){
echo"<font color='red'>checkout field is empty.</font><br/>";
}
if(empty($guestNo)){
echo"<font color='red'>guestNo field is empty.</font><br/>";
}
if(empty($roomsNo)){
echo"<font color='red'>roomsNo field is empty.</font><br/>";
}
//link to the previous page
echo"<br/><a href='javascript:self.history.back();'>Go Back</a>";
}else{
// if all the fields are filled (not empty)
//insert data to database
$result=mysqli_query($connection,"INSERT INTO
booking(status,checkin,checkout,guestNo,roomsNo)
VALUES('$status','$checkin','$checkout','$guestNo','$roomsNo')");

//display success message
echo"<font color='green'>Data added successfully.";
echo"<br/><a href='home.php'>Return to home</a>";
}
}
?>
</body>
</html>
```

Edit_booking.php

```
<?php
// including the database connection file
include_once("func/DB_config.php");
$connection=mysqli_connect('localhost','root','');
mysqli_select_db($connection,'bookings');

$sql="select * from booking ";
$result=mysqli_query($connection,$sql);

if(isset($_POST['update']))
{
$status=$_POST['status'];
$checkin=$_POST['checkin'];
$checkout=$_POST['checkout'];
$guestNo=$_POST['guestNo'];
$roomsNo=$_POST['roomsNo'];
$b_id=$_POST['b_id'];

// checking empty fields
if(empty($status)||empty($checkin)||empty($checkout)||empty($guestNo)||empty($roomsNo)){
if(empty($status)){
echo"<font color='red'>status field is empty.</font><br/>";
}

if(empty($checkin)){
echo"<font color='red'>checkin field is empty.</font><br/>";
}

if(empty($checkout)){
echo"<font color='red'>checkout field is empty.</font><br/>";
}
```

```
if(empty($guestNo)){
echo"<font color='red'>guestNo field is empty.</font><br/>";
}
if(empty($roomsNo)){
echo"<font color='red'>roomsNo field is empty.</font><br/>";
}
}else{
//updating the table
$sqlq="UPDATE booking SET
status='".$status."',checkin='".$checkin."',checkout='".$checkout."',guestNo='".$guestNo.
"',roomsNo='".$roomsNo."' WHERE b_id=".$b_id;
$result=mysqli_query($connection,$sqlq);

//redirectig to the display page. In our case, it is index.php
header("Location: home.php");
}
}
?>
<?php
//getting id from url
$b_id=$_GET['id'];

//selecting data associated with this particular id
$result=mysqli_query($connection,"SELECT * FROM booking WHERE b_id=".$b_id);

while($res=mysqli_fetch_array($result))
{
$status=$res['status'];
$checkin=$res['checkin'];
$checkout=$res['checkout'];
$guestNo=$res['guestNo'];
$roomsNo=$res['roomsNo'];
}

?>
<html>
<head>
<title>Edit Data</title>
</head>

<body>
<ahref="index.php">Home</a>
<br/><br/>

<formname="form1"method="post"action="edit_booking.php">
<tableborder="0">
<tr>
<td>Status</td>
<td><inputtype="text"name="status"value="<?phpecho$status;?>"></td>
</tr>
<tr>
<td>Checkin</td>
<td><inputtype="text"name="checkin"value="<?phpecho$checkin;?>"></td>
</tr>
<tr>
<td>Checkout</td>
<td><inputtype="text"name="checkout"value="<?phpecho$checkout;?>"></td>
</tr>
<tr>
<td>Guest Number</td>
<td><inputtype="text"name="guestNo"value="<?phpecho$guestNo;?>"></td>
</tr>
<tr>
<td>Rooms Number</td>
<td><inputtype="text"name="roomsNo"value="<?phpecho$roomsNo;?>"></td>
</tr>
```

```
<tr>
<td><inputtype="hidden"name="b_id"value=<?phpecho$_GET['id'];?>></td>
<td><inputtype="submit"name="update"value="Update"></td>
</tr>
</table>
</form>
</body>
</html>
```

Delete_booking.php

```
<html>
<body>


<?php

//including the database connection file
$connection=mysqli_connect('localhost','root','');
mysqli_select_db($connection,'bookings');
include("DB_config.php");

//getting id of the data from url
$id=$_GET['id'];

//deleting the row from table
$result=mysqli_query($conn,"DELETE FROM booking WHERE b_id=".$id);

//redirecting to the display page (index.php in our case)
header("Location:home.php");
?>
</body>
</html>
```

Travelagency.php

```
<?php


$connection=mysqli_connect('localhost','root','');
mysqli_select_db($connection,'bookings');

$sql="select * from travel_agency";
$result=mysqli_query($connection,$sql);



?>

<html>
<head>
<title>Homepage</title>
</head>

<body>
<ahref="add_travelagency.html">Add New Data</a><br/><br/>
<tablewidth='100%'border=0>
<trbgcolor='#CCCCCC'>
<td>Name</td>
<td>Telephone</td>
<td>Update</td>
</tr>
<?php
//while($res = mysql_fetch_array($result)) { // mysql_fetch_array is deprecated, we need
to use mysqli_fetch_array
while($res=mysqli_fetch_array($result)){
```

```
echo"<tr>";
echo"<td>".$res['t_name']."</td>";
echo"<td>".$res['t_telephone']."</td>";
echo"<td><a href=\"edit_travelagency.php?id=$res[t_id]\">Edit</a> |
<a href=\"delete.php?id=$res[t_id]\" onClick=\"return confirm('Are you sure you want to
delete?')\">Delete</a></td>";

}
?>
</table>
</body>
</html>
```

Add_travelagency.html

```
<html>
<head>
<title>Add Data</title>
</head>

<body>
<ahref="travelagency.php">Home</a>
<br/><br/>

<formaction="add_travelagency.php"method="post"name="form1">
<tablewidth="25%"border="0">
<tr>
<td>Name</td>
<td><inputtype="text"name="t_name"></td>
</tr>
<tr>
<td>Telephone</td>
<td><inputtype="text"name="t_telephone"></td>
</tr>
<tr>

<td></td>
<td><inputtype="submit"name="Submit"value="Add_travelagency"></td>
</tr>
</table>
</form>
</body>
</html>
```

Add_travelagency.php

```
<html>
<head>
<title>Add Data</title>
</head>

<body>
<?php
$connection=mysqli_connect('localhost','root','');
mysqli_select_db($connection,'bookings');
//including the database connection file
include_once("func/DB_config.php");

if(isset($_POST['Submit'])){
$t_name=$_POST['t_name'];
$t_telephone=$_POST['t_telephone'];

// checking empty fields
if(empty($name)||empty($t_telephone)){
if(empty($t_name)){
echo"<font color='red'>Name field is empty.</font><br/>";
```

```
}

if(empty($t_telephone)){
echo"<font color='red'>type field is empty.</font><br/>";
}

//link to the previous page
echo"<br/><a href='javascript:self.history.back();'>Go Back</a>";
}else{
// if all the fields are filled (not empty)
//insert data to database
$result=mysqli_query($connection,"INSERT INTO travel_agency(t_name,t_telephone)
VALUES('$t_name','$t_telephone')");

//display success message
echo"<font color='green'>Data added successfully.";
echo"<br/><a href='home.php'>Return to home</a>";
}
}
?>
</body>
</html>
```

Edit_travelagency.php

```
<?php
// including the database connection file
include_once("func/DB_config.php");
$connection=mysqli_connect('localhost','root','');
mysqli_select_db($connection,'bookings');

$sql="select * from travel_agency ";
$result=mysqli_query($connection,$sql);

if(isset($_POST['update']))
{
$t_name=$_POST['t_name'];
$telephone=$_POST['telephone'];
$t_id=$_POST['t_id'];

// checking empty fields
if(empty($t_name)||empty($telephone)){
if(empty($name)){
echo"<font color='red'>Name field is empty.</font><br/>";
}

if(empty($telephone)){
echo"<font color='red'>type field is empty.</font><br/>";
}
}else{
//updating the table
$sqlq="UPDATE travel_agency SET t_name='".$t_name."',telephone='".$telephone."'WHERE
t_id=".$t_id;
$result=mysqli_query($connection,$sqlq);

//redirectig to the display page. In our case, it is index.php
header("Location: home.php");
}
}
?>
<?php
//getting id from url
$t_id=$_GET['id'];

//selecting data associated with this particular id
$result=mysqli_query($connection,"SELECT * FROM travel_agency WHERE t_id=".$t_id);
```

```
while($res=mysqli_fetch_array($result))
{
$t_name=$res['t_name'];
$t_telephone=$res['t_telephone'];

}
?>
<html>
<head>
<title>Edit Data</title>
</head>

<body>
<ahref="index.php">Home</a>
<br/><br/>

<formname="form1"method="post"action="edit travel_agency.php">
<tableborder="0">
<tr>
<td>Name</td>
<td><inputtype="text"name="t_name"value="<?phpecho$t_name;?>"></td>
</tr>
<tr>
<td>Telephone</td>
<td><inputtype="text"name="telephone"value="<?phpecho$telephone;?>"></td>
</tr>
<tr>
<td><inputtype="hidden"name="t_id"value=<?phpecho$_GET['id'];?>></td>
<td><inputtype="submit"name="update"value="Update"></td>
</tr>
</table>
</form>
</body>
</html>
```

Delete_travelagency.php

```
<html>
<body>


<?php

//including the database connection file
$connection=mysqli_connect('localhost','root','');
mysqli_select_db($connection,'bookings');
include("DB_config.php");

//getting id of the data from url
$id=$_GET['id'];

//deleting the row from table
$result=mysqli_query($conn,"DELETE FROM travel_agency WHERE t_id=".$id);

//redirecting to the display page (index.php in our case)
header("Location:home.php");
?>
</body>
</html>
```

ΒΙΒΛΙΟΓΡΑΦΙΑ

1. Εφημερίδα "Η ΚΑΘΗΜΕΡΙΝΗ, ΕΛΛΗΝΙΚΗ ΟΙΚΟΝΟΜΙΑ", 26.08.2017 (http://www.kathimerini.gr/924133/article/oikonomia/ellhnikh-oikonomia/alpha-bank-sto-20-h-symvolh-toy-toyrismoy-sto-aep)

xamp

2. https://blog.udemy.com/xampp-tutorial/
3. https://en.wikipedia.org/wiki/XAMPP

php

4. http://www.w3schools.com/php/default.asp
5. http://www.freestuff.gr/forums/viewtopic.php?t=19080

notepad

6. https://notepad-plus-plus.org

violet uml

7. http://alexdp.free.fr/violetumleditor/page.php

phpMyAdmin

8. https://en.wikipedia.org/wiki/PhpMyAdmin
9. https://www.phpmyadmin.net

Βάσεις Δεδομένων

10. Βασίλειος Τ. Ταμπακάς «Εισαγωγή στις βάσεις δεδομένων»

11. Silberschatz, Korth & Sudarshan, «Συστήματα Βάσεων Δεδομένων», 4η έκδοση, 2004.

12. C. Date, «Εισαγωγή στα Συστήματα Βάσεων Δεδομένων»

13. Elmasri and Navathe, «Θεμελιώδεις Αρχές Συστημάτων Βάσεων Δεδομένων».