ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ &

ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

ΑΝΑΦΟΡΑ ΕΞΑΜΙΝΙΑΙΑΣ ΕΡΓΑΣΙΑΣ

ΒΑΣΕΙΣ ΔΕΔΟΜΕΝΩΝ

(7ου εξαμήνου, Ακ. Έτος: 2014-2015)

ΟΜΑΔΑ: 55

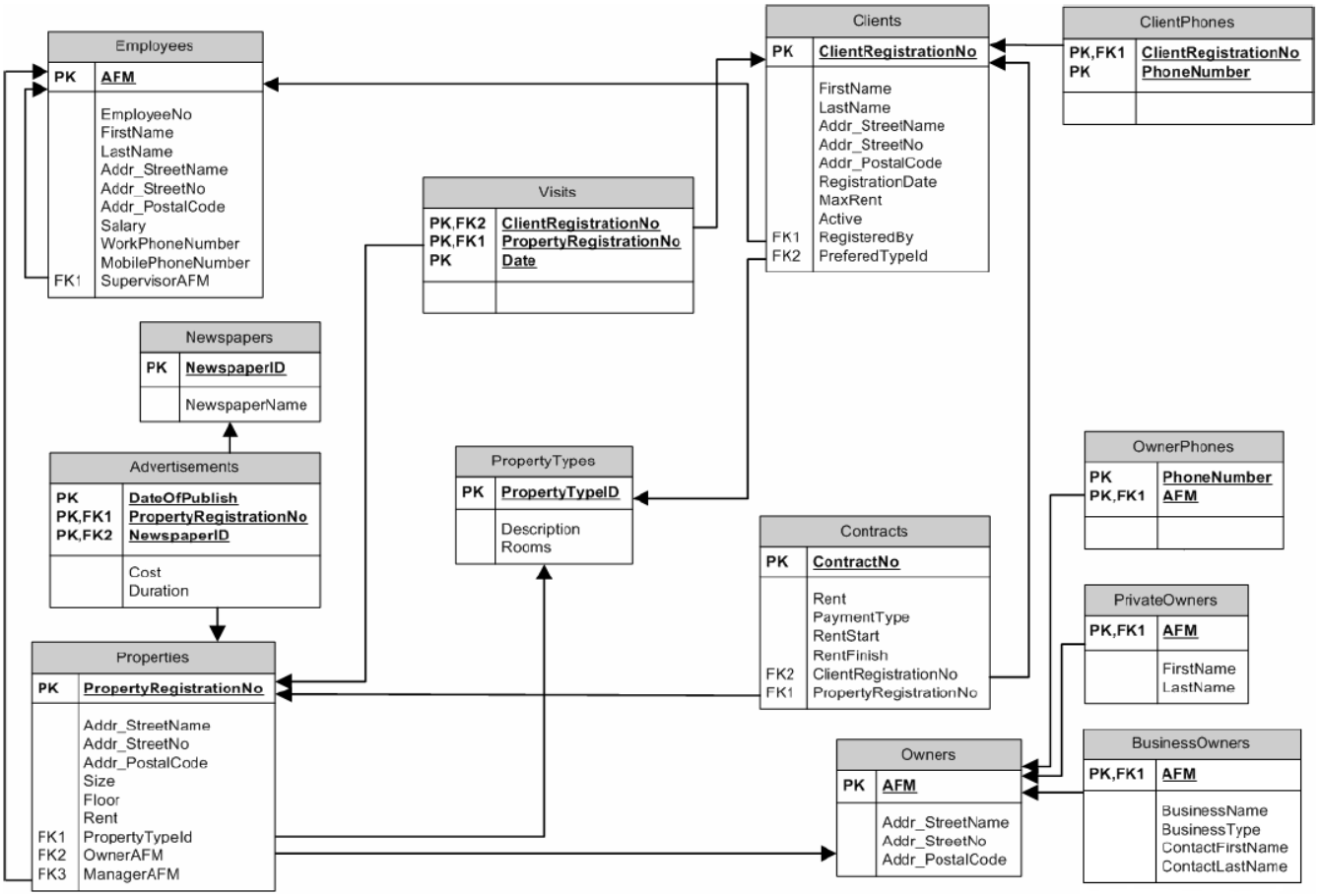
ΑΝΔΡΟΥΛΙΔΑΚΗΣ ΙΩΑΝΝΗΣ (03111030)

ΒΑΣΙΛΑΚΗΣ ΓΕΩΡΓΙΟΣ (03111061)

ΘΕΟΔΩΡΑΚΗΣ ΡΑΦΑΗΛ-ΓΕΩΡΓΙΟΣ (03111071)

➢ Σχεσιακό Μοντέλο

Ως σχεσιακό μοντέλο για την ανάπτυξη της εφαρμογής χρησιμοποιήθηκε το μοντέλο που περιλαμβάνεται στην ενδεικτική λύση της πρώτης άσκησης.



Το Microsoft Visual Studio είναι ένα ολοκληρωμένο περιβάλλον ανάπτυξης (IDE) από τη Microsoft. Χρησιμοποιείται για την ανάπτυξη προγραμμάτων ηλεκτρονικών υπολογιστών για τα Microsoft Windows, καθώς και ιστοσελίδες, διαδικτυακές εφαρμογές και υπηρεσίες web.Το Visual Studio χρησιμοποιεί πλατφόρμες ανάπτυξης λογισμικού της Microsoft, όπως το Windows API, Windows Forms, Windows Presentation Foundation, το Windows Store και το Microsoft Silverlight. Μπορεί να παράγει τόσο εγγενή κώδικα, όσο και διαχειριζόμενο κώδικα.

Το VS περιλαμβάνει έναν επεξεργαστή κώδικα με υποστήριξη IntelliSense. Το ολοκληρωμένο πρόγραμμα εντοπισμού σφαλμάτων (debugger) μας βοήθησε στον εντοπισμό σφαλμάτων τόσο σε επίπεδο πηγαίου κώδικα και όσο και σε επίπεδο μηχανής. Άλλα ενσωματωμένα εργαλεία μας διευκόλυναν ως προς το βασικό σχεδιασμό του γραφικού περιβάλλοντος της εφαρμογής που βλέπει ο χρήστης (GUI.

Το Visual Studio υποστηρίζει διαφορετικές γλώσσες προγραμματισμού και επιτρέπει την επεξεργασία κώδικα και τον εντοπισμό σφαλμάτων για σχεδόν οποιαδήποτε γλώσσα προγραμματισμού, εφόσον έχουμε εγκαταστήσει τις κατάλληλες βιβλιοθήκες.

Οι Built-in γλώσσες περιλαμβάνουν C, C ++, VB.NET, C # και F#. Ακόμη υπάρχει υποστήριξη για άλλες γλώσσες, όπως η Python, Ruby, XML / XSLT, HTML / XHTML, JavaScript και CSS. Εμείς χρησιμοποιήσαμε τη C# για την εφαρμογή μας.

- **Ελεύθερο λογισμικό**

Η express έκδοση ή παλαιότερες εκδόσεις του VS δίνονται ελεύθερα για χρήση, χωρίς κανένα κόστος.

- **Περιορισμοί Βάσης Δεδομένων**

Περιορισμοί αναφορικής ακεραιότητας (περιορισμοί ξένων κλειδιών)

Οι περιορισμοί ξένων κλειδιών δημιουργήθηκαν πάνω στο σχεσιακό μοντέλο στο οποίο βασίστηκε η εφαρμογή.

Περιλαμβάνονται στα DDL κατά τη δημιουργία των πινάκων των σχέσεων με τη σύνταξη:

CONSTRAINT <όνομα περιορισμού>

FOREIGN KEY (<όνομα γνωρίσματος στον οριζόμενο πίνακα>)

REFERENCES <όνομα πίνακα στον οποίο αναφέρεται> (<όνομα γνωρίσματος στον αναφερόμενο πίνακα>)

ON DELETE <ενέργεια που εκτελείται κατά τη διαγραφή στοιχείου από τον αναφερόμενο πίνακα>

ON UPDATE <ενέργεια που εκτελείται κατά τη ενημέρωση στοιχείου από τον αναφερόμενο πίνακα>

Οι ενέργειες που χρησιμοποιήθηκαν για να υποστηρίζονται οι περιορισμοί κατά τις αλλαγές είναι:

- **CASCADE** : Χρησιμοποιήθηκε ως αποτέλεσμα σε διαγραφή. Στην περίπτωση της διαγραφής στοιχείου από τον αναφερόμενο πίνακα, διαγράφονται και όσες καταχωρήσεις έχουν την τιμή της καταχώρησης που διαγράφηκε.

Περιορισμοί ακεραιότητας οντότητας

- **πρωτεύοντα κλειδιά** : Οι περιορισμοί πρωτευόντων κλειδιών δημιουργήθηκαν σύμφωνα με το σχεσιακό μοντέλο που χρησιμοποιήθηκε. Ορίζονται κατά την δημιουργία των πινάκων των σχέσεων με τη σύνταξη:

  PRIMARY KEY (<όνομα γνωρίσματος>)

- **NOT NULL** : Ο περιορισμός αυτός υποδεικνύει ότι η τιμή ενός γνωρίσματος δεν μπορεί να είναι NULL. Το σύστημα Βάσεων Δεδομένων δεν επιτρέπει αυτόματα να πραγματοποιηθεί οποιαδήποτε ενέργεια (εισαγωγή ή ενημέρωση παραβιάζει των περιορισμό). Οι περιορισμοί αυτοί δημιουργήθηκαν με βάση τη λογική σε γνωρίσματα (εκτός των κλειδιών για τα οποία ισχύει ανεξάρτητα ο περιορισμός) τα οποία θεωρήθηκε ότι είναι απαραίτητο να είναι γνωστά για τη σωστή λειτουργία της εφαρμογής.

Περιορισμοί πεδίου τιμών

Για τη δημιουργία περιορισμών πεδίου τιμών χρησιμοποιήθηκαν triggers, καθώς η MySQL δεν υποστηρίζει τον όρο check.

Χρησιμοποιήθηκαν ώστε να εξασφαλίζεται ότι:

- Γνωρίσματα όπως πρωτεύοντα κλειδιά (AFM, EmployeeNo κπλ.) θα είναι μη αρνητικοί αριθμοί
- Οικονομικά μεγέθη όπως η τιμή του ενοικίου θα είναι μη αρνητικοί αριθμοί
- Κάποια γνωρίσματα μπορούν να πάρουν μόνο συγκεκριμένες τιμές. Για παράδειγμα το γνώρισμα Active μπορεί να λάβει τις τιμές True και False.

Οι ορισμοί των triggers περιλαμβάνονται στο παράρτημα.

- **Ευρετήρια**

Το VS δημιουργεί αυτόματα ευρετήρια στα γνωρίσματα τα οποία είναι πρωτεύοντα και ξένα κλειδιά. Συνεπώς δεν θεωρήθηκε απαραίτητο να ορισθούν επιπλέον ευρετήρια, καθώς οι περισσότερες αναζητήσεις που γίνονται κατά τα ερωτήματα ή τις ενημερώσεις χρησιμοποιούν αυτά τα γνωρίσματα και ωφελούνται σε ταχύτητα από την ύπαρξη των ευρετηρίων.

➢ SQL Queries

Η επιλογή των ερωτημάτων έγινε με βάση την λειτουργικότητα τους, την χρησιμότητά τους σε μία πραγματική εφαρμογή και σύμφωνα με τους περιορισμούς που θεωρήσαμε ότι χρειαζόταν. Ένας επιπρόσθετος λόγος για την επιλογή των ερωτημάτων είναι το μεγάλο φάσμα των δυνατοτήτων της γλώσσας που προσφέρουν.

```
1.SELECT *
FROM Properties,Employees
where Employees.AFM= Properties.ManagerAFM and Properties.Rent>=Number
```

Το ερώτημα επιστρέφει από την βάση δεδομένων τα επιλεγμένα από τον χρήστη γνωρίσματα των πινάκων Properties και Employee με το ίδιο AFM των οποίων το Rent πληροί μια συνθήκη ορισμένη από τον χρήστη. Με αυτόν τον τρόπο κάποιος μπορεί να μάθει ποιο υποσύνολο των εργαζομένων είχε υπό την ευθύνη ακίνητα με ενοίκιο μεγαλύτερο ή ίσο του προκαθορισμένου.

```
2.SELECT *
FROM Properties,Newspapers,Advertisements
where Newspapers.NewspaperId= Advertisements.NewspaperId and
Properties.PropertyRegistrationNo= Advertisements.PropertyRegistrationNo
and Newspapers.NewspaperName ='Kathimerini'
```

Το ερώτημα επιστρέφει από την βάση δεδομένων τα επιλεγμένα από τον χρήστη γνωρίσματα των πινάκων Properties, Newspapers και Advertisements για όλες τις αγγελίες που είναι καταχωρημένες στην εφημερίδα Kathimerini.

```
3. SELECT min(Rent),max(Rent), avg(Rent)
 FROM Contracts
```

Επιστρέφει την ελάχιστη, την μέγιστη και τον μέσο όρο των Ενοικίων των συμβολαίων, χρήσιμα στοιχεία για το Οικονομικό τμήμα.

```
4. SELECT COUNT(Contracts.ContractNo),Employees.AFM
FROM Employees,Contracts,Clients
WHERE Employees.AFM = Clients.RegisteredBy AND
Clients.ClientRegistrationNo = Contracts.ClientRegistrationNo GROUP BY
Employees.AFM
```

Μετράει τα συμβόλαια που έχει κλείσει ο κάθε υπάλληλος ,χρήσιμο στοιχείο για την αξιολόγηση της απόδοσης των υπαλλήλων.

```
5. SELECT Owners.AFM,COUNT(Properties.PropertyRegistrationNo)
FROM Owners,Properties
WHERE Owners.AFM = Properties.OwnerAFM GROUP BY Owners.AFM HAVING
COUNT(Properties.PropertyRegistrationNo) > "+textBox1.Text;
```

Εμφανίζει στο χρήστη τους ιδιοκτήτες και τον αριθμό των ακινήτων τους , που έχουν στην κατοχή τους αριθμό ακινήτων μεγαλύτερο από αυτόν που επέλεξε ο χρήστης.

```
6. SELECT sum(Advertisements.Cost)

from  Advertisements, Properties

where  Properties.PropertyRegistrationNo=
Advertisements.PropertyRegistrationNo GROUP BY
Properties.PropertyRegistrationNo having
sum(Advertisements.Cost)>="+textBox1.Text+"  ORDER BY
sum(Advertisements.Cost) "
```

Εμφανίζει στο χρήστη τα ακίνητα με συνολικό κόστος διαφήμισης μεγαλύτερο ή ίσο από ένα επιλεγμένο νούμερο και ταξινομεί τα αποτελέσματα σύμφωνα με το κόστος αυτό σε αύξουσα σειρά.

```
7. SELECT Employees.AFM,Employees.Salary

from  Employees

where Employees.Salary > ( Select Salary from Employees where
AFM="+textBox1.Text +")" ;
```

Επιλέγει από τον πίνακα Employees τα πεδία AFM και Salary για τους υπαλλήλους με Μισθό μεγαλύτερο από αυτόν που επιλέγει ο χρήστης.

```
8. SELECT * from Visits
where (Visits.ClientRegistrationNo ="+textBox1.Text+")
```

Αυτό το Query εμφανίζει όλα τα πεδία από τον πίνακα των Επισκέψεων για ένα συγκεκριμένο πελάτη που επιλέγει ο χρήστης.

```
9. SELECT * from Clients where LastName like '%"+textBox1.Text+"%'
```

Βοηθάει το χρήστη να βρει το όνομα ενός πελάτη σύμφωνα με κάποιο γράμμα ή ένα κομμάτι από το επίθετο του.

```
10. SELECT distinct Owners.AFM,Properties.PropertyRegistrationNo
from Owners,Properties
where Properties.OwnerAFM=Owners.AFM AND Owners.AFM NOT IN (select
Owners.AFM from PrivateOwners,Owners where Owners.AFM=PrivateOwners.AFM)
```

Το τελευταίο Query επιστρέφει τα ακίνητα μαζί με το ΑΦΜ των Ιδιοκτητών που είναι Εταιρία και όχι ιδιώτης.

➢ Όψεις

```
CREATE VIEW [dbo].[View]
    AS SELECT ClientRegistrationNo,FirstName,LastName,MaxRent FROM [Clients]
```

Η όψη αυτή είναι ενημερώσιμη και μας δείχνει μόνο τα σημαντικότερα στοιχεία από τον πίνακα των πελατών και βοηθάει τον χρήστη να δει γρηγορότερα τα στοιχεία αυτά.

```
CREATE VIEW [dbo].[view_example]
    AS SELECT e.FirstName,e.LastName,e.WorkPhoneNumber,p.*
    FROM [Employees] as e
    INNER JOIN [Properties] AS p ON e.AFM = p.ManagerAFM
```

Η όψη αυτή είναι μη ενημερώσιμη (αφού κάνουμε INNER JOIN μεταξύ 2 πινάκων) και μας δείχνει 3 βασικά στοιχεία του υπαλλήλου(e.FirstName,e.LastName,e.WorkPhoneNumber) μαζί με όλα τα στοιχεία του ακινήτου του οποίου είναι υπεύθυνος.

➢ Triggers

```
CREATE TRIGGER [Trigger1]
    ON [dbo].[Employees]
    AFTER INSERT
    AS
    BEGIN
         UPDATE Employees SET SupervisorAFM= 102 ;
    END
```

Το trigger αυτό ενεργοποιείται μετά από ένα INSERT στον πίνακα Employees

και βάζει αρχικά για οποιοδήποτε υπάλληλο να έχει υπεύθυνο του τον υπάλληλο με AFM=102 που είναι ο υπεύθυνος προσωπικού. Στη συνέχεια ο χρήστης μπορεί να αλλάξει τον υπεύθυνο ενός υπαλλήλου μέσω της λειτουργίας του UPDATE.

```
CREATE TRIGGER [Trigger_delete_Client]
    ON [dbo].[Clients]
    FOR DELETE
    AS
    BEGIN
         SET NOCOUNT ON
         DELETE FROM [ClientPhones ] WHERE ClientRegistrationNo IN(SELECT
deleted.ClientregistrationNo FROM deleted)
         DELETE FROM [Visits] WHERE ClientRegistrationNo IN(SELECT
deleted.ClientregistrationNo FROM deleted)
         DELETE FROM [Contracts] WHERE ClientRegistrationNo IN(SELECT
deleted.ClientregistrationNo FROM deleted)
    END
```

Αυτό το trigger μετά από κάθε DELETE στον πίνακα Client ελέγχει στους πίνακες [Visits] και [Contracts] αν υπάρχουν καταχωρήσεις σε αυτούς για αυτόν τον πελάτη και τις διαγράφει.

➢ DDL Δημιουργίας Βάσης Δεδομένων

```
CREATE TABLE [dbo].[Advertisements] (
    [PropertyRegistrationNo] INT        NOT NULL,
    [NewspaperID]            INT        NOT NULL,
    [DateOfPublish]          NCHAR (20) NOT NULL,
    [Cost]                   MONEY      NOT NULL,
    [Duration]               NCHAR (20) NOT NULL,
    PRIMARY KEY CLUSTERED ([NewspaperID] ASC, [PropertyRegistrationNo]
ASC, [DateOfPublish] ASC),
    FOREIGN KEY ([NewspaperID]) REFERENCES [dbo].[Newspapers]
([NewspaperID]),
    FOREIGN KEY ([PropertyRegistrationNo]) REFERENCES [dbo].[Properties]
([PropertyRegistrationNo]),
    CHECK ([Cost]>=(0)),
    CHECK ([NewspaperID]>(0))
);

CREATE TABLE [dbo].[BusinessOwners] (
    [AFM]              INT        NOT NULL,
```

```
    [BusinessName]     NCHAR (50) NOT NULL,
    [BusinessType]     NCHAR (50) NULL,
    [ContactFirstName] NCHAR (50) NOT NULL,
    [ContactLastName]  NCHAR (50) NOT NULL,
    PRIMARY KEY CLUSTERED ([AFM] ASC),
    FOREIGN KEY ([AFM]) REFERENCES [dbo].[Owners] ([AFM])
);


CREATE TABLE [dbo].[ClientPhones ] (
    [ClientRegistrationNo] INT        NOT NULL,
    [PhoneNumber]          NCHAR (10) NOT NULL,
    PRIMARY KEY CLUSTERED ([ClientRegistrationNo] ASC, [PhoneNumber] ASC),
    FOREIGN KEY ([ClientRegistrationNo]) REFERENCES [dbo].[Clients]
([ClientRegistrationNo]) ON DELETE CASCADE
);


CREATE TABLE [dbo].[Clients] (
    [ClientRegistrationNo] INT        NOT NULL,
    [FirstName]            NCHAR (50) NOT NULL,
    [LastName]             NCHAR (50) NOT NULL,
    [Addr_StreetName]      NCHAR (50) NOT NULL,
    [Addr_StreetNo]        NCHAR (50) NOT NULL,
    [Addr_PostalCode]      NCHAR (50) NOT NULL,
    [RegistrationDate]     NCHAR (50) NOT NULL,
    [MaxRent]              MONEY      NOT NULL,
    [Active]               NCHAR (50) NULL,
    [RegisteredBy]         INT        NOT NULL,
    [PreferedTypeID]       INT        NOT NULL,
    PRIMARY KEY CLUSTERED ([ClientRegistrationNo] ASC),
    UNIQUE NONCLUSTERED ([ClientRegistrationNo] ASC),
    FOREIGN KEY ([RegisteredBy]) REFERENCES [dbo].[Employees] ([AFM]),
    FOREIGN KEY ([PreferedTypeID]) REFERENCES [dbo].[PropertyTypes]
([PropertyTypeID]),
    CHECK ([Active] = 'True'
           OR [Active] = 'False'),
    CHECK ([MaxRent] >= (0))
);


CREATE TABLE [dbo].[Contracts] (
```

```
    [ContractNo]              INT        NOT NULL,
    [Rent]                    MONEY      NOT NULL,
    [PaymentType]             NCHAR (50) NULL,
    [RentStar]                NCHAR (50) NOT NULL,
    [RentFinish]              NCHAR (50) NOT NULL,
    [ClientRegistrationNo]    INT        NOT NULL,
    [PropertyRegistrationNo] INT        NOT NULL,
    PRIMARY KEY CLUSTERED ([ContractNo] ASC),
    FOREIGN KEY ([ClientRegistrationNo]) REFERENCES [dbo].[Clients]
([ClientRegistrationNo]) ON DELETE CASCADE,
    FOREIGN KEY ([PropertyRegistrationNo]) REFERENCES [dbo].[Properties]
([PropertyRegistrationNo]),
Unique ([PropertyRegistrationNo])
);


CREATE TABLE [dbo].[Employees] (
    [AFM]               INT        NOT NULL,
    [EmployeeNo]        INT        NOT NULL,
    [FirstName]         NCHAR (50) NOT NULL,
    [LastName]          NCHAR (50) NOT NULL,
    [Addr_StreetName]   NCHAR (50) NOT NULL,
    [Addr_StreetNo]     NCHAR (50) NOT NULL,
    [Addr_PostCode]     INT        NOT NULL,
    [Salary]            MONEY      NOT NULL,
    [WorkPhoneNumber]   NCHAR (20) NULL,
    [MobilePhoneNumber] NCHAR (20) NULL,
    [SupervisorAFM]     NCHAR (10) NOT NULL,
    PRIMARY KEY CLUSTERED ([AFM] ASC),
    UNIQUE NONCLUSTERED ([EmployeeNo] ASC),
    CHECK ([Salary] >= (0)),
    CHECK ([AFM] > (0)),
    CHECK ([EmployeeNo] > (0))
);


CREATE TABLE [dbo].[Newspapers] (
    [NewspaperID]   INT        NOT NULL,
    [NewspaperName] NCHAR (30) NOT NULL,
    PRIMARY KEY CLUSTERED ([NewspaperID] ASC)
);
```

```
CREATE TABLE [dbo].[OwnerPhones] (
    [AFM]         INT        NOT NULL,
    [PhoneNumber] NCHAR (10) NOT NULL,
    PRIMARY KEY CLUSTERED ([PhoneNumber] ASC, [AFM] ASC),
    FOREIGN KEY ([AFM]) REFERENCES [dbo].[Owners] ([AFM])
);
```

```
CREATE TABLE [dbo].[Owners] (
    [AFM]              INT        NOT NULL,
    [Addr_StreetName] NCHAR (40) NOT NULL,
    [Addr_StreetNo]   NCHAR (50) NOT NULL,
    [Addr_PostalCode] NCHAR (40) NOT NULL,
    PRIMARY KEY CLUSTERED ([AFM] ASC),
    CHECK ([AFM]>=(0))
);
```

```
CREATE TABLE [dbo].[PrivateOwners] (
    [AFM]       INT        NOT NULL,
    [FirstName] NCHAR (40) NOT NULL,
    [LastName]  NCHAR (40) NOT NULL,
    PRIMARY KEY CLUSTERED ([AFM] ASC),
    FOREIGN KEY ([AFM]) REFERENCES [dbo].[Owners] ([AFM])
);
```

```
CREATE TABLE [dbo].[Properties] (
    [PropertyRegistrationNo] INT        NOT NULL,
    [Size]                   NCHAR (20) NOT NULL,
    [Floor]                  INT        NOT NULL,
    [Rent]                   MONEY      NOT NULL,
    [Addr_StreetName]        NCHAR (20) NOT NULL,
    [Addr_StreetNo]          NCHAR (20) NOT NULL,
    [Addr_PostalCode]        NCHAR (20) NULL,
    [PropertyTypeID]         INT        NOT NULL,
    [OwnerAFM]               INT        NOT NULL,
    [ManagerAFM]             INT        NOT NULL,
```

```
    PRIMARY KEY CLUSTERED ([PropertyRegistrationNo] ASC),
    FOREIGN KEY ([PropertyTypeID]) REFERENCES [dbo].[PropertyTypes]
([PropertyTypeID]),
    FOREIGN KEY ([ManagerAFM]) REFERENCES [dbo].[Employees] ([AFM]),
    FOREIGN KEY ([OwnerAFM]) REFERENCES [dbo].[Owners] ([AFM]),
    CHECK ([PropertyRegistrationNo]>=(0)),
    CHECK ([Floor]>=(0))
);


CREATE TABLE [dbo].[PropertyTypes] (
    [PropertyTypeID] INT   NOT NULL,
    [Description]    NTEXT NULL,
    [Rooms]          INT   NOT NULL,
    PRIMARY KEY CLUSTERED ([PropertyTypeID] ASC)
);


CREATE TABLE [dbo].[Visits] (
    [PropertyRegistrationNo] INT        NOT NULL,
    [ClientRegistrationNo]   INT        NOT NULL,
    [DateOfVisit]            NCHAR (50) NOT NULL,
    PRIMARY KEY CLUSTERED ([ClientRegistrationNo] ASC, [DateOfVisit] ASC,
[PropertyRegistrationNo] ASC),
    FOREIGN KEY ([PropertyRegistrationNo]) REFERENCES [dbo].[Properties]
([PropertyRegistrationNo]),
    FOREIGN KEY ([ClientRegistrationNo]) REFERENCES [dbo].[Clients]
([ClientRegistrationNo]) ON DELETE CASCADE
);
```