UNIVERSITATEA TEHNICĂ “GHEORGHE ASACHI” IAŞI
FACULTATEA DE AUTOMATICĂ ŞI CALCULATOARE
SPECIALIZAREA CALCULATOARE ŞI TEHNOLOGIA INFOR MAŢIEI DISCIPLINA : BAZE DE DATE PROIECT

GESTIUNEA ACTIVITATII INTR-O AFACERE DE CAR DETAILING

Proiect la disciplina Baze de date (BD)

Coordonator: Mironeanu Catalin

Paraschiv Stefan

Introducere

Analiza, proiectarea si implementarea unei baze de date si a aplicatiei aferente care sa modeleze activitatea unei afaceri de car detailing cu privire la gestiunea angajatilor si a programarilor pentru a fi o afacere functionala.

Descrierea cerintelor si modul de organizare al proiectului

Obiectivul proiectului este de a crea o baza de date cat mai simpla si eficienta pentru gestionarea unei afaceri de car detailing, care se diferentiaza de o spalatorie normala de masini prin faptul ca ofera servicii premium si mult mai detaliate .O astfel de afacere presupune multa munca si atentie deoarece trebuie gestionati angajatii corect, spatiul in care se desfasoara activitatile dar si programarile. Daca aceste lucruri simple nu sunt implementate corect afacerea va putea da faliment sau va putea sa piarda potentiali clienti din cauza erorilor tehnice.

Principalele date gestionate sunt cele legate de :

- **Angajatii firme**i: pentru salarii si serviciile prestate dar si pentru a putea gasi responsabilul in cazul unei probleme.
- **Programari**:pentru a stabili daca este posibila indeplinirea serviciului respectiv sau nu la monentul respectiv
- **Clientii**:pentru a sti unde sa se dezvolte afacerea in viitor, pentru asta datele unui client sunt stocate si analizate.
- **Salar:** unde se tine evidenta banilor pentru fiecare angajat
- **Boxe:** unde sunt gestionate boxele, ce dimensiuni au
- **List_servicii**: contine toate serviciile si detalii despre ele

Descrierea functională a aplicației

Principalele functii care se pot intalni intr-o afacere de car detailing sunt:

- Evidenta clientilor
- Gestiunea programarilor
- Evidenta angajatilor
- Evidenta spatiului de lucru

Descrierea detaliata a entitatilor si a relatiilor dintre tabele din modelul logic

Tabelele din aceasta aplicatie sunt:

Angajati;

Programari;

Boxe;

List_Servicii;

Clienti;

Salar;

Entitatile din modelul logic sunt:

- Clienti

Aceasta entitate se ocupa de pastrarea informatiilor despre fiecare client si contine urmatoarele atribute si constrangeri.

- ID_client (NUMERIC) - Primary Key (autoincrement)
- Nume_client(VARCHAR(30)) - Mandatory si constrangere ca lungimea sa fie mai mare ca 1.
- Oras(VARCHAR(10))-Mandatory si constrangere ca lungimea sa fie mai mare ca 2.
- Birthdate(DATE)-Mandatory si constrangere ca sa fie nascut dupa '1 jan 1950'.

- Programari

Aceasta entitate se ocupa de realizarea unei programari, momentan nu s-a putut face verificarea datelor care intra (angajtul sa fie disponibil, boxa sa fie disponibila la ora x, insa va fi facuta in tema de casa prin verificarea variabilei, de exemplu daca vrem sa verificam daca se poate face o programare la 15 o sa verific daca se afla in intervalul orar la care sunt porgramari, daca este atunci dau return la ceva mai mare ca 1 altfel 0).Programari are urmatoarele atribute si constrangeri.

- ID_prog(NUMERIC) -Primary Key (autoincrement)
- Data(DATA)- Mandatory cu constrangerea ca data sa fie mai mare ca data curenta
- Ora(FLOAT)-Mandatory cu constrangerea sa fie intre 0 si 24, aceasta baza de date are ore de forma 14.5 (14:30) etc.
- ID_client(NUMERIC) - Foreign Key din CLIENTI

- ID_boxa(NUMERIC)- Foreign Key din BOXE
- ID_serviciu(NUMERIC)- Foreign Key din LIST_SERVICII
- ID_angajat(NUMERIC)- Foreign Key din ANGAJATI
- In aceasta tabela mai este o constrangere unique intre data si ora.

- List_servicii

Aceasta entitate defineste serviciile care exista, care sunt descrise prin tipul serviciului care pentru una din cele 4 clase de masini au preturi diferite, astfel tabela are urmatoare atribute si constrangeri.

- ID_serviciu(NUMERIC)-Primary Key (autoincrement)
- Nume_serviciu(VARCHAR(30))-Mandatory cu constrangerea sa fie in lista aceasta: 'detailing interior','detailing motor','detailing exterior'
- Clasa(VARCHAR(2))-Mandatory cu constrangerea sa fie:'A','B','C' sau 'D'
- Durata(FLOAT)-Mandatory cu constrangerea sa fie intre 0 si 8
- Pret(NUMERIC(4))-Mandatory cu constrangerea sa fie mai mare ca 0

- Boxe

Aceasta entitate descrie boxele care sunt si dimensiunile, are urmatoarele atribute si constrangeri.

- ID_Boxa(NUMERIC)-Primary Key (autoincrement)
- Lungime(NUMERIC(2))-Mandatory cu constrangerea sa fie pozitiva dimensiunea
- Latime(NUMERIC(2))-Mandatory cu constrangerea sa fie pozitiva dimensiunea

- Angajati

Aceasta entitate descrie angajatii, si ce bonus de bani au primit, acest bonus se acorda odata la 5 servicii, este iar o situatie ca la programari cu verificare prin SELECT, iar tot asa odata pe luna isi da reset la ore lucrate si "bonus bani".Tabela are urmatoarele atribute si constrangeri.

- ID_Angajat(NUMERIC)-Primary Key
- Nume(VARCHAR(20))-Mandatory cu constrangerea sa fie mai lung ca 1.
- Ore_lucrate(FLOAT(2))- Mandatory cu constrangerea sa fie mai mare sau egal cu 0.
- Bonus_bani(NUMERIC(10))-Cu constrangerea sa fie mai mare sau egal cu 0.

- Salar

Aceasta entitate tine evidenta banilor pentru angajati.

- ID_anagajat(NUMERIC)-Primary Foreign Key
- Salar(NUMERIC(20))-Mandatory cu constrangerea sa fie mai mare ca 1000.
- Bonus(NUMERIC(10))-Cu constrangerea sa fie mai mare ca 0.
- Total(NUMERIC(30))-Mandatory cu constrangrea sa fie mai mare ca 1000.

In aceasta baza de date avem urmatoarele relatii:

- Relatii 1:1:

  - Intre entitatile ANGAJATI si SALAR exista o relatie one_to_one prin care entitatea SALAR mosteneste cheia primara din ANGAJATI (ID_angajat), astfel fiecarui angajat ii corespunde "un tabel" din salar.

- Relatii 1:n:

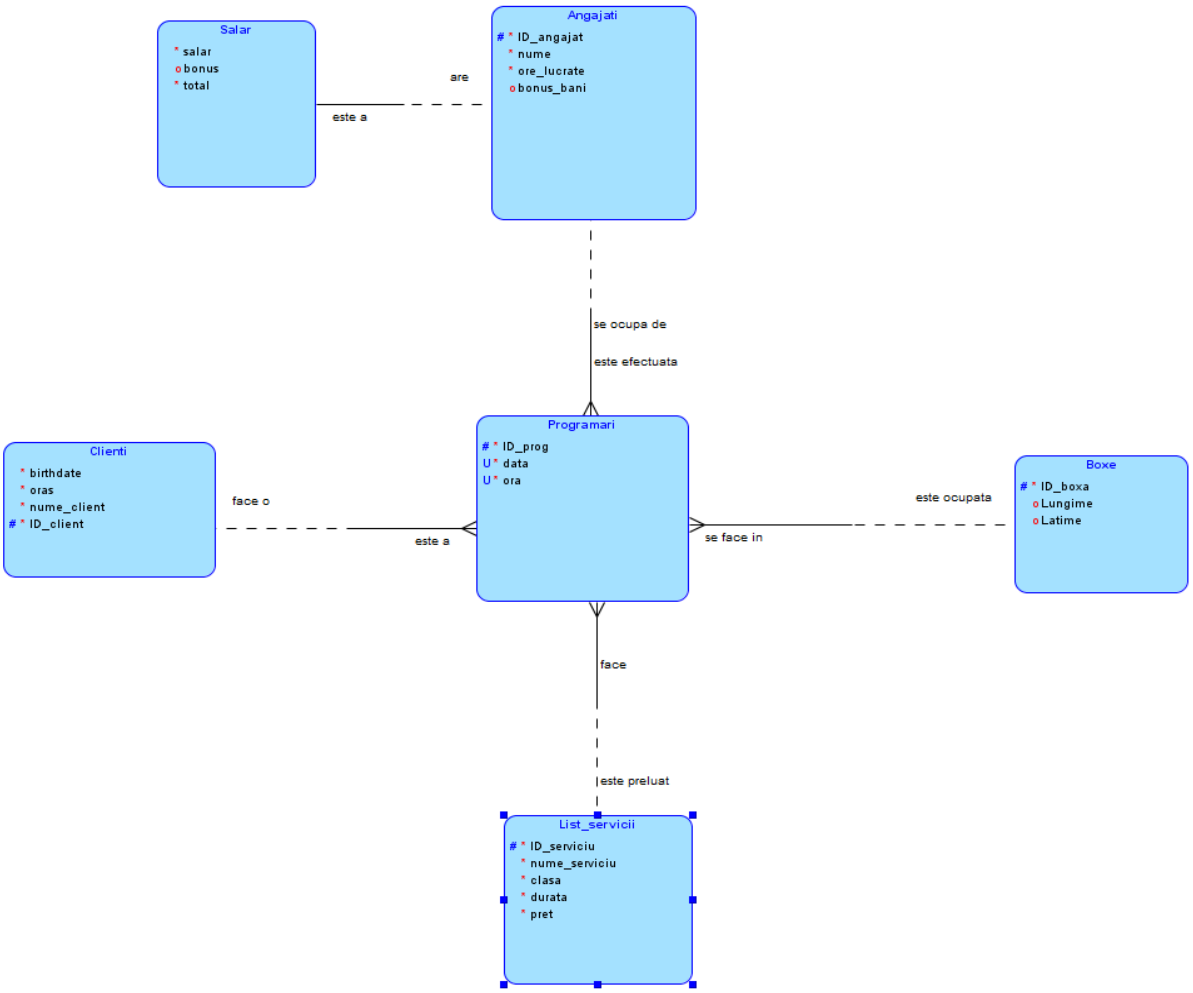
  - Intre PROGRAMARI si ANGAJATI exista o relatie one_to_many pentru ca un anagajat poate avea mai multe programari insa o programare corespunde unui singur angajat.Legatura este facuta prin propagarea cheii primare(ID_angajat) din ANGAJATI, entitatea parinte in PROGRAMARI, entitatea copil.

  - Intre PROGRAMARI si LIST_SERVICII exista o relatie one_to_many pentru ca un serviciu poate fi efectuat in mai multe programari insa intr-o programare se efectueaza un singur serviciu.Legatura este facuta prin propagarea cheii primare (ID_serviciu) din LIST_SERVICII, entitatea parinte in PROGRAMARI, entitatea copil.

  - Intre PROGRAMARI si CLIENTI exista o relatie one_to_many pentru ca un client poate face mai multe programari insa o programare este a unui client.Legatura este facuta prin propagarea cheii primare (ID_client) din CLIENTI, entitatea parinte in PROGRAMARI, entitatea copil.

- Intre PROGRAMARI si BOXE exista o relatie one_to_many pentru ca mai multe boxe corespunde mai multor programari insa o programare corespunde unei singure boxe. Legatura este facuta prin propagarea cheii primare (ID_boxa) din BOXE, entitatea parinte in PROGRAMARI, entitatea copil.

MODEL LOGIC:

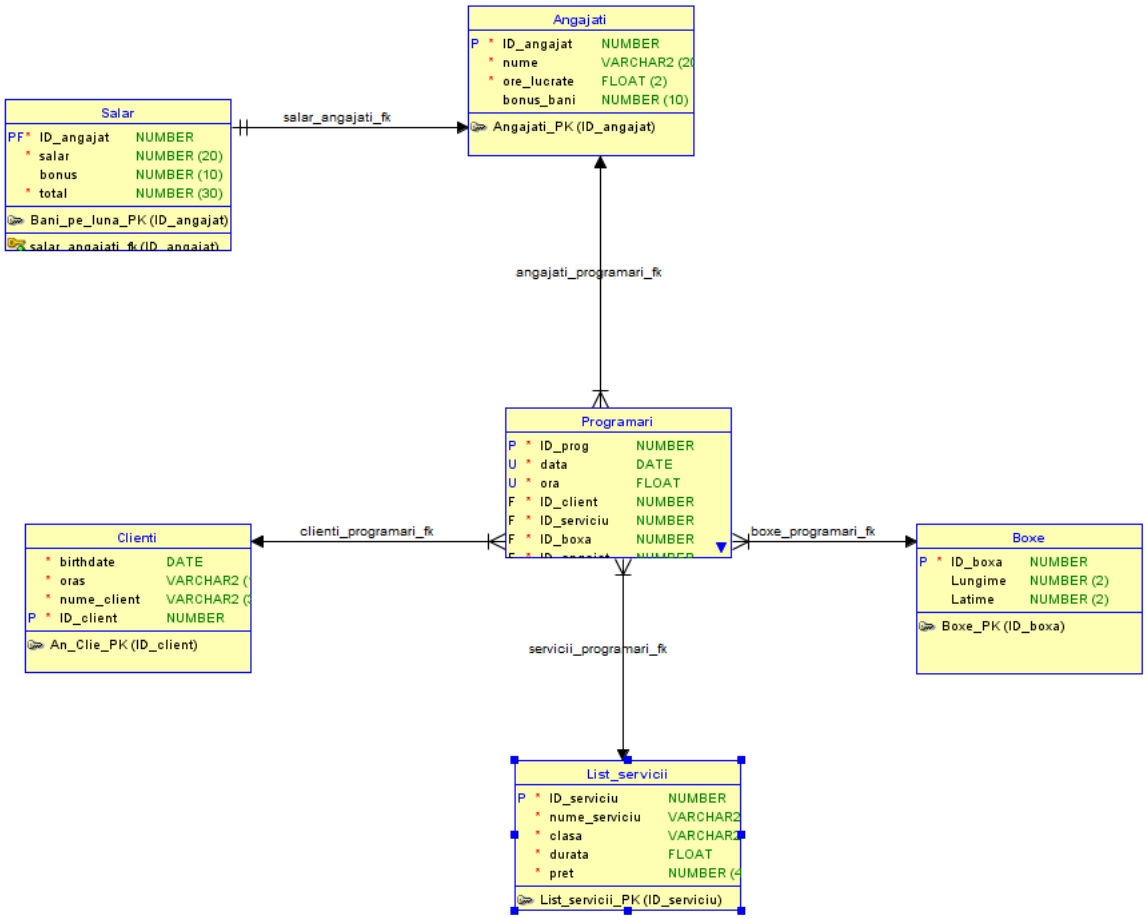


Descrierea detaliata a entitatilor si a relatiilor dintre tabele din modelul fizic

In modelul fizic se pot vedea mult mai clar cheile care sunt mostenite si care sunt relatiile din tabele, aici este adaugat si autoincrementul pentru cheile primare din : ANGAJATI,PROGRAMARI,CLIENTI,BOXE si LIST_SERVICII dar si un trigger pentru a verifica in tabela PROGRMARI daca data introdusa este mai mare decat cea curenta, cu urmatorul cod:

```
BEGIN

IF( :new.data < SYSDATE )

THEN

RAISE_APPLICATION_ERROR( -20001,

'Data invalida: ' || TO_CHAR( :new.data, 'DD.MM.YYYY' ) || ' trebuie sa fie mai mare decat data curenta.' );

END IF;

END;
```

MODEL FIZIC:


Angajati
P * ID_angajat NUMBER
* nume VARCHAR2 (20
* ore_lucrate FLOAT (2)
bonus_bani NUMBER (10)
Angajati_PK (ID_angajat)
Salar
PF* ID_angajat NUMBER
* salar NUMBER (20)
bonus NUMBER (10)
* total NUMBER (30)
Bani_pe_luna_PK (ID_angajat)
salar_angajati_fk
angajati_programari_fk
Programari
P * ID_prog NUMBER
U * data DATE
U * ora FLOAT
F * ID_client NUMBER
F * ID_serviciu NUMBER
F * ID_boxa NUMBER
Clienti
* birthdate DATE
* oras VARCHAR2 (
* nume_client VARCHAR2 (
P * ID_client NUMBER
An_Clie_PK (ID_client)
clienti_programari_fk
boxe_programari_fk
Boxe
P * ID_boxa NUMBER
Lungime NUMBER (2)
Latime NUMBER (2)
Boxe_PK (ID_boxa)
servicii_programari_fk
List_servicii
P * ID_serviciu NUMBER
* nume_serviciu VARCHAR2
* clasa VARCHAR2
* durata FLOAT
* pret NUMBER (4
List_servicii_PK (ID_serviciu)