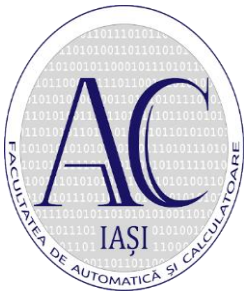
UNIVERSITATEA TEHNICĂ ”GHEORGHE ASACHI” IAȘI

FACULTATEA DE AUTOMATICĂ ȘI CALCULATOARE



DISCIPLINA BAZE DE DATE

Gestiunea unui magazin de instrumente muzicale

Coordonator,

Mironeanu Cătălin

Student,

Epure Andrei-Ioan

Grupa 1309A

2022

Titlu proiect: Gestiunea unui magazin de instrumente muzicale

Analiza, proiectarea și implementarea unei baze de date care să modeleze activitatea unui magazin cu instrumente muzicale .

Pentru a implementa această bază de date este necesar să avem informații despre : categoria instrumentelor ;tipul instrumentelor din categoriile precizate anterior;producătorul ce a fabricat instrumentele respective; ofertele disponibile ;specificațiile instrumentelor muzicale ,datele clienților si comenzile acestora.

Descrirea modului de organizare al proiectului

În realizarea acestei baze de date s-a ținut cont de urmatoarele informații:

-category : dorim sa cunoaștem in ce categorie se încadreaza intrumentul dorit de client

(instrument cu corzi ,cu clape,etc.);

-type: dorim să cunoaștem ce tip de intrument și-ar dori clientul,de exemplu :pentru categoria "instrumente cu corzi" ,clientul ar putea dori o chitară,o harpă ,o vioară,etc.;

-instrument: în acest caz ne concentrăm pe obiectul propriu-zis și întram in detalii(precum, denumira exactă a instrumentului,preț,stocuri,producător,tipul,categoria,dacă sunt sau nu oferte la acel instrument),aici clientul își poate selecta produsul dorit

-manufacturer: conține detalii despre denumirea producătorului și locația acestuia

-orders: conține comenzile fiecarui client,mai precis,ce a comandat și statusul comenzii(care poate lua una din valorile "In process"," Delivered" sau "Returned")

-oferta:contine data de la care începe o anumita oferta ,data la care se termină oferta respectivă si discount-ul oferit

-client:in cazul clienților dorim sa cunoastem informații precum:numele acestora,

numărul de telefon,email-ul acestora,țara si orașul unde se află;

Descriere constrângeri:

Category:

-category-id:primary-key(e obligatoriu, nu poate fi null,e unic) este generat printr-un mecanism de autoincrement

-category_name: să fie unic,să nu fie null,să fie format doar din litere(poate să conțină și spații) și să aibă lungimea mai mare ca 0

Type:

-type-id:primary-key(e obligatoriu,nu poate fi null,e unic) și este generat printr-un mecanism de autoincrement

-type_name: să fie unic,să nu fie null,să fie format doar din litere(poate să conțină și spații) și să aibă lungimea mai mare ca 0

Manufacturer:

-manufacturer-id:primary-key(e obligatoriu ,nu poate fi null,e unic) și este generat printr-un mecanism de autoincrement

-manufacturer_name: să fie unic,să nu fie null,să fie format doar din litere(poate să conțină și spații) și să aibă lungimea mai mare ca 0

-country:e mandatory , poate conține doar litere și să aibă lungimea mai mare ca 0

Oferta:

-oferta-id: primary-key (e obligatoriu ,nu poate fi null,e unic) și este generat printr-un mecanism de autoincrement

-start_date: trebuie să fie mai mare decât data curentă ,nu poate fi null

-end_date: trebuie să fie mai mare decât data curentă și mai mare decât start_date,nu poate fi null

-discount: nu poate fi null și trebuie trebuie să fie mai mare decat 0

Instrument:

-instrument-id:primary-key (e obligatoriu ,nu poate fi null,e unic) și este generat printr-un mecanism de autoincrement

-instrument_name: să fie unic,să nu fie null și să aibă lungimea mai mare ca 0

-price: nu poate fi null și trebuie să fie mai mare ca 0

-stock: nu poate fi null și trebuie să fie mai mare ca 0

Orders:

-order-id:primary-key (e obligatoriu ,nu poate fi null,e unic) și este generat printr-un mecanism de autoincrement

-status:poate fi “In process”,”Returned”,”Delivered”

orders_instrument (obtinuță prin prin legatura de tip many-to-many)**:**

-quantity: nu poate fi null și trebuie să fie mai mare ca 0

Client:

-client-id:primary-key (e obligatoriu ,nu poate fi null,e unic) și este generat printr-un mecanism de autoincrement

-first_name: să nu fie null,să fie format doar din litere(poate să conțina si spații) și să aibă lungimea mai mare ca 0

-last_name: să nu fie null,să fie format doar din litere(poate să conțină și spații) și să aibă lungimea mai mare ca 0

-email : să fie de forma a@b.c și să fie unic și să aibă lungimea mai mare ca 4

-phone_number: să fie unic și de forma XXXXXXXXXX sau XXX-XXXXXXX sau

XXX-XXX-XXXX

-country:e mandatory și poate conține doar litere și să aibă lungimea mai mare ca 0

-city: e mandatory și poate conține doar litere și să aibă lungimea mai mare ca 0

Descrierea detaliată a entităților și a relațiilor dintre ele:

Entitățtile din această baza de date sunt:

- Category
- Type
- Instrument
- Manufacturer
- Oferta
- Client
- Orders
- orders_instrument(obtinuță prin prin legatura de tip many-to-many)

În această bază de date regăsim tipuri de relație : 1:1 , 1:N si M:N.

Între Category si Type se stabilește o legătură one-to-many.Legătura se realizează prin câmpul CATEGORY_ID.Părintele este Category deoarece o categorie (ex :instrument cu corzi) poate avea mai multe tipuri(chitara,harpa,etc),dar un tip nu poate apartine mai multor categorii.

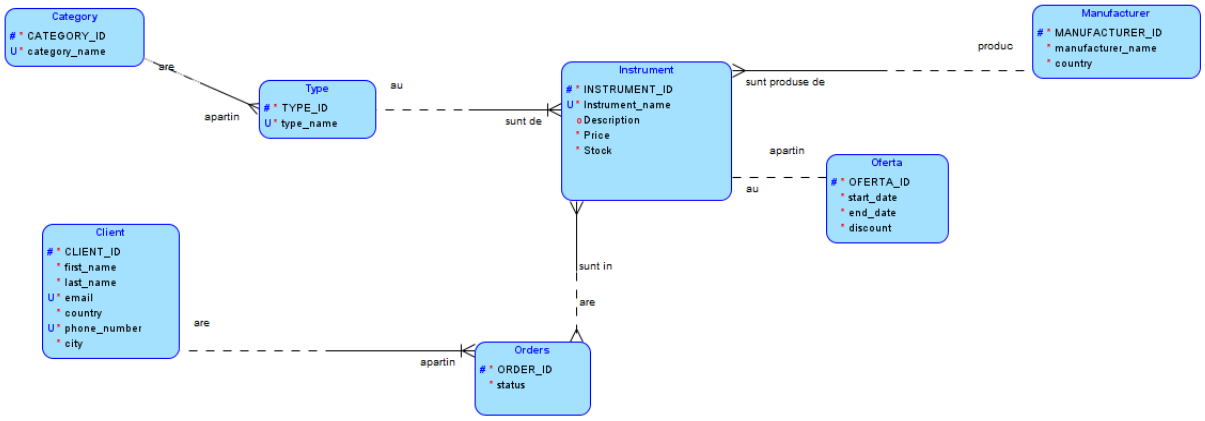
Între Type si Instrumente stabilește o legătură one-to-many.Legătura se realizează prin câmpul TYPE_ID. Părintele este Type deoarece un tip de instrument(ex chitara) poate avea mai multe instrumente propriu-zise,dar un instrument nu poate apartine mai multor tipuri.

Între Manufacturer si Instrumente stabilește o legătură one-to-many. Legătura se realizează prin câmpul MANUFACTURER_ID. Părintele este Manufacturer deoarece un instrument poate avea un singur producător,dar un producător poate produce mai multe instrumente.

Între Oferta si Instrumente se stabilește o legătură one-to-one, deoarece un instrument poate sa aibă (sau nu) o ofertă,iar oferta poate fi doar a unui singur produs.Legătura se realizează prin câmpul OFERTA_ID.

Între Client si Orders se stabilește o legătură one-to-many.Legătura se realizeaza prin câmpul CLIENT_ID. Părintele este Client deoarece un client poate avea mai multe comenzi,dar lista cu comenzi nu poate aparține mai multor clienti.

Între Orders si Instrument se stabilește o legătura many-to-many deoarece un instrument poate să apară in lista de produse comandate a mai multor clienti,dar o lista de produse comandate poate conține mai multe instrumente.



Aspecte legate de normalizare

Tabela category este în a treia formă normală deoarece : pentru fiecare category_id există o singură valoare de category_name (astfel conține valori atomice din domeniul său și nu sunt grupuri de astfel de valori),nu conține grupuri care se repetă,atributul non-cheie category_name depinde in totalitate de cheia candidat category_id si direct (non-tranzitiv) dependentă de aceasta.<category _id>-> category _name

Tabela type este în a treia formă normală deoarece : pentru fiecare type_id există o singură valoare de type_name și o singură valoare de category_id,nu conține grupuri care se repetă,atributele non-cheie depind in totalitate de cheile candidat și sunt direct (non-tranzitiv) dependente de acestea.<type_id>->type_name, <type_id>->category_id,<category_id>->category_name.

Tabela manufacturer este în a treia formă normală deoarece : pentru fiecare manufacturer_id există o singura valoare de manufacturer _name și country ,nu conține grupuri care se repetă,atributele non-cheie depind in totalitate de cheile candidat și sunt direct (non-tranzitiv) dependente de acestea. < manufacturer _id>-> manufacturer _name, < manufacturer _id>->country

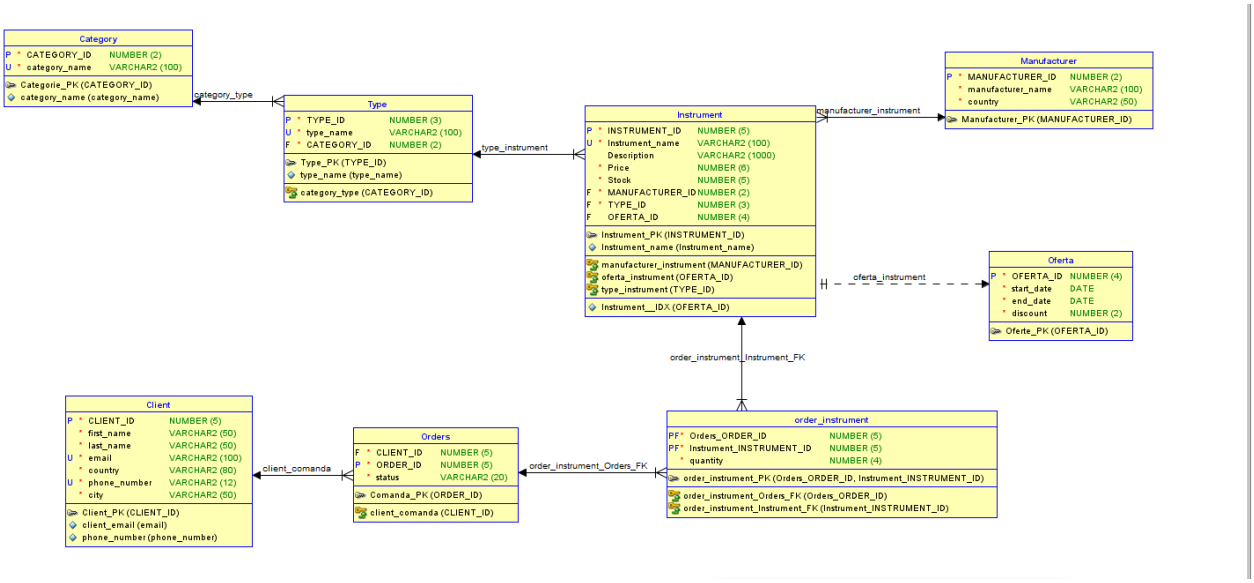
Tabela oferta este în a treia formă normală deoarece : pentru fiecare oferta _id există o singura valoare de start_date,end_date și dicount (,nu conține grupuri care se repetă atributele non-cheie depind in totalitate de cheile candidat și sunt direct (non-tranzitiv) dependente de acestea. < oferta _id>-> start_date, < oferta _id>-> end_date, < oferta _id>-> discount

Tabela client este în a treia formă normală deoarece : pentru fiecare client _id există o singura valoare de first_name,last_name,email,country,phone_number,city,nu conține grupuri care se repetă,atributele non-cheie depind in totalitate de cheile candidat și sunt direct (non-tranzitiv) dependente de acestea. < client _id>-> first_name, < client _id>-> last_name, < client _id>-> email, < client _id>-> country, < client _id>-> phone_number, < client _id>-> city

Tabela orders este în a treia formă normală deoarece : pentru fiecare order _id există o singura valoare de client_id,status ,nu conține grupuri care se repetă,atributele non-cheie depind in totalitate de cheile candidat și sunt direct (non-tranzitiv) dependente de acestea. < order _id>-> client_id, < order _id>-> status, < client _id>-> first_name, < client _id>-> last_name, ...

Tabela order_instrument este în a treia formă normală. pentru fiecare order _id există o singura valoare de instrument_id și quantity ,nu conține grupuri care se repetă,atributele non-cheie depind in totalitate de cheile candidat și sunt direct (non-tranzitiv) dependente de acestea.

Tabela instrument este în a treia formă normală deoarece : pentru fiecare instrument _id există o singura valoare de instrument _name,description,price,stock,manufacturer_id și type_id ,nu conține grupuri care se repetă,atributele non-cheie depind in totalitate de cheile candidat și sunt direct (non-tranzitiv) dependente de acestea.



Tehnologiile folosite :

Aplicatia web a fost realizata prin utilizarea framework-ului Flask din python,prin acesta realizandu-se comunicarea intre interfata si baza de date.Interfata a fost creata folosind HTML, CSS si JavaScript.Accesul la baza de date se face prin modulul cx_Oracle.

Observatie .
Pentru a se realiza conexiunea pe serverul facultattii trebuie descarcat Oracle Instant Client .

Realizarea conexiunii :

```
cx_Oracle.init_oracle_client(lib_dir=r"C:\Users\Andrei\Desktop\Epure_Andrei_Ioan_1309A\instantclient_21_7")
con_tns = cx_Oracle.makedsn('bd-dc.cs.tuiasi.ro', '1539', service_name='orcl')
try:
    con = cx_Oracle.connect('bd096', 'bd096', dsn=con_tns)

except Exception:
    print('Unexpected error')
else:
    print('Conexiune stabilita')
```

Functia select in tabela client se realizeaza astfel:

```
@app.route('/client')
def client_fct():
    clients = []
    cur = con.cursor()
    cur.execute('select * from client  order by client_id')
    for result in cur:
        client = {}
        client['id'] = result[0]
        client['first_name'] = result[1]
        client['last_name'] = result[2]
        client['email'] = result[3]
        client['country'] = result[4]
        client['phone_number'] = result[5]
        client['city'] = result[6]
        clients.append(client)

    return render_template('client.html', client=clients)
```

Clienti

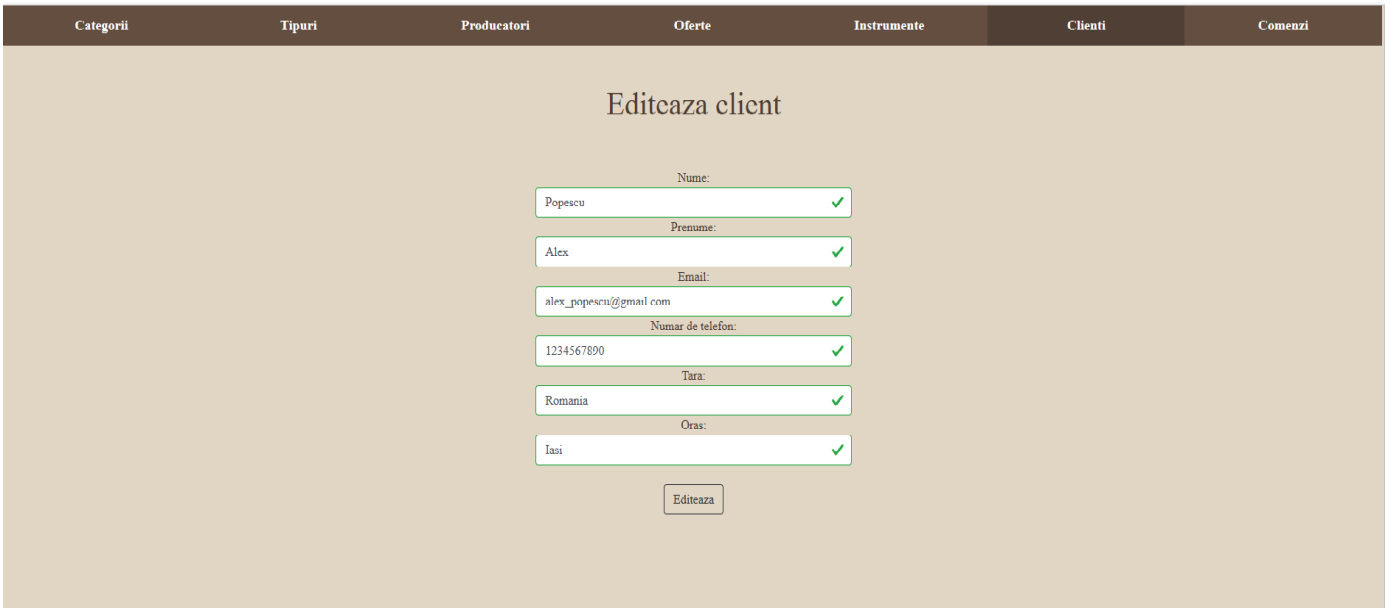
ID	Nume	Prenume	Email	Numar de telefon	Tara	Orasul	
1	Popescu	Alex	alex_popescu@gmail.com	1234567890	Romania	Iasi	Editeaza Sterge
2	Johnson	Cristian	cristian_johnson@gmail.com	1352775345	Germania	Berlin	Editeaza Sterge
3	Garcia	Leo	leo_garcia@gmail.com	3674812414	Canada	Toronto	Editeaza Sterge
4	Miller	Steve	steve_miller@gmail.com	2746001783	USA	New York	Editeaza Sterge
5	Williams	Max	max_williams@gmail.com	7873512461	Argentina	Salta	Editeaza Sterge
6	Smith	John	john_smith2@gmail.com	5553472134	USA	San Diego	Editeaza Sterge
7	Jones	Jack	jack_jones@gmail.com	6654361296	USA	Chicago	Editeaza Sterge
8	Brown	Thomas	thomas_brown3@gmail.com	8726766545	Mexico	Durango	Editeaza Sterge

Functia insert in tabela client se realizeaza astfel:

```
@app.route('/add_client', methods=['POST'])
def add_client_fct():
    if request.method == 'POST':
        cur = con.cursor()
        values = []
        values.append("'" + request.form['first_name'] + "'")
        values.append("'" + request.form['last_name'] + "'")
        values.append("'" + request.form['email'] + "'")
        values.append("'" + request.form['country'] + "'")
        values.append("'" + request.form['phone_number'] + "'")
        values.append("'" + request.form['city'] + "'")

        query = 'insert into client values(NULL,' + ', '.join(values) + ')'

        cur.execute(query)

        cur.execute('commit')
        cur.close()
        return redirect('/client')
```

Functia update in tabela client se realizeaza astfel:

```
@app.route('/edit_client', methods=['POST'])
def edit_client_fct():
    id = "'" + request.form['id'] + "'"
    first_name = "'" + request.form['first_name'] + "'"
    last_name = "'" + request.form['last_name'] + "'"
    email = "'" + request.form['email'] + "'"
    country = "'" + request.form['country'] + "'"
    phone_number = "'" + request.form['phone_number'] + "'"
    city = "'" + request.form['city'] + "'"

    cur = con.cursor()
    query = "update client set first_name=%s,last_name=%s,email=%s,country=%s,phone_number=%s,city=%s where " \
            "client_id=%s" % (first_name, last_name, email, country, phone_number, city, id)
    cur.execute(query)
    cur.execute('commit')
    cur.close()
    return redirect('/client')
```



Functia delete in tabela client se realizeaza astfel:

```
@app.route('/del_client', methods=['POST'])
def del_client_fct():
    cur = con.cursor()
    cur.execute('delete from client where client_id=' + request.form['id'])
    cur.execute('commit')
    cur.close()
    return redirect('/client')
```