ΒΑΣΕΙΣ ΔΕΔΟΜΕΝΩΝ

Αναφορά εξαμηνιαίας εργασίας

Φοιτητές:

Αγιοργίτης ‘Αγγελος 03108178

Γεωργούλιας Αγησίλαος 03108158

ΠΕΡΙΕΧΟΜΕΝΑ

➢ Θέμα

Για την άσκηση αυτή κληθήκαμε να φτιάξουμε ένα σύστημα βάσης δεδομένων για την εταιρεία ΙΔΑΝΙΚΟ-ΣΠΙΤΙ, μια επιχείρηση που διευκολύνει τη διαδικασία ενοικίασης ακινήτων. Οι βασικές λειτουργίες αυτής της βάσης είναι να αποθηκεύει και να επεξεργάζεται τα δεδομένα που αφορούν στους υπάλληλους, τους πελάτες, τα συμβόλαια και άλλα στοιχεία που είναι απαραίτητα για τη σωστή λειτουργία της επιχείρησης.

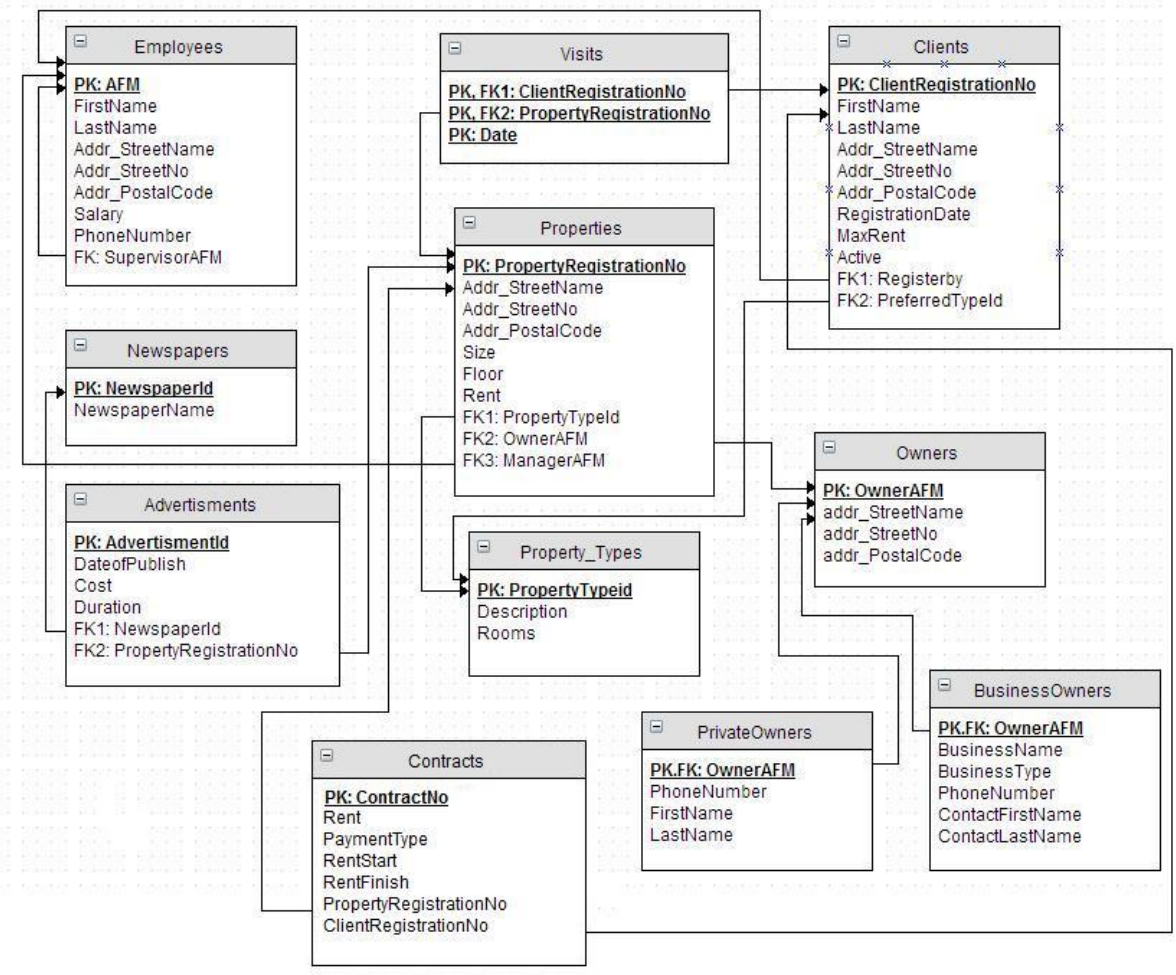
➢ Επιλογή Περιβάλλοντος

Για το SQL κομμάτι της εργασίας χρησιμοποιήσαμε τη MySQL καθώς είναι από τις πιο γνωστές και καθιερωμένες εμπορικές εφαρμογές και διατίθεται δωρεάν.

Για το γραφικό περιβάλλον επιλέξαμε τη Java και το IDE Eclipse διότι θεωρούμε πως αυτός ο συνδυασμός είναι ένα πιο ολοκληρωμένο προγραμματιστικό εργαλείο απ’ότι η PHP ή κάποιο άλλο περιβάλλον.

➢ Σχεδιασμός της Βάσης στο Σχεσιακό Μοντέλο

Για το σχεδιασμό της βάσης χρησιμοποιήσαμε το Σχεσιακό μοντέλο που αναρτήθηκε στο Mycourses, με ορισμένες τροποποιήσεις. Ακολουθεί, στην επόμενη σελίδα, το σχήμα του Σχεσιακού μοντέλου:



➢ Constraints-Integrity Issues

○ Entity Integrity (Ακεραιότητα Οντότητας)

Για να εξασφαλίσουμε αυτόν τον περιορισμό βεβαιωθήκαμε πως όλες οι σχέσεις της βάσης μας έχουν μία η παραπάνω ιδιότητες ορισμένες ως primary key, το οποίο δεν μπορεί να έχει την τιμή NULL:

// *OwnerAFM int unsigned* ***not null primary key*** //

// *ClientRegistrationNo int unsigned* ***not null,***

PropertyRegistrationNo int unsigned ***not null,***

primary key *(ClientRegistrationNo,PropertyRegistrationNo)* //

○ Referential Integrity (Αναφορική Ακεραιότητα)

Πρέπει να υπάρχει συνέπεια μεταξύ 2 πινάκων που συνδέονται με foreign key(s). Για να το πετύχουμε αυτό, κατά τη δημιουργία των πινάκων δηλώνουμε τυχόν ξένα κλειδιά μαζί με τον πίνακα στον οποίο αναφέρονται:

// ***FOREIGN KEY (RegisteredBy) REFERENCES Employees(AFM)*** *ON UPDATE CASCADE ON DELETE SET NULL,* //

Αυτό μας εξασφαλίζει ότι η MySQL δεν θα δεχτεί εισαγωγές τιμών στα πεδία-ξένα κλειδιά τα οποία δεν υπάρχουν στα referenced tables, αλλά ούτε και διαγραφές ή updates που θα δημιουργήσουν «ξεκρέμαστα» ξένα κλειδιά.

Επειδή όμως δεν θα ήταν λογικό να μην επιτρέπουμε διαγραφές σε ορισμένα πεδία-ξένα κλειδιά, μπορούμε να χρησιμοποιήσουμε τις εντολές CASCADE ή SET NULL σε περίπτωση update ή delete έτσι ώστε να διατηρείται η ακεραιότητα της βάσης:

// *FOREIGN KEY (NewspaperId) REFERENCES Newspapers(NewspaperId)* ***ON UPDATE CASCADE ON DELETE SET NULL*** //

Με αυτή την υλοποίηση λέμε στην SQL πως όταν αλλάξουμε ή διαγράψουμε μια εγγραφή από τον πίνακα Newspapers, να αλλάξει το πεδίο NewspaperId στον πίνακα Advertisments αν πρόκειται για update, ή να το κάνει set null αν πρόκειται για delete.

Γενικά στα περισσότερα foreign keys προσπαθήσαμε να χρησιμοποιήσουμε την υλοποίηση που είναι πιο λογική για το χρήστη με σκοπό να επιτρέψουμε όσο πιο πολλά updates και deletes γίνεται. Ο μόνος περιορισμός που θέσαμε είναι πως δεν μπορεί να γίνει διαγραφή πελάτη (table: Clients) ή ακινήτου (table: Properties) εάν η εγγραφή υπάρχει στον πίνακα με τα συμβόλαια. Στην περίπτωση αυτή θα επιστραφεί SQL error.

// *FOREIGN KEY (ClientRegistrationNo) REFERENCES Clients(ClientRegistrationNo) ON UPDATE CASCADE,*

FOREIGN KEY (PropertyRegistrationNo) REFERENCES Properties(PropertyRegistrationNo) ON UPDATE CASCADE //

○ Περιορισμοί

Για τους περιορισμούς πεδίου τιμών φροντίσαμε να δηλώσουμε κατά τη δημιουργία πινάκων για κάθε εγγραφή τι είδους τιμές να δέχεται. Χρησιμοποιήσαμε ευρέως τους τύπους int, unsigned και varchar ενώ λιγότερο τους τύπους date και decimal. Χρησιμοποιήσαμε επίσης και τον τύπο enum που επιτρέπει συγκεκριμένες τιμές στα πεδία του.

// *addr_StreetNo* ***int unsigned*** *not null*

Active ***enum('Yes','No')*** //

Για τους περιορισμούς στηλών εργαστήκαμε σε επίπεδο Java. Το ότι τα τηλέφωνα, για παράδειγμα, είναι unsigned int συχνά δεν είναι αρκετό γιατί ο χρήστης μπορεί κατά

λάθος να δώσει πολύ μικρό αριθμό. Έτσι το σύστημα καλό θα ήταν να μπορεί να καταλαβαίνει τα λάθη αυτά και να τα επισημαίνει στο χρήστη με κάποιο error message.

Αυτό το πετύχαμε ορίζοντας μια Boolean συνάρτηση type_validation η οποία θέτει περιορισμούς στις τιμές των πεδίων PhoneNumber και addr_PostalCode που εμφανίζονται συχνά στους πίνακες μας.

Το τηλέφωνο δεν μπορεί να είναι 5ψήφιο (πολύ μικρό) ενώ ο ταχυδρομικός κωδικός δεν μπορεί να είναι 6ψήφιος (πολύ μεγάλος).

```
phone = Integer.parseInt(input[7].getText());
postalcode = Integer.parseInt(input[5].getText());

if ((phone < 100000)|| (postalcode > 99999)){
                    JOptionPane.showMessageDialog(null,"Error on table
Employees : Phone number or postal code are incorrect","error",
JOptionPane.ERROR_MESSAGE);
                    return false;
}
return true;
```

Τέλος, για τους user-defined περιορισμούς εργαστήκαμε και σε επίπεδο Mysql και σε επίπεδο Java.

```
salary = Integer.parseInt(input[6].getText());

if (salary < 680){
                    JOptionPane.showMessageDialog(null,"Error on table
Employees : Salary is lower than minimum wage","error",
JOptionPane.ERROR_MESSAGE);
                    return false;
}

return true;
```

Ο περιορισμός αυτός εξασφαλίζει ότι ο μισθός των εργαζομένων δεν είναι μικρότερος απ'τον κατώτατο μισθό. Βέβαια ο κατώτατος μισθός αλλάζει, όμως όχι τόσο συχνά ώστε να καταντάει κουραστικό το να αλλάζει manually κάποιος την τιμή της συνθήκης.

Επίσης με τα triggers που ακολουθούν στην επόμενη σελίδα καλύπτουμε τις εξής περιπτώσεις:

Πριν την εισαγωγή νέας εγγραφής στον πίνακα Contracts, βεβαιωνόμαστε πως η ημερομηνία λήξης του συμβολαίου είναι μεταγενέστερη απ'την ημερομηνία έναρξης. Εάν δεν είναι τότε εμφανίζουμε error message στο χρήστη ειδοποιώντας τον ότι έχει κάνει λάθος στις ημερομηνίες.

Πριν τη μετατροπή του πίνακα Employees, εάν ο υπάλληλος πήρε προαγωγή (είχε SupervisorAFM ενώ τώρα δεν έχει) τότε ελέγχουμε να δούμε αν ο νέος του μισθός είναι ανώτερος απ'τον προηγούμενο του. Αν όχι τότε εμφανίζουμε error message ειδοποιώντας

το χρήστη ότι οι υπάλληλοι που παίρνουν προαγωγή πρέπει να παίρνουν μεγαλύτερο μισθό.

```
create trigger date_trigger before insert on Contracts
        for each row
        begin
                declare msg varchar(255);
                if (new.RentFinish < new.RentStart) then
                        set msg = 'Error on table Contracts : Rent must finish later than the day it starts';
                        signal sqlstate '45000' set message_text = msg;
                end if;
        end
//
```

```
create trigger employee_trigger before update on Employees
        for each row
        begin
                declare msg varchar(255);
                if ((new.SupervisorAFM IS NULL) and (old.SupervisorAFM > 0) and (old.Salary > new.Salary)) then
                        set msg = 'Error on table Employees : Promoted employees must receive a higher salary';
                        signal sqlstate '45000' set message_text = msg;
                end if;
        end
//
```

- **Indexes**

Η MySQL δημιουργεί αυτόματα indexes στα primary και foreign keys των πινάκων. Τα indexes αυτά είναι τύπου BTREE. Γενικά καλή ιδέα είναι να χρησιμοποιεί κανείς indexes σε πίνακες με πολλές εγγραφές και στους οποίους γίνονται συχνά queries, αλλά κατά προτίμηση όχι συχνά inserts, updates και deletes καθώς αυτό κοστίζει αφού πρέπει να ενημερώνεται κάθε φορά το ευρετήριο. Με αυτά τα κριτήρια λοιπόν εμείς δημιουργήσαμε indexes στο Employees.LastName και στο Clients.LastName διότι σε αυτά τα πεδία η συχνότητα των queries είναι δυσανάλογα υψηλή ως προς τη συχνότητα

inserts/updates/deletes σε σύγκριση με τα υπόλοιπα πεδία των ίδιων και τον υπολοίπων πινάκων.

// CREATE INDEX employee_index

ON Employees(LastName);

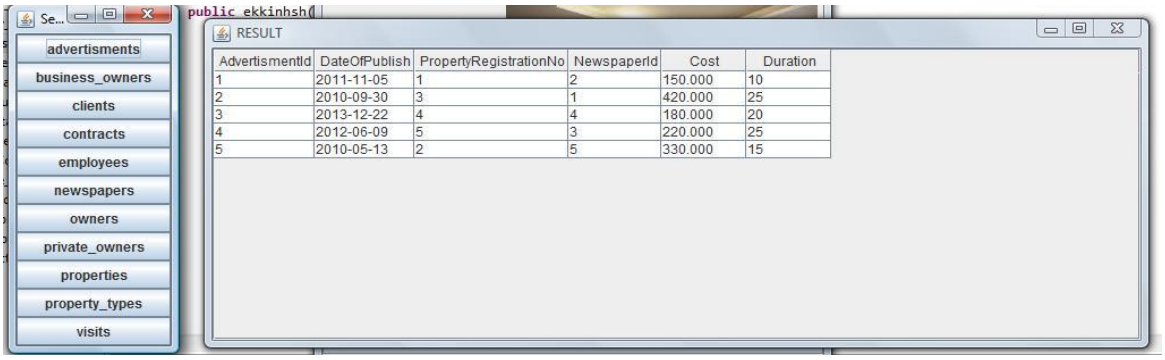
CREATE INDEX client_index

ON Clients(LastName); //

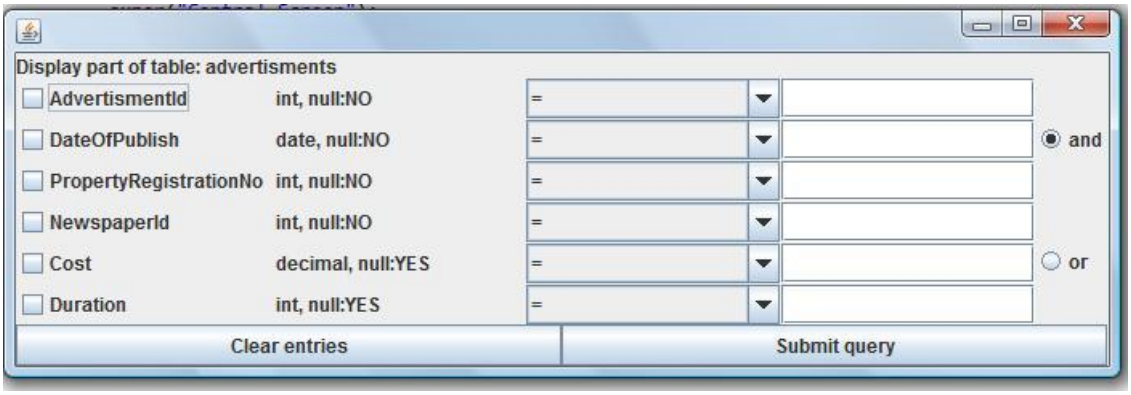
- **Παρουσίαση interface και γραφικού περιβάλλοντος**

Η υλοποίηση μας κάνει εκτενή αξιοποίηση στοιχείων όπως JPanels και JButtons καθώς και εικόνες για να διευκολύνει τη χρήση ενός user που δεν έχει γνώση των εσωτερικών μηχανισμών της βάσης και ούτε θέλει να θυμάται ασήμαντες πληροφορίες όπως τα διάφορα ID που χρησιμοποιούνται για την ενδοεπικοινωνία τον πινάκων.

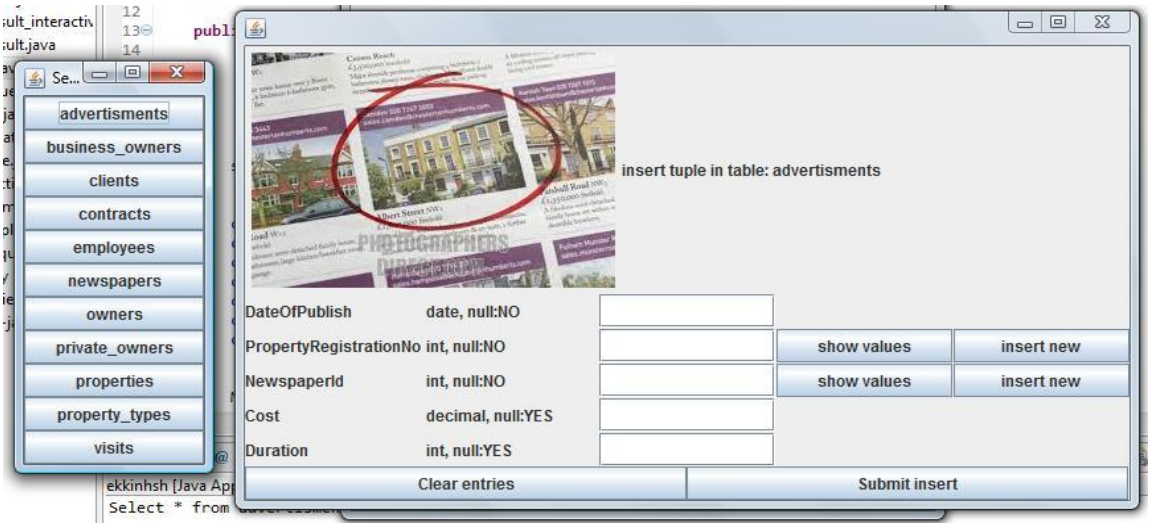
Η αρχική οθόνη δίνει στο χρήστη όλες τις επιλογές που χρειάζεται. Πατώντας την πρώτη επιλογή θα του εμφανίσει τη λίστα με τους πίνακες, και επιλέγοντας έναν πίνακα θα δει όλες τις εγγραφές του.



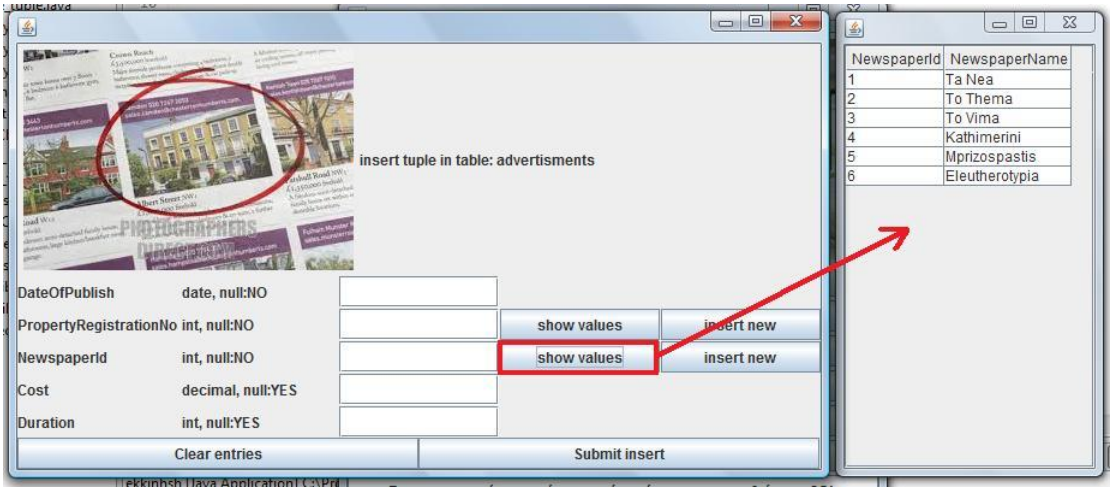
Εάν επιλέξει να δεί ένα μέρος του table θα εμφανιστεί ένα παράθυρο που του δίνει τη δυνατότητα να επιλέξει τις τιμές των πεδίων των εγγραφών που τον ενδιαφέρουν για το συγκεκριμένο table.



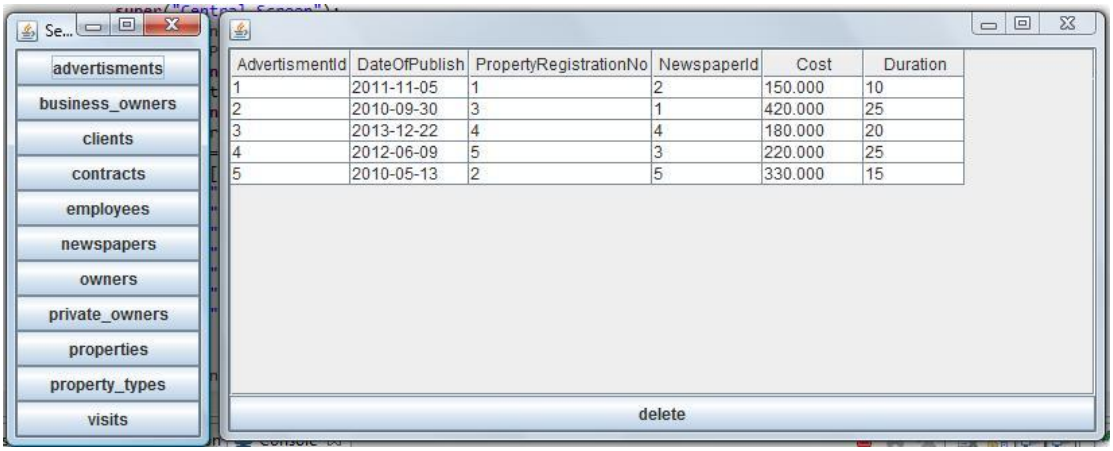
Επιλέγοντας να κάνει εισαγωγή μιας καινούριας εγγραφής θα πρέπει να επιλέξει σε ποιον πίνακα θέλει να γίνει η εισαγωγή και στη συνέχεια θα του εμφανιστεί το παράθυρο με τα πεδία καθώς και μια εικόνα (η εικόνα έχει ως στόχο το visual confirmation ότι ο χρήστης επέλεξε το σωστό table).



Επειδή όμως ο χρήστης δεν χρειάζεται να θυμάται λεπτομέρειες όπως το NewspaperID, μπορεί σε συγκεκριμένα πεδία να επιλέξει show values ώστε να δει τις επιλογές που έχει.

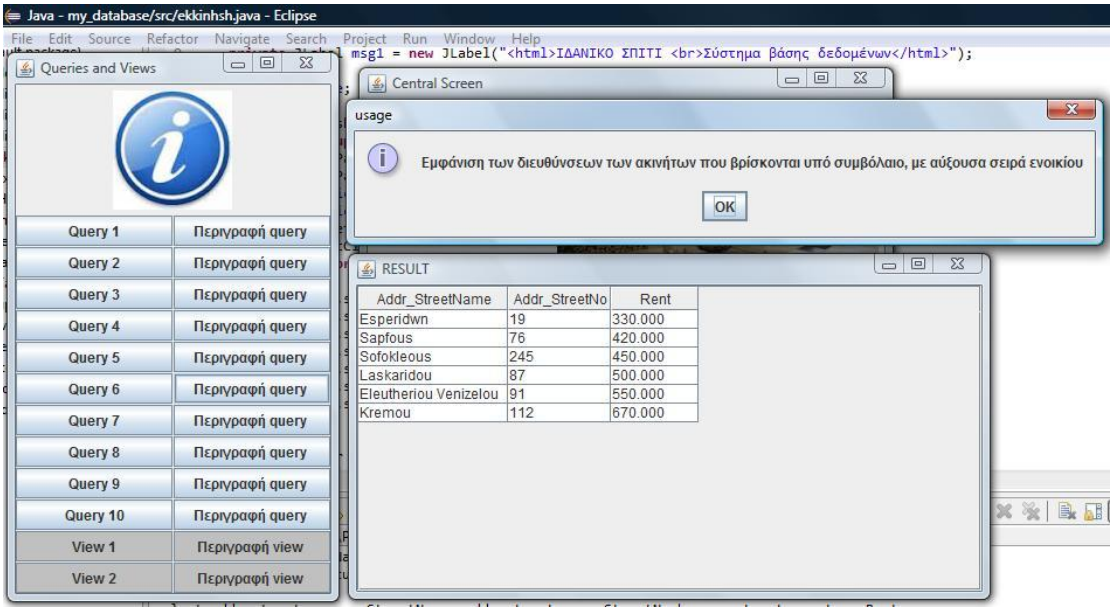


Για διαγραφή, αφού ο χρήστης επιλέξει το table απ'το οποίο θέλει να διαγράψει μια εγγραφή, αρκεί απλά να επιλέξει την εγγραφή και να πατήσει delete για να διαγραφεί.



Το update λειτουργεί παρόμοια με το delete μόνο που μετά το πάτημα της επιλογής update ανοίγει νέο παράθυρο όμοιο με αυτό της εισαγωγής εγγραφής ώστε ο χρηστής να δώσει τα νέα πεδία της εγγραφής που αλλάζει.

Τέλος, ο χρήστης έχει τη δυνατότητα να εκτελέσει έτοιμα queries και views επιλέγοντας το αντίστοιχο option στο αρχικό μενού. Μπορεί επίσης να δει την περιγραφή του query ή του view.



- **DLLs**
  - **Triggers**

Παραθέσαμε ήδη στις σελίδες 5&6 τα 2 εκ των 3 triggers μαζί με περιγραφή της λειτουργίας τους. Το 3^{ο} trigger που χρησιμοποιήσαμε εκτελείται μετά την εισαγωγή νέας εγγραφής στα συμβόλαια και εξασφαλίζει ότι οι πελάτες που βρίσκονται υπό συμβόλαιο έχουν στο πεδίο Active τιμή ‘Yes’.

```
create trigger client_trigger after insert on Contracts
  for each row
  begin
        update Clients set Active='Yes'
  where ClientRegistrationNo=new.ClientRegistrationNo;
  end
//
```

  - **Views**

Μη ενημερώσιμο view: αριθμός πελατών που προτιμούν το κάθε είδος σπιτιού

```
CREATE VIEW view1 (description, zitisi)
AS
select Description, COUNT(*)
from Property_Types p, Clients c
```

where p.PropertyTypeId = c.PreferredTypeId

group by Description ;

Ενημερώσιμο view: ποιοί πελάτες είναι active

CREATE VIEW view2 (FirstName,LastName,RegistrationDate,Active)

AS

select FirstName,LastName,RegistrationDate,Active

from Clients

where Active="Yes";

- **Queries**

Τα ονόματα και οι διευθύνσεις των ιδιωτών που έχουν καταχωρήσει το σπίτι τους για ενοικίαση

select FirstName, LastName, Addr_StreetName as StreetName, Addr_StreetNo as StreetNumber

from private_owners natural join properties;

Τα ονόματα ιδιωτών ΚΑΙ επιχειρήσεων που έχουν καταχωρήσει το σπίτι τους για ενοικίαση

(select FirstName, LastName, Addr_StreetName as StreetName, Addr_StreetNo as StreetNumber

from private_owners natural join properties)

union

(select ContactFirstName, ContactLastName, Addr_StreetName as StreetName, Addr_StreetNo as StreetNumber

from business_owners natural join properties);

Εμφάνιση του ελάχιστου, του μέγιστου και του μέσου όρου μισθού για τους προιστάμενους και τους υφιστάμενους

(select min(salary), avg(salary), max(salary)

from Employees where SupervisorAFM is null)

union

(select min(salary), avg(salary), max(salary)

from Employees where SupervisorAFM is not null);

Εμφάνιση της διεύθυνσης όλων των ακινήτων και του είδους τους

select addr_Streetname as StreetName, addr_StreetNo as StreetNo, Description

from Property_types natural join properties;

Εμφάνιση του μέσου όρου ενοικίου ανά είδος ακινήτου

select Description, avg(rent) as Rent

from property_types natural join properties

group by PropertyTypeId;

Εμφάνιση των διευθύνσεων των ακινήτων που βρίσκονται υπό συμβόλαιο, με αύξουσα σειρά ενοικίου

select addr_streetname as StreetName, addr_streetno as StreetNumber, contracts.rent as Rent

from contracts, properties

where contracts.PropertyRegistrationNo = properties.PropertyRegistrationNo

order by contracts.rent;

Εμφάνιση των συμβολαίων με ενοίκιο μεγαλύτερο από αυτό που δηλώθηκε για το ακίνητο κατά την καταχώρηση στη βάση δεδομένων

select addr_streetname as StreetName, addr_StreetNo as StreetNumber, contracts.rent as ContractRent, properties.rent as PropertyRent

from contracts,properties

where contracts.PropertyRegistrationNo = properties.PropertyRegistrationNo

having contracts.rent > properties.rent;

Εμφάνιση του συνολικού αριθμού συμβολαίων και του συνολικού αριθμού καταχωρημένων ακινήτων

select count(distinct(contracts.PropertyRegistrationNo)) as TotalContracts, count(distinct(properties.PropertyRegistrationNo)) as TotalProperties

from contracts,properties;

Εμφάνιση του αριθμού των διαφημίσεων των οποίων το διαφημιζόμενο ακίνητο τελικά ενοικιάστηκε

select count(*) as SuccessfulAdvertisments

from advertisments natural join

(select contracts.propertyregistrationno

from properties,contracts

where contracts.PropertyRegistrationNo = properties.PropertyRegistrationNo) as temp;

Εμφάνιση των επισκέψεων που τελικά ο ίδιος πελάτης κατέληξε να αγοράσει το σπίτι

select temp.FirstName, temp.LastName, temp.addr_Streetname as StreetName, temp.addr_StreetNo as StreetNumber, visits.date as Date

from visits,

(select contracts.clientregistrationno,contracts.propertyregistrationno, clients.FirstName, clients.LastName, properties.addr_Streetname, properties.addr_StreetNo

from contracts,properties,clients

where contracts.clientregistrationno = clients.clientregistrationno and contracts.propertyregistrationno = properties.propertyregistrationno) as temp

where visits.propertyregistrationno = temp.propertyregistrationno AND visits.clientregistrationno = temp.clientregistrationno;

- **Source files**

Το αρχείο create_DB.txt δημιουργεί τη βάση, τους πίνακες, τα indexes, τα views και τα triggers:

```
drop database my_project;
create database my_project;
use my_project;

create table Employees (
   AFM int unsigned not null primary key,
   FirstName varchar(30) not null,
   LastName varchar(30) not null,
   addr_StreetName varchar(30) not null,
   addr_StreetNo int unsigned not null,
   addr_PostalCode int unsigned not null,
   Salary decimal(13,3) not null,
   PhoneNumber int unsigned not null,
   SupervisorAFM int unsigned,
   FOREIGN KEY (SupervisorAFM) REFERENCES Employees(AFM) ON UPDATE CASCADE
ON DELETE SET NULL
)ENGINE=INNODB;

create table Property_Types(
   PropertyTypeId int unsigned not null auto_increment primary key,
   Description varchar(100) not null,
   Rooms int unsigned not null
)ENGINE=INNODB;

create table Clients (
   ClientRegistrationNo int unsigned not null auto_increment primary key,
   FirstName varchar(30) not null,
   LastName varchar(30) not null,
   addr_StreetName varchar(30) not null,
   addr_StreetNo int unsigned not null,
   addr_PostalCode int unsigned not null,
   RegistrationDate date,
   MaxRent int unsigned,
   Active enum('Yes','No'),
```

```
   RegisteredBy int unsigned,
   PreferredTypeId int unsigned,
   FOREIGN KEY (RegisteredBy) REFERENCES Employees(AFM) ON UPDATE CASCADE
ON DELETE SET NULL,
   FOREIGN KEY (PreferredTypeId) REFERENCES Property_Types(PropertyTypeId) ON
UPDATE CASCADE ON DELETE SET NULL
)ENGINE=INNODB;

create table Owners (
   OwnerAFM int unsigned not null primary key,
   addr_StreetName varchar(30) not null,
   addr_StreetNo int unsigned not null,
   addr_PostalCode int unsigned not null
)ENGINE=INNODB;

create table Private_Owners (
   OwnerAFM int unsigned not null primary key,
   PhoneNumber int unsigned not null,
   FirstName varchar(30) not null,
   LastName varchar(30) not null,
   FOREIGN KEY (OwnerAFM) REFERENCES Owners(OwnerAFM) ON UPDATE CASCADE
ON DELETE CASCADE
)ENGINE=INNODB;

create table Business_Owners(
   OwnerAFM int unsigned not null primary key,
   BusinessName varchar(30) not null,
   BusinessType varchar(30),
   PhoneNumber int unsigned not null,
   ContactFirstName varchar(30) not null,
   ContactLastName varchar(30) not null,
   FOREIGN KEY (OwnerAFM) REFERENCES Owners(OwnerAFM) ON UPDATE CASCADE
ON DELETE CASCADE
)ENGINE=INNODB;

create table Properties (
   PropertyRegistrationNo int unsigned not null auto_increment primary key,
   Addr_StreetName varchar(30) not null,
   Addr_StreetNo int unsigned not null,
   Addr_PostalCode int unsigned not null,
   Size decimal(13,3) not null,
   Floor int unsigned not null,
   Rent decimal(13,3) unsigned,
   PropertyTypeId int unsigned,
   ManagerAFM int unsigned,
   OwnerAFM int unsigned,
```

```
   FOREIGN KEY (PropertyTypeId) REFERENCES Property_Types(PropertyTypeId) ON
UPDATE CASCADE ON DELETE SET NULL,
   FOREIGN KEY (ManagerAFM) REFERENCES Employees(AFM) ON UPDATE CASCADE
ON DELETE SET NULL,
   FOREIGN KEY (OwnerAFM) REFERENCES Owners(OwnerAFM) ON UPDATE CASCADE
ON DELETE SET NULL
)ENGINE=INNODB;

create table Contracts (
   ContractNo int unsigned not null auto_increment primary key,
   Rent decimal(13,3) not null,
   PaymentType enum('Cash','Check'),
   RentStart date,
   RentFinish date,
   ClientRegistrationNo int unsigned not null,
   PropertyRegistrationNo int unsigned not null,
   FOREIGN KEY (ClientRegistrationNo) REFERENCES Clients(ClientRegistrationNo) ON
UPDATE CASCADE,
   FOREIGN KEY (PropertyRegistrationNo) REFERENCES
Properties(PropertyRegistrationNo) ON UPDATE CASCADE
)ENGINE=INNODB;

create table Visits (
   ClientRegistrationNo int unsigned not null,
   PropertyRegistrationNo int unsigned not null,
   Date date,
   FOREIGN KEY (ClientRegistrationNo) REFERENCES Clients(ClientRegistrationNo) ON
UPDATE CASCADE ON DELETE CASCADE,
   FOREIGN KEY (PropertyRegistrationNo) REFERENCES
Properties(PropertyRegistrationNo) ON UPDATE CASCADE ON DELETE CASCADE,
   primary key (ClientRegistrationNo,PropertyRegistrationNo)
)ENGINE=INNODB;

create table Newspapers (
   NewspaperId int unsigned not null auto_increment primary key,
   NewspaperName varchar(30) not null
)ENGINE=INNODB;

create table Advertisments(
   AdvertismentId int unsigned not null auto_increment primary key,
   DateOfPublish date not null,
   PropertyRegistrationNo int unsigned not null,
   NewspaperId int unsigned not null,
   Cost decimal(13,3),
   Duration int unsigned,
   FOREIGN KEY (PropertyRegistrationNo) REFERENCES
Properties(PropertyRegistrationNo) ON UPDATE CASCADE ON DELETE CASCADE,
```

```
   FOREIGN KEY (NewspaperId) REFERENCES Newspapers(NewspaperId) ON UPDATE
CASCADE ON DELETE CASCADE
)ENGINE=INNODB;


/*mh enhmerwsimh: αριθμός πελατών που προτιμούν το κάθε είδος σπιτιού*/
CREATE VIEW view1 (description, zitisi)
AS
select Description, COUNT(*)
from Property_Types p, Clients c
where p.PropertyTypeId = c.PreferredTypeId
group by Description ;


/*enhmerwsimh: ποιοί πελάτες είναι active*/
CREATE VIEW view2 (FirstName,LastName,RegistrationDate,Active)
AS
select FirstName,LastName,RegistrationDate,Active
from Clients
where Active="Yes";

CREATE INDEX employee_index
ON Employees(LastName);

CREATE INDEX client_index
ON Clients(LastName);

delimiter //

create trigger date_trigger before insert on Contracts
   for each row
   begin
            declare msg varchar(255);
            if (new.RentFinish < new.RentStart) then
                     set msg = 'Error on table Contracts : Rent must finish later than the
day it starts';
                     signal sqlstate '45000' set message_text = msg;
            end if;
   end
//

create trigger employee_trigger before update on Employees
   for each row
   begin
            declare msg varchar(255);
            if ((new.SupervisorAFM IS NULL) and (old.SupervisorAFM > 0) and (old.Salary
> new.Salary)) then
```

```
                    set msg = 'Error on table Employees : Promoted employees must
receive a higher salary';
                    signal sqlstate '45000' set message_text = msg;
            end if;
   end
//

create trigger client_trigger after insert on Contracts
   for each row
   begin
            update Clients set Active='Yes' where
ClientRegistrationNo=new.ClientRegistrationNo;
   end
//

delimiter ;
```

Το αρχείο populateDB.txt εισάγει τις εγγραφές στους πίνακες

```
use my_project;

delete from Employees;
delete from Property_Types;
delete from Clients;
delete from Owners;
delete from Private_Owners;
delete from Business_Owners ;
delete from Properties ;
delete from Contracts ;
delete from Visits ;
delete from Newspapers ;
delete from Advertisments ;


alter table Employees auto_increment=1;
alter table Property_Types auto_increment=1;
alter table Clients auto_increment=1;
alter table Properties auto_increment=1;
alter table Contracts auto_increment=1;
alter table Visits auto_increment=1;
alter table Newspapers auto_increment=1;
alter table Advertisments auto_increment=1;


insert into Property_Types(PropertyTypeId, Description, Rooms) values
(null,'Garsoniera',1),
(null,'Dyari',2),
```

(null,'Triari',3),
(null,'Tessari',4),
(null,'Garage',1);

insert into Employees(AFM, FirstName,LastName,
addr_StreetName,addr_StreetNo,addr_PostalCode,Salary,PhoneNumber,SupervisorAF
M) values
('244891','Ned','Stark','Laskaridou','22','17676','2000','46823670', null),
('471290','Catelyn','Tully','Kremou','45','18743','2500','27016385', null),
('570922','Jaime','Lannister','Papagou','112','19822','3000','47925150', null),
('974500','Cersei','Lannister','Kolokotroni','89','85628','2200','62981490', null),
('288311','Stannis','Baratheon','Aleksandreias','33','78120','1900','14945829', null),
('689541','Robert','Baratheon','Themistokleous','44','20987','2200','51390567','974500
'),
('394273','Margaery','Tyrell','Sivitanidou','90','97452','2900','58924519','974500'),
('536201','Daenerys','Targaryen','Filaretou','22','34087','1400','52803589','689541'),
('309812','Davos','Seaworth','3is
Septemvriou','98','65922','1200','28901754','288311'),
('462382','Oberyn','Martell','Syntagmatos','24','72299','2200','68245901','974500');

insert into
Clients(ClientRegistrationNo,FirstName,LastName,addr_StreetName,addr_StreetNo,a
ddr_PostalCode,RegistrationDate,MaxRent,Active,RegisteredBy,PreferredTypeId)
values
(null,'Sansa','Stark','Laskaridou','22','17676','2010-12-12','500','Yes','244891','1'),
(null,'Arya','Stark','Kremou','45','18743','2014-6-12','420','Yes','244891','3'),
(null,'Loras','Tyrell','Sivitanidou','90','97452','2013-10-12','750','Yes','471290','2'),
(null,'Joffrey','Lannister','Kolokotroni','89','85628','2012-8-12','500','Yes','471290','4'),
(null,'Myrcella','Baratheon','Filaretou','22','34087','2015-7-
12','1200','Yes','689541','1'),
(null,'John','Snow','3is Septemvriou','98','65922','2011-1-12','600','Yes','689541','2'),
(null,'Tommen','Baratheon','Aleksandreias','33','78120','2012-2-
12','400','No','689541','3'),
(null,'Asha','Greyjoy','Syntagmatos','24','72299','2014-4-12','550','No','536201','5'),
(null,'Khal','Drogo','Themistokleous','44','20987','2013-5-12','700','No','536201','4');

insert into Owners(OwnerAFM, addr_StreetName,addr_StreetNo,addr_PostalCode)
values
('472390','Aleksandreias','33','78120'),
('578932','Filaretou','22','34087'),
('905781','Laskaridou','22','17676'),
('579123','Sivitanidou','90','97452'),
('390461','3is Septemvriou','98','65922'),

('214622','Syntagmatos','24','72299');

insert into
Business_Owners(OwnerAFM,BusinessName,BusinessType,PhoneNumber,ContactFirst
Name,ContactLastName) values
('472390','Castle Rock','Enexyrodaneisthrio','64247156','Tyrion','Lannister'),
('578932','Winterfell','Skylotrofeio','42078371','Maester','Aemon'),
('905781','Highgarden','Anthopwleio','89367190','Olenna','Tyrell');

insert into Private_Owners(OwnerAFM,PhoneNumber,FirstName,LastName) values
('579123','62975178','Jorah','Mormont'),
('390461','78419752','Rhaegar','Targaryen'),
('214622','71459714','Petyr','Baelish');

insert into Properties(PropertyRegistrationNo, Addr_StreetName, Addr_StreetNo,
Addr_PostalCode,Size,Floor,Rent,PropertyTypeId,ManagerAFM,OwnerAFM) values
(null,'Laskaridou','87','17676','105','1','420','1','244891','472390'),
(null,'Kremou','112','19062','82','3','500','2','244891','578932'),
(null,'Sapfous','76','18752','71','3','450','4','974500','579123'),
(null,'Eleutheriou Venizelou','91','15678','67','1','330','2','244891','390461'),
(null,'Esperidwn','19','42042','132','0','670','3','536201','472390'),
(null,'Sofokleous','245','13377','52','1','550','2','974500','578932'),
(null,'Lykourgou','4','64128','98','5','890','1','536201','905781'),
(null,'28is Oktovriou','58','21095','87','4','285','5','974500','214622');

insert into Contracts(ContractNo,Rent,PaymentType,RentStart,RentFinish,
ClientRegistrationNo,PropertyRegistrationNo) values
(null,'420','Cash','2014-10-4','2015-10-14','1','3'),
(null,'500','Check','2011-5-22','2015-2-4','2','1'),
(null,'450','Cash','2012-4-20','2016-11-5','3','6'),
(null,'330','Cash','2015-3-6','2016-3-21','4','5'),
(null,'670','Check','2012-6-17','2016-6-30','5','2'),
(null,'550','Check','2014-11-16','2015-8-9','6','4');

insert into Visits(ClientRegistrationNo,PropertyRegistrationNo,Date) values
('2','1','2014-9-29'),
('1','2','2011-2-1'),
('5','3','2012-4-19'),
('1','4','2015-2-25'),

('3','5','2012-5-11'),
('2','6','2014-10-24');

insert into Newspapers(NewspaperId, NewspaperName) values
(null,'Ta Nea'),
(null,'To Thema'),
(null,'To Vima'),
(null,'Kathimerini'),
(null,'Mprizospastis'),
(null,'Eleutherotypia');

insert into
Advertisments(AdvertismentId,DateOfPublish,PropertyRegistrationNo,NewspaperId,Cost,Duration) values
(null,'2011-11-5','1','2','150','10'),
(null,'2010-9-30','3','1','420','25'),
(null,'2013-12-22','4','4','180','20'),
(null,'2012-6-9','5','3','220','25'),
(null,'2010-5-13','2','5','330','15');