ΒΑΣΕΙΣ ΔΕΔΟΜΕΝΩΝ

ΕΞΑΜΗΝΙΑΙΑ ΕΡΓΑΣΙΑ

Ακαδημαϊκό έτος 2020-2021

Ομάδα ΒΤ

Αγγελική Εμμανουέλα Συρρή | ΑΜ : 03118811

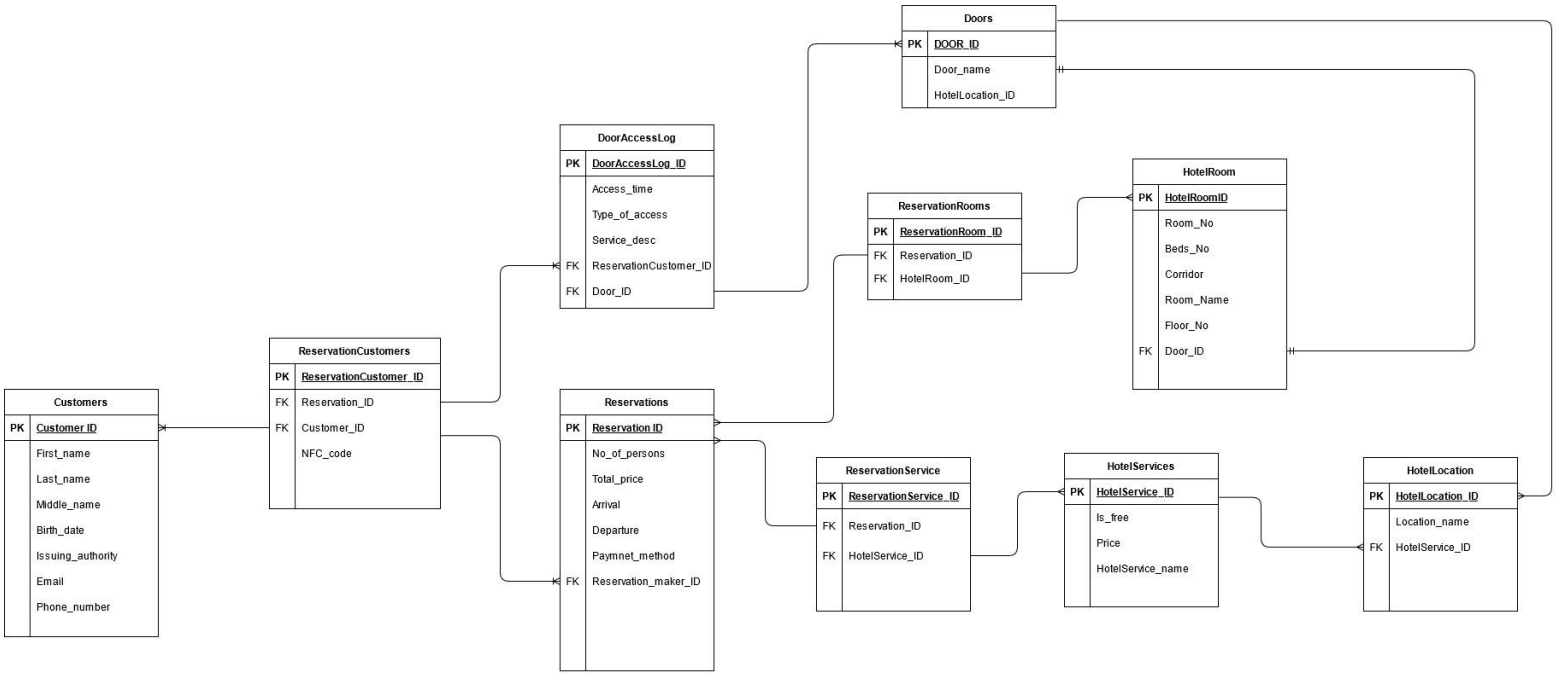
Γιώργος Παπαδούλης | ΑΜ : 03118003

Νικόλαος Μπέλλος | ΑΜ : 03118183

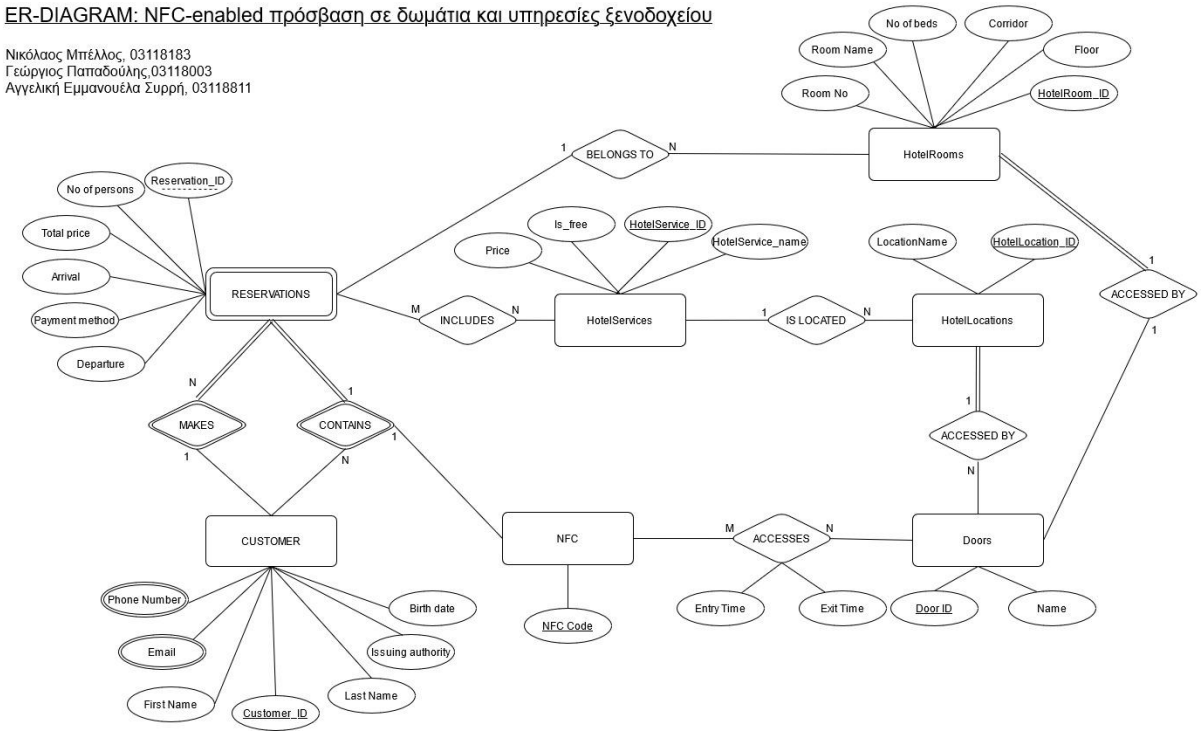
NFC-enabled πρόσβαση σε δωμάτια και υπηρεσίες ξενοδοχείου

Η ζητούμενη ξενοδοχειακή μονάδα αναπτύχθηκε με γνώμονα την ελαχιστοποίηση των επαφών των πελατών του ξενοδοχείου. Το παραπάνω αίτημα υποβοηθείται και από την χρήση του NFC-enabled bracelet, το οποίο καταγράφει όλες τις κινήσεις του πελάτη μέσα στο ξενοδοχείο και επομένως σε περίπτωση θετικής διάγνωσης βοηθάει στην ιχνηλάτηση και στην εύρεση άλλων πιθανών θετικών κρουσμάτων.

a. Το παρακάτω αποτελεί το σχεσιακό διάγραμμα που αντιστοιχεί στο ER διάγραμμα που παραδώσαμε.



Παρακάτω παραθέτουμε και το ER διάγραμμα, επειδή έχουν γίνει κάποιες μικρές αλλαγές σε αυτό.



b. Για την δημιουργία των παραπάνω πινάκων πινάκων στην SQL χρησιμοποιήσαμε το χαρακτηριστικό NOT NULL, όταν το χαρακτηριστικό αυτό θεωρούνταν απαραίτητο για την αρχικοποίηση και την φυσική υπόσταση του στοιχείου. Φυσικά τα primary keys δεν γίνεται να είναι NULL. Τα foreign keys τα ορίζουμε ως NOT NULL, ώστε να υπάρχει συνέπεια στην βάση και να συμφωνούν με τα primary keys των σχέσεων στις οποίες ανήκουν.

Query για τη δημιουργία της βάσης:

```
IF NOT EXISTS (SELECT name FROM master.dbo.sysdatabases
WHERE name = 'HotelManagement')
CREATE DATABASE [HotelManagement] COLLATE Greek_CI_AI -- Collatiom
: CI = Case Insensitive, AI = Accent Insensitive
GO
```

Query για τη δημιουργία του πίνακα Customers:

```
CREATE TABLE Customers
(
      Customer_ID int IDENTITY(1,1) PRIMARY KEY,
      First_name varchar (50) not null,
      Last_name varchar (50) not null,
```

```
      Birth_date datetime,
      Issuing_authority varchar (50) not null,
      Email varchar (50),
      Phone varchar (50)
)
```

Εδώ το Customer_ID είναι Primary Key, και αυτό συμβαίνει, διότι είναι το attribute εκείνο το οποίο είναι μοναδικό και ξεχωριστό για κάθε πελάτη.

Query για τη δημιουργία του πίνακα Reservations:

```
CREATE TABLE Reservations
(
      Reservation_ID int IDENTITY(1,1) PRIMARY KEY,
      Arrival datetime,
      Departure datetime,
      Total_price float,
      Payment_method varchar (50),
      No_persons int
)
```

```
ALTER TABLE Reservations
ADD Reservation_maker_ID int FOREIGN KEY REFERENCES
Customers(Customer_ID)
```

Εδώ το Reservation_ID είναι Primary Key, και αυτό συμβαίνει, διότι είναι το attribute εκείνο το οποίο είναι μοναδικό και ξεχωριστό για κάθε κράτηση. Επίσης, στον πίνακα Reservations το Reservation_maker_ID είναι foreign key για τον πίνακα Customer και αντιστοιχίζει τους δύο πίνακες σε σχέση πολλά(Reservations) προς ένα(Customer).

Query για τη δημιουργία του πίνακα Doors:

```
CREATE TABLE Doors
(
      Door_ID int IDENTITY(1,1) PRIMARY KEY,
      Door_name varchar (50)
)

ALTER TABLE Doors
ADD HotelLocation_ID int FOREIGN KEY REFERENCES
HotelLocations(HotelLocation_ID)
```

Στον παραπάνω πίνακα Doors, το Door_ID είναι το primary key, διότι αυτό το χαρακτηριστικό είναι ξεχωριστό και μοναδικό για κάθε πόρτα. Στον πίνακα Doors το

HotelLocation_ID είναι foreign key για τον πίνακα HotelLocations αντιστοιχίζοντας σε σχέση πολλά(Doors) προς ένα(HotelLocations) του δύο πίνακες.

Query για τη δημιουργία του πίνακα HotelServices:

```
CREATE TABLE HotelServices
(
      HotelService_ID int IDENTITY(1,1) PRIMARY KEY,
      HotelService_name varchar (50),
      Is_free bit,
      Price float
)
```

Στον παραπάνω πίνακα `HotelServices`, το `HotelService_ID` είναι το primary key, διότι αυτό το χαρακτηριστικό είναι ξεχωριστό και μοναδικό για κάθε υπηρεσία.

Query για τη δημιουργία του πίνακα HotelLocations:

```
CREATE TABLE HotelLocations
(
      HotelLocation_ID int IDENTITY(1,1) PRIMARY KEY,
      Location_name varchar (50)
)

ALTER TABLE HotelLocations
ADD HotelService_ID int FOREIGN KEY REFERENCES
HotelServices(HotelService_ID)
```

Στον παραπάνω πίνακα `HotelLocations`, το `HotelLocation_ID` είναι το primary key, διότι αυτό το χαρακτηριστικό είναι ξεχωριστό και μοναδικό για κάθε τοποθεσία. Επίσης, το HotelService_ID είναι foreign key στον πίνακα HotelServices αντιστοιχίζοντας του δύο πίνακες με μία σχέση πολλά (HotelLocations) σε ένα (HotelServices).

Query για τη δημιουργία του πίνακα HotelRooms:

```
CREATE TABLE HotelRooms
(
      HotelRoom_ID int IDENTITY(1,1) PRIMARY KEY,
      Room_No int,
      Room_name varchar (20),
      No_beds int,
      Corridor_No int,
      Floor_No int
)
```

```
ALTER TABLE HotelRooms
ADD Door_ID int FOREIGN KEY REFERENCES Doors(Door_ID)
```

Στον πίνακα HotelRooms το HotelRoom_Id είναι το primary key του καθώς αυτό χαρακτιρίζει μοναδικά κάθε εγγραφή σε αυτόν τον πίνακα. Επίσης, το Door_ID είναι Foreign key στον πίνακα Doors αντιστοιχίζοντας τους δύο πίνακες με μία σχέση πολλά (HotelRooms) προς ένα(Doors).

Query για τη δημιουργία του πίνακα ReservationCustomers:

```
CREATE TABLE ReservationCustomers
(
      ReservationCustomer_ID int IDENTITY(1,1) PRIMARY KEY,
      Customer_ID int FOREIGN KEY REFERENCES
Customers(Customer_ID),
      Reservation_ID int FOREIGN KEY REFERENCES
Reservations(Reservation_ID),
      NFC_code int
)
```

Ο πίνακας ReservationCustomers είναι μία σχέση (πολλά προς πολλά) που ενώνει τις οντότητες Customers και Reservations. Για αυτό το λόγο περιέχει εκτός από το primary key ReservationCustomer_ID που χαρακτηρίζει μοναδικά κάθε εγγραφή του, και δύο foreign keys ένα προς το Customers και ένα προς το Reservations. Επίσης, συμπεριλαμβάνει και το NFC_code κάθε πελάτη, το οποίο χαρακτηρίζει σε ένα χρονικό διάστημα (άφιξη - αναχώρηση) τον συγκεκριμένο πελάτη.

Query για τη δημιουργία του πίνακα DoorAccessLog:

```
CREATE TABLE DoorAccessLog
(
      DoorAccessLog_ID int IDENTITY(1,1) PRIMARY KEY,
      Entry_time datetime,
      Exit_time datetime,
      Service_desc varchar (255),
      ReservationCustomer_ID int FOREIGN KEY REFERENCES
ReservationCustomers(ReservationCustomer_ID),
      Door_ID int FOREIGN KEY REFERENCES Doors(Door_ID)
)
```

Ο πίνακας DoorAccessLog είναι μία σχέση (πολλά προς πολλά) που ενώνει τις οντότητες `ReservationCustomers` και `Doors`. Για αυτό το λόγο περιέχει εκτός από το primary key `DoorAccessLog_ID` που χαρακτηρίζει μοναδικά κάθε εγγραφή του, και δύο foreign keys ένα προς το `ReservationCustomers` και ένα προς το `Doors`. Επίσης, συμπεριλαμβάνει και τα `Entry_time, Exit_time, Service_desc` κάθε μετακίνησης (πρόσβασης ή εξόδου από κάθε χώρο).

Query για τη δημιουργία του πίνακα ReservationServices:

```
CREATE TABLE ReservationServices
(
      ReservationService_ID int IDENTITY(1,1) PRIMARY KEY,
      Reservation_ID int FOREIGN KEY REFERENCES
Reservations(Reservation_ID),
      HotelService_ID int FOREIGN KEY REFERENCES
HotelServices(HotelService_ID)
)
```

Ο πίνακας `ReservationServices` είναι μία σχέση (πολλά προς πολλά) που ενώνει τις οντότητες `Reservations` και `HotelServices`. Για αυτό το λόγο περιέχει εκτός από το primary key `ReservationService_ID` που χαρακτηρίζει μοναδικά κάθε εγγραφή του, και δύο foreign keys ένα προς το `Reservations` και ένα προς το `HotelServices`.

Query για τη δημιουργία του πίνακα ReservationRooms:

```
CREATE TABLE ReservationRooms
(
      ReservationRoom_ID int IDENTITY(1,1) PRIMARY KEY,
      Reservation_ID int FOREIGN KEY REFERENCES
Reservations(Reservation_ID),
      HotelRoom_ID int FOREIGN KEY REFERENCES
HotelRooms(HotelRoom_ID)
)
```

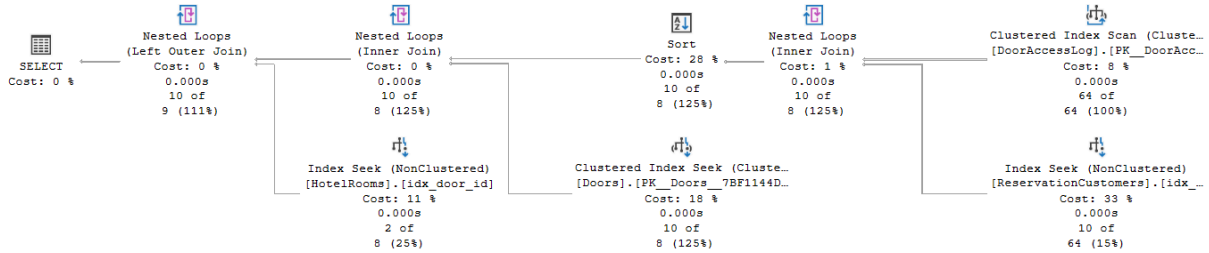
Ο πίνακας `ReservationRooms`είναι μία σχέση (πολλά προς πολλά) που ενώνει τις οντότητες `Reservations` και `HotelRooms`. Για αυτό το λόγο περιέχει εκτός από το primary key `ReservationService_ID` που χαρακτηρίζει μοναδικά κάθε εγγραφή του, και δύο foreign keys ένα προς το `Reservations` και ένα προς το `HotelRooms`.

b. Έχουν οριστεί 5 ευρετήρια, τα οποία ορίστηκαν με γνώμονα την ελαχιστοποίηση του χρόνου που απαιτείται για την υλοποίηση των queries.

```
CREATE INDEX idx_reservation_id ON
ReservationCustomers(Reservation_ID)
CREATE INDEX idx_nfc ON ReservationCustomers(NFC_Code)
CREATE INDEX idx_door_id ON DoorAccessLog(Door_ID)
CREATE INDEX idx_hotellocation_id ON Doors(HotelLocation_ID)
```

```
CREATE INDEX idx_door_id ON HotelRooms(Door_ID)
```

Για να καταλήξουμε στις βέλτιστες επιλογές για indexes, ενεργοποιούμε στο Workbench την επιλογή για ενσωμάτωση του Actual Execution Plan, το οποίο μας ενημερώνει για τους πόρους επί τις εκατό που καταναλώνει ένα query για την εκτέλεση της κάθε sql εντολής (πχ inner/left join). Το παρακάτω είναι ένα execution plan για το Query 9, και παρατηρούμε ότι γίνεται χρήση 2 NonClustered Indexes, τα οποία βοηθούν σημαντικά στην μείωση υπολογιστικών πόρων.



Τα ευρετήρια επιλέγονται με βάση του χαρακτηριστικού εκείνου, σύμφωνα με το οποίο γίνονται οι περισσότερες αναζητήσεις ή εκτελούνται οι συναρτήσεις της SQL.

c. Τεχνολογία ανάπτυξης εφαρμογής

Για την διαχείριση και ανάπτυξη της βάσης χρησιμοποιήθηκε το Microsoft SQL Express Server 2019 και για DBMS χρησιμοποιήθηκε SQL Server Management Studio. Για το στήσιμο του web server χρησιμοποιείται flask (Python) και για την σύνδεση μεταξύ βάσης και του server χρησιμοποιείται ένας `pypyodbc connector`. Για το UI χρησιμοποιήθηκε HTML,CSS και Java Script. Για την επικοινωνία μεταξύ `backend` και `frontend` χρησιμοποιούνται μέθοδοι `GET` και `Post`.
`Version:`

- sql server 2019
- Flask 2.0.1
- pypyodbc 1.3.4

d. Βήματα εγκατάστασης σε λογισμικό Windows.

1. Απαιτείται η εγκατάσταση του Microsoft SQL Express Server 2019 Download.
2. Αφού έχει γίνει η εγκατάσταση συνδεόμαστε στον server μέσω μιας DBMS και προτείνουμε την χρήση του SQL Server Management Studio. Συνδεόμαστε με SQL Server Authentication, με Server Name το όνομα του υπολογιστή και κάνουμε login με username sa, για να έχουμε όλα τα δικαιώματα, και τον κωδικό που επιθυμούμε.

3. Αφού συνδεθούμε επιτυχώς, μπορούμε να πάμε να δημιουργήσουμε όλους τους πίνακες και τις απαραίτητες συνδέσεις μεταξύ τους. Στην εργασία έχουμε επισυνάψει τα SQL Scripts που περιέχουν μέσα τα Scripts για την δημιουργία των παραπάνω.
   Για ευκολία έχουμε επισυνάψει .bak αρχείο, με το οποίο μπορεί κανείς άμεσα κάνοντας restore την βάση δεδομένων, να λάβει όλους τους πίνακες, με τις συνδέσεις και τα δεδομένα.
4. Στην συνέχεια, βάζουμε τα δεδομένα στην βάση. Τα δεδομένα μας αρχικά τα βάλαμε σε ένα αρχείο .xlsx και για να μην υπάρξουν συγκρούσεις στην χρήση των foreign keys, βάζουμε τους πίνακες με μια συγκεκριμένη σειρά μέσα στην βάση. Χρησιμοποιούμε το Import / Export wizard του Microsoft Management Studio και τοποθετούμε τους πίνακες με την ακόλουθη σειρά:

Customers
Reservations
HotelServices
HotelLocations
Doors
HotelRooms
ReservationCustomers
ReservationServices
ReservationRooms
DoorAccessLog

   Παράλληλα με την εισαγωγή των πινάκων φροντίζουμε να έχει ενεργοποιηθεί το identity insert, το οποίο μπορεί κανείς να βρεί στην επιλογή Edit Mappings.

5. Αφού έχουν οριστεί οι πίνακες, έχει γίνει εισαγωγή των δεδομένων, έχουν δημιουργηθεί τα απαραίτητα ευρετήρια και τα views, προχωράμε στο στήσιμο του περιβάλλοντος πάνω στο οποίο τρέχει η εφαρμογή μας.
   Εγκαθιστούμε τις απαραίτητες βιβλιοθήκες:

```
pip install Flask
pip install pypyodbc
```

Ύστερα ανοίγουμε τα αρχεία που επισυνάπτονται σε κάποιον editor και αντικαθιστούμε στο αρχείο app.py, τα credentials που ορίσαμε στο βήμα 2. Πιο συγκεκριμένα:

```
ql_user = '**'
sql_password = '****'
sql_server_name = '*******'
sql_database_name = 'HotelManagement'
```

Αφού τοποθετήσουμε, όλα τα αρχεία και σιγουρευτούμε ότι είναι στους σωστούς φακέλους, μπορούμε να τρέξουμε την εφαρμογή μας σε κάποιον localhost. Για να μπορούμε επιτυχώς να το κάνουμε αυτό, πρέπει να βρισκόμαστε στον ίδιο φάκελο που είναι και το αρχείο app.py (Projects) και εκτελούμε τις εντολές:

```
cd Project
python -m flask run
```

Το project θα τρέχει στον localhost που θα υποδειχθεί στο terminal, όταν εκτελεστούν οι παραπάνω εντολές.