**Proiect Baze de Date**

**Problema 6**

**Monitorizarea evenimentelor unei firme de paza**

Studenta Frenț Paraschiva-Cristina, 3.1

Facultatea de Automatică și Calculatoare

Ingineria Sistemelor, anul II

**Enunț:**

Se considera o aplicatie pentru monitorizarea evenimentelor unei firme de paza.

Baza de date Oracle va contine urmatoarele informatii:

* Nume client
* CNP
* Adresa
* Eveniment (descriere)
* De plata
* data producerii evenimentului
* agentul care a constatat evenimentul
* timpul de reactie la eveniment (in cat timp a ajuns agentul la fata locului)

Stiind ca Nume client are maxim 14 caractere, CNP are exact 13 caractere, adresa are maxim 10 caractere, eveniment are maxim 20 caractere, agentul este de maxim 10 caractere, de plata este un numar cu 2 zecimale, pozitiv, timpul de reactie este pozitiv si exprimat in secunde, se cere:

1. Să se realizeze proiectarea bazei de date aferente (structura de tabele, structura de coloane a fiecărei tabele, constrângeri).
2. Sa se scrie comenzile SQL pentru tabelele proiectate la punctul anterior
3. Să se scrie comenzile SQL pentru popularea bazei de date cu urmatoarele informatii: 4 clienti, 2 avand cate 3 evenimente, unul un eveniment si altul niciun eveniment.
4. Să se scrie o procedura care sa permita adaugarea unui eveniment - procedura va fi apelata cu parametrii: CNP, eveniment, data, agent si timp de reactie. Procedura va actualiza si suma de plata a clientului, stiind ca un eveniment pe an este gratuit, si ca restul costa 10 RON daca timpul a fost sub 300 secunde si 5 RON altfel.
5. Sa se genereze un raport care sa cuprinda numele, CNP-ul si numarul total de evenimente, precum si timpul mediu de reactie al tuturor agentilor pentru acele evenimente.
6. Sa se genereze un raport detaliat care sa cuprinda numele, CNP-ul, evenimentul, data, agentul, timpul de reactie, ordonat dupa nume client, agent in sens crescator si dupa timpul de reactie in sens descrescator.
7. Sa se scrie un trigger care dupa adaugarea unui eveniment, sa actualizeze suma de plata.
8. Sa se scrie o functie care sa primeasca ca si parametru CNP-ul si sa returneze timpul mediu de reactie la toate evenimentele acelui client.
9. Sa se afiseze clientul cu cele mai multe evenimente, precizand: numele clientului, CNPul, adresa, evenimentul, agentul.
10. Sa se afiseze agentul cu cele mai multe evenimente, precizand agentul, evenimentul, timpul de reactie si ponderea timpului de reactie pe fiecare caz in timpul total de reactie al agentului.

**Rezolvare:**

**1. Să se realizeze proiectarea bazei de date aferente (structura de tabele, structura de coloane a fiecărei tabele, constrângeri).**

*Am impartit baza de date in 2 tabele, cu urmatoarea structura:*

* Tabela Clienti:

-CNP char(13) PRIMARY KEY

-Nume varchar(14) NOT NULL

-Adresa varchar(10) NOT NULL

-De\_plata number(7,2) NOT NULL CHECK (De\_plata >= 0)

* Tabela Evenimente:

-CNP char(13) NOT NULL REFERENCES Clienti (CNP)

-Eveniment varchar(20) NOT NULL

-Data\_eveniment date NOT NULL

-Agent varchar(10) NOT NULL

-Timp\_de\_reactie integer NOT NULL CHECK (Timp\_de\_reactie > 0)

**2. Sa se scrie comenzile SQL pentru tabelele proiectate la punctul anterior.**

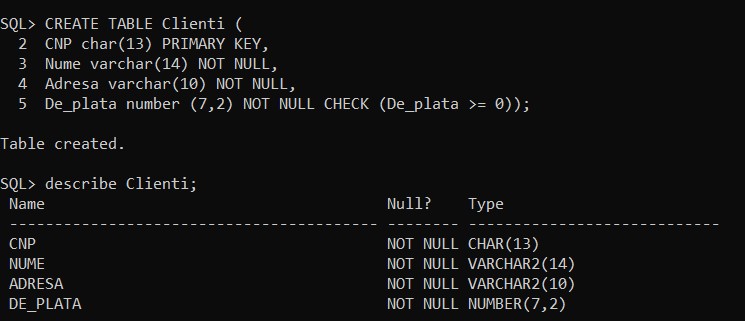
CREATE TABLE Clienti (

CNP char(13) PRIMARY KEY,

Nume varchar(14) NOT NULL,

Adresa varchar(10) NOT NULL,

De\_plata number (7,2) NOT NULL CHECK (De\_plata >= 0));



CREATE TABLE Evenimente (

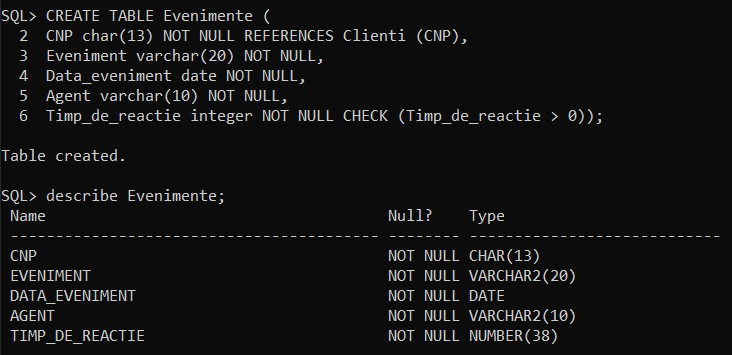
CNP char(13) NOT NULL REFERENCES Clienti (CNP),

Eveniment varchar(20) NOT NULL,

Data\_eveniment date NOT NULL,

Agent varchar(10) NOT NULL,

Timp\_de\_reactie integer NOT NULL CHECK (Timp\_de\_reactie > 0));



*Legatura dintre cele 2 tabele se face prin campul “CNP”, fiind o legatura de tipul “references”.*

*Legatura dintre cele 2 tabele provoaca urmatoarea restrictie : o linie din tabela Clienti nu poate fi stearsa pana cand toate liniile din tabela Evenimente care contin CNP-ul liniei care se doreste a fi stearsa nu sunt sterse.*

**3. Să se scrie comenzile SQL pentru popularea bazei de date cu urmatoarele informatii: 4 clienti, 2 avand cate 3 evenimente, unul un eveniment si altul niciun eveniment.**

SET AUTOCOMMIT ON;

SELECT sysdate FROM dual;

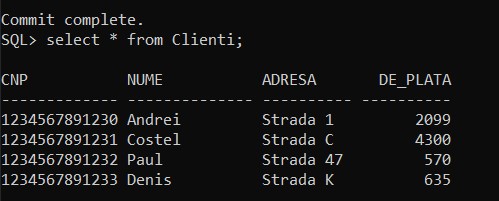
ALTER session SET nls\_date\_format='DD-MM-YYYY';

INSERT INTO Clienti VALUES ('1234567891230', 'Andrei ', 'Strada 1', 2099);

INSERT INTO Clienti VALUES ('1234567891231', 'Costel', 'Strada C', 4300);

INSERT INTO Clienti VALUES ('1234567891232', 'Paul', 'Strada 47', 570);

INSERT INTO Clienti VALUES ('1234567891233', 'Denis', 'Strada K', 635);



INSERT INTO Evenimente VALUES ('1234567891230', 'bataie in bar', '14-05-2021',

'Agentul 7', 35);

INSERT INTO Evenimente VALUES ('1234567891230', 'nunta', '08-12-2023', 'Agent Q',

10);

INSERT INTO Evenimente VALUES ('1234567891230', 'majorat', '17-10-2021', 'Agent

431', 20);

INSERT INTO Evenimente VALUES ('1234567891231', 'botez', '25-02-2019', 'Agent

Ion', 40);

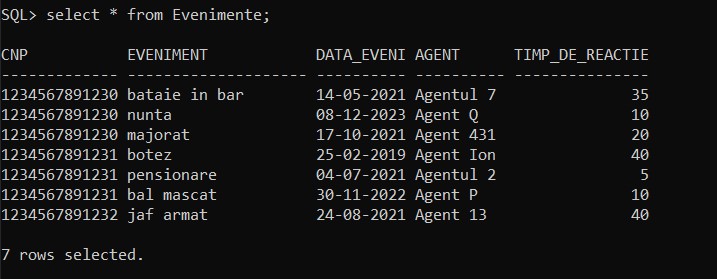
INSERT INTO Evenimente VALUES ('1234567891231', 'pensionare', '04-07-2021',

'Agentul 2', 5);

INSERT INTO Evenimente VALUES ('1234567891231', 'bal mascat', '30-11-2022', 'Agent P', 10);

INSERT INTO Evenimente VALUES ('1234567891232', 'jaf armat', '24-08-2021', 'Agent

13', 40);



**4. Să se scrie o procedura care sa permita adaugarea unui eveniment - procedura va fi apelata cu parametrii: CNP, eveniment, data, agent si timp de reactie. Procedura va actualiza si suma de plata a clientului, stiind ca un eveniment pe an este gratuit, si ca restul costa 10 RON daca timpul a fost sub 300 secunde si 5 RON altfel.**

create or replace procedure Adaugare\_Eveniment(CNP1 varchar, eveniment1 varchar,

data\_eveniment1 date, agent1 varchar, timp\_de\_reactie1 integer)

as

pret numeric(7,2); k integer:=0; BEGIN

insert into Evenimente (CNP,Eveniment,Data\_eveniment,Agent,Timp\_de\_reactie)

VALUES (CNP1,Eveniment1,Data\_eveniment1,agent1,timp\_de\_reactie1); select count(CNP) into k from Evenimente WHERE CNP1=CNP; select de\_plata into Pret from Clienti WHERE CNP1=CNP; if(pret=0 AND k=0) THEN

pret:=0;

elsif timp\_de\_reactie1<300 THEN pret:=pret+10;

elsif timp\_de\_reactie1>=300 THEN

pret:=pret+5; end if;

update Clienti SET de\_plata=pret

WHERE CNP1=CNP;

end;

/

*Pasii procedurii:*

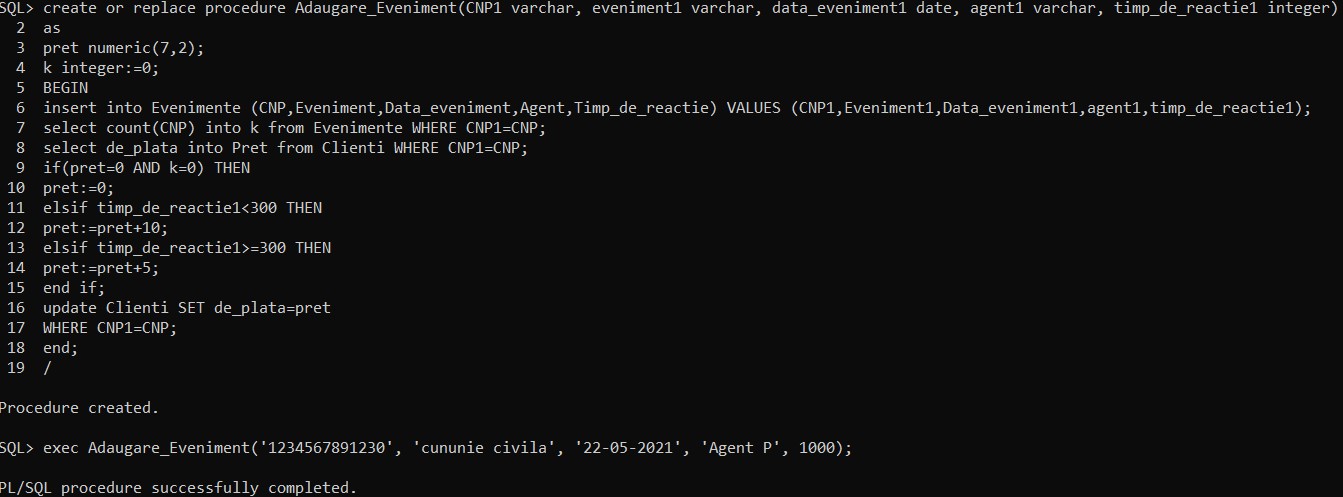
*-introduce datele primite in tabela Evenimente*

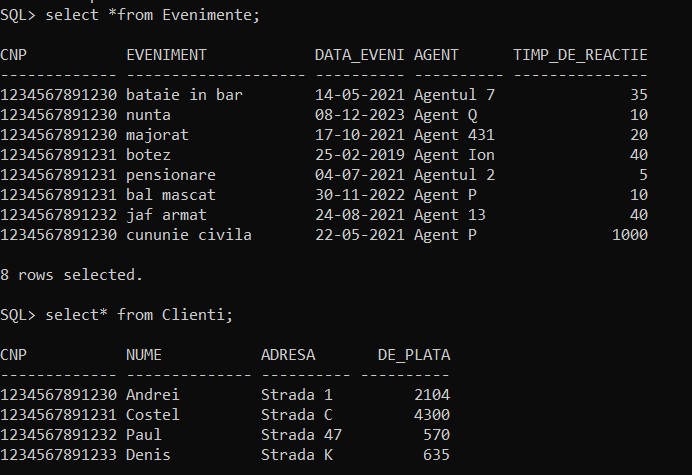
*-numara de cate ori a fost servit un client, pentru a stabili daca interventia este sau nu gratuita, rezultatul fiind pus in variabila k*

*-se cauta in tabela Clienti pretul serviciului corespunzator clientului cu CNP-ul primit ca parametru si se pune in variabila pret. Daca serviciul este gratuit campul “de\_plata” va fi actualizat cu val 0, iar daca nu acesta se va actualiza cu pretul corespunzator timpului de interventie (10 pt timp<300 si 5 altfel)*

*Pentru executarea procedurii se foloseste comanda:*

exec Adaugare\_Eveniment('1234567891230', 'cununie civila', '22-05-2021', 'Agent P', 1000);





**5. Sa se genereze un raport care sa cuprinda numele, CNP-ul si numarul total de evenimente, precum si timpul mediu de reactie al tuturor agentilor pentru acele evenimente.**

select A.CNP, A.Nume, COUNT(B.CNP) AS Numar\_Evenimente,

SUM(B.timp\_de\_reactie)/COUNT(B.CNP) AS Timp\_Mediu\_De\_Reactie

FROM Clienti A, Evenimente B

WHERE A.CNP=B.CNP

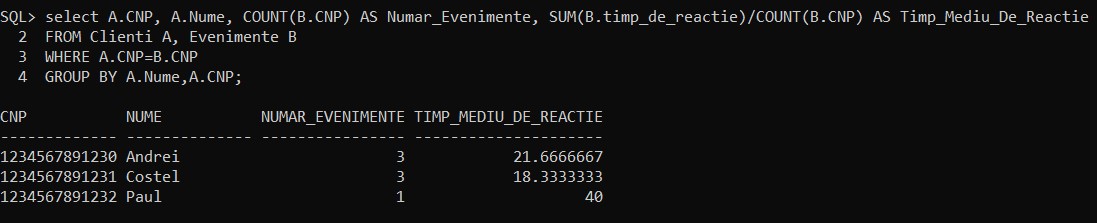
GROUP BY A.Nume,A.CNP;

*Pasii raportului:*

*-calculeaza suma timpilor de reactie (cu functia “SUM”) si o imparte la numarul de evenimente care au avut loc (numarate cu functia “COUNT”)*

*-grupeaza datele dupa numele si CNP-ul clientului*

*Conditia A.CNP=B.CNP are rolul de a sincroniza cele 2 tabele, abesnta acesteia ducand la rezultate gresite.*



1. **Sa se genereze un raport detaliat care sa cuprinda numele, CNP-ul, evenimentul, data, agentul, timpul de reactie, ordonat dupa nume client, agent in sens crescator si dupa timpul de reactie in sens descrescator.**

select A.CNP, A.Nume, B.Eveniment, B.Data\_Eveniment, B.Agent, B.Timp\_de\_reactie

FROM Clienti A, Evenimente B

WHERE A.CNP=B.CNP

ORDER BY A.Nume ASC, B.Agent ASC, B.Timp\_de\_reactie DESC;

*Sunt selectate campurile cerute, este aplicata conditia de sincronizare a tabelelor (A.CNP=B.CNP), iar apoi sunt ordonate dupa criteriile cerute, timpul de reactie fiind ordonat descrescator prin clauza “DESC”.*



1. **Sa se scrie un trigger care dupa adaugarea unui eveniment, sa actualizeze suma de plata.**

create or replace trigger Suma\_Plata before insert on Evenimente for each row declare pret NUMERIC(7,2);

k INTEGER :=0; CNP2 VARCHAR(13); timp\_de\_reactie2 INTEGER;

begin

CNP2:=:NEW.CNP;

timp\_de\_reactie2:=:NEW.timp\_de\_reactie; select count(CNP) into k from Evenimente WHERE CNP2=CNP; select de\_plata into pret from Clienti WHERE CNP2=CNP; if (pret=0 AND k=0) THEN

pret:=0;

elsif (timp\_de\_reactie2<300) THEN pret:=pret+10;

elsif (timp\_de\_reactie2>=300) THEN

pret:=pret+5; end if; update Clienti set de\_plata =pret *WHERE CNP2=Clienti.CNP;* *end;*

*/*

*Pasii Trigger-ului:*

* *se salveaza mai intai cnp-ul in variabila temporara CNP2 (CNP –ul este luat din campul :new.cnp care reprezinta CNP-ul dat intr-o comanda de tip insert pe tabela evenimente)*

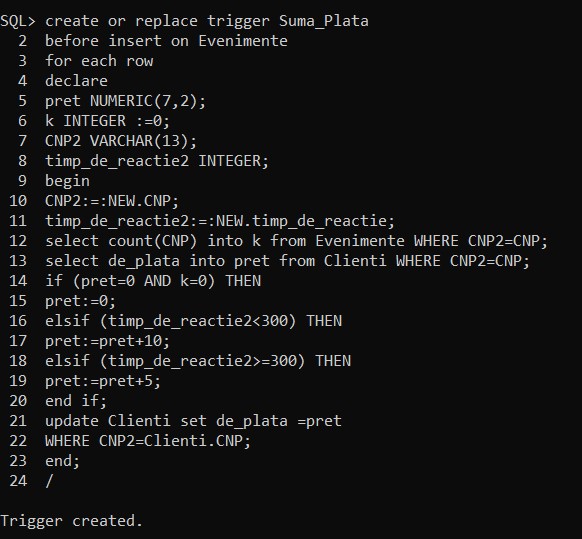
*-se salveaza campul timp\_de\_reactie din :new.timp\_de\_reactie in variabila timp\_de\_reactie2*

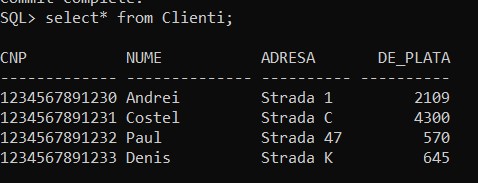
* *se numara de cate ori a fost servit clientul cu cnp-ul dat -se determina costul pe serviciu pentru clientul curent - se fac urmatoarele verificari:*

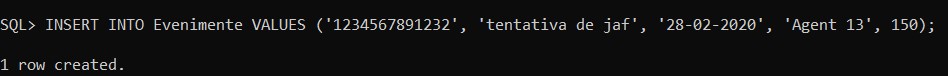
*\*daca nu a mai fost servit si pretul costul serviciului este 0 atunci pretul final este 0*

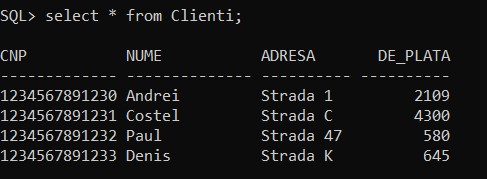
*\* daca aceasta nu este prima oara cand este servit , se calculeaza pretul final astfel: pretul pe serviciu + pretul pe timpul d interventie*

* *se actualizeaza campul de\_plata al tabelei clienti cu pretul rezultat la pasul anterior.*









*Se observa ca a fost adaugata suma de 10 lei clientului de pe a 3-a linie.*

**8. Sa se scrie o functie care sa primeasca ca si parametru CNP-ul si sa returneze timpul mediu de reactie la toate evenimentele acelui client.**

create or replace function Timp\_Mediu(CNP1 VARCHAR) return NUMERIC

as

timp\_mediu NUMERIC(10,2); begin

select sum(timp\_de\_reactie)/count(CNP1) into timp\_mediu from Evenimente

WHERE CNP1=CNP group by CNP1; return timp\_mediu; end;

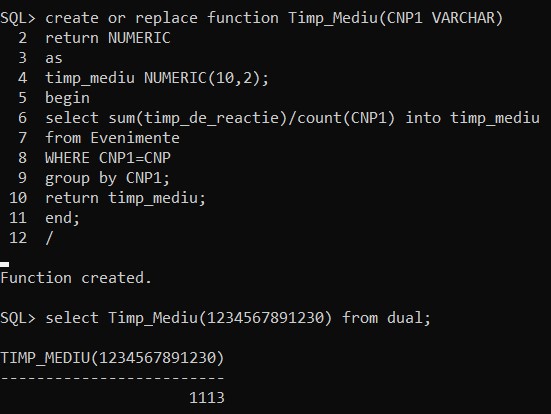
/

Pasii functiei:

*- se determina timpul mediu de reactie pentru un cnp dat calculandu-se media timpilor de reactie si se salveza in variabila timp\_mediu*

*-returneaza val lui timp\_mediu*

*Functia este apelata prin comanda: SELECT Timp\_Mediu(CNP-ul dorit) from dual;*



**9. Sa se afiseze clientul cu cele mai multe evenimente, precizand: numele clientului, CNP-ul, adresa, evenimentul, agentul.**

SELECT A.CNP, A.Nume, A.Adresa, B.Eveniment, B.Agent

FROM Clienti A, Evenimente B where A.CNP=B.CNP AND B.CNP in( SELECT CNP

from Evenimente

GROUP BY CNP

HAVING COUNT(\*) = (

SELECT MAX(COUNT(Eveniment))

from Evenimente

GROUP BY CNP));

*Explicatie:*

*-am determinat clientul cu numarul maxim de evenimente, folosind CNP-ul sau din tabelul Evenimente (B.CNP)*

B.CNP in(

SELECT CNP

from Evenimente

GROUP BY CNP

HAVING COUNT(\*) = (

SELECT MAX(COUNT(Eveniment))

from Evenimente

GROUP BY CNP)

-*am pus conditia “A.CNP=B.CNP”, pentru a face legatura intre cele 2 tabele, prin CNP-*

*ul clientului*

*-la final am imbinat comenzile, afisand astfel clientul cu cele mai multe evenimente*



**10. Sa se afiseze agentul cu cele mai multe evenimente, precizand agentul, evenimentul, timpul de reactie si ponderea timpului de reactie pe fiecare caz in timpul total de reactie al agentului.**

select Agent, Eveniment, timp\_de\_reactie, (timp\_de\_reactie\*100)/

(select SUM(timp\_de\_reactie) from Evenimente where Agent in( select Agent from Evenimente

GROUP BY Agent HAVING COUNT(\*) = ( select MAX(COUNT(Eveniment)) from Evenimente

GROUP BY Agent))) as Pondere

from Evenimente where Agent in( select Agent from Evenimente

GROUP BY Agent HAVING COUNT(\*) = ( select MAX(COUNT(Eveniment))

from Evenimente GROUP BY Agent));

*Explicatie:*

*-am determinat agentul cu numarul maxim de evenimente:*

*select Agent from Evenimente*

*GROUP BY Agent*

*HAVING COUNT(\*) = (*

*select MAX(COUNT(Eveniment)) //comanda pentru a determina numarul maxim de evenimente from Evenimente GROUP BY Agent)*

*-am determinat suma totala a timpului de reactie (la care am impartit apoi timpul de reactie inmultit cu 100, pentru a stabili ponderea)*

*select SUM(timp\_de\_reactie) from Evenimente where Agent in( select Agent from Evenimente*

*GROUP BY Agent HAVING COUNT(\*) = ( select MAX(COUNT(Eveniment))*

*from Evenimente*

*GROUP BY Agent*

*-la final, am imbinat comenzile de mai sus, pentru a afisa agentul cu cele mai multe evenimente, impreuna cu ponderea acestuia.*

