Ανάλυση Σεναρίου – Αναγκαιότητα δημιουργίας Βάσης Δεδομένων

Εισαγωγή

Η αύξηση της εργασίας (όλο και περισσότερα άτομα αγοράζουν αυτοκίνητα, με αποτέλεσμα να αυξάνονται και οι αριθμοί επισκευών), η παροχή 3 υπηρεσιών από την αντιπροσωπεία αυτοκινήτων GoCar και η ανάγκη μείωσης χρόνου των εργασιών για τη σωστή λειτουργία της επιχείρησης είναι οι πιο σημαντικοί από τους λόγους βελτίωσης της διαχείρισης και αποθήκευσης των πληροφοριών των πελατών. Η λύση που προτείνεται είναι η δημιουργία μιας ολοκληρωμένης βάσης δεδομένων, η οποία θα βοηθάει σε καθημερινή βάση τους πωλητές στην ταχεία ενημέρωση των απαραίτητων δεδομένων αλλά και στην εύκολη μετάβαση από τον κλασικό τρόπο αποθήκευσής τους στον σύγχρονο τρόπο με τη χρήση ηλεκτρονικών υπολογιστών.

Οι Σύμβουλοί μας προσφέρουν τη καλύτερη λύση για αυτό το πρόβλημα. Η πλήρης εφαρμογή της βάσης δεδομένων θα πάρει λιγότερο από 5 βδομάδες. Το διάστημα αυτό αρκεί, ώστε να εκπαιδευτεί το προσωπικό που θα την χρησιμοποιεί. Ύστερα, από την δημιουργία της βάσης, κάθε πωλητής θα έχει τη δυνατότητα να διαχειρίζεται τα δεδομένα για κάποιον πελάτη, αγορά ή επισκευή, από τον προσωπικό του υπολογιστή. Με κάθε τηλέφωνο ή email που δέχεται, θα μπορεί να βρει οποιαδήποτε πληροφορία χρειαστεί για να εξυπηρετήσει τον εκάστοτε πελάτη.

Οι πωλητές θα έχουν την δυνατότητα να προσθέτουν νέους πελάτες και να δημιουργούν λίστες πελατών για τη προώθηση συγκεκριμένων υπηρεσιών ή αυτοκινήτων. Η διαδικασία εγγραφής / τροποποίησης των στοιχείων θα είναι άμεση, με αποτέλεσμα την ελάχιστη ταλαιπωρία του πελάτη, αφού θα έχει τη δυνατότητα να μιλήσει απευθείας με τον πωλητή και να πραγματοποιήσει οποιαδήποτε ενέργεια. Θα σταματήσουν, λοιπόν, άσκοπα τηλέφωνα, κακές συνεννοήσεις και χρονοβόρες διαδικασίες.

Επιχείρηση και Αποστολή

Η αντιπροσωπεία αυτοκινήτων GoCar είναι μια μικρή αλλά γρήγορα αναπτυσσόμενη επιχείρηση, που επιθυμεί να δώσει στον πελάτη το καλύτερο δυνατό αποτέλεσμα. Αποστολή της είναι η γρήγορη και σωστή εξυπηρέτηση των πελατών της. Ένα σύγχρονο, ενημερωμένο και φιλικό προς το χρήστη περιβάλλον είναι κάτι που θα συμβάλλει σίγουρα στην επίτευξη αυτού του στόχου. Οι πελάτες θα νιώθουν ασφαλείς με τις αγορές τους, η

διαδικασία θα είναι πιο εύκολη για αυτούς και τους πωλητές και θα επιτυγχάνονται ακόμα καλύτερα αποτελέσματα.

Απαιτήσεις Πληροφοριών

Για την ορθή λειτουργία του συγκεκριμένου συστήματος, απαραίτητες είναι ορισμένες πληροφορίες. Χρειάζονται όλα τα μέχρι τώρα δεδομένα πελατών, όπως το όνομά τους, το τηλέφωνό τους, τις αγορές/επισκευές που έχουν κάνει και άλλα στοιχεία επικοινωνίας. Επίσης, χρειάζονται πληροφορίες για τα αυτοκίνητα που έχουν ήδη πωληθεί, όπως το μοντέλο τους, η τιμή τους, η χρονολογία τους, καθώς και οι αντίστοιχες πληροφορίες για τα τωρινά διαθέσιμα αυτοκίνητα. Ακόμα, χρειάζονται πληροφορίες για τις υπηρεσίας επισκευής αυτοκινήτων, όπως ο τίτλος της εργασίας, το εύρος τιμής της, το χρονικό διάστημα που απαιτείται για την ολοκλήρωσή της.

Επιχειρησιακοί κανόνες

Θα πρέπει να υπάρχει η δυνατότητα ο κάθε πωλητής να μπορεί να εξυπηρετήσει τους πελάτες είτε αυτοί θελήσουν να εξυπηρετηθούν δια ζώσης είτε μέσω τηλεφώνου ή email. Άρα, θα πρέπει να υπάρχει η αντίστοιχη εξοικείωση και γνώση των πωλητών με το σύστημα βάσης δεδομένων, ώστε να μπορούν να αποθηκεύουν και τροποποιούν δεδομένα σχετικά με τις ανάγκες του κάθε πελάτη. Να μπορούν, δηλαδή, να αποθηκεύουν μία νέα αγορά αυτοκινήτου, από υπάρχοντα ή νέο πελάτη, να μπορούν να αποθηκεύουν μία νέα επισκευή αυτοκινήτου, από υπάρχοντα ή νέο πελάτη, αλλά και να μπορούν να κάνουν οποιαδήποτε αλλαγή κριθεί απαραίτητη για ήδη κλεισμένες συμφωνίες.

Υποθέσεις

Οι πελάτες γνωρίζουν ότι όσα δεδομένα περαστούν στη βάση, θα είναι διαθέσιμα από όλους τους πωλητές και θα είναι επίσης διαθέσιμα προς επεξεργασία για τις προτιμήσεις τους και για διαφημιστικά mails. Ακόμα, με αυτό το σύστημα δίνεται η δυνατότητα ένας πελάτης να επικοινωνήσει με πάνω από έναν πωλητή για την ολοκλήρωση μιας συμφωνίας. Πρέπει, δηλαδή, ο πελάτης να είναι ενήμερος για το συγκεκριμένο σενάριο. Επίσης, πρέπει να γνωρίζει ότι με την αποθήκευση των ήδη διαθέσιμων (σε χειρόγραφη

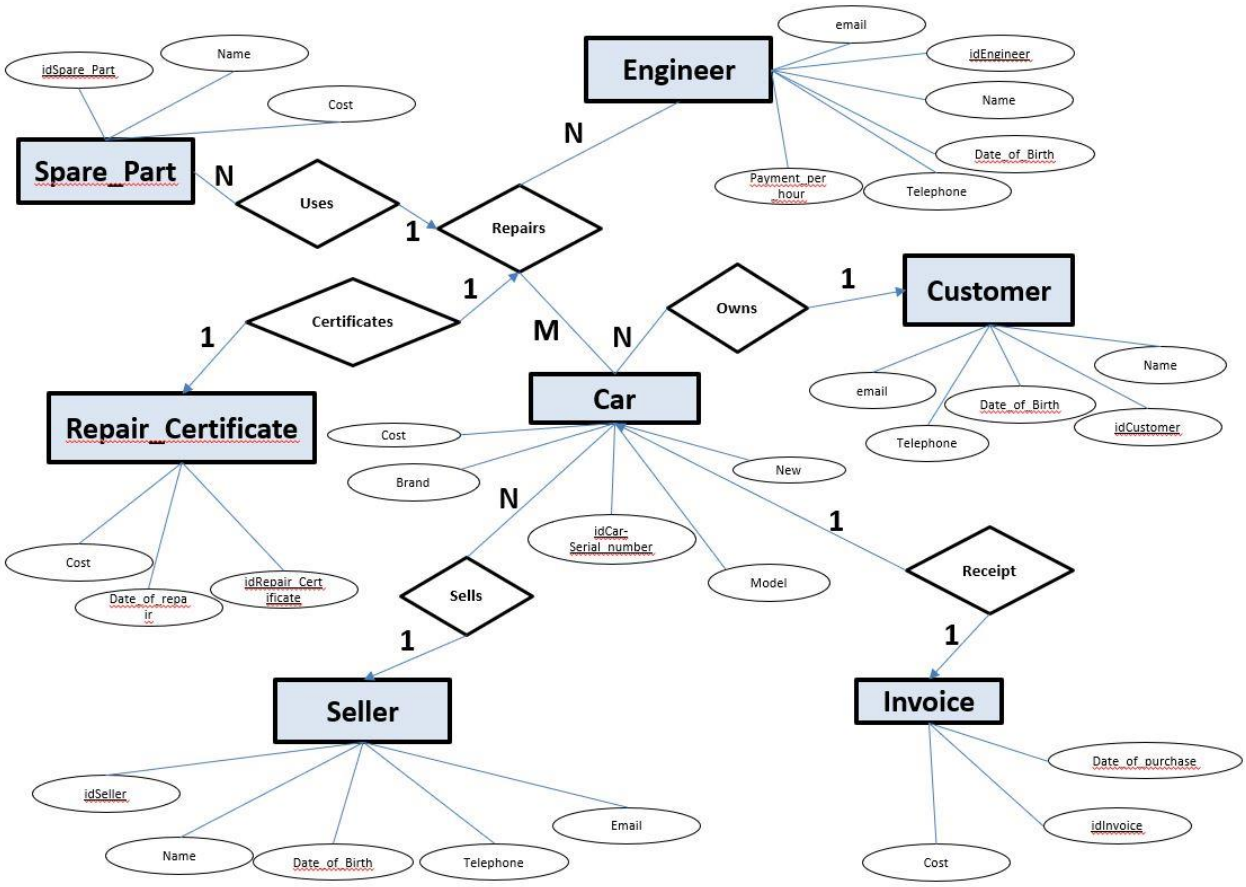
μορφή) δεδομένων, διατηρούνται σε ηλεκτρονικό αρχείο όσες πληροφορίες έχουν ήδη γίνει γνωστές στην επιχείρηση.

Διάγραμμα ERD

Επεξήγηση διαγράμματος

Στο αρχείο finalERD.pptx βρίσκεται το διάγραμμα ERD, με τη χρήση των σχημάτων και συμβόλων του παραδείγματος που υπήρχε. Έχουν δημιουργηθεί οι πίνακες Car, Customer, Seller, Engineer, Invoice, Repair_Certificate και Spare_Part, αφού αυτοί κρίθηκαν απαραίτητοι από την εκφώνηση του σεναρίου για την επιχείρηση GoCar. Στο διάγραμμα φαίνονται αναλυτικά όλες οι συσχετίσεις μεταξύ των οντοτήτων, τα πεδία της κάθε ενότητας, καθώς και ο πληθικός λόγος κάθε συσχέτισης.

Επίσης, το διάγραμμα φαίνεται και στη παρακάτω εικόνα.



Παρουσίαση των Πινάκων και των Χαρακτηριστικών τους

Οι συσχετίσεις των παρακάτω πινάκων φαίνονται στο προηγούμενο σχήμα

Car (idCar_Serial_number, Model, New, Brand, Cost)

Seller (idSeller, Name, Date_of_Birth, Telephone, Email)

Engineer (idEngineer, Name, Date_of_Birth, Email, Payment_per_hour)

Customer (idCustomer, Name, Date_of_Birth, Email, Telephone)

Invoice (idInvoice, Cost, Date_of_purchase)

Spare_Part (idSpare_Part, Name, Cost)

Repair_Certificate (idRepair_Certificate, Date_of_repair, Cost)

Μεταφορά ERD σε SQL

Στο αρχείο GoCar.sql βρίσκεται ο κώδικας για την αναπαράσταση του παραπάνω διαγράμματος, σε μια sql βάση δεδομένων. Οι εντολές που βρίσκονται σε αυτό το αρχείο δημιουργούν τους απαραίτητους πίνακες, συνδέσεις και πεδία για την ολοκλήρωση της βάσης δεδομένων GoCar. Οι εντολές δημιουργίας των πινάκων βρίσκονται, επίσης, στις παρακάτω εικόνες.

- Car

```
CREATE TABLE IF NOT EXISTS `GoCar`.`Car` (
  `idCar-Serial_number` INT NOT NULL,
  `Brand` VARCHAR(45) NOT NULL,
  `Model` VARCHAR(45) NOT NULL,
  `Release_Year` INT NOT NULL,
  `New` TINYINT NOT NULL,
  `Cost` INT NOT NULL,
  `Seller_idSeller` INT NOT NULL,
  `Customer_idCustomer` INT NOT NULL,
  `Invoice_idInvoice` INT NOT NULL,
  PRIMARY KEY (`idCar-Serial_number`, `Seller_idSeller`, `Customer_idCustomer`),
  INDEX `fk_Car_Seller_idx` (`Seller_idSeller` ASC) VISIBLE,
  INDEX `fk_Car_Customer1_idx` (`Customer_idCustomer` ASC) VISIBLE,
  INDEX `fk_Car_Invoice1_idx` (`Invoice_idInvoice` ASC) VISIBLE,
  CONSTRAINT `fk_Car_Seller`
    FOREIGN KEY (`Seller_idSeller`)
    REFERENCES `GoCar`.`Seller` (`idSeller`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_Car_Customer1`
    FOREIGN KEY (`Customer_idCustomer`)
    REFERENCES `GoCar`.`Customer` (`idCustomer`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_Car_Invoice1`
    FOREIGN KEY (`Invoice_idInvoice`)
    REFERENCES `GoCar`.`Invoice` (`idInvoice`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;
```

- Engineer

```
CREATE TABLE IF NOT EXISTS `GoCar`.`Engineer` (
  `idEngineer` INT NOT NULL,
  `Name` VARCHAR(45) NOT NULL,
  `Date_of_Birth` DATE NOT NULL,
  `Telephone` VARCHAR(15) NOT NULL,
  `email` VARCHAR(45) NOT NULL,
  `Payment_per_hour` INT NOT NULL,
  PRIMARY KEY (`idEngineer`))
ENGINE = InnoDB;
```

- Customer

```
CREATE TABLE IF NOT EXISTS `GoCar`.`Customer` (
  `idCustomer` INT NOT NULL,
  `Name` VARCHAR(45) NOT NULL,
  `Date_of_Birth` DATE NOT NULL,
  `Telephone` VARCHAR(15) NOT NULL,
  `email` VARCHAR(45) NOT NULL,
  PRIMARY KEY (`idCustomer`))
ENGINE = InnoDB;
```

- Seller

```
CREATE TABLE IF NOT EXISTS `GoCar`.`Seller` (
  `idSeller` INT NOT NULL,
  `Name` VARCHAR(45) NOT NULL,
  `Date_of_Birth` DATE NOT NULL,
  `Telephone` VARCHAR(15) NOT NULL,
  `email` VARCHAR(45) NOT NULL,
  PRIMARY KEY (`idSeller`))
ENGINE = InnoDB;
```

- Spare_Part

```
CREATE TABLE IF NOT EXISTS `GoCar`.`Spare_Part` (
  `idSpare_Part` INT NOT NULL,
  `Name` VARCHAR(45) NOT NULL,
  `Cost` INT NOT NULL,
  `Engineer_repairs_Car_idRepair` INT NOT NULL,
  PRIMARY KEY (`idSpare_Part`),
  INDEX `fk_Spare_Part_Engineer_repairs_Car1_idx` (`Engineer_repairs_Car_idRepair` ASC) VISIBLE,
  CONSTRAINT `fk_Spare_Part_Engineer_repairs_Car1`
    FOREIGN KEY (`Engineer_repairs_Car_idRepair`)
    REFERENCES `GoCar`.`Engineer_repairs_Car` (`idRepair`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;
```

- Invoice

```
CREATE TABLE IF NOT EXISTS `GoCar`.`Invoice` (
  `idInvoice` INT NOT NULL,
  `Cost` INT NOT NULL,
  `Date_of_purchase` DATE NOT NULL,
  PRIMARY KEY (`idInvoice`))
ENGINE = InnoDB;
```

- Repair_Certificate

```
CREATE TABLE IF NOT EXISTS `GoCar`.`Repair_Certificate` (
  `idRepair_Certificate` INT NOT NULL,
  `Cost` INT NOT NULL,
  `Date_of_repair` DATE NOT NULL,
  PRIMARY KEY (`idRepair_Certificate`))
ENGINE = InnoDB;
```

- Engineer_repairs_Car

```
CREATE TABLE IF NOT EXISTS `GoCar`.`Engineer_repairs_Car` (
  `idRepair` INT NOT NULL,
  `Engineer_idEngineer` INT NOT NULL,
  `Car_idCar-Serial_number` INT NOT NULL,
  `Car_Seller_idSeller` INT NOT NULL,
  `Car_Invoice_idInvoice` INT NOT NULL,
  `Car_Customer_idCustomer` INT NOT NULL,
  `Working_Hours` INT NOT NULL,
  `Repair_Certificate_idRepair_Certificate` INT NOT NULL,
  PRIMARY KEY (`idRepair`),
  INDEX `fk_Engineer_has_Car_Car1_idx` (`Car_idCar-Serial_number` ASC, `Car_Seller_idSeller` ASC, `Car_Invoice_idInvoice` ASC,
  INDEX `fk_Engineer_has_Car_Engineer1_idx` (`Engineer_idEngineer` ASC) VISIBLE,
  INDEX `fk_Engineer_repairs_Car_Repair_Certificate1_idx` (`Repair_Certificate_idRepair_Certificate` ASC) VISIBLE,
  CONSTRAINT `fk_Engineer_has_Car_Engineer1`
    FOREIGN KEY (`Engineer_idEngineer`)
    REFERENCES `GoCar`.`Engineer` (`idEngineer`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_Engineer_has_Car_Car1`
    FOREIGN KEY (`Car_idCar-Serial_number` , `Car_Seller_idSeller` , `Car_Customer_idCustomer`)
    REFERENCES `GoCar`.`Car` (`idCar-Serial_number` , `Seller_idSeller` , `Customer_idCustomer`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_Engineer_repairs_Car_Repair_Certificate1`
    FOREIGN KEY (`Repair_Certificate_idRepair_Certificate`)
    REFERENCES `GoCar`.`Repair_Certificate` (`idRepair_Certificate`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;
```

Εισαγωγή δεδομένων

Στο αρχείο Insert.sql βρίσκεται ο κώδικας για την εισαγωγή 3 εγγραφών στη βάση δεδομένων gocar. Η 1η εισαγωγή προσθέτει έναν πελάτη, η 2η προσθέτει έναν μηχανικό και η 3η προσθέτει έναν πωλητή. Για περισσότερες λεπτομέρειες μπορεί να μελετηθεί το αρχείο με τον κώδικα, όπως φαίνεται και στη παρακάτω εικόνα.

```sql
-- The following commands insert registrations to the database

INSERT INTO customer
VALUES (1, 'Kostas Papadopoulos', DATE '2021-06-05', '2101234567', 'test@gmail.com');

INSERT INTO engineer
VALUES (2, 'Giannis Papadakis', DATE '2020-06-05', '6912345678', 'test1@gmail.com', 10);

INSERT INTO seller
VALUES (3, 'Dimitris Papaloukas', DATE '2019-06-05', '6987654321', 'test2@gmail.com');

```

Ερωτήματα επιλογής

Στο αρχείο Select.sql βρίσκεται ο κώδικας για την επιλογή των εγγραφών, που περιγράφονται στην εκφώνηση. Ο κώδικας περιλαμβάνει σχόλια και αναλυτικά τις εντολές για την επιλογή των δεδομένων που ζητούνται. Τα ερωτήματα φαίνονται ένα-ένα και στις παρακάτω εικόνες.

- Ερώτημα 1

```sql
-- 1st query
SELECT *
FROM car c
WHERE c.Release_Year >= 2000 AND c.Release_Year <= 2010 AND Brand = 'Ford';
```

- Ερώτημα 2

```sql
-- 2nd query
SELECT c.Seller_idSeller, count(*) AS 'Sales'
FROM car c, invoice i
WHERE c.Invoice_idInvoice = i.idInvoice AND i.Date_of_Purchase = 2020
GROUP BY c.Seller_idSeller;
```

- Ερώτημα 3

```
-- 3rd query
SELECT e.idRepair
FROM engineer_repairs_car e, repair_certificate r
WHERE e.Repair_Certificate_idRepair_Certificate = r.idRepair_Certificate AND r.Cost = (
    SELECT max(Cost)
    FROM repair_certificate r
    WHERE r.Date_of_repair >= DATE '2021-04-01' AND r.Date_of_repair <= DATE '2021-04-30');

SELECT e.idRepair
FROM engineer_repairs_car e, repair_certificate r
WHERE e.Repair_Certificate_idRepair_Certificate = r.idRepair_Certificate AND r.Cost = (
    SELECT min(Cost)
    FROM repair_certificate r
    WHERE r.Date_of_repair >= DATE '2021-04-01' AND r.Date_of_repair <= DATE '2021-04-30');

SELECT e.idRepair
FROM engineer_repairs_car e, repair_certificate r
WHERE e.Repair_Certificate_idRepair_Certificate = r.idRepair_Certificate AND r.Cost = (
    SELECT avg(Cost)
    FROM repair_certificate r
    WHERE r.Date_of_repair >= DATE '2021-04-01' AND r.Date_of_repair <= DATE '2021-04-30');
```

- Ερώτημα 4

```
-- 4th query
SELECT count(*) AS 'Number of spare_parts'
FROM engineer_repairs_car e, spare_part s
WHERE s.Engineer_repairs_Car_idRepair = e.idRepair AND e.Repair_Certificate_idRepair_Certificate IN (
    SELECT r.idRepair_Certificate
    FROM repair_certificate r
    WHERE r.Date_of_repair >= DATE '2021-05-01' AND r.Date_of_repair <= DATE '2021-05-31');
```

- Ερώτημα 5

```
-- 5th query
SELECT c.idCustomer, c.Name
FROM engineer_repairs_car e, customer c, repair_certificate r
WHERE e.Car_Customer_idCustomer = c.idCustomer AND r.idRepair_Certificate = e.Repair_Certificate_idRepair_Certificate
AND r.Date_of_Repair >= DATE '2021-01-01' AND r.Date_of_Repair <= DATE '2021-12-31';
```

- Ερώτημα 6

```
-- 6th query
SELECT avg(s.Cost) AS 'Average cost of spare_parts'
FROM spare_part s, engineer_repairs_car e, repair_certificate r
WHERE s.Engineer_repairs_Car_idRepair = e.idRepair AND e.Repair_Certificate_idRepair_Certificate = r.idRepair_Certificate
AND r.Date_of_repair >= DATE '2020-01-01' AND r.Date_of_repair <= DATE '2020-12-31';
```

- Ερώτημα 7

```
-- 7th query
SELECT n.Name, count(*) AS 'Number of cars fixed'
FROM engineer_repairs_car e, engineer n
WHERE e.Engineer_idEngineer = n.idEngineer AND e.Repair_Certificate_idRepair_Certificate IN (
    SELECT r.idRepair_Certificate
    FROM repair_certificate r
    WHERE r.Date_of_repair >= DATE '2020-01-01' AND r.Date_of_repair <= DATE '2020-12-31')
GROUP BY e.Engineer_idEngineer;
```