Magazin de piese auto

Student: Heghea Mihail-Cristian

Grupa: 1309A

Coordonator: Cătălin Mironeanu

Tema baze de date – Magazin de piese auto

Enunt:

Sa se construiasca o aplicatie pentru gestiunea unui magazin de piese auto. Specificatii:

- Sa se poata adauga/sterge modele/marci de masini;
- Sa se poata adauga produse si stock pentru ele;
- Sa se poata adauga categorii de produse;
- Sa se poata crea comenzi, folosind cateva informatii despre client;
- Sa se poata vizualiza comenzile facuta in trecut.

Pentru realizarea functionalitatilor minimale ale unei astfel de aplicatii, baza de date contine urmatoarele tabele:

- ANGAJATI;
- CLIENTI;
- PRODUSE;
- CATEGORII;
- MARCI_MASINI;
- MODELE_MASINI;
- COMENZI.

Tabelul ANGAJATI contine informatii despre angajati(nume, data_nasterii, data_angajarii)

Tabelul CLIENTI contine informatii despre fiecare clienti(nume, telefon, serie buletin, numar buletin).

Tabelul PRODUSE contine informatii despre produse(denumire, pret, stock, etc).

Tabelul CATEGORII contine categoriile din care fac parte piesele, exemplu: ‘sistem racire’, ‘suspensii’, etc.

Tabelele MARCI_MASINI, MODELE_MASINI contine marcile, respeciv modelele de masini pentru care se gasesc piese in magazin.

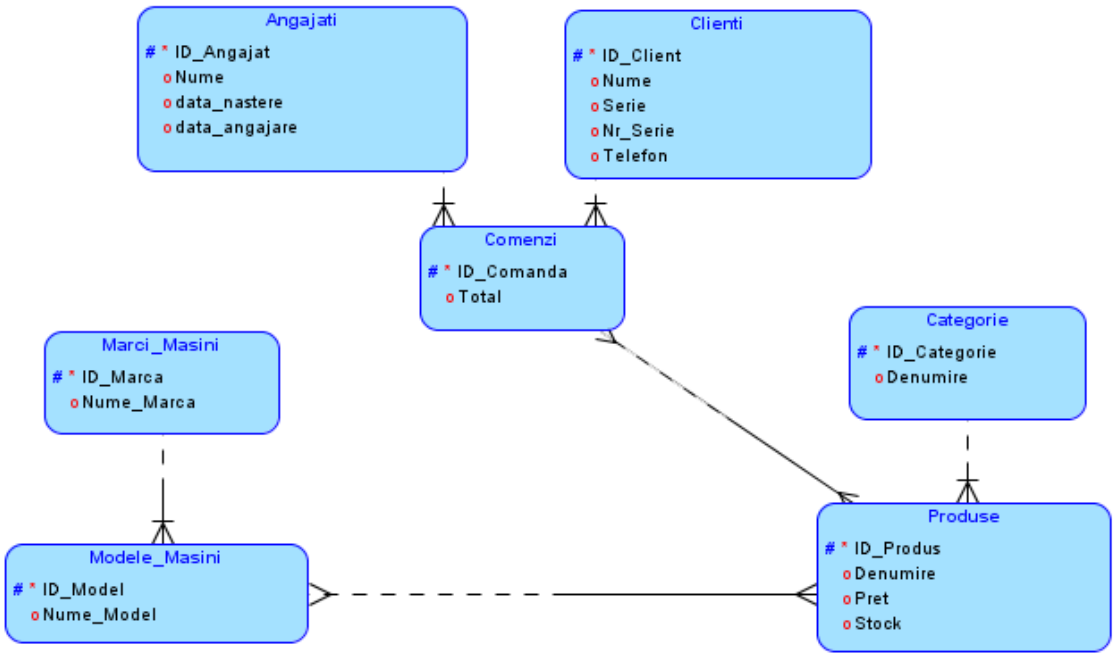
Tabelul COMENZI face legatura intre client si produsele cumparate de acesta.

Aplicatia va pune la dispozitie o interfata din care se pot adauga produse, categorii, masini se pot vizualiza continutul tabelelor prin intermediul operatiilor de adaugare, de exemplu putem vedea ce categorii de produse exista, atunci cand vrem sa adaugam un produs nou, sau le putem vedea din meniul de stergere a categoriilor. Se pot crea comenzi (care ar trebuii sa fie

pregatite la deposit pentru a putea fi ridicate de clienti). O sa existe un cos de cumparaturi pentru comenzi, care poate fi modificat pana cand comanda este plasata.

Pentru realizarea acestei aplicatii folosim o baza de date oracle, iar pentru interfata cu utilizatorul se foloseste limbajul Python. Tot Python este folosit si pentru partea de back-end care gestioneaza baza de date.

ERD initial:



Se observa relatia 1:N intre Marci_Masini si Modele_Masini, deoarece o marca de masini poate sa contina mai multe modele, iar un model poate sa apartine unei singure marci.

Categorii – Produse (1:N), o categorie poate sa contina unul sau mai multe produse, iar un produs poate sa apartina unei singure categorii.

Angajati – Comenzi (1:N), un angajat poate sa realizeze una sau mai multe comenzi, iar o comanda poate sa fie realizata de un singur angajat.

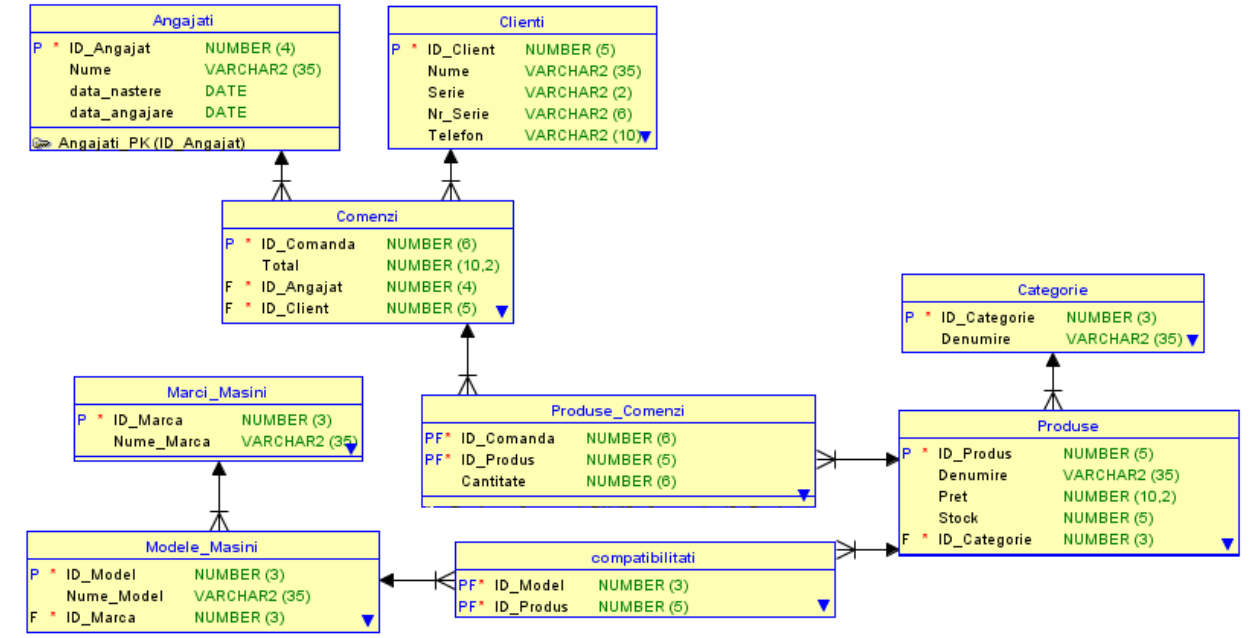
Clienti – Comenzi (1:N), un client poate sa faca una sau mai multe comenzi, insa o comanda nu poate sa apartina mai multor clienti, deci are un singur client.

Comenzi – Produse (N:M), fiecare comanda poate sa contina unul sau mai multe produse, iar un produs poate sa apartina uneia sau mai multor comenzi, deoarece este stocat in baza de date ca stock, nu ca produs in sine.

Modele_Masini – Produse, fiecare model este compatibil cu unul sau mai multe produse de acelasi tip, un produs poate sa fie compatibil cu mai multe modele de masini, de exemplu: Antigel.

Ultimele 2 relatii trebuie normalizate, prin adaugarea a 2 tabele noi, ‘Compatibilitati’ pentru Modele_Masini-Produse, respective ‘Produse_Comenzi’ pentru Comenzi – Produse.

ERD final:



Se poate observa ca acele 2 relatii au fost normalizate prin 2 table noi.

Tabelul ‘Compatibilitati’, va retine id-ul fiecarui produs si id-ul modelului de masina cu care este compatibil. Daca un produse este compatibil cu mai multe modele, atunci va exista cate o inregistrare pentru fiecare model. Cele 2 atribute formeaza cheia primara si sunt chei straine preluate din tabelele ‘Produse’ si ‘Modele_Masini’.

Tabelul ‘Produse_Comenzi’, contine id-ul fiecarei comenzi si id-ul ficarui produs, attribute care formeaza cheia primara a tebelei, ele sunt si chei straine din tabelele ‘Comenzi’ si ‘Produse’. Mai exista un atribut in care se retine cantitatea pentru fiecare produs din comanda.

Constrangeri:

Constrangerile primare care se vad in ERD-ul final, notate cu ‘P’ obliga ca acel atribut sa fie unic, adica sa nu mai exite o inregistrare cu aceasi valoare. Cu ajutor acestei/acestor chei, este garantat ca poti identifica in mod unic o inregistrare.

Constrangerile straine(foreign), notate cu ‘F’, obliga ca valorile acelui atribut sa existe in tabelul din care este ‘imprumutata’ cheia. Astefel se realizeaza o legatura intre tabele.

In cadrul acestei teme mai sunt folosite constrangeri de tip ‘NOT NULL’, care impiedica ca respectivul atribut sa fie lasate pe null, adica sa nu continta o valoare.

Constrangerile de tip check sunt folosite pentru a restrictiona domeniul acelui atribut, de exemplu, sa fie mai mare ca 10 (CHECK(atr > 10)).

In cadrul aplicatiei am adaugat o constrangere de tip check pe atributul ‘data_angajare’ din angajati, sa fie mai mare ca ‘data_nasterii’.

O alta constrangere ‘check’ a fost adaugata pe atributul ‘Total’ din tabela ‘Comenzi’, deoarece totalul trebuie sa fie mai mare ca 0, insa acest lucru este asigurat si prin aplicatie, deoarece daca nu se adauga nici un produs, comanda nu poate fi creata.

Alte 2 constrangeri ‘check’ sunt pe atributele ‘pret’ si ‘stock’ din tabela ‘Produse’. Pretul trebuie sa fie mai mare strict ca 0, insa stock-ul poate sa fie si 0, sis a fie adaugat pe viitor.

Numarul de telefon din tabela ‘Clienti’ trebuie sa aiba 10 caractere, iar acest lucru este asigurat de aplicatie, la fel si ‘Serie’, ‘Nr_Serie’, care trebuie sa aiba max 2 caractere, respective 6.

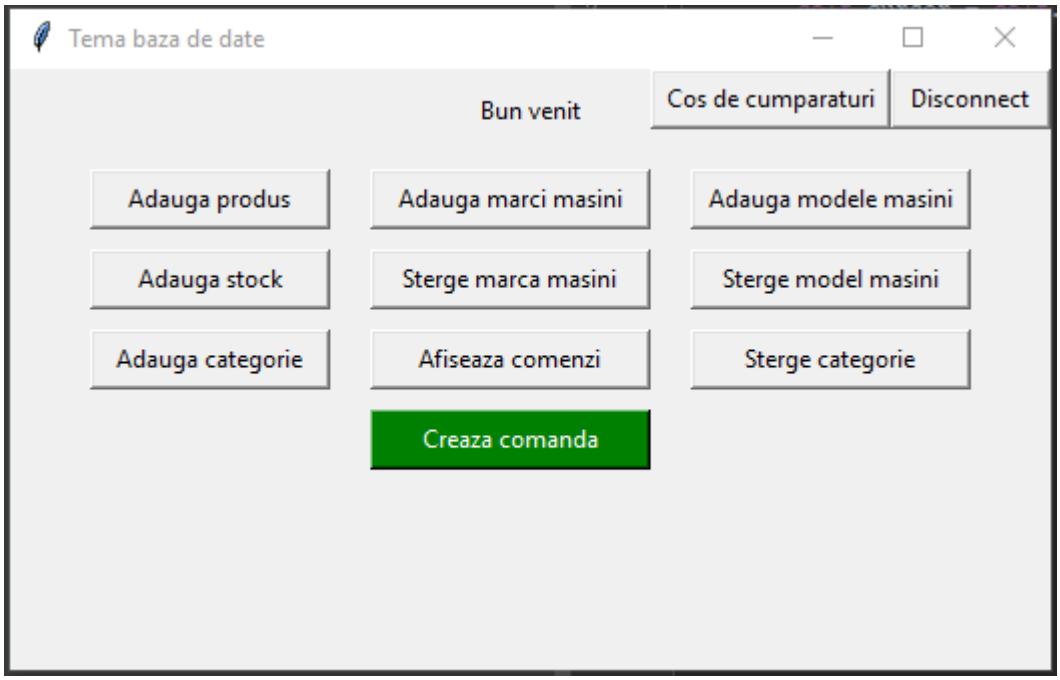
Conectarea la baza de date:

Pentru a realiza conexiunea la baza de date, folosesc modulul ‘cx_Oracle’. Instructiunea care realizeaza conexiunea este urmatoarea: ‘cx_Oracle.connect('dbname/dbpass@127.0.0.1/xe')’

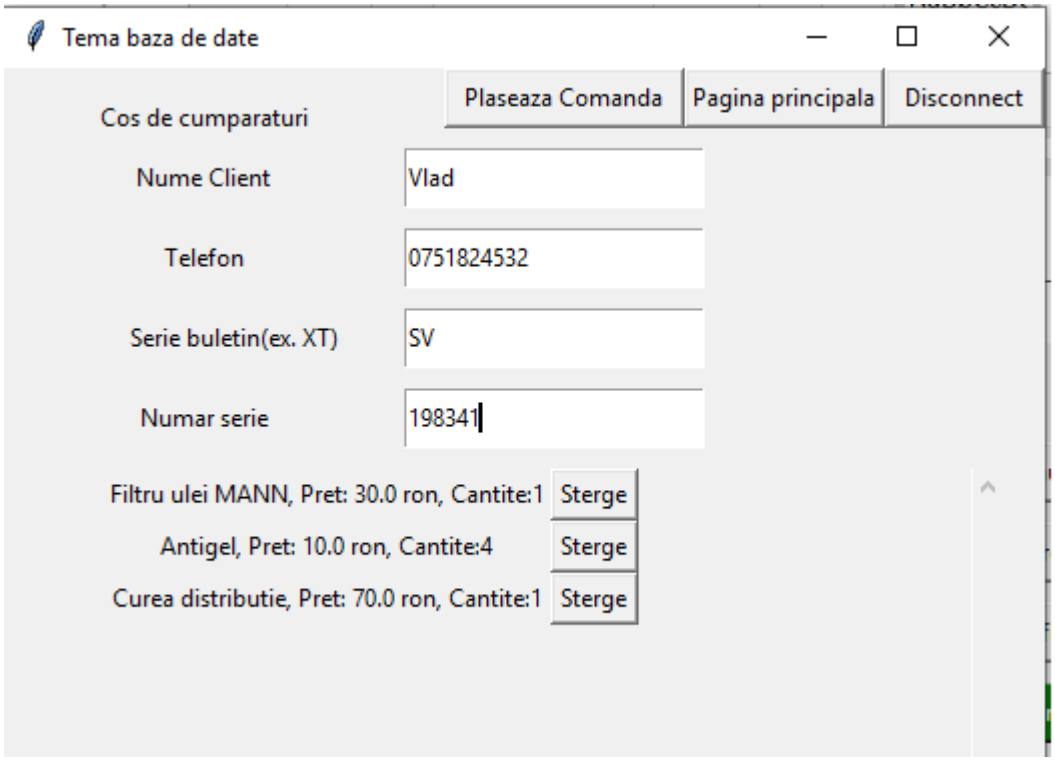
Am creat o clasa numita ‘Connection’ pentru partea de conexiune cu baza de date, aceasta contine conexiunea prorpiu-zisa si un ‘cursor’ cu ajutorul caruia se pot apela instructiunile SQL.

Capturi de ecran si exemple cod:

Pagina principala:



Cosul de cumparaturi:



Functia de afisare (din pagina ‘CreateOrderPage’) a paginii cu produsele din cosul de cumparaturi

```
def showCart(self):
    try:
        y = 0
        frame = self.controller.frames["CartPage"]
        prods = self.controller.bdapp.products
        for widget in frame.prodFrame.winfo_children():
            widget.destroy()
        for i in range(len(prods)):
            s = "SELECT * FROM produse WHERE ID_Produs = " + str(prods[i])
            self.controller.bdapp.db.cursor.execute(s)
            prod = self.controller.bdapp.db.cursor.fetchall()[0]
            amount = self.controller.bdapp.nrOfProducts
            text = str(prod[1]) + ", Pret: " + str(prod[2])+" ron, Cantite:" + str(amount[i])
            label = tk.Label(frame.prodFrame, text=text)
            label.grid(row=y, column=0)
            ckbutton = tk.Button(frame.prodFrame, text="Sterge", command=lambda: self.deleteProd(i))
            ckbutton.grid(row=y, column=1)
            y += 1
        self.controller.geometry("520x600")
        self.controller.show_frame("CartPage")
    except:
```

Functia de afisare a paginii care contine comenzile finalizate:

```
def showOrdersPage(self):
    try:
        y = 0
        frame = self.controller.frames["ShowOrdersPage"]
        for widget in frame.ordersFrame.winfo_children():
            widget.destroy()
        s = "SELECT ID_Comanda, Total, ID_Angajat, ID_Client from Comenzi"
        self.controller.bdapp.db.cursor.execute(s)
        orders = self.controller.bdapp.db.cursor.fetchall()
        for i in range(len(orders)):
            s = "SELECT Nume, Serie, Nr_Serie, Telefon FROM Clienti WHERE ID_Client = " + str(orders[i][3])
            self.controller.bdapp.db.cursor.execute(s)
            client = self.controller.bdapp.db.cursor.fetchall()[0]
            s = "SELECT Nume,Data_angajare FROM Angajati WHERE ID_Angajat = " + str(orders[i][2])
            self.controller.bdapp.db.cursor.execute(s)
            employee = self.controller.bdapp.db.cursor.fetchall()[0]
            text = str(client[0]) + ", Telefon: " + str(client[3])+", Serie: " + str(client[1]+"("+str(client[2]+"), Total: "+str(orders[i]
            label = tk.Label(frame.ordersFrame, text=text)
            label.grid(row=y, column=0)
            s = "SELECT ID_Produs, Cantitate FROM Produse_Comenzi WHERE ID_Comanda = " + str(orders[i][0])
            self.controller.bdapp.db.cursor.execute(s)
            prods = self.controller.bdapp.db.cursor.fetchall()
            y += 1
            for j in range(len(prods)):
                amount = prods[j][1]
                s = "SELECT Denumire, Pret FROM produse WHERE ID_Produs = " + str(prods[j][0])
                self.controller.bdapp.db.cursor.execute(s)
                prod = self.controller.bdapp.db.cursor.fetchall()[0]
                s = "SELECT ID_Produs, ID_Model FROM compatibilitati WHERE ID_Produs = " +str(prods[j][0])
                self.controller.bdapp.db.cursor.execute(s)
                idModel = self.controller.bdapp.db.cursor.fetchall()[0]
                s = "SELECT Nume_Model, ID_Marca FROM Modele_Masini WHERE ID_Model = " +str(idModel[1])
                self.controller.bdapp.db.cursor.execute(s)
                model = self.controller.bdapp.db.cursor.fetchall()[0]
                s = "SELECT Nume_Marca FROM Marci_Masini WHERE ID_Marca = " +str(model[1])
                self.controller.bdapp.db.cursor.execute(s)
                brand = self.controller.bdapp.db.cursor.fetchall()[0]
                text = "------"+str(prod[0]) + "("+str(brand[0])+","+str(model[0])+ "), Pret: " + str(prod[1]) + ", Cantitate" + str(amoun
                label = tk.Label(frame.ordersFrame, text=text)
                label.grid(row=y, column=0)
                y += 1
            label = tk.Label(frame.ordersFrame, text="")
            label.grid(row=y, column=0)
            y += 1
        self.controller.geometry("520x600")
        self.controller.show_frame("ShowOrdersPage")
    except:
        pass
```

Pentru a porni aplicatia, rulati ‘appStart.py’, trebuie modificat numele si parola bazei de date din acelasi fisier (appStart.py), din clasa bdApp -> Connection(numeBD, parolaBD). Baza de date trebuie sa fie pe localhost.