

Baze de date

-TEMA-

Studenti:

Butnaru Raimond-Eduard 1306B

Guriuc Vlad-Ionut 1306B

Profesor coordonator:

Avram Sorin

Cuprins

Cuprins	2
Introducere	3
Programe si biblioteci folosite	4
Frontend:	4
Backend:	6
Diagramele ER	7
Constrangeri utilizate	8
Conectarea la baza de date	8

Introducere

Aplicația constă în managerierea unui sistem hotelier, fie acesta de către un utilizator oarecare fie de către un manager de hotel.

În aceasta aplicație se pot vedea toate înregistrările fiecărei tabele din baza de date la acțiunea butoanelor ce conțin titlul tabelului.

Aplicația noastră este concepută pentru a putea fi realizată gestiunea unei baze de date a unui hotel. În baza de date au fost create următoarele tabele:

1. Clienți, având coloanele: id_client, nume, prenume, telefon, email, varsta;
2. Detalii_camera, având coloanele: id_camera, pret_noapte, etaj, tip_camera, capacitate, vedere_la_mare, balcon;
3. Rezervare, având coloanele: id_rezervare, id_client, id_camera, numar_nopti, total_plata, data_rezervare, data_eliberare;
4. Camera, având coloanele: id_camera, numar_camera.

În această aplicație am implementat, cu ajutorul unei interfețe, principalele funcții de interogare prezente în structura oricărei baze de date. Principalele funcții folosite de noi sunt: insert, modify, select, update, delete. Aplicația a fost structurată astfel încât datele prezente în baza de date a hotelului să poată fi manipulate atât de client cât și de angajat/angajator. Datele pot fi editate și vizualizate în mod diferit în funcție de fiecare categorie de utilizator. Aplicația beneficiază de o serie de constrângeri ce facilitează navigarea utilizatorilor mai puțin experimentați (constrângeri de tip unique, check, primary key, foreign key). În procesul de rezervare clientul trebuie să introducă o serie de date cu caracter personal în interfața grafică și să le salveze. În cazul în care datele sunt incorect introduse sau rezervarea este indisponibilă în perioada selectată, se vor afișa mesaje de informare explicite.

Totodată, datorită unor verificări amănunțite, aplicația tinde să evite conflictele de date ce pot apărea la nivelul introducerii unor date noi sau la modificarea celor existente. Spre exemplu:

- nu se poate face check-in și checkout la hotel în aceeași zi
- nu se pot face rezervări anterior datei curente
- nu se poate selecta o dată de check-in sau check-out care se intercalează cu alt interval pe aceeași cameră.
- nu pot exista 2 camere cu același număr
- nu pot exista 2 clienți diferiți cu numere de telefon la fel
- nu se pot face rezervări pe camere inexistente sau neinregistrate
- id-uri de clienți inexistente nu pot face rezervări
- numărul maxim de persoane pe cameră este de 4

Programe si biblioteci folosite

Programul nostru este dezvoltat în python și folosește bibliotecile tkinter, tkcalendar, cx_Oracle si datetime. Acestea au fost folosite pentru a crea interfața grafică a aplicației, pentru a manipula bazele de date și pentru a accesa și memora momente de timp, finisand astfel procesul de rezervare a camerelor.

Modulele de tkinter și tkcalendar sunt utilizate strict pentru preluarea de date din campurile de intrare ale interfeței. Interfața propriu-zisă este realizata cu tkinter, însă aceasta conține și elemente de tkcalendar pentru preluarea și managerierea interfețelor cu calendarul.

Frontend:

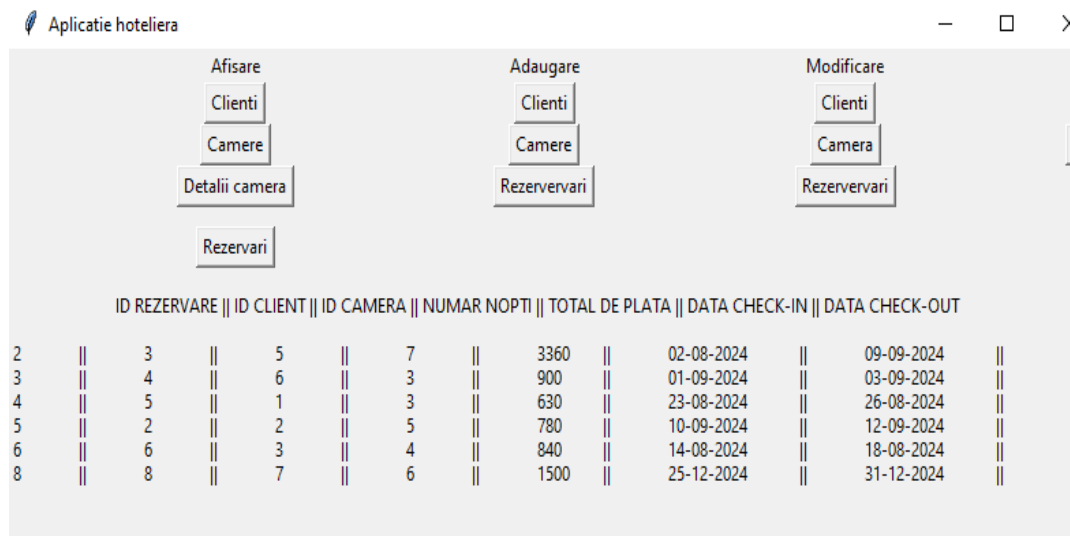
Partea de frontend este realizata în python cu modulele mentionate anterior. Acest front urmează o interfața simplă pentru utilizator, avand butoane puse pe coloane care reprezinta, afisarea, adaugarea, modificarea și respectiv stergerea inregistrarilor.

Sub aceste butoane exista un spațiu mare și liber unde datele urmează sa fie prezentate.

Pentru operații mai complexe, spre exemplu adaugarea, ștergerea sau modificarea intrărilor în baza de date, se utilizează ferestre cu interfețe suplimentare.

Frontend realizat de: Butnaru Raimond

Exemplu de afișare rezervari



ID REZERVARE ID CLIENT ID CAMERA NUMAR NOPTI TOTAL DE PLATA DATA CHECK-IN DATA CHECK-OUT													
2		3		5		7		3360		02-08-2024		09-09-2024	
3		4		6		3		900		01-09-2024		03-09-2024	
4		5		1		3		630		23-08-2024		26-08-2024	
5		2		2		5		780		10-09-2024		12-09-2024	
6		6		3		4		840		14-08-2024		18-08-2024	
8		8		7		6		1500		25-12-2024		31-12-2024	

Interfața de adaugare rezervare:

Adaugare rezervare

ID client (Numar):

ID Camera (Numar):

Numar de nopti rezervate (Numar):

Total de plata (Numar):

Data check-in:

January 2024						
Mon	Tue	Wed	Thu	Fri	Sat	Sun
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31	1	2	3	4
5	6	7	8	9	10	11

Data checkout:

January 2024						
Mon	Tue	Wed	Thu	Fri	Sat	Sun
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31	1	2	3	4
5	6	7	8	9	10	11

Adaugare rezervare

Interfata de adaugare camera:

Adaugare camera

Numarul camerei (Numar):

Pret per noapte (Numar):

Etajul (Numar):

Tipul Camerei:

Capacitate:

Vedere la mare? ☐

Tip balcon:

Adauga

Mentiune: În interfața de mai sus campurile “Tipul Camerei”, “Capacitate” și “Tip Balcon” sunt niste liste de tip dropdown care permit doar selecția anumitor date. Campul “Vedere la mare” este un obiect checkbox care o data bifat semnifica “Da aceasta camera are și vedere spre mare” (care în baza de date este doar “DA”) iar lasat nebifat reprezinta contrariul (adică se regaseste ca și “NU” în baza de date)

Backend:

Backend-ul aplicației este realizat cu ajutorul modului `cx_Oracle`. Acesta necesită o conexiune în urma careia este necesară crearea unui cursor ce execută interogările, inserarea, stergerile și modificările la baza de date. După fiecare operație pe care o execută cursorul, se face o operație de “fetch” în memorie, care stochează datele aduse de cursor.

```
cursor = self.connection.cursor()
cursor.execute("SELECT TELEFON FROM CLIENTI")
rows = cursor.fetchall()
for row in rows:
    if telefonEntry.get() == row[0]:
        messagebox.showerror( title: "Eroare", message: "Numarul de telefon introdus este deja folosit")
        return None
```

În codul exemplu de mai sus se realizează interogarea la baza de date, după care se stochează rezultatul în variabila “rows”. Tipul de date returnat în urma interogării este o listă de tuple. Spre exemplu primul rând fiind de forma:

`[('numar de telefon 1 ',), (' numar de telefon 2 ',) ...]`

Virgulele de după numerele de telefon sunt datorită volatilității unui tuplu, acestea pot fi extinse până la n date.

Tot în exemplul anterior se realizează verificarea unicității numărului de telefon introdus de către utilizator, cazul în care numărul de telefon exista deja, acesta va fi respins și procesul de adăugare sau de modificare al clientului se încheie imediat.

Diagramele ER

BD014.DETALII_CAMERA		
F	* ID_CAMERA	NUMBER (6)
	* PRET_NOAPTE	NUMBER (6)
	* ETAJ	NUMBER (2)
	* TIP_CAMERA	VARCHAR2 (20 BYTE)
	* CAPACITATE	NUMBER (3)
	* VEDERE_LA_MARE	CHAR (3 BYTE)
	BALCON	VARCHAR2 (9 BYTE)
DETALII_CAMERA_FK (ID_CAMERA)		



BD014.CAMERA		
P	* ID_CAMERA	NUMBER (6)
	* NUMAR_CAMERA	VARCHAR2 (4 BYTE)
CAMERA_PK (ID_CAMERA)		
CAMERA_PK (ID_CAMERA)		

BD014.CLIENTI		
P	* ID_CLIENT	NUMBER (6)
	* NUME	VARCHAR2 (20 BYTE)
	* PRENUME	VARCHAR2 (20 BYTE)
U	* TELEFON	CHAR (10 BYTE)
	* EMAIL	VARCHAR2 (100 BYTE)
	* VARSTA	NUMBER (6)
CLIENT_PK (ID_CLIENT)		
CLIENT_TELEFON_UN (TELEFON)		
CLIENT_PK (ID_CLIENT)		
CLIENT_TELEFON_UN (TELEFON)		

BD014.REZERVARE		
	* ID_REZERVARE	NUMBER (6)
	* ID_CLIENT	NUMBER (6)
	* ID_CAMERA	NUMBER (6)
	* NUMAR_NOPTI	NUMBER (3)
	* TOTAL_PLATA	NUMBER (6)
	* DATA_REZERVARE	DATE
	* DATA_ELEIBERARE	DATE
IDX_ID_REZERVARE (ID_REZERVARE)		

În vederea optimizării aplicației, regasim astfel o serie de elemente comune în cele 4 tabele create pentru care am aplicat constrângeri. Aceste constrângeri au în primul rând rolul de a facilita navigarea între tabele în momentul interogărilor și introducerii de valori. Clientul are atribuit în tabela CLIENTI un id_client(primary key) ce este folosit pentru a face corespondență cu rezervarea pe care o face. În aceeași tabelă regasim un câmp destinat numărului de telefon al clientului ce are o constrângere de tip unique. Acestea, împreună cu coloana ID REZERVARE aflată în tabela REZERVARE, asigură unicitatea fiecărei rezervări de la fiecare client și sporește considerabil gradul de organizare al sistemului hotelier, diminuând șansele apariției de confuzie sau incidente.

IDX_ID_REZERVARE(ID_REZERVARE) este un index simplu asociat tablei REZERVARI. Cu ajutorul acestui index, baza de date poate fi accesată mai rapid pe baza coloanei ID_REZERVARE, cu dezavantajul că acest index va ocupa mai mult spațiu în baza de date.

Crearea tabelelor, a constrângerilor și a popularea tabelelor realizată de: Guriuc Vlad

Constrângeri utilizate

- id_client- primary key în tabela CLIENTI ;
- telefon- unique în tabela CLIENTI deoarece fiecare client poate avea un singur număr de telefon și lungimea sa egală cu 10;
- id_camera- primary key în tabela CAMERA și foreign key în tabela DETALII_CAMERA;
- varsta - în tabela CLIENTI cu constrângere de tip check >18;
- email- în tabela CLIENTI constrângere de tip check ce acceptă doar tiparul specific(____@_.);
- vedere_la_mare-coloana cu constrângere în tabela DETALII_CAMERA ce acceptă numai valorile 'DA' sau 'NU';
- Balcon - coloana cu constrângere în tabela DETALII_CAMERA ce acceptă doar valorile 'PROPRIU' sau 'COMUN'.

Conectarea la baza de date

Conectarea la baza de date Oracle se realizează prin intermediul unui "instant client" de la Oracle. În aplicația noastră, calea spre acest client este menționată în intermediul codului pentru a lăsa modulul de cx_Oracle să găsească un fel de "poartă" spre serverul SQLplus. O dată menționată aceasta "poartă" programului îi mai necesită doar datele de logare care sunt scrise în interiorul programului pentru a se loga la baza noastră de date creată anterior.

Mai jos este exemplul de cum se realizeaza conexiunea la baza de date. Se inițializează clientul instant apoi se seteaza credentialele și hostul bazei de date, după care se inițializează interfața cu cei 3 parametrii.

```
if __name__ == "__main__":  
    cx_Oracle.init_oracle_client(lib_dir="E:\\Facultate\\BD\\instantclient_21_12")  
  
    username = "bd014"  
    password = "bd014"  
    dsn = "bd-dc.cs.tuiasi.ro:1539/orcl"  
    deschidere = Deschidere(username, password, dsn)  
    deschidere.mainloop()
```

Bibliografie:

<https://docs.oracle.com/javadb/10.8.3.0/ref/rrefsqlj26498.html>

https://www.techonthenet.com/oracle/tables/alter_table.php

<https://www.geeksforgeeks.org/dropdown-menus-tkinter/>

<https://www.geeksforgeeks.org/create-a-date-picker-calendar-tkinter/>

<https://www.programiz.com/python-programming/datetime/current-datetime>

<https://www.freecodecamp.org/news/python-datetime-now-how-to-get-todays-date-and-time/>

<https://www.geeksforgeeks.org/python-tkinter-entry-widget/>

<https://www.geeksforgeeks.org/python-tkinter-messagebox-widget/>