

Universitatea din Bucureşti
Facultatea de Matematică-Informatică

PROIECT

SISTEME DE GESTIUNE

A BAZELOR DE DATE

Voiculescu Alina-Virginia

Grupa 231

Cuprins

1. [Prezentare utilitate bazei de date.](#)
2. [Diagramă entitate-relație.](#)
3. [Diagramă conceptuală.](#)
4. [Creare tabele.](#)
5. [Inserare date.](#)
6. [Subprogram stocat – colecții.](#)
7. [Subprogram stocat – cursor.](#)
8. [Subprogram stocat de tip funcție.](#)
9. [Subprogram stocat de tip procedură.](#)
10. [Trigger de tip LMD la nivel de comandă.](#)
11. [Trigger de tip LMD la nivel de linie.](#)
12. [Trigger de tip LDD.](#)
13. [Pachet – subpuncte anterioare.](#)
14. [Pachet – obiecte necesare unui flux de acțiuni integrate.](#)



1

Prezentați, pe scurt, baza de date (utilitatea ei).

Descrierea modelului real, a utilității acestuia și a regulilor de funcționare

Furturile bancare sunt, din păcate, din ce în ce mai practicate în cotidian. Din această cauză, dar și datorită dorinței de a prinde răufăcătorii, este necesar a fi ținută evidență furturilor bancare. Astfel, modelul real propus se numește Gestiunea furturilor bancare. El surprinde băncile afectate, clienții acestora, martorii la jaf, hoții, sentința pe care trebuie să o deservească și unde va fi petrecută, dar și situația în care se află în momentul actual – arestați sau nu.

Prezentarea constrângerilor(restricții, reguli) impuse asupra modelului

Modelul prezentat conține o serie de restricții și de reguli. Acestea vor fi prezentate în paragraful următor:

Pot exista orașe fără închisori.

Toate băncile prevăzute în model au fost jefuite.

O bancă poate să nu aibă niciun client.

Nu este obligatoriu ca un jaf să aibă martori.

Hoțul de bănci, uneori, nu este găsit în momentul interogării modelului. Asta înseamnă că acesta nu este încă arestat.

Unii hoți nu primesc sentință, deoarece aceasta se referă la închisoare. Ei sunt cei cu probleme psihice grave, fiind duși în diverse centre și spitale.

Pot exista secții de poliție în care nu lucrează niciun polițist. Acestea sunt secții vechi, desființate.

Descrierea entităților, incluzând precizarea cheii primare.

ENTITATE	CHEIE PRIMARĂ	DESCRIERE
ȚARĂ	id_țară	Entitatea reprezintă țările în care pot exista încisori.
ORĂȘ	id_oraș	Entitatea reprezintă orașele în care pot exista încisori.
ÎNCHISOARE	id_închisoare	Entitatea reprezintă clădirile (încisorile) în care este/va fi petrecut timpul precizat în sentință.
SENTINȚĂ	id_sentință	Entitatea reprezintă timpul care trebuie petrecut în închisoare (sentrtele) acordat(e) hoților de către judecători.
BANCĂ	id_bancă	Entitatea reprezintă clădirile (bâncile) care au fost jefuite de-a lungul timpului.
CLIENT	id_client	Entitatea reprezintă persoanele care au realizat acțiuni în bâncile jefuite și care nu sunt angajați (clientii unei bânci jefuite).
HOT_DE_BĂNCI	id_hoț	Entitatea reprezintă persoana care săvârșește actul infracțional (furtul bancar).
JUDECĂTOR	id_judecător	Entitatea reprezintă persoana care are dreptul să dea hoților o sentință.
JAF	id_jaf	Entitatea reprezintă actul infracțional săvârșit de hoții de bânci.
MARTOR	id_martor	Entitatea reprezintă persoanele care au participat la jaf, dar nu sunt hoții.
POLIȚIST	id_polițist	Entitatea reprezintă persoanele care au dreptul de a aresta hoții.
SECTIE_POLIȚIE	id_sectie	Entitatea reprezintă clădirile (secțiile) în care lucrează polițistii.

Descrierea relațiilor și precizarea cardinalității acestora

RELATIE	DESCRIERE	CARDINALITATE
ȚARĂ – ORAȘ	Relația se numește “se află în” și face conexiunea între fiecare oraș și țara în care se află.	one-to-many
ORAȘ – ÎNCHISOARE	Relația se numește “are” și întocmește o legătură între fiecare închisoare și orașul în care se află.	one-to-many
ÎNCHISOARE – SENTINȚĂ	Relația se numește “va fi petrecută” și face conexiunea între fiecare sentință și locul în care va fi petrecută.	one-to-many
ORAȘ – BANCĂ	Relația se numește “se situează în” și întocmește o legătură între fiecare bancă și orașul în care se situează.	one-to-many
BANCĂ – CLIENT	Relația se numește “are” și face conexiunea între fiecare client al unei bânci și, desigur, banca.	one-to-many
BANCĂ – JAF	Relația se numește “face parte din” și face conexiunea între fiecare jaf și banca unde s-a petrecut.	one-to-many
MARTOR – JAF	Relația se numește “a văzut” și întocmește o legătură între fiecare martor și fiecare jaf pe care l-a vizualizat.	many-to-many

HOT_DE_BĂNCI – JAF	Relația se numește “face parte la” și face conexiunea între fiecare bancă jefuită și fiecare hoț de către care a fost jefuită.	many-to-many
HOT_DE_BĂNCI – POLIȚIST	Relația se numește “arestează” și întocmește o legătură între fiecare hoț de bănci și fiecare polițist de către care a fost arestat.	many-to-many
SECTIE_POLIȚIE – POLIȚIST	Relația se numește “lucrează la” și face conexiunea între fiecare polițist și secția de poliție la care lucrează.	one-to-many
SENTINTĂ – JUDECĂTOR – HOT_DE_BĂNCI	Relația se numește “primește” și se referă la sentința primită de un hoț de bănci, acordată de un judecător.	Relație de tip 3 (many-to-many-to-many)

Descrierea atributelor, incluzând tipul de date și eventualele constrângeri, valori implicate, valori posibile ale atributelor

Atributele descrise mai jos au nume simple, sugestive, pentru ca diagrama entitate-relație, diagrama conceptuală și viitoarea bază de date să fie ușor de înțeles de către cei care o vizualizează.

Entitatea ȚARĂ:

ATRIBUT	TIP DE DATE	CONSTRÂNGERE	CHEIE PRIMARĂ
id_țară	VARCHAR2(5)	NOT NULL	Da
denumire_țară	VARCHAR2(30)	NOT NULL	Nu

Entitatea ORAȘ:

ATRIBUT	TIP DE DATE	CONSTRÂNGERE	CHEIE PRIMARĂ
id oraș	NUMBER(6)	NOT NULL	Da
denumire oraș	VARCHAR2(30)	NOT NULL	Nu

Entitatea ÎNCHISOARE:

ATRIBUT	TIP DE DATE	CONSTRÂNGERE	CHEIE PRIMARĂ
id_închisoare	NUMBER(6)	NOT NULL	Da
denumire_închisoare	VARCHAR2(30)	NOT NULL	Nu

Entitatea SENTINȚĂ:

ATRIBUT	TIP DE DATE	CONSTRÂNGERE	CHEIE PRIMARĂ
id_sentință	NUMBER(6)	NOT NULL	Da
luni_de_închisoare	NUMBER(10, 2)	NOT NULL	Nu

Entitatea BANCĂ:

ATRIBUT	TIP DE DATE	CONSTRÂNGERE	CHEIE PRIMARĂ
id_bancă	NUMBER(6)	NOT NULL	Da
denumire_bancă	VARCHAR2(30)	NOT NULL	Nu

Entitatea JUDECĂTOR:

ATRIBUT	TIP DE DATE	CONSTRÂNGERE	CHEIE PRIMARĂ
id_judecător	NUMBER(6)	NOT NULL	Da
nume_judecător	VARCHAR2(30)	NOT NULL	Nu
prenume_judecător	VARCHAR2(30)	NOT NULL	Nu
vârstă	NUMBER(6)	-	Nu
denumire_tribunal	VARCHAR2(30)	NOT NULL	Nu

Entitatea CLIENT:

ATRIBUT	TIP DE DATE	CONSTRÂNGERE	CHEIE PRIMARĂ
id_client	NUMBER(6)	NOT NULL	Da
nume_client	VARCHAR2(30)	NOT NULL	Nu
prenume_client	VARCHAR2(30)	-	Nu
vârstă_client	NUMBER(6)	-	Nu

Entitatea HOT_DE_BĂNCI:

ATRIBUT	TIP DE DATE	CONSTRÂNGERE	CHEIE PRIMARĂ
id_hot	NUMBER(6)	NOT NULL	Da
nume_hot	VARCHAR2(30)	NOT NULL	Nu
prenume_hot	VARCHAR2(30)	-	Nu
vârstă_hot	NUMBER(6)	-	Nu

Entitatea MARTOR:

ATRIBUT	TIP DE DATE	CONSTRÂNGERE	CHEIE PRIMARĂ
id_martor	NUMBER(6)	NOT NULL	Da
nume_martor	VARCHAR2(30)	NOT NULL	Nu
prenume_martor	VARCHAR2(30)	-	Nu
vârstă_martor	NUMBER(6)	-	Nu

Entitatea JAF:

ATRIBUT	TIP DE DATE	CONSTRÂNGERE	CHEIE PRIMARĂ
id_jaf	NUMBER(6)	NOT NULL	Da
dată_jaf	DATE	NOT NULL	Nu

Entitatea POLIȚIST:

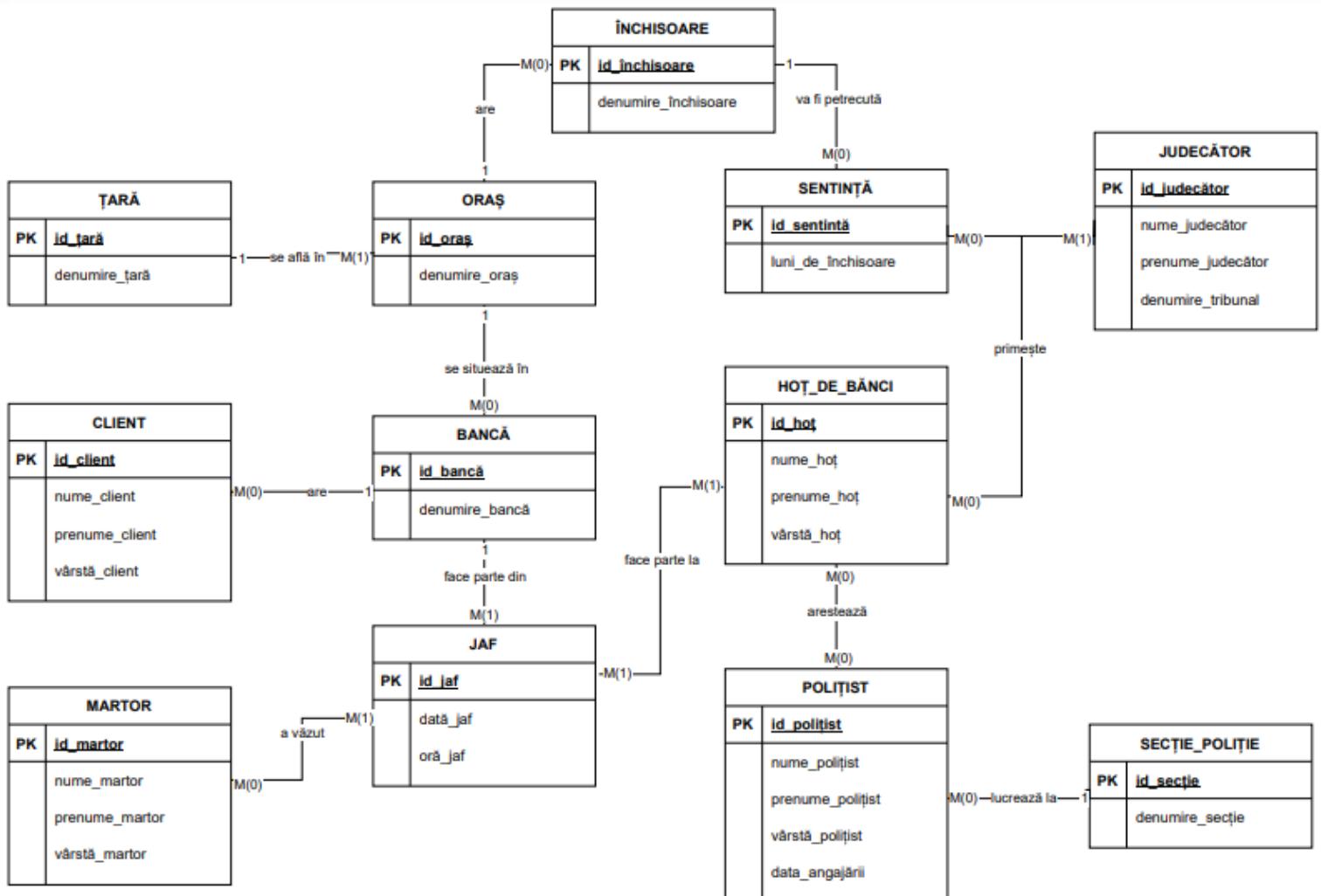
ATRIBUT	TIP DE DATE	CONSTRÂNGERE	CHEIE PRIMARĂ
id_polițist	NUMBER(6)	NOT NULL	Da
nume_polițist	VARCHAR2(30)	NOT NULL	Nu
prenume_polițist	VARCHAR2(30)	-	Nu
vârstă_polițist	NUMBER(6)	-	Nu
data_angajării	DATE	-	Nu

Entitatea SECTIE_POLITIE:

ATRIBUT	TIP DE DATE	CONSTRÂNGERE	CHEIE PRIMARĂ
id_sectie	NUMBER(6)	NOT NULL	Da
denumire_sectie	VARCHAR2(35)	NOT NULL	Nu

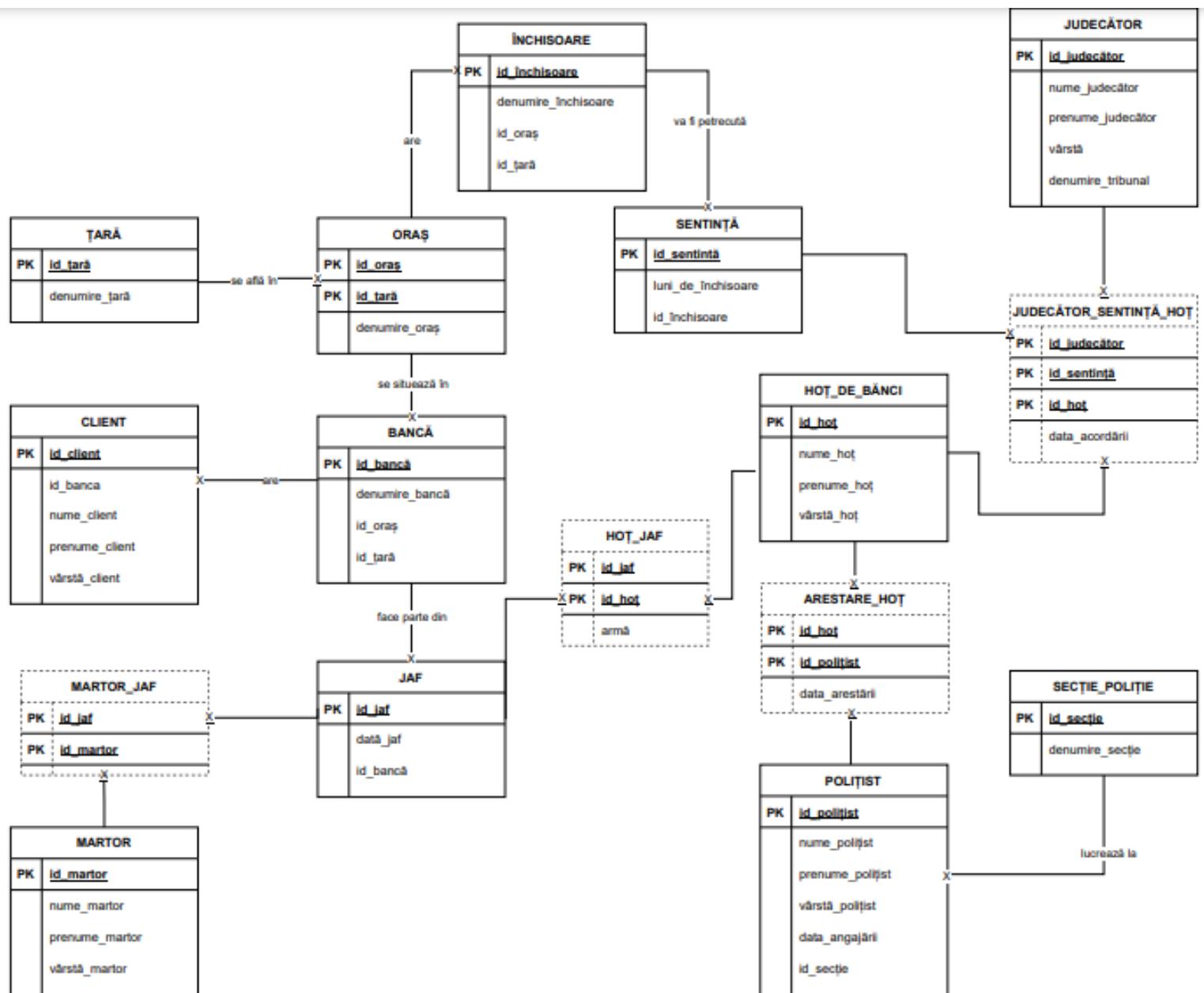
2

Realizați diagrama entitate-relație (ERD).





Pornind de la diagrama entitate-relație, realizați diagrama conceptuală a modelului propus, integrând toate atributele necesare.



4

Implementați în Oracle diagrama conceptuală realizată: definiți toate tabelele, implementând toate constrângerile de integritate necesare (chei primare, cheile externe etc).

Acest exercițiu se află și în fișierul text 231_Voiculescu_Alina_sursa.txt.

Crearea tabelelor:

Cod SQL:

```
create table tara(
    id_tara varchar2(5) primary key,
    denumire_tara varchar2(30) not null
);

create table oras(
    id_oras number(6),
    id_tara varchar2(5),
    denumire_oras varchar2(30) not null
);

create table inchisoare(
    id_inchisoare number(6) primary key,
    denumire_inchisoare varchar2(30) not null,
    id_oras number(6),
    id_tara varchar(5)
);

create table sentinta(
    id_sentinta number(6) primary key,
    luni_de_inchisoare number(10, 2) not null,
    id_inchisoare number(6)
);
```

```

create table judecator(
    id_judecator number(6) primary key,
    nume_judecator varchar2(30) not null,
    prenume_judecator varchar2(30) not null,
    varsta number(6),
    denumire_tribunal varchar2(30) not null
);

create table hot_de_banci(
    id_hot number(6) primary key,
    nume_hot varchar2(30) not null,
    prenume_hot varchar2(30),
    varsta_hot number(6)
);

create table judecator_sentinta_hot(
    id_judecator number(6),
    id_sentinta number(6),
    id_hot number(6),
    data_acordarii date
);

create table banca(
    id_banca number(6) primary key,
    denumire_banca varchar2(30) not null,
    id_oras number(6),
    id_tara varchar(5)
);

create table client(
    id_client number(6) primary key,
    id_banca number(6),
    nume_client varchar2(30) not null,
    prenume_client varchar2(30),
    varsta_client number(6)
);

create table jaf(
    id_jaf number(6) primary key,
    id_banca number(6),
    data_jaf date not null
);

create table hot_jaf(
    id_jaf number(6),
    id_hot number(6),
    arma varchar2(30)
);

create table martor(
    id_martor number(6) primary key,
    nume_martor varchar2(30) not null,
    prenume_martor varchar2(30),
    varsta_martor number(6)
);

create table sectie_politie(
    id_sectie number(6) primary key,

```

```

        denumire_sectie varchar2(35) not null
);

create table politist(
    id_politist number(6) primary key,
    nume_politist varchar2(30) not null,
    prenume_politist varchar2(30),
    varsta_politist number(6),
    data_angajarii date,
    id_sectie number(6)
);

create table martor_jaf(
    id_jaf number(6),
    id_martor number(6)
);

create table arestare_hot(
    id_hot number(6),
    id_politist number(6),
    data_arestarii date
);

alter table oras
add constraint fk_oras_tara foreign key (id_tara) references
tara(id_tara);

alter table oras
add constraint pk_oras primary key (id_oras, id_tara);

alter table inchisoare
add constraint fk_inchisoare_oras foreign key (id_oras, id_tara)
references oras(id_oras, id_tara);

alter table sentinta
add constraint fk_sentinta_inchisoare foreign key (id_inchisoare)
references inchisoare(id_inchisoare);

alter table judecator_sentinta_hot
add constraint fk_judecator_sentinta_hot foreign key (id_judecator)
references judecator(id_judecator);

alter table judecator_sentinta_hot
add constraint fk_sentinta_judecator_hot foreign key (id_sentinta)
references sentinta(id_sentinta);

alter table judecator_sentinta_hot
add constraint fk_hot_judecator_sentinta foreign key (id_hot)
references hot_de_banci(id_hot);

alter table judecator_sentinta_hot
add constraint pk_judecator_sentinta_hot primary key (id_judecator,
id_sentinta, id_hot);

```

```

alter table banca
add constraint fk_banca_oras foreign key (id_oras, id_tara) references
oras(id_oras, id_tara);

alter table client
add constraint fk_client_banca foreign key (id_banca) references
banca(id_banca);

alter table jaf
add constraint fk_jaf_banca foreign key (id_banca) references
banca(id_banca);

alter table hot_jaf
add constraint fk_jaf_hot foreign key (id_jaf) references jaf(id_jaf);

alter table hot_jaf
add constraint fk_hot_jaf foreign key (id_hot) references
hot_de_banci(id_hot);

alter table hot_jaf
add constraint pk_hot_jaf primary key (id_jaf, id_hot);

alter table politist
add constraint fk_politist_sectie foreign key (id_sectie) references
sectie_politie(id_sectie);

alter table martor_jaf
add constraint fk_jaf_martor foreign key (id_jaf) references
jaf(id_jaf);

alter table martor_jaf
add constraint fk_martor_jaf foreign key (id_martor) references
martor(id_martor);

alter table martor_jaf
add constraint pk_martor_jaf primary key (id_jaf, id_martor);

alter table arestare_hot
add constraint fk_hot_arestare foreign key (id_hot) references
hot_de_banci(id_hot);

alter table arestare_hot
add constraint fk_arestare_hot foreign key (id_politist) references
politist(id_politist);

alter table arestare_hot
add constraint pk_arestare_hot primary key (id_hot, id_politist);

commit;

```

SQL Worksheet History

Worksheet Query Builder

```
create table inchisoare(
    id_inchisoare number(6) primary key,
    denumire_inchisoare varchar2(30) not null,
    id_oras number(6),
    id_tara varchar2(5)
);

create table sentinta(
    id_sentinta number(6) primary key,
    luni_de_inchisoare number(10, 2) not null,
    id_inchisoare number(6)
);

create table judecator(
    id_judecator number(6) primary key,
    nume_judecator varchar2(30) not null,
    prenume_judecator varchar2(30) not null,
    varsta number(6),
    denumire_tribunal varchar2(30) not null
);
```

Script Output x

Task completed in 0.041 seconds

```
Table ARESTARE_HOT created.

Table ORAS altered.

Table ORAS altered.
```



5

Adăugați informații coerente în tabelele create (minim 5 înregistrări pentru fiecare entitate independentă; minim 10 înregistrări pentru tabelele asociative).

Acest exercițiu se află și în fișierul text 231_Voiculescu_Alină_sursa.txt.

Inserarea datelor:

Cod SQL:

```
begin
    insert into tara (id_tara, denumire_tara)
    values ('RO', 'Romania');

    insert into tara (id_tara, denumire_tara)
    values ('BE', 'Belgia');

    insert into tara (id_tara, denumire_tara)
    values ('ES', 'Spania');

    insert into tara (id_tara, denumire_tara)
    values ('JP', 'Japonia');

    insert into tara (id_tara, denumire_tara)
    values ('DE', 'Germania');

    insert into oras (id_oras, id_tara, denumire_oras)
    values (1, 'JP', 'Osaka');

    insert into oras (id_oras, id_tara, denumire_oras)
    values (2, 'RO', 'Bucuresti');

    insert into oras (id_oras, id_tara, denumire_oras)
    values (3, 'DE', 'Berlin');
```

```

insert into oras (id_oras, id_tara, denumire_oras)
values (4, 'BE', 'Saint-Gilles');

insert into oras (id_oras, id_tara, denumire_oras)
values (5, 'ES', 'Madrid');

insert into inchisoare (id_inchisoare, denumire_inchisoare,
id_oras, id_tara)
values (124, 'Inchisoarea Aranjuez', 5, 'ES');

insert into inchisoare (id_inchisoare, denumire_inchisoare,
id_oras, id_tara)
values (125, 'Inchisoarea Osaka', 1, 'JP');

insert into inchisoare (id_inchisoare, denumire_inchisoare,
id_oras, id_tara)
values (126, 'Inchisoarea St. Gilles', 4, 'BE');

insert into inchisoare (id_inchisoare, denumire_inchisoare,
id_oras, id_tara)
values (127, 'Penitenciarul Bucuresti-Jilava', 2, 'RO');

insert into inchisoare (id_inchisoare, denumire_inchisoare,
id_oras, id_tara)
values (128, 'Inchisoarea Plotzensee', 3, 'DE');

insert into sentinta (id_sentinta, luni_de_inchisoare,
id_inchisoare)
values (475, 120, 124);

insert into sentinta (id_sentinta, luni_de_inchisoare,
id_inchisoare)
values (476, 135, 128);

insert into sentinta (id_sentinta, luni_de_inchisoare,
id_inchisoare)
values (477, 200, 126);

insert into sentinta (id_sentinta, luni_de_inchisoare,
id_inchisoare)
values (478, 80, 125);

insert into sentinta (id_sentinta, luni_de_inchisoare,
id_inchisoare)
values (479, 110, 127);

insert into judecator (id_judecator, nume_judecator,
prenume_judecator, varsta, denumire_tribunal)
values (100, 'Gordon', 'Gilbert', 43, 'Tribunalul Bucuresti');

```

```

    insert into judecator (id_judecator, nume_judecator,
prenume_judecator, varsta, denumire_tribunal)
values (101, 'Vincent', 'Horace', 32, 'Tribunalul Bruxelles');

    insert into judecator (id_judecator, nume_judecator,
prenume_judecator, varsta, denumire_tribunal)
values (102, 'Villanova', 'Renata', 36, 'Tribunalul Madrid');

    insert into judecator (id_judecator, nume_judecator,
prenume_judecator, varsta, denumire_tribunal)
values (103, 'Lehmann', 'Emmerich', 58, 'Tribunalul Berlin');

    insert into judecator (id_judecator, nume_judecator,
prenume_judecator, varsta, denumire_tribunal)
values (104, 'Takayuki', 'Kawakami', 40, 'Tribunalul Tokyo');

    insert into hot_de_banci (id_hot, nume_hot, prenume_hot,
varsta_hot)
values (405, 'Osborne', 'Kizzy', 45);

    insert into hot_de_banci (id_hot, nume_hot, prenume_hot,
varsta_hot)
values (406, 'Armando', 'Anika', 54);

    insert into hot_de_banci (id_hot, nume_hot, prenume_hot,
varsta_hot)
values (407, 'Derichs', 'Roelof', 42);

    insert into hot_de_banci (id_hot, nume_hot, prenume_hot,
varsta_hot)
values (408, 'Spitznagel', 'Tania', 19);

    insert into hot_de_banci (id_hot, nume_hot, prenume_hot,
varsta_hot)
values (409, 'Jerome', 'Veronika', 48);

    insert into judecator_sentinta_hot (id_judecator, id_sentinta,
id_hot, data_acordarii)
values (100, 475, 405, to_date('17-Jun-00', 'DD-MON-RR'));

    insert into judecator_sentinta_hot (id_judecator, id_sentinta,
id_hot, data_acordarii)
values (100, 476, 406, to_date('26-Mar-73', 'DD-MON-YY'));

    insert into judecator_sentinta_hot (id_judecator, id_sentinta,
id_hot, data_acordarii)
values (100, 475, 407, to_date('13-Jan-98', 'DD-MON-YY'));

    insert into judecator_sentinta_hot (id_judecator, id_sentinta,
id_hot, data_acordarii)
values (101, 477, 406, to_date('7-Dec-85', 'DD-MON-YY'));

```

```

    insert into judecator_sentinta_hot (id_judecator, id_sentinta,
id_hot, data_acordarii)
    values (102, 479, 409, to_date('15-Nov-95', 'DD-MON-YY'));

    insert into judecator_sentinta_hot (id_judecator, id_sentinta,
id_hot, data_acordarii)
    values (103, 478, 406, to_date('10-Apr-02', 'DD-MON-YY'));

    insert into judecator_sentinta_hot (id_judecator, id_sentinta,
id_hot, data_acordarii)
    values (104, 476, 409, to_date('12-Oct-08', 'DD-MON-YY'));

    insert into judecator_sentinta_hot (id_judecator, id_sentinta,
id_hot, data_acordarii)
    values (104, 477, 405, to_date('17-Jul-20', 'DD-MON-YY'));

    insert into judecator_sentinta_hot (id_judecator, id_sentinta,
id_hot, data_acordarii)
    values (104, 477, 406, to_date('25-May-09', 'DD-MON-YY'));

    insert into judecator_sentinta_hot (id_judecator, id_sentinta,
id_hot, data_acordarii)
    values (104, 479, 408, to_date('20-Jan-20', 'DD-MON-YY'));

```



```

insert into banca (id_banca, denumire_banca, id_oras, id_tara)
values (500, 'Caixabank', 5, 'ES');

insert into banca (id_banca, denumire_banca, id_oras, id_tara)
values (501, 'BCR', 2, 'RO');

insert into banca (id_banca, denumire_banca, id_oras, id_tara)
values (502, 'Commerzbank', 3, 'DE');

insert into banca (id_banca, denumire_banca, id_oras, id_tara)
values (503, 'Mizuho Bank', 1, 'JP');

insert into banca (id_banca, denumire_banca, id_oras, id_tara)
values (504, 'Banque CPH', 4, 'BE');

```



```

insert into client (id_client, id_banca, nume_client,
prenume_client, varsta_client)
values (300, 502, 'Baaiman', 'Nadine', 47);

insert into client (id_client, id_banca, nume_client,
prenume_client, varsta_client)
values (301, 501, 'Gabrielli', 'Jaimie', 25);

insert into client (id_client, id_banca, nume_client,
prenume_client, varsta_client)
values (302, 503, 'Boucher', 'Denise', 56);

insert into client (id_client, id_banca, nume_client,
prenume_client, varsta_client)
values (303, 500, 'Foster', 'Paul', 72);

```

```

    insert into client (id_client, id_banca, nume_client,
prenume_client, varsta_client)
values (304, 504, 'Oliviero', 'Renato', 30);

insert into jaf (id_jaf, id_banca, data_jaf)
values (800, 503, to_date('17-Jan-73', 'DD-MON-YY'));

insert into jaf (id_jaf, id_banca, data_jaf)
values (801, 504, to_date('5-Dec-85', 'DD-MON-YY'));

insert into jaf (id_jaf, id_banca, data_jaf)
values (802, 502, to_date('13-Nov-95', 'DD-MON-YY'));

insert into jaf (id_jaf, id_banca, data_jaf)
values (803, 500, to_date('7-Apr-02', 'DD-MON-YY'));

insert into jaf (id_jaf, id_banca, data_jaf)
values (804, 501, to_date('9-Oct-08', 'DD-MON-YY'));

insert into jaf (id_jaf, id_banca, data_jaf)
values (805, 504, to_date('20-May-09', 'DD-MON-YY'));

insert into jaf (id_jaf, id_banca, data_jaf)
values (806, 502, to_date('15-Jul-20', 'DD-MON-YY'));


insert into hot_jaf (id_jaf, id_hot, arma)
values (800, 406, 'pistol');

insert into hot_jaf (id_jaf, id_hot, arma)
values (801, 406, 'cutit');

insert into hot_jaf (id_jaf, id_hot, arma)
values (802, 409, 'mitraliera');

insert into hot_jaf (id_jaf, id_hot, arma)
values (802, 407, 'pusca');

insert into hot_jaf (id_jaf, id_hot, arma)
values (802, 405, 'pistol');

insert into hot_jaf (id_jaf, id_hot, arma)
values (803, 406, 'cutit');

insert into hot_jaf (id_jaf, id_hot, arma)
values (804, 409, 'electrosoc');

insert into hot_jaf (id_jaf, id_hot, arma)
values (805, 406, 'electrosoc');

insert into hot_jaf (id_jaf, id_hot, arma)
values (806, 405, 'pistol');

insert into hot_jaf (id_jaf, id_hot, arma)
values (806, 408, 'spray paralizant');

```

```

    insert into martor (id_martor, nume_martor, prenume_martor,
varsta_martor)
        values (305, 'Norman', 'Valerie', 23);

    insert into martor (id_martor, nume_martor, prenume_martor,
varsta_martor)
        values (306, 'Snow', 'Clarence', 68);

    insert into martor (id_martor, nume_martor, prenume_martor,
varsta_martor)
        values (307, 'Mendez', 'Andrea', 40);

    insert into martor (id_martor, nume_martor, prenume_martor,
varsta_martor)
        values (308, 'Wu', 'Viktoria', 34);

    insert into martor (id_martor, nume_martor, prenume_martor,
varsta_martor)
        values (309, 'Mendez', 'Erik', 49);

    insert into martor (id_martor, nume_martor, prenume_martor)
values (310, 'Dumitrescu', 'Dragos');




    insert into sectie_politie (id_sectie, denumire_sectie)
values (600, 'Politia municipală Madrid');

    insert into sectie_politie (id_sectie, denumire_sectie)
values (601, 'Departamentul de politie Berlin');

    insert into sectie_politie (id_sectie, denumire_sectie)
values (602, 'Sectia de politie Saint Gilles');

    insert into sectie_politie (id_sectie, denumire_sectie)
values (603, 'Sectia de politie Osaka');

    insert into sectie_politie (id_sectie, denumire_sectie)
values (604, 'Sectia 2 Politie');




    insert into politist (id_politist, id_sectie, nume_politist,
prenume_politist, varsta_politist, data_angajarii)
        values (605, 600, 'Gimenez', 'Alejandro', 47, to_date('14-Jan-92',
'DD-MON-YY'));

    insert into politist (id_politist, id_sectie, nume_politist,
prenume_politist, varsta_politist, data_angajarii)
        values (606, 601, 'Herrmann', 'Loreley', 75, to_date('25-Feb-64',
'DD-MON-YY'));

    insert into politist (id_politist, id_sectie, nume_politist,
prenume_politist, varsta_politist, data_angajarii)
        values (607, 602, 'Piette', 'Janne', 63, to_date('1-Jun-76', 'DD-
MON-YY'));

```

```

    insert into politist (id_politist, id_sectie, nume_politist,
prenume_politist, varsta_politist, data_angajarii)
    values (608, 603, 'Iwamoto', 'Nishikawa', 48, to_date('11-Feb-91',
'DD-MON-YY'));

    insert into politist (id_politist, id_sectie, nume_politist,
prenume_politist, varsta_politist, data_angajarii)
    values (609, 604, 'Voiculescu', 'Alina', 30, to_date('1-Mar-09',
'DD-MON-YY'));

    insert into politist (id_politist, id_sectie, nume_politist,
prenume_politist, varsta_politist, data_angajarii)
    values (610, 604, 'Dumitrescu', 'Dragos', 38, to_date('1-Jun-05',
'DD-MON-YY'));


insert into martor_jaf (id_jaf, id_martor)
values (800, 306);

insert into martor_jaf (id_jaf, id_martor)
values (801, 306);

insert into martor_jaf (id_jaf, id_martor)
values (802, 307);

insert into martor_jaf (id_jaf, id_martor)
values (802, 309);

insert into martor_jaf (id_jaf, id_martor)
values (803, 305);

insert into martor_jaf (id_jaf, id_martor)
values (804, 308);

insert into martor_jaf (id_jaf, id_martor)
values (804, 305);

insert into martor_jaf (id_jaf, id_martor)
values (805, 308);

insert into martor_jaf (id_jaf, id_martor)
values (806, 308);

insert into martor_jaf (id_jaf, id_martor)
values (806, 309);


insert into arestare_hot (id_hot, id_politist, data_arestarii)
values (405, 605, to_date('30-Jun-00', 'DD-MON-YY'));

insert into arestare_hot (id_hot, id_politist, data_arestarii)
values (405, 609, to_date('20-Jul-20', 'DD-MON-YY'));

insert into arestare_hot (id_hot, id_politist, data_arestarii)
values (406, 606, to_date('5-Apr-73', 'DD-MON-YY'));

```

```

insert into arestare_hot (id_hot, id_politist, data_arestarii)
values (406, 607, to_date('30-Dec-85', 'DD-MON-YY'));

insert into arestare_hot (id_hot, id_politist, data_arestarii)
values (406, 605, to_date('25-Apr-02', 'DD-MON-YY'));

insert into arestare_hot (id_hot, id_politist, data_arestarii)
values (406, 609, to_date('26-May-09', 'DD-MON-YY'));

insert into arestare_hot (id_hot, id_politist, data_arestarii)
values (407, 608, to_date('29-Jan-98', 'DD-MON-YY'));

insert into arestare_hot (id_hot, id_politist, data_arestarii)
values (408, 609, to_date('27-Jan-20', 'DD-MON-YY'));

insert into arestare_hot (id_hot, id_politist, data_arestarii)
values (409, 608, to_date('15-Dec-95', 'DD-MON-YY'));

insert into arestare_hot (id_hot, id_politist, data_arestarii)
values (409, 609, to_date('15-Oct-08', 'DD-MON-YY'));

commit;

end;

```

ÎNAINTE:

The screenshot shows an SQL worksheet interface with the following details:

- Toolbar:** Includes icons for file operations (New, Open, Save, Print, Copy, Paste, Find, Replace, Undo, Redo), zoom, and font size.
- Tab Bar:** Shows "Worksheet" and "Query Builder".
- Script Area:** Contains the SQL code provided above, including the insert statements, commit, end, and select * from inchisoare; command.
- Status Bar:** Shows "Script Output x" and "Query Result x". Below it, it says "SQL | All Rows Fetched: 0 in 0.004 seconds".
- Result Area:** Displays a table header row with columns: ID_INCHISOARE, DENUMIRE_INCHISOARE, ID_ORAS, and ID_TARA.

ÎN TIMPUL INSERĂRII:

The screenshot shows the Oracle SQL Developer interface. The top part displays a PL/SQL script in the 'Worksheet' tab. The script inserts multiple rows into the 'arestare_hot' table, setting the 'data_arestarii' column to specific dates using the 'to_date' function. It includes a 'commit' statement and ends with a 'select * from inchisoare' command. The bottom part shows the 'Script Output' tab with the message 'PL/SQL procedure successfully completed.' and a note about the task completion time.

```
SQL Worksheet | History
Worksheet | Query Builder | SQL | PL/SQL | Data | Scripts | Help | Aa |
```

```
insert into arestare_hot (id_hot, id_politist, data_arestarii)
values (406, 607, to_date('30-Dec-85', 'DD-MON-YY'));

insert into arestare_hot (id_hot, id_politist, data_arestarii)
values (406, 605, to_date('25-Apr-02', 'DD-MON-YY'));

insert into arestare_hot (id_hot, id_politist, data_arestarii)
values (406, 609, to_date('26-May-09', 'DD-MON-YY'));

insert into arestare_hot (id_hot, id_politist, data_arestarii)
values (407, 608, to_date('29-Jan-98', 'DD-MON-YY'));

insert into arestare_hot (id_hot, id_politist, data_arestarii)
values (408, 609, to_date('27-Jan-20', 'DD-MON-YY'));

insert into arestare_hot (id_hot, id_politist, data_arestarii)
values (409, 608, to_date('15-Dec-95', 'DD-MON-YY'));

insert into arestare_hot (id_hot, id_politist, data_arestarii)
values (409, 609, to_date('15-Oct-08', 'DD-MON-YY'));

commit;

end;
/
select * from inchisoare;
```

Script Output | Query Result | Task completed in 0.199 seconds

PL/SQL procedure successfully completed.

DUPĂ INSERARE:

The screenshot shows a SQL Worksheet interface with a toolbar at the top and two main panes below. The top pane contains a script for inserting data into the 'arestare_hot' table and then selecting all rows from the 'inchisoare' table. The bottom pane displays the results of the 'select' query.

```
SQL Worksheet History
Worksheet Query Builder
values (407, 608, to_date('29-Jan-98', 'DD-MON-YY'));

insert into arestare_hot (id_hot, id_politist, data_arestariei)
values (408, 609, to_date('27-Jan-20', 'DD-MON-YY'));

insert into arestare_hot (id_hot, id_politist, data_arestariei)
values (409, 608, to_date('15-Dec-95', 'DD-MON-YY'));

insert into arestare_hot (id_hot, id_politist, data_arestariei)
values (409, 609, to_date('15-Oct-08', 'DD-MON-YY'));

commit;

end;
/

select * from inchisoare;
```

Script Output x | Query Result x

SQL | All Rows Fetched: 5 in 0.064 seconds

ID_INCHISOARE	DENUMIRE_INCHISOARE	ID_ORAS	ID_TARA
1	124 Inchisoarea Aranjuez	5 ES	
2	125 Inchisoarea Osaka	1 JP	
3	126 Inchisoarea St. Gilles	4 BE	
4	127 Penitenciarul Bucuresti-Jilava	2 RO	
5	128 Inchisoarea Plotzensee	3 DE	

6

Formulați în limbaj natural o problemă pe care să o rezolvați folosind un subprogram stocat care să utilizeze două tipuri de colecție studiate. Apelați subprogramul.

Acest exercițiu se află și în fișierul text 231_Voiculescu_Alina_sursa.txt.

Implementarea și rularea codului:

Exercițiu

Afișați numele, prenumele, vârsta, data angajării și toți hoții de bănci arestați de polițiștii care au peste 45 de ani. De asemenea, în cazul în care au fost angajați în zi de weekend, să se modifice data angajării, astfel încât să fie înregistrată în baza de date următoarea zi de luni de la data actuală a angajării.

Cod SQL:

```
create or replace procedure politisti_peste45 as
    type hoti is varray(10) of varchar2(30);
    type despre_politisti is record
        (id_politist.id_politist%type,
         nume_politist.nume_politist%type,
         prenume_politist.prenume_politist%type,
         varsta_politist.varsta_politist%type,
         data_angajarii_politist.data_angajarii%type);

    type hoti_prinsi is table of hoti index by pls_integer;
    type politisti is table of despre_politisti index by pls_integer;

    v_hoti hoti_prinsi;
    v_politisti politisti;
```

```

begin
    select id_politist, nume_politist, prenume_politist,
          varsta_politist, data_angajarii
    bulk collect into v_politisti
    from politist
    where varsta_politist > 45;

    for i in v_politisti.first..v_politisti.last loop
        select nume_hot "Nume hot"
        bulk collect into v_hoti(i)
        from hot_de_banci h
        join arestare_hot a on (a.id_hot = h.id_hot)
        join politist p on (p.id_politist = a.id_politist)
        where p.id_politist = v_politisti(i).id;
    end loop;

    for i in v_politisti.first..v_politisti.last loop
        dbms_output.put_line(v_politisti(i).nume || ' ' ||
                             v_politisti(i).prenume || ', ' ||
                             v_politisti(i).varsta ||
                             ' ani, angajat la data de ' ||
                             v_politisti(i).data_angajarii);

        if v_hoti(i).count = 0 then
            dbms_output.put_line('Nu exista niciun hot arestat de
aceasta persoana.');
        else
            dbms_output.put_line('Hotii arestati:');

            for j in v_hoti(i).first..v_hoti(i).last loop
                dbms_output.put_line(v_hoti(i)(j));
            end loop;
        end if;

        dbms_output.new_line;
    end loop;

    forall i in v_politisti.first..v_politisti.last
        update politist
        set data_angajarii = next_day(data_angajarii, 'Monday')
        where id_politist = v_politisti(i).id
        and (to_char(data_angajarii, 'Day') like 'Saturday%'
        or to_char(data_angajarii, 'Day') like 'Sunday%');
    end politisti_peste45;
/

--apelare
begin
    politisti_peste45;
end;
/

```

SQL Worksheet | History

Worksheet Query Builder

```
end politisti_pest45;
/
--apelare
begin
    politisti_pest45;
end;
```

Script Output x | Task completed in 0.031 seconds

```
Procedure POLITISTI_PESTE45 compiled

PL/SQL procedure successfully completed.
```

Dbsm Output

+ Buffer Size: 20000 |

DB final project 1st year x

```
Gimenez Alejandro, 47 ani, angajat la data de 14-JAN-92
Hotii arestati:
Osborne
Armando

Herrmann Loreley, 75 ani, angajat la data de 25-FEB-64
Hotii arestati:
Armando

Piette Janne, 63 ani, angajat la data de 01-JUN-76
Hotii arestati:
Armando

Iwamoto Nishikawa, 48 ani, angajat la data de 12-FEB-91
Hotii arestati:
Derichs
Jerome
```



7

Formulați în limbaj natural o problemă pe care să o rezolvați folosind un subprogram stocat care să utilizeze un tip de cursor studiat. Apelați subprogramul.

Acest exercițiu se află și în fișierul text 231_Voiculescu_Alină_sursa.txt.

Implementarea și rularea codului:

Exercițiu

Să se afișeze, pentru fiecare bancă, jafurile care au avut loc de-a lungul timpului, alături de numele și prenumele martorilor care au asistat la acestea.

Cod SQL:

```
create or replace procedure banca_jafuri_martori as
  type refcursor is ref cursor;
  cursor c_banca is select id_banca, denumire_banca,
    cursor (select id_jaf
      from jaf j
      where j.id_banca = b.id_banca)
    from banca b;

  cursor c_martori(v_id_jaf jaf.id_jaf%type) is select nume_martor,
    prenume_martor
    from martor_jaf mj,
      martor m
    where mj.id_martor =
      m.id_martor
    and mj.id_jaf =
      v_id_jaf;

  v_id_banca banca.id_banca%type;
  v_denumire_banca banca.denumire_banca%type;
```

```

v_cursor refcursor;
v_id_jaf jaf.id_jaf%type;
begin
    open c_banca;

loop
    fetch c_banca into v_id_banca, v_denumire_banca, v_cursor;
    exit when c_banca%notfound;

    dbms_output.put_line('Banca: ' || v_denumire_banca);

    loop
        fetch v_cursor into v_id_jaf;
        exit when v_cursor%notfound;

        if v_cursor%notfound and v_cursor%rowcount = 0 then
            dbms_output.put_line('--Pentru aceasta banca nu sunt
inregistrare jafuri.');
        end if;

        dbms_output.put_line('--Jaf: ' || v_id_jaf);

        dbms_output.put_line('----Martori: ');
        for i in c_martori(v_id_jaf) loop
            dbms_output.put_line('-----' || i.nume_martor || ' '
                                || i.prenume_martor);
        end loop;

    end loop;

    dbms_output.new_line;
    dbms_output.new_line;

end loop;

close c_banca;
end banca_jafuri_martori;
/
--apelare
begin
    banca_jafuri_martori;
end;
/

```

SQL Worksheet | History

Worksheet | Query Builder

```
banca_jafuri_martori;
```

Script Output | Task completed in 0.034 seconds

```
Procedure BANCA_JAFURI_MARTORI compiled

PL/SQL procedure successfully completed.
```

Dbms Output

DB final project 1st year ×

```
Banca: Caixabank
--Jaf: 803
----Martori:
-----Norman Valerie

Banca: BCR
--Jaf: 804
----Martori:
-----Norman Valerie
-----Wu Viktoria

Banca: Commerzbank
--Jaf: 802
----Martori:
-----Mendez Andrea
-----Mendez Erik
--Jaf: 806
----Martori:
-----Wu Viktoria
-----Mendez Erik

Banca: Mizuho Bank
--Jaf: 800
----Martori:
-----Snow Clarence

Banca: Banque CPH
--Jaf: 801
----Martori:
-----Snow Clarence
--Jaf: 805
----Martori:
-----Wu Viktoria
```

8

Formulați în limbaj natural o problemă pe care să o rezolvați folosind un subprogram stocat de tip funcție care să utilizeze într-o singură comandă SQL 3 dintre tabelele definite. Tratați toate excepțiile care pot apărea. Apelați subprogramul astfel încât să evidențiați toate cazurile tratate.

Acest exercițiu se află și în fișierul text 231_Voiculescu_Alina_sursa.txt.

Implementarea și rularea codului:

Exercițiu

Să se creeze o funcție care returnează vârsta primului răufăcător prins de polițistul primit ca parametru (numele și prenumele polițistului). Tratați toate cazurile posibile.

Cod SQL:

```
create or replace function varsta_prim_hot
    (v_nume_politist politist.nume_politist%type,
     v_prenume_politist politist.prenume_politist%type)
return varchar2 is
    v_varsta_hot number;
    exista_politist number;

    no_cop_found exception;
    pragma exception_init(no_cop_found, -20002);
begin
    select count(*)
    into exista_politist
    from politist
    where nume_politist = v_nume_politist
        and prenume_politist = v_prenume_politist;
    if exista_politist = 0 then
        raise no_cop_found;
    end if;
    select vîrstă
    into v_varsta_hot
    from politist
    where nume_politist = v_nume_politist
        and prenume_politist = v_prenume_politist
        and id_răufăcător = (select id_răufăcător
                               from răufăcător
                               where id_politist = exista_politist);
end;
```

```

from politist
where nume_politist = v_nume_politist
and prenume_politist = v_prenume_politist;

if exista_politist = 0 then
    raise_application_error(-20002, 'Nu exista niciun politist cu
                                    acest nume.');
end if;

select varsta_hot
into v_varsta_hot
from politist p
join arestare_hot a on (a.id_politist = p.id_politist)
join hot_de_banci h on (a.id_hot = h.id_hot)
where h.id_hot = (select id_hot
                    from arestare_hot
                    where data_arestarii =
                        (select DataArest
                            from (select id_politist,
                                         min(data_arestarii) as DataArest
                                from arestare_hot
                                group by id_politist)
                           where id_politist = p.id_politist))
and nume_politist = v_nume_politist
and prenume_politist = v_prenume_politist
group by nume_politist, varsta_hot;

return 'Primul raufacator prins de politistul dat are ' ||
                   to_char(v_varsta_hot) || ' ani.';

exception
when NO_DATA_FOUND then
    return 'Nu exista niciun raufacator prins de ' ||
           v_nume_politist || ' ' || v_prenume_politist || '.';

when TOO_MANY_ROWS then
    return 'Politistul ' || v_nume_politist || ' ' ||
           v_prenume_politist || ' a arestat mai mult de un
           "prim" infractor.';

when others then
    return 'Alta eroare: ' || sqlerrm;
end varsta_prim_hot;
/

--mesaj returnat: "Primul raufacator prins de politistul dat are 45
ani."
begin
    dbms_output.put_line(varsta_prim_hot('Gimenez', 'Alejandro'));
end;
/

--mesaj returnat: "Nu exista niciun raufacator prins de Dumitrescu
Dragos."
begin
    dbms_output.put_line(varsta_prim_hot('Dumitrescu', 'Dragos'));
end;
/

```

```
--mesaj returnat: "Alta eroare: ORA-20002: Nu exista niciun politist cu
acest nume."
begin
    dbms_output.put_line(varsta_prim_hot('Nume', 'Prenume'));
end;
/

```

OK

The screenshot shows the Oracle SQL Developer interface with the following details:

- SQL Worksheet:** Contains the PL/SQL code for the function VARSTA_PRIM_HOT.
- Script Output:** Shows the successful compilation of the function and completion of the task.
- Dbms Output:** Displays the message "Primul raufacator prins de politistul dat are 45 ani."

```

SQL Worksheet | History
Worksheet | Query Builder
return 'Nu exista niciun raufacator prins de ' || v_nume_politist || ' ' || v_prenume_politist || '.';
when TOO_MANY_ROWS then
    return 'Politistul ' || v_nume_politist || ' ' || v_prenume_politist || ' a arestat mai mult de un "prim" infractor.';
when others then
    return 'Alta eroare: ' || sqlerrm;
end varsta_prim_hot;
/

--mesaj returnat: "Primul raufacator prins de politistul dat are 45 ani."
begin
    dbms_output.put_line(varsta_prim_hot('Gimenez', 'Alejandro'));
end;
/

```

Script Output x
Task completed in 0.268 seconds

Function VARSTA_PRIM_HOT compiled

PL/SQL procedure successfully completed.

Dbms Output x
Buffer Size: 20000
Oracle Academy Club project x DB final project 1st year x

Primul raufacator prins de politistul dat are 45 ani.

NO DATA FOUND

SQL Worksheet | History

Worksheet | Query Builder

```
return 'Politistul ' || v_nume_politist || ' ' || v_prenume_politist || ' a arestat mai mult de un "prim" infractor.';

when others then
    return 'Alta eroare: ' || sqlerrm;
end varsta_prim_hot;
/

--mesaj returnat: "Primul raufacator prins de politistul dat are 45 ani."
begin
    dbms_output.put_line(varsta_prim_hot('Gimenez', 'Alejandro'));
end;
/

--mesaj returnat: "Nu exista niciun raufacator prins de Dumitrescu Dragos."
begin
    dbms_output.put_line(varsta_prim_hot('Dumitrescu', 'Dragos'));
end;
/
```

Script Output | Task completed in 0.078 seconds

```
PL/SQL procedure successfully completed.
```

Dbms Output | Buffer Size: 20000

```
DB final project 1st year
Nu exista niciun raufacator prins de Dumitrescu Dragos.
```

OTHER ERROR

The screenshot shows the Oracle SQL Developer interface. In the top navigation bar, 'SQL Worksheet' is selected. Below it, the 'Worksheet' tab is active, showing a PL/SQL block:

```
--mesaj returnat: "Primul raufacator prins de politistul dat are 45 ani."  
begin  
    dbms_output.put_line(varsta_prim_hot('Gimenez', 'Alejandro'));  
end;  
/  
  
--mesaj returnat: "Nu exista niciun raufacator prins de Dumitrescu Dragos."  
begin  
    dbms_output.put_line(varsta_prim_hot('Dumitrescu', 'Dragos'));  
end;  
/  
  
--mesaj returnat: "Alta eroare: ORA-20002: Nu exista niciun politist cu acest nume."  
begin  
    dbms_output.put_line(varsta_prim_hot('Nume', 'Prenume'));  
end;  
/
```

In the 'Script Output' tab, the message 'Task completed in 0.041 seconds' is displayed, followed by 'PL/SQL procedure successfully completed.'

In the 'Dbms Output' tab, the error message 'Alta eroare: ORA-20002: Nu exista niciun politist cu acest nume.' is shown.

9

Formulați în limbaj natural o problemă pe care să o rezolvați folosind un subprogram stocat de tip procedură care să utilizeze într-o singură comandă SQL 5 dintre tabelele definite. Tratați toate excepțiile care pot apărea, incluzând excepțiile NO_DATA_FOUND și TOO_MANY_ROWS. Apelați subprogramul astfel încât să evidențiați toate cazurile tratate.

Acest exercițiu se află și în fișierul text 231_Voiculescu_Alina_sursa.txt.

Implementarea și rularea codului:

Exercițiu

Să se afișeze denumirea țărilor în care se află închisorile în care au mers/vor merge hoții a căror sentință a fost stabilită de judecătorul dat ca parametru, alături de numele fiecărui. Afipați, apoi, numele celui mai periculos hoț (cel cu cea mai mare sentință) judecat de persoana primită ca parametru. În cazul în care există mai mulți hoți cu sentință egală, va fi afișat cel mai Tânăr dintre ei. De asemenea, creșteți cu 1 vârstă tuturor hoților judecați de persoana dată ca parametru. Tratați toate excepțiile/cazurile care pot apărea, returnând un mesaj corespunzător.

Cod SQL:

```
create or replace procedure cel_mai_periculos_hot
    (v_nume_judecator judecator.nume_judecator%type,
     v_prenume_judecator judecator.prenume_judecator%type)
as
    type nume is table of varchar2(50);
    type luni is table of number;
```

```

v_numé_hoti nume := nume();
v_prenume_hoti nume := nume();
v_denumire_tari nume := nume();
v_sentinta luni := luni();

v_numé_hot_periculos nume := nume();
v_prenume_hot_periculos nume := nume();

sentinta_max sentinta.luni_de_inchisoare%type := -1;
exista_judecator number := 0;

no_judge_found exception;
pragma exception_init(no_judge_found, -20001);
begin
    select count(*)
    into exista_judecator
    from judecator
    where nume_judecator = v_numé_judecator
    and prenume_judecator = v_prenume_judecator;

    if exista_judecator = 0 then
        raise_application_error(-20001, 'Nu există niciun judecător cu
                                         acest nume.');
    end if;

    select nume_hot, prenume_hot, denumire_tara, luni_de_inchisoare
    bulk collect into v_numé_hoti, v_prenume_hoti,
                           v_denumire_tari, v_sentinta
    from tara
    join oras using (id_tara)
    join inchisoare using (id_oras)
    join sentinta using (id_inchisoare)
    join judecator_sentinta_hot using (id_sentinta)
    join judecator using (id_judecator)
    join hot_de_banci using (id_hot)
    where nume_judecator = v_numé_judecator
    and prenume_judecator = v_prenume_judecator
    order by varsta_hot;

    select max(luni_de_inchisoare)
    into sentinta_max
    from tara
    join oras using (id_tara)
    join inchisoare using (id_oras)
    join sentinta using (id_inchisoare)
    join judecator_sentinta_hot using (id_sentinta)
    join judecator using (id_judecator)
    join hot_de_banci using (id_hot)
    where nume_judecator = v_numé_judecator
    and prenume_judecator = v_prenume_judecator;

    if cardinality(v_numé_hoti) <> 0 then
        dbms_output.put_line('Judecătorul ' || v_numé_judecator ||
                           ' ' || v_prenume_judecator || ':');
    end if;

    if v_numé_hoti.exists(v_numé_hoti.first) then

```

```

        for i in v_nume_hoti.first..v_nume_hoti.last loop
            dbms_output.put_line(v_nume_hoti(i) || ' ' ||
                                v_prenume_hoti(i) || ', ' ||
                                v_denumire_tari(i));

        if v_sentinta(i) = sentinta_max then
            v_nume_hot_periculos.extend;
            v_nume_hot_periculos(v_nume_hot_periculos.last)
                := v_nume_hoti(i);

            v_prenume_hot_periculos.extend;
            v_prenume_hot_periculos(v_prenume_hot_periculos.last)
                := v_prenume_hoti(i);
        end if;
    end loop;
end if;

dbms_output.new_line;

if v_nume_hoti.exists(v_nume_hoti.first) then
    forall i in v_nume_hoti.first..v_nume_hoti.last
        update hot_de_banci
        set varsta_hot = varsta_hot + 1
        where nume_hot = v_nume_hoti(i)
        and prenume_hot = v_prenume_hoti(i);
end if;

if v_nume_hot_periculos.count = 1 then
    dbms_output.put_line('Cel mai periculos hot: ' ||
                        v_nume_hot_periculos(1) || ' ' ||
                        v_prenume_hot_periculos(1));
elsif v_nume_hot_periculos.count = 0 or sentinta_max = -1 then
    raise NO_DATA_FOUND;
else
    raise TOO_MANY_ROWS;
end if;
exception
    when NO_DATA_FOUND then
        dbms_output.put_line('Nu exista niciun hot judecat de ' ||
                            v_nume_judecator || ' ' ||
                            v_prenume_judecator || '.');
    when TOO_MANY_ROWS then
        dbms_output.put_line('Exista mai multi hoti la fel de
                            periculosi judecati de ' ||
                            v_nume_judecator || ' ' ||
                            v_prenume_judecator || '.');
        dbms_output.put_line('Cel mai periculos este, totusi: ' ||
                            v_nume_hot_periculos(1) || ' ' ||
                            v_prenume_hot_periculos(1));
    when others then
        dbms_output.put_line('Alta eroare: ' || sqlerrm);
end cel_mai_periculos_hot;
/
--apelare: 1 hot cu sentinta maxima

```

```

select nume_hot, prenume_hot, denumire_tara, luni_de_inchisoare,
varsta_hot
from tara
join oras using (id_tara)
join inchisoare using (id_oras)
join sentinta using (id_inchisoare)
join judecator_sentinta_hot using (id_sentinta)
join judecator using (id_judecator)
join hot_de_banci using (id_hot)
where nume_judecator = 'Gordon'
and prenume_judecator = 'Gilbert'
order by varsta_hot asc, luni_de_inchisoare desc;

begin
    cel_mai_periculos_hot('Gordon', 'Gilbert');
end;
/

select nume_hot, prenume_hot, denumire_tara, luni_de_inchisoare,
varsta_hot
from tara
join oras using (id_tara)
join inchisoare using (id_oras)
join sentinta using (id_inchisoare)
join judecator_sentinta_hot using (id_sentinta)
join judecator using (id_judecator)
join hot_de_banci using (id_hot)
where nume_judecator = 'Gordon'
and prenume_judecator = 'Gilbert'
order by varsta_hot asc, luni_de_inchisoare desc;

rollback;

--apelare: 2 hoti cu sentinta maxima (TOO_MANY_ROWS)
select nume_hot, prenume_hot, denumire_tara, luni_de_inchisoare,
varsta_hot
from tara
join oras using (id_tara)
join inchisoare using (id_oras)
join sentinta using (id_inchisoare)
join judecator_sentinta_hot using (id_sentinta)
join judecator using (id_judecator)
join hot_de_banci using (id_hot)
where nume_judecator = 'Takayuki'
and prenume_judecator = 'Kawakami'
order by varsta_hot asc, luni_de_inchisoare desc;

begin
    cel_mai_periculos_hot('Takayuki', 'Kawakami');
end;
/

select nume_hot, prenume_hot, denumire_tara, luni_de_inchisoare,
varsta_hot
from tara
join oras using (id_tara)
join inchisoare using (id_oras)
join sentinta using (id_inchisoare)

```

```

join judecator_sentinta_hot using (id_sentinta)
join judecator using (id_judecator)
join hot_de_banci using (id_hot)
where nume_judecator = 'Takayuki'
and prenume_judecator = 'Kawakami'
order by varsta_hot asc, luni_de_inchisoare desc;

rollback;

--apelare: niciun hot judecat (NO_DATA_FOUND)
begin
    cel_mai_periculos_hot('Lapusan', 'Livia');
end;
/
--apelare: nu exista judecatorul (OTHER ERROR)
begin
    cel_mai_periculos_hot('Nume', 'Prenume');
end;
/

```

OK

ÎNAINTE:

The screenshot shows the Oracle SQL Developer interface. In the top navigation bar, 'SQL Worksheet' is selected. Below it, there are various toolbar icons. The main area has two tabs: 'Worksheet' and 'Query Builder'. The 'Worksheet' tab contains the following PL/SQL code:

```

when others then
    dbms_output.put_line('Alta eroare: ' || sqlerrm);
end cel_mai_periculos_hot;
/

--apelare: 1 hot cu sentinta maxima
select nume_hot, prenume_hot, denumire_tara, luni_de_inchisoare, varsta_hot
from tara
join oras using (id_tara)
join inchisoare using (id_oras)
join sentinta using (id_inchisoare)
join judecator_sentinta_hot using (id_sentinta)
join judecator using (id_judecator)
join hot_de_banci using (id_hot)
where nume_judecator = 'Gordon'
and prenume_judecator = 'Gilbert'
order by varsta_hot asc, luni_de_inchisoare desc;

```

In the 'Query Result' tab, the output of the query is displayed in a table:

NUME_HOT	PRENUME_HOT	DENUMIRE_TARA	LUNI_DE_INCHISOARE	VARSTA_HOT
1 Derichs	Roelof	Spania	120	42
2 Osborne	Kizzy	Spania	120	45
3 Armando	Anika	Germania	135	54

AFIȘARE:

SQL Worksheet | History

Worksheet Query Builder

```
end cel_mai_periculos_hot;
/
--apelare: 1 hot cu sentinta maxima
select nume_hot, prenume_hot, denumire_tara, luni_de_inchisoare, varsta_hot
from tara
join oras using (id_tara)
join inchisoare using (id_oras)
join sentinta using (id_inchisoare)
join judecator_sentinta_hot using (id_sentinta)
join judecator using (id_judecator)
join hot_de_banci using (id_hot)
where nume_judecator = 'Gordon'
and prenume_judecator = 'Gilbert'
order by varsta_hot asc, luni_de_inchisoare desc;

begin
    cel_mai_periculos_hot('Gordon', 'Gilbert');
end;
/
```

Script Output x | Query Result x

Task completed in 0.034 seconds

PL/SQL procedure successfully completed.

Dbms Output x

+ | Buffer Size: 20000 |

DB final project 1st year x

Judecatorul Gordon Gilbert:
Derichs Roelof, Spania
Osborne Kizzy, Spania
Armando Anika, Germania

Cel mai periculos hot: Armando Anika

DUPĂ:

SQL Worksheet History

Worksheet Query Builder

```
order by varsta_hot asc, luni_de_inchisoare desc;

begin
    cel_mai_periculos_hot('Gordon', 'Gilbert');
end;
/

select nume_hot, prenume_hot, denumire_tara, luni_de_inchisoare, varsta_hot
from tara
join oras using (id_tara)
join inchisoare using (id_oras)
join sentinta using (id_inchisoare)
join judecator_sentinta_hot using (id_sentinta)
join judecator using (id_judecator)
join hot_de_banci using (id_hot)
where nume_judecator = 'Gordon'
and prenume_judecator = 'Gilbert'
order by varsta_hot asc, luni_de_inchisoare desc;
```

Script Output x Query Result x

SQL All Rows Fetched: 3 in 0.003 seconds

NUME_HOT	PRENUME_HOT	DENUMIRE_TARA	LUNI_DE_INCHISOARE	VARSTA_HOT
1 Derichs	Roelof	Spania	120	43
2 Osborne	Kizzy	Spania	120	46
3 Armando	Anika	Germania	135	55

TOO MANY ROWS

ÎNAINTE:

The screenshot shows a SQL worksheet interface with the following details:

SQL Worksheet History tab is selected.

Worksheet tab is selected.

The query code is:

```
rollback;

--apelare: 2 hoti cu sentinta maxima (TOO_MANY_ROWS)
select nume_hot, prenume_hot, denumire_tara, luni_de_inchisoare, varsta_hot
from tara
join oras using (id_tara)
join inchisoare using (id_oras)
join sentinta using (id_inchisoare)
join judecator_sentinta_hot using (id_sentinta)
join judecator using (id_judecator)
join hot_de_banci using (id_hot)
where nume_judecator = 'Takayuki'
and prenume_judecator = 'Kawakami'
order by varsta_hot asc, luni_de_inchisoare desc;
```

Script Output tab is selected.

Query Result tab is selected.

Message: All Rows Fetched: 4 in 0.003 seconds

	NUME_HOT	PRENUME_HOT	DENUMIRE_TARA	LUNI_DE_INCHISOARE	VARSTA_HOT
1	Spitznagel	Tania	Romania	110	19
2	Osborne	Kizzy	Belgia	200	45
3	Jerome	Veronika	Germania	135	48
4	Armando	Anika	Belgia	200	54

AFIŞARE:

The screenshot shows the Oracle SQL Developer interface with the following components visible:

- SQL Worksheet**: Contains the PL/SQL code for selecting information about hoti and executing a stored procedure.
- Script Output**: Shows the message "PL/SQL procedure successfully completed."
- Dbms Output**: Displays the results of the stored procedure execution, listing several tuples.

```
--apelare: 2 hoti cu sentinta maxima (TOO_MANY_ROWS)
select nume_hot, prenume_hot, denumire_tara, luni_de_inchisoare, varsta_hot
from tara
join oras using (id_tara)
join inchisoare using (id_oras)
join sentinta using (id_inchisoare)
join judecator_sentinta_hot using (id_sentinta)
join judecator using (id_judecator)
join hot_de_banci using (id_hot)
where nume_judecator = 'Takayuki'
and prenume_judecator = 'Kawakami'
order by varsta_hot asc, luni_de_inchisoare desc;

begin
    cel_mai_periculos_hot('Takayuki', 'Kawakami');
end;
/
```

Script Output x | Task completed in 0.034 seconds

PL/SQL procedure successfully completed.

Dbms Output x | Buffer Size: 20000

DB final project 1st year x

```
Judecatorul Takayuki Kawakami:
Spitznagel Tania, Romania
Osborne Kizzy, Belgia
Jerome Veronika, Germania
Armando Anika, Belgia

Exista mai multi hoti la fel de periculosi judecati de Takayuki Kawakami.
Cel mai periculos este, totusi: Osborne Kizzy
```

DUPĂ:

SQL Worksheet History

Worksheet Query Builder

```
order by varsta_hot asc, luni_de_inchisoare desc,
```

```
begin
    cel_mai_periculos_hot('Takayuki', 'Kawakami');
end;
/

select nume_hot, prenume_hot, denumire_tara, luni_de_inchisoare, varsta_hot
from tara
join oras using (id_tara)
join inchisoare using (id_oras)
join sentinta using (id_inchisoare)
join judecator_sentinta_hot using (id_sentinta)
join judecator using (id_judecator)
join hot_de_banci using (id_hot)
where nume_judecator = 'Takayuki'
and prenume_judecator = 'Kawakami'
order by varsta_hot asc, luni_de_inchisoare desc;
```

Script Output x Query Result x

SQL | All Rows Fetched: 4 in 0.003 seconds

	NUME_HOT	PRENUME_HOT	DENUMIRE_TARA	LUNI_DE_INCHISOARE	VARSTA_HOT
1	Spitznagel	Tania	Romania	110	20
2	Osborne	Kizzy	Belgia	200	46
3	Jerome	Veronika	Germania	135	49
4	Armando	Anika	Belgia	200	55

NO DATA FOUND

```
--apelare: niciun hot judecat (NO_DATA_FOUND)
begin
    cel_mai_periculos_hot('Lapusan', 'Livia');
end;
/

```

Script Output x | Query Result x
Task completed in 0.03 seconds

PL/SQL procedure successfully completed.

Dbms Output x
Buffer Size: 20000
DB final project 1st year x

Nu exista niciun hot judecat de Lapusan Livia.

OTHER ERROR

```
--apelare: nu exista judecatorul (OTHER ERROR)
begin
    cel_mai_periculos_hot('Nume', 'Prenume');
end;
/

```

Script Output x | Query Result x
Task completed in 0.041 seconds

PL/SQL procedure successfully completed.

Dbms Output x
Buffer Size: 20000
DB final project 1st year x

Alta eroare: ORA-20001: Nu exista niciun judecator cu acest nume.

10

Definiți un trigger de tip LMD la nivel de comandă.
Declanșați trigger-ul.

Acest exercițiu se află și în fișierul text 231_Voiculescu_Alina_sursa.txt.

Implementarea și rularea codului:

Exercițiu

Definiți un declanșator de tip LMD la nivel de comandă care permite inserarea, actualizarea și ștergerea datelor din tabela SENTINȚĂ doar în intervalul orar 6:00 – 20:59, de luni până vineri. De asemenea, să se permită ștergerea informațiilor din tabelul SENTINȚĂ doar utilizatorului ADMIN. Comenzile care s-au executat cu succes să se introducă în tabelul INFO_SENTINȚĂ, alături de user-ul care a executat comanda, data executării și denumirea comenzii.

Cod SQL:

```
create sequence seq_info
increment by 1
start with 1
nocache
nocycle;

create table info_sentinta (id int primary key,
                            utilizator varchar2(30),
                            data date,
                            comanda varchar2(100));

create or replace trigger restrictie_interval_orar_si_user
for insert or update or delete on sentinta
compound trigger
before statement is
```

```

begin
    if (to_char(sysdate, 'D') = 7) or (to_char(sysdate, 'D') = 1)
        or (to_char(sysdate, 'HH24') not between 6 and 20) then
            raise_application_error(-20003, 'Tabelul SENTINTA nu poate
                                         fi actualizat in afara
                                         orelor de program!');

    end if;

    if deleting then
        if user <> 'ADMIN' then
            raise_application_error(-20004, 'Dvs sunteti ' ||
                                      user || '. Nu aveti voie sa
                                      stergeti informatii din acest
                                      tabel. Doar ADMIN-ul are
                                      voie.');
        end if;
    end if;
end before statement;

after statement is
begin
    if inserting then
        insert into info_sentinta
        values (seq_info.nextval, sys.login_user, sysdate,
                'INSERT');
    elsif updating then
        insert into info_sentinta
        values (seq_info.nextval, sys.login_user, sysdate,
                'UPDATE');
    elsif deleting then
        insert into info_sentinta
        values (seq_info.nextval, sys.login_user, sysdate,
                'DELETE');
    end if;
end after statement;
end;
/
insert into sentinta (id_sentinta, luni_de_inchisoare, id_inchisoare)
values (480, 500, 124);

delete from sentinta
where id_sentinta = 480
and luni_de_inchisoare = 500
and id_inchisoare = 124;

select * from info_sentinta;

```

INSERARE ÎN INTERVALUL ORAR 6:00 – 20:59

The screenshot shows the Oracle SQL Developer interface. In the 'Worksheet' tab, a PL/SQL script is run:

```
insert into info_sentinta
values (seq_info.nextval, sys.login_user, sysdate, 'INSERT');
elsif updating then
    insert into info_sentinta values (seq_info.nextval, sys.login_user, sysdate, 'UPDATE');
elsif deleting then
    insert into info_sentinta values (seq_info.nextval, sys.login_user, sysdate, 'DELETE');
end if;
end after statement;
end;
/
insert into sentinta (id_sentinta, luni_de_inchisoare, id_inchisoare)
values (480, 500, 124);
```

The output shows the sequence was created, the table was created, and the trigger was compiled. One row was inserted.

```
Sequence SEQ_INFO created.

Table INFO_SENTINTA created.

Trigger RESTRICTIE_INTERVAL_ORAR_SI_USER compiled

1 row inserted.
```

Tabelul INFO_SENTINTĂ:

```
41
42 insert into sentinta (id_sentinta, luni_de_inchisoare, id_inchisoare)
43 values (480, 500, 124);
44
45 delete from sentinta
46 where id_sentinta = 480
47 and luni_de_inchisoare = 500
48 and id_inchisoare = 124;
49
50 select * from info_sentinta;
```

The screenshot shows the 'Query Result' tab with the following output:

```
All Rows Fetched: 1 in 0.01 seconds
```

ID	UTILIZATOR	DATA	COMANDA
1	BLXQN	06-JAN-22	INSERT

INSERARE ÎN AFARA INTERVALULUI ORAR 6:00 – 20:59

SQL Worksheet History

Worksheet Query Builder

```
35      elsif deleting then
36          insert into info_sentinta values (seq_info.nextval, sys.login_user, sysdate, 'DELETE');
37      end if;
38  end after statement;
39end;
/
41
42 insert into sentinta (id_sentinta, luni_de_inchisoare, id_inchisoare)
43 values (480, 500, 124);
44
45 delete from sentinta
46 where id_sentinta = 480
47 and luni_de_inchisoare = 500
48 and id_inchisoare = 124;
49
50 select * from info_sentinta;
```

Script Output | Query Result | Task completed in 0.038 seconds

```
Error starting at line : 42 in command -
insert into sentinta (id_sentinta, luni_de_inchisoare, id_inchisoare)
values (480, 500, 124)
Error report -
ORA-20003: Tabelul SENTINTA nu poate fi actualizat in afara orelor de program!
ORA-06512: at "BLXQN.RESTRICTIE_INTERVAL_ORAR_SI_USER", line 5
ORA-04088: error during execution of trigger 'BLXQN.RESTRICTIE_INTERVAL_ORAR_SI_USER'
```

STERGERE

SQL Worksheet History

Worksheet Query Builder

```
35      elsif deleting then
36          insert into info_sentinta values (seq_info.nextval, sys.login_user, sysdate, 'DELETE');
37      end if;
38  end after statement;
39end;
/
41
42 insert into sentinta (id_sentinta, luni_de_inchisoare, id_inchisoare)
43 values (480, 500, 124);
44
45 delete from sentinta
46 where id_sentinta = 480
47 and luni_de_inchisoare = 500
48 and id_inchisoare = 124;
```

Script Output | Query Result | Task completed in 0.067 seconds

```
Error starting at line : 45 in command -
delete from sentinta
where id_sentinta = 480
and luni_de_inchisoare = 500
and id_inchisoare = 124
Error report -
ORA-20004: Dvs sunteti BLXQN. Nu aveti voie sa stergeti informatii din acest tabel. Doar ADMIN-ul are voie.
ORA-06512: at "BLXQN.RESTRICTIE_INTERVAL_ORAR_SI_USER", line 10
ORA-04088: error during execution of trigger 'BLXQN.RESTRICTIE_INTERVAL_ORAR_SI_USER'
```

11

Definiți un trigger de tip LMD la nivel de linie.
Declanșați trigger-ul.

Acest exercițiu se află și în fișierul text 231_Voiculescu_Alina_sursa.txt.

Implementarea și rularea codului:

Exercițiu 1

Definiți un declanșator de tip LMD la nivel de linie care să nu permită judecarea unui hot de mai mult de două ori de aceeași persoană (același judecător).

Cod SQL:

```
create or replace trigger hot_rejudecat
before insert on judecator_sentinta_hot
for each row
declare
    v_nr_judecari number := 0;
begin
    select count(*)
    into v_nr_judecari
    from judecator_sentinta_hot
    where id_hot = :NEW.id_hot
    and id_judecator = :NEW.id_judecator
    group by id_hot, id_judecator;

    if v_nr_judecari = 2 then
        raise_application_error(-20005, 'Hotul trebuie rejudecat de
                                   alta persoana. Un judecator
                                   poate da sentinte de maxim 2
                                   ori unui anumit hot!');
    end if;
end;
/
```

```
--La momentul actual, hotul cu id-ul 405 a fost judecat o singura data
de judecatorul cu id-ul 100
--Dovada:
select count(*) "Numar judecari"
from judecator_sentinta_hot
where id_hot = 405
and id_judecator = 100
group by id_hot, id_judecator;

--a doua judecare: OK!
insert into judecator_sentinta_hot
values (100, 478, 405, sysdate);

--a treia judecare: EROARE!
insert into judecator_sentinta_hot
values (100, 479, 405, sysdate);
```

La momentul actual, hoțul cu id-ul 405 a fost judecat o singură dată de judecătorul cu id-ul 100.

The screenshot shows a SQL Worksheet window with the following details:

- Toolbar:** Includes icons for New, Open, Save, Print, Copy, Paste, Find, Replace, and Font Size.
- Worksheet Tab:** Labeled "Worksheet" and "Query Builder".
- Script Area:** Displays the following PL/SQL code:


```
11 and id_judecator = .NEW.id_judecator
12 group by id_hot, id_judecator;
13
14 if v_nr_judecari = 2 then
15   raise_application_error(-20005, 'Hotul trebuie rejudicat de alta persoana. Un judecator poate da sentinte de maxim 2 ori unui acuzat!');
16 end if;
17 end;
18 /
19
20 --La momentul actual, hotul cu id-ul 405 a fost judecat o singura data de judecatorul cu id-ul 100
21 --Dovada:
22 select count(*) "Numar judecari"
23 from judecator_sentinta_hot
24 where id_hot = 405
25 and id_judecator = 100
26 group by id_hot, id_judecator;
```
- Output Area:** Labeled "Script Output" and "Query Result".
 - Script Output:** Shows the PL/SQL code above.
 - Query Result:** Shows the output of the query:

Numar judecari
1

All Rows Fetched: 1 in 0.006 seconds

A doua judecare: OK!

SQL Worksheet | History

Worksheet | Query Builder

```
15    raise_application_error(-20005, 'NOTUL trebuie rejudicat de alta persoana. Un judecator poate
16    end if;
17 end;
18 /
19
20 --La momentul actual, hotul cu id-ul 405 a fost judecat o singura data de judecatorul cu id-ul 100
21 --Dovada:
22 select count(*) "Numar judecari"
23 from judecator_sentinta_hot
24 where id_hot = 405
25 and id_judecator = 100
26 group by id_hot, id_judecator;
27
28 --a doua judecare: OK!
29 insert into judecator_sentinta_hot
30 values (100, 478, 405, sysdate);
```

Script Output | Query Result | Task completed in 0.034 seconds

Trigger HOT_REJUDECAT compiled

1 row inserted.

A treia judecare: EROARE!

SQL Worksheet History

Worksheet Query Builder

```
19
20 --La momentul actual, hotul cu id-ul 405 a fost judecat o singura data de judecatorul cu id-ul 100
21 --Dovada:
22 select count(*) "Numar judecari"
23 from judecator_sentinta_hot
24 where id_hot = 405
25 and id_judecator = 100
26 group by id_hot, id_judecator;
27
28 --a doua judecare: OK!
29 insert into judecator_sentinta_hot
30 values (100, 478, 405, sysdate);
31
32 --a treia judecare: EROARE!
33 insert into judecator_sentinta_hot
34 values (100, 479, 405, sysdate);
```

Script Output x | Query Result x
Task completed in 0.037 seconds

Error starting at line : 33 in command -
insert into judecator_sentinta_hot
values (100, 479, 405, sysdate)
Error report -
ORA-20005: Hotul trebuie rejudicat de alta persoana. Un judecator poate da sentinte de maxim 2 ori unui acuzat hot!
ORA-06512: at "BLXQN.HOT_REJUDECAT", line 12
ORA-04088: error during execution of trigger 'BLXQN.HOT_REJUDECAT'

Exercițiul 2

Definiți un declanșator de tip LMD la nivel de linie care să nu permită micșorarea vârstei clienților, și nici mărirea vârstei cu mai mult de un an. De asemenea, să nu se poată să se introducă un client cu o vârstă mai mică de 18 ani.

Cod SQL:

```
create or replace trigger restrictii_clienti
before insert or update of varsta_client on client
for each row
when (NEW.varsta_client < 18 or
      NEW.varsta_client < nvl(OLD.varsta_client,NEW.varsta_client - 1)
      or
      NEW.varsta_client > nvl(OLD.varsta_client,NEW.varsta_client - 1)
                  + 1)
declare
    v_varsta number;
begin
    raise_application_error(-20005, 'Varsta clientului nu poate avea
                                   acea valoare!');
end;
/

--inserare client cu varsta de 10 ani: EROARE
insert into client
values (305, 500, 'Nume', 'Prenume', 10);

--marire varsta client cu mai mult de 1 an: EROARE
update client
set varsta_client = varsta_client + 3
where id_client = 304;

--micsorare varsta client: EROARE
update client
set varsta_client = varsta_client - 1
where id_client = 304;
```

INSERARE CLIENT CU VÂRSTA DE 10 ANI: EROARE

```
1 create or replace trigger restrictii_clienti
2 before insert or update of varsta_client on client
3 for each row
4 when (NEW.varsta_client < 18 or
5     NEW.varsta_client < nvl(OLD.varsta_client,NEW.varsta_client - 1) or
6     NEW.varsta_client > nvl(OLD.varsta_client,NEW.varsta_client - 1) + 1)
7 declare
8     v_varsta number;
9 begin
10    raise_application_error(-20005, 'Varsta clientului nu poate avea aceea valoare!');
11 end;
12 /
13
14 --inserare client cu varsta de 10 ani: EROARE
15 insert into client
16 values (305, 500, 'Nume', 'Prenume', 10);
```

Script Output x Query Result x
Task completed in 0.044 seconds

Trigger RESTRICTII_CLIENTI compiled

Error starting at line : 15 in command -
insert into client
values (305, 500, 'Nume', 'Prenume', 10)
Error report -
ORA-20005: Varsta clientului nu poate avea aceea valoare!
ORA-06512: at "BLXQN.RESTRICTII_CLIENTI", line 4
ORA-04088: error during execution of trigger 'BLXQN.RESTRICTII_CLIENTI'

MĂRIRE VÂRSTĂ CLIENT CU MAI MULT DE 1 AN: EROARE

SQL Worksheet History

Worksheet Query Builder

```
6      NEW.varsta_client > nvl(OLD.varsta_client,NEW.varsta_client - 1) + 1)
7 declare
8     v_varsta number;
9 begin
10    raise_application_error(-20005, 'Varsta clientului nu poate avea acea valoare!');
11 end;
12 /
13
14 --inserare client cu varsta de 10 ani: EROARE
15 insert into client
16 values (305, 500, 'Nume', 'Prenume', 10);
17
18 --marire varsta client cu mai mult de 1 an: EROARE
19 update client
20 set varsta_client = varsta_client + 3
21 where id_client = 304;
```

Script Output x | Query Result x
Task completed in 0.041 seconds

```
Error starting at line : 19 in command -
update client
set varsta_client = varsta_client + 3
where id_client = 304
Error report -
ORA-20005: Varsta clientului nu poate avea acea valoare!
ORA-06512: at "BLXQN.RESTRICTII_CLIENTI", line 4
ORA-04088: error during execution of trigger 'BLXQN.RESTRICTII_CLIENTI'
```

MICȘORARE VÂRSTĂ CLIENT: EROARE

The screenshot shows a SQL Worksheet window in Oracle SQL Developer. The code attempts to insert a new client record and then update an existing one to decrease their age by 1 year. An error occurs at the update step due to a trigger constraint.

```
SQL Worksheet History
Worksheet Query Builder

11 end;
12 /
13
14 --inserare client cu varsta de 10 ani: EROARE
15 insert into client
16 values (305, 500, 'Nume', 'Prenume', 10);
17
18 --marire varsta client cu mai mult de 1 an: EROARE
19 update client
20 set varsta_client = varsta_client + 3
21 where id_client = 304;
22
23 --micsorare varsta client: EROARE
24 update client
25 set varsta_client = varsta_client - 1
26 where id_client = 304;
```

Script Output x Query Result x
Task completed in 0.035 seconds

```
Error starting at line : 24 in command -
update client
set varsta_client = varsta_client - 1
where id_client = 304
Error report -
ORA-20005: Varsta clientului nu poate avea acea valoare!
ORA-06512: at "BLXQN.RESTRICTII_CLIENTI", line 4
ORA-04088: error during execution of trigger 'BLXQN.RESTRICTII_CLIENTI'
```

12

Definiți un trigger de tip LDD.
Declanșați trigger-ul.

Acest exercițiu se află și în fișierul text 231_Voiculescu_Alina_sursa.txt.

Implementarea și rularea codului:

Exercițiu

Să se definească un declanșator care are rolul de a bloca orice eventuală modificare asupra bazei de date. Se va defini un tabel de tip audit numit INFO_INCERCARI, care va conține toate încercările de modificare a bazei de date (user-ul, evenimentul încercat, data încercării și numele obiectului).

Cod SQL:

```
create sequence seq_info
increment by 1
start with 1
nocache
nocycle;

create table info_incercari(id int primary key,
                           utilizator varchar2(30),
                           eveniment varchar2(20),
                           nume_object varchar2(100),
                           data date);
/

create or replace trigger fara_alte_modificari
after create or drop or alter on database
declare
    pragma autonomous_transaction;
begin
    insert into info_incercari
    values (seq_info.nextval, sys.login_user, sys.sysevent,
            sys.dictionary_obj_name, sysdate);
```

```

        commit;

        raise_application_error(-20006, 'Baza de date este deja in forma
                                    finala, nemainputand fi editata.');

end;
/
drop table martor cascade constraints;
select * from info_incercari;

```

SQL Worksheet History

Worksheet Query Builder

```

10      nume_object varchar2(100),
11      data date);
12 /
13
14 create or replace trigger fara_alte_modificari
15 after create or drop or alter on database
16 declare
17     pragma autonomous_transaction;
18 begin
19     insert into info_incercari
20     values (seq_info.nextval, sys.login_user, sys.sysevent, sys.dictionary_obj_name, sysdate);
21
22     commit;
23
24     raise_application_error(-20006, 'Baza de date este deja in forma finala, nemainputand fi editata.');
25 end;
26 /
27
28 drop table martor cascade constraints;

```

Script Output x | Query Result x

| Task completed in 0.039 seconds

Sequence SEQ_INFO created.

Table INFO_INCERCARI created.

Trigger FARA_ALTE_MODIFICARI compiled

DROP TABLE

```
SQL Worksheet History
Worksheet Query Builder
18 begin
19   insert into info_incercari
20   values (seq_info.nextval, sys.login_user, sys.sysevent, sys.dictionary_obj_name, sysdate);
21
22   commit;
23
24   raise_application_error(-20006, 'Baza de date este deja in forma finala, nemainputand fi editata.');
25 end;
26 /
27
28 drop table martor cascade constraints;
```

Script Output | Query Result | Task completed in 0.038 seconds

Error starting at line : 28 in command -
drop table martor cascade constraints
Error report -
ORA-00604: error occurred at recursive SQL level 1
ORA-04088: error during execution of trigger 'BLXQN.FARA_ALTE_MODIFICARI'
ORA-00604: error occurred at recursive SQL level 2
ORA-20006: Baza de date este deja in forma finala, nemainputand fi editata.
ORA-06512: at line 9
00604. 00000 - "error occurred at recursive SQL level %s"
*Cause: An error occurred while processing a recursive SQL statement
(a statement applying to internal dictionary tables).
*Action: If the situation described in the next error on the stack
can be corrected, do so; otherwise contact Oracle Support.

Tabelul INFO_ÎNCERCĂRI:

```
SQL Worksheet History
Worksheet Query Builder
19 insert into info_incercari
20   values (seq_info.nextval, sys.login_user, sys.sysevent, sys.dictionary_obj_name, sysdate);
21
22   commit;
23
24   raise_application_error(-20006, 'Baza de date este deja in forma finala, nemainputand fi editata.');
25 end;
26 /
27
28 drop table martor cascade constraints;
29
30 select * from info_incercari;
```

Script Output | Query Result | All Rows Fetched: 1 in 0.005 seconds

ID	UTILIZATOR	EVENIMENT	NUME_OBJECT	DATA
1	BLXQN	ALTER	MARTOR	06-JAN-22

13

Definiți un pachet care să conțină toate obiectele definite în cadrul proiectului.

Acest exercițiu se află și în fișierul text 231_Voiculescu_Alina_sursa.txt.

Implementarea și rularea codului:

Cod SQL:

```
create or replace package pkg as
procedure politisti_pest45;

procedure banca_jafuri_martori;

function varsta_prim_hot
    (v_nume_politist politist.nume_politist%type,
     v_prenume_politist politist.prenume_politist%type)
return varchar2;

procedure cel_mai_periculos_hot
    (v_nume_judecator judecator.nume_judecator%type,
     v_prenume_judecator judecator.prenume_judecator%type);
end pkg;
/

create or replace package body pkg as
procedure politisti_pest45 as
    type hoti is varray(10) of varchar2(30);
    type despre_politisti is record
        (id politist.id_politist%type,
         nume politist.nume_politist%type,
         prenume politist.prenume_politist%type,
         varsta politist.varsta_politist%type,
         data_angajarii politist.data_angajarii%type);

    type hoti_prinsi is table of hoti index by pls_integer;
    type politisti is table of despre_politisti index by
```

```

        pls_integer;

v_hoti hoti_prinsi;
v_politisti politisti;

begin
    select id_politist, nume_politist, prenume_politist,
           varsta_politist, data_angajarii
      bulk collect into v_politisti
     from politist
    where varsta_politist > 45;

    for i in v_politisti.first..v_politisti.last loop
        select nume_hot "Nume hot"
          bulk collect into v_hoti(i)
         from hot_de_banci h
        join arestare_hot a on (a.id_hot = h.id_hot)
        join politist p on (p.id_politist = a.id_politist)
       where p.id_politist = v_politisti(i).id;
    end loop;

    for i in v_politisti.first..v_politisti.last loop
        dbms_output.put_line(v_politisti(i).nume || ' ' ||
                             v_politisti(i).prenume || ', ' ||
                             v_politisti(i).varsta || ' ani,
                                         angajat la data de ' ||
                             v_politisti(i).data_angajarii);

        if v_hoti(i).count = 0 then
            dbms_output.put_line('Nu exista niciun hot arestat de
                                 aceasta persoana.');
        else
            dbms_output.put_line('Hotii arestati:');
            for j in v_hoti(i).first..v_hoti(i).last loop
                dbms_output.put_line(v_hoti(i)(j));
            end loop;
        end if;

        dbms_output.new_line;
    end loop;

    forall i in v_politisti.first..v_politisti.last
        update politist
        set data_angajarii = next_day(data_angajarii, 'Monday')
       where id_politist = v_politisti(i).id
         and (to_char(data_angajarii, 'Day') like 'Saturday%'
              or to_char(data_angajarii, 'Day') like 'Sunday%');
end politisti_peste45;
-----
```

```

procedure banca_jafuri_martori as
    type refcursor is ref cursor;
    cursor c_banca is select id_banca, denumire_banca,
```

```

cursor (select id_jaf
        from jaf j
        where j.id_banca = b.id_banca)
        from banca b;

cursor c_martori(v_id_jaf jaf.id_jaf%type) is
    select nume_martor, prenume_martor
    from martor_jaf mj, martor m
    where mj.id_martor = m.id_martor
    and mj.id_jaf = v_id_jaf;
v_id_banca banca.id_banca%type;
v_denumire_banca banca.denumire_banca%type;
v_cursor refcursor;
v_id_jaf jaf.id_jaf%type;
begin
open c_banca;

loop
fetch c_banca into v_id_banca, v_denumire_banca, v_cursor;
exit when c_banca%notfound;

dbms_output.put_line('Banca: ' || v_denumire_banca);

loop
fetch v_cursor into v_id_jaf;
exit when v_cursor%notfound;

if v_cursor%notfound and v_cursor%rowcount = 0 then
    dbms_output.put_line('--Pentru aceasta banca nu
                           sunt inregistrare jafuri.');
end if;

dbms_output.put_line('--Jaf: ' || v_id_jaf);

dbms_output.put_line('----Martori: ');
for i in c_martori(v_id_jaf) loop
    dbms_output.put_line('-----' || i.nume_martor ||
                         ' ' || i.prenume_martor);
end loop;

end loop;

dbms_output.new_line;
dbms_output.new_line;

end loop;

close c_banca;
end banca_jafuri_martori;

```

```

function varsta_prim_hot
    (v_nume_politist politist.nume_politist%type,
     v_prenume_politist politist.prenume_politist%type)

```

```

return varchar2 is
    v_varsta_hot number;
    exista_politist number;

    no_cop_found exception;
    pragma exception_init(no_cop_found, -20002);
begin
    select count(*)
    into exista_politist
    from politist
    where nume_politist = v_numef_politist
    and prenume_politist = v_prenume_politist;

    if exista_politist = 0 then
        raise_application_error(-20002, 'Nu exista niciun politist
                                         cu acest nume.');
    end if;

    select varsta_hot
    into v_varsta_hot
    from politist p
    join arestare_hot a on (a.id_politist = p.id_politist)
    join hot_de_banci h on (a.id_hot = h.id_hot)
    where h.id_hot =
        (select id_hot
         from arestare_hot
         where data_arestarii = (select DataArest
                                   from (select id_politist,
                                              min(data_arestarii) as DataArest
                                   from arestare_hot
                                   group by id_politist)
                                   where id_politist = p.id_politist))
    and nume_politist = v_numef_politist
    and prenume_politist = v_prenume_politist
    group by nume_politist, varsta_hot;

    return 'Primul raufacator prins de politistul dat are ' ||
           to_char(v_varsta_hot) || ' ani.';
exception
    when NO_DATA_FOUND then
        return 'Nu exista niciun raufacator prins de ' ||
               v_numef_politist || ' ' || v_prenume_politist || '.';
    when TOO_MANY_ROWS then
        return 'Politistul ' || v_numef_politist || ' ' ||
               v_prenume_politist || ' a arestat mai mult de un
"prim" infractor.';

    when others then
        return 'Alta eroare: ' || sqlerrm;
end varsta_prim_hot;

```

```

procedure cel_mai_periculos_hot
    (v_numef_judecator judecator.nume_judecator%type,
     v_prenume_judecator judecator.prenume_judecator%type)
as
```

```

type nume is table of varchar2(50);
type luni is table of number;

v_nume_hoti nume := nume();
v_prenume_hoti nume := nume();
v_denumire_tari nume := nume();
v_sentinta_luni := luni();

v_nume_hot_periculos nume := nume();
v_prenume_hot_periculos nume := nume();

sentinta_max sentinta.luni_de_inchisoare%type := -1;
exista_judecator number := 0;

no_judge_found exception;
pragma exception_init(no_judge_found, -20001);

begin
    select count(*)
    into exista_judecator
    from judecator
    where nume_judecator = v_nume_judecator
    and prenume_judecator = v_prenume_judecator;

    if exista_judecator = 0 then
        raise_application_error(-20001, 'Nu exista niciun judecator
                                         cu acest nume.');
    end if;

    select nume_hot, prenume_hot, denumire_tara, luni_de_inchisoare
    bulk collect into v_nume_hoti, v_prenume_hoti,
                           v_denumire_tari, v_sentinta
    from tara
    join oras using (id_tara)
    join inchisoare using (id_oras)
    join sentinta using (id_inchisoare)
    join judecator_sentinta_hot using (id_sentinta)
    join judecator using (id_judecator)
    join hot_de_banci using (id_hot)
    where nume_judecator = v_nume_judecator
    and prenume_judecator = v_prenume_judecator
    order by varsta_hot;

    select max(luni_de_inchisoare)
    into sentinta_max
    from tara
    join oras using (id_tara)
    join inchisoare using (id_oras)
    join sentinta using (id_inchisoare)
    join judecator_sentinta_hot using (id_sentinta)
    join judecator using (id_judecator)
    join hot_de_banci using (id_hot)
    where nume_judecator = v_nume_judecator
    and prenume_judecator = v_prenume_judecator;

    if cardinality(v_nume_hoti) <> 0 then

        dbms_output.put_line('Judecatorul ' || v_nume_judecator ||
                           ' ' || v_prenume_judecator || ':');
    end if;
end;

```

```

end if;

if v_nume_hoti.exists(v_nume_hoti.first) then
    for i in v_nume_hoti.first..v_nume_hoti.last loop
        dbms_output.put_line(v_nume_hoti(i) || ' ' ||
                             v_prenume_hoti(i) || ', ' ||
                             v_denumire_tari(i));

        if v_sentinta(i) = sentinta_max then
            v_nume_hot_periculos.extend;
            v_nume_hot_periculos(v_nume_hot_periculos.last)
                := v_nume_hoti(i);

            v_prenume_hot_periculos.extend;
            v_prenume_hot_periculos(v_prenume_hot_periculos.last)
                := v_prenume_hoti(i);
        end if;
    end loop;
end if;

dbms_output.new_line;

if v_nume_hoti.exists(v_nume_hoti.first) then
    forall i in v_nume_hoti.first..v_nume_hoti.last
        update hot_de_banci
        set varsta_hot = varsta_hot + 1
        where nume_hot = v_nume_hoti(i)
              and prenume_hot = v_prenume_hoti(i);
end if;

if v_nume_hot_periculos.count = 1 then
    dbms_output.put_line('Cel mai periculos hot: ' ||
                         v_nume_hot_periculos(1) || ' ' ||
                         v_prenume_hot_periculos(1));
elsif v_nume_hot_periculos.count = 0 or sentinta_max = -1 then
    raise NO_DATA_FOUND;
else
    raise TOO_MANY_ROWS;
end if;
exception
when NO_DATA_FOUND then
    dbms_output.put_line('Nu exista niciun hot judecat de ' ||
                         v_nume_judecator || ' ' ||
                         v_prenume_judecator || '.');
when TOO_MANY_ROWS then
    dbms_output.put_line('Există mai multi hoti la fel de
                           pericolosi judecati de ' ||
                           v_nume_judecator || ' ' ||
                           v_prenume_judecator || '.');
    dbms_output.put_line('Cel mai periculos este, totusi: ' ||
                         v_nume_hot_periculos(1) || ' ' ||
                         v_prenume_hot_periculos(1));

```

```

        when others then
            dbms_output.put_line('Alta eroare: ' || sqlerrm);
    end cel_mai_periculos_hot;
end pkg;
/

--apelarea in diverse moduri (SQL si PL/SQL)
exec pkg.politisti_pest45;

execute pkg.banca_jafuri_martori;

select pkg.varsta_prim_hot('Gimenez', 'Alejandro') "Varsta primului hot
prins de politistul dat"
from dual;

begin
    pkg.cel_mai_periculos_hot('Takayuki', 'Kawakami');
end;
/

```

The screenshot shows the Oracle SQL Developer interface. The top window is titled 'SQL Worksheet' and contains the PL/SQL code from the previous snippet. The 'Script Output' tab at the bottom of the worksheet shows the message 'PL/SQL procedure successfully completed.' Below this is the 'Dbms Output' tab, which displays the results of the procedure execution. The output is organized into sections based on bank names:

- Banca: Caixabank**
 - Jaf: 803
 - Martori:
 - Norman Valerie
- Banca: BCR**
 - Jaf: 804
 - Martori:
 - Norman Valerie
 - Wu Viktoria
- Banca: Commerzbank**
 - Jaf: 802
 - Martori:
 - Mendez Andrea
 - Mendez Erik
 - Jaf: 806
 - Martori:
 - Wu Viktoria
 - Mendez Erik

SQL Worksheet | History

Worksheet | Query Builder

```
267      when others then
268          dbms_output.put_line('Alta eroare: ' || sqlerrm);
269      end cel_mai_periculos_hot;
270  end pkg;
271 /
272
273 --apelarea in diverse moduri (SQL si PL/SQL)
274 exec pkg.politisti_peste45;
```

Script Output | Query Result

Task completed in 0.041 seconds

PL/SQL procedure successfully completed.

Dbms Output | Buffer Size: 20000

DB final project 1st year

```
Gimenez Alejandro, 47 ani, angajat la data de 14-JAN-92
Hotii arestati:
Osborne
Armando
```

```
Herrmann Loreley, 75 ani, angajat la data de 25-FEB-64
Hotii arestati:
Armando
```

```
Piette Janne, 63 ani, angajat la data de 01-JUN-76
Hotii arestati:
Armando
```

```
Iwamoto Nishikawa, 48 ani, angajat la data de 12-FEB-91
Hotii arestati:
Derichs
Jerome
```

SQL Worksheet History

Worksheet Query Builder

```
279 | select pkg.varsta_prim_hot('Gimenez', 'Alejandro') "Varsta primului hot prins de politistul dat"
280 | from dual;
```

Script Output x Query Result x

SQL | All Rows Fetched: 1 in 0.008 seconds

Varsta primului hot prins de politistul dat

1 Primul raufacator prins de politistul dat are 46 ani.

SQL Worksheet History

Worksheet Query Builder

```
269 |     when others then
270 |         dbms_output.put_line('Alta eroare: ' || sqlerrm);
271 |     end cel_mai_periculos_hot;
272 | end pkg;
273 |
274 --apelarea in diverse moduri (SQL si PL/SQL)
275 exec pkg.politisti_peste45;
276
277 execute pkg.banca_jafuri_martori;
278
279 select pkg.varsta_prim_hot('Gimenez', 'Alejandro') "Varsta primului hot prins de politistul dat"
280 from dual;
281
282 begin
283     pkg.cel_mai_periculos_hot('Takayuki', 'Kawakami');
284 end;
285 /
```

Script Output x Query Result x

Task completed in 0.039 seconds

PL/SQL procedure successfully completed.

Dbms Output x

Buffer Size: 20000

DB final project 1st year x

Judecatorul Takayuki Kawakami:
 Spitznagel Tania, Romania
 Osborne Kizzy, Belgia
 Jerome Veronika, Germania
 Armando Anika, Belgia

Există mai mulți hoti la fel de periculoși judecati de Takayuki Kawakami.
 Cel mai periculos este, totusi: Osborne Kizzy

14

Definiți un pachet care să conțină toate obiectele definite în cadrul proiectului.

Acest exercițiu se află și în fișierul text 231_Voiculescu_Alina_sursa.txt.

Implementarea și rularea codului:

Să se definească un pachet (denumit UTILITIES) prin care să se poată realiza următoarele acțiuni:

a) Adăugarea unui jaf

- denumire bancă*
- data ➔ default ziua inserării*

b) Adăugarea unui hoț la un anumit jaf

- nume*
- prenume*
- denumire bancă*
- data jafului ➔ default ziua inserării*

c) Adăugarea unui martor la un anumit jaf

- nume*
- prenume*
- denumire bancă*
- data jafului ➔ default ziua inserării*

d) Arrestarea unui hoț

- nume hoț*
- prenume hoț*
- nume polițist*
- prenume polițist*
- data arestării ➔ default ziua inserării*

e) Adăugarea unei sentințe

- nume hoț
- prenume hoț
- nume judecător
- prenume judecător
- timp de închisoare ➔ în luni
- denumire închisoare
- data acordării ➔ default ziua inserării

f) Verificarea existenței unui anumit client la o anumită bancă

- nume
- prenume
- denumire bancă

g) Verificarea existenței a cel puțin unui martor la un anumit jaf

- denumire bancă
- data jafului ➔ default ziua curentă

h) Verificarea existenței unui anumit martor la un anumit jaf

- nume
- prenume
- denumire bancă
- data jafului ➔ default ziua curentă

i) Afisarea orașului și a țării unde se află o anumită bancă

- denumire bancă

j) Afisarea orașului și a țării unde se află o anumită închisoare

- denumire închisoare

k) Afisarea sentințelor (lunilor de închisoare) săvârșite de / pe care le va săvârși un anumit hoț

- nume
- prenume

l) Afisarea hoților care nu sunt încă prinși

m) Afisarea secției la care lucrează un anumit polițist

- nume
- prenume

n) Afisarea polițiștilor care lucrează la o anumită secție

- denumire secție

o) Afisarea tuturor tribunalelor în care a fost judecat un anumit hoț

- nume
- prenume

p) Afisarea tuturor polițiștilor care au prins, de-a lungul timpului, un anumit hoț

- nume
- prenume

q) Afisarea tuturor judecătorilor care au judecat, de-a lungul timpului, un anumit hoț

- nume
- prenume

r) Afisarea numărului de jafuri la care a participat un anumit hoț

- nume
- prenume

s) Afişarea tuturor băncilor pe care le-a jefuit un anumit hoț

- nume
- prenume

t) Afişarea tuturor orașelor și țărilor în care există bănci jefuite de un anumit hoț

- nume
- prenume

u) Afişarea detaliilor martorilor la un anumit jaf

- denumire bancă
- data jafului → default ziua curentă

v) Afişarea detaliilor clienților unei anumite bănci

- denumire bancă

Cod SQL:

```
create or replace package utilities as
type sentinte is table of number;

type ids is table of number index by pls_integer;

type nume is table of varchar(62) index by pls_integer;

procedure adaugare_jaf(v_denumire_banca banca.denumire_banca%type,
                       v_data_jaf jaf.data_jaf%type default
                       sysdate);

procedure adaugare_hot(v_nume_hot hot_de_banci.nume_hot%type,
                       v_prenume_hot hot_de_banci.prenume_hot%type,
                       v_denumire_banca banca.denumire_banca%type,
                       v_data_jaf jaf.data_jaf%type default
                       sysdate);

procedure adaugare_martor(v_nume_martor martor.nume_martor%type,
                           v_prenume_martor martor.prenume_martor%type,
                           v_denumire_banca banca.denumire_banca%type,
                           v_data_jaf jaf.data_jaf%type default
                           sysdate);

procedure arestare_hot_politist
  (v_nume_hot hot_de_banci.nume_hot%type,
   v_prenume_hot hot_de_banci.prenume_hot%type,
   v_nume_politist politist.nume_politist%type,
   v_prenume_politist politist.prenume_politist%type,
   v_data_arestarii arestare_hot.data_arestarii%type default
   sysdate);

procedure adaugare_sentinta
  (v_nume_hot hot_de_banci.nume_hot%type,
   v_prenume_hot hot_de_banci.prenume_hot%type,
   v_nume_judecator judecator.nume_judecator%type,
   v_prenume_judecator judecator.prenume_judecator%type,
   v_luni_sentinta sentinta.luni_de_inchisoare%type,
   v_denumire_inchisoare inchisoare.denumire_inchisoare%type,
   v_data_acordarii judecator_sentinta_hot.data_acordarii%type
   default sysdate);
```

```

function exista_client(v_nume_client client.nume_client%type,
                      v_prenume_client client.prenume_client%type,
                      v_denumire_banca banca.denumire_banca%type)
return number;

function exista_martori_la_jaf
    (v_denumire_banca banca.denumire_banca%type,
     v_data_jaf jaf.data_jaf%type default sysdate)
return number;

function exista_martor(v_nume_martor martor.nume_martor%type,
                      v_prenume_martor martor.prenume_martor%type,
                      v_denumire_banca banca.denumire_banca%type,
                      v_data_jaf jaf.data_jaf%type default
                      sysdate)
return number;

function locatie_banca(v_denumire_banca banca.denumire_banca%type)
return varchar2;

function locatie_inchisoare
    (v_denumire_inchisoare inchisoare.denumire_inchisoare%type)
return varchar2;

procedure sentinte_hot
    (v_nume_hot hot_de_banci.nume_hot%type,
     v_prenume_hot hot_de_banci.prenume_hot%type);

procedure hoti_neprinsi;

function sectie_politist
    (v_nume_politist politist.nume_politist%type,
     v_prenume_politist politist.prenume_politist%type)
return varchar2;

procedure politisti_din_sectie
    (v_denumire_sectie sectie_politie.denumire_sectie%type);

procedure tribunale_hot
    (v_nume_hot hot_de_banci.nume_hot%type,
     v_prenume_hot hot_de_banci.prenume_hot%type);

procedure politisti_hot
    (v_nume_hot hot_de_banci.nume_hot%type,
     v_prenume_hot hot_de_banci.prenume_hot%type);

procedure judecatori_hot
    (v_nume_hot hot_de_banci.nume_hot%type,
     v_prenume_hot hot_de_banci.prenume_hot%type);

function numar_jafuri_hot
    (v_nume_hot hot_de_banci.nume_hot%type,
     v_prenume_hot hot_de_banci.prenume_hot%type)
return number;

procedure banci_jefuite_hot
    (v_nume_hot hot_de_banci.nume_hot%type,

```

```

        v_prenume_hot hot_de_banci.prenume_hot%type);

procedure orase_tari_jefuite_hot
    (v_nume_hot hot_de_banci.nume_hot%type,
     v_prenume_hot hot_de_banci.prenume_hot%type);

procedure martori(v_denumire_banca banca.denumire_banca%type,
                  v_data_jaf jaf.data_jaf%type default sysdate);

procedure clienti(v_denumire_banca banca.denumire_banca%type);
end utilities;
/

create or replace package body utilities as
---a) Adaugarea unui jaf
procedure adaugare_jaf(v_denumire_banca banca.denumire_banca%type,
                       v_data_jaf jaf.data_jaf%type default
                       sysdate) is
    v_id_banca banca.id_banca%type;
    v_id_jaf jaf.id_jaf%type;
    v_nr_banci number;
begin
    select count(*)
    into v_nr_banci
    from banca
    where denumire_banca = v_denumire_banca;

    if v_nr_banci <> 0 then
        select id_banca
        into v_id_banca
        from banca
        where denumire_banca = v_denumire_banca;
    else
        select nvl(max(id_banca),0) + 1
        into v_id_banca
        from banca;
    end if;

    insert into banca (id_banca, denumire_banca)
    values (v_id_banca, v_denumire_banca);

    select nvl(max(id_jaf),0) + 1
    into v_id_jaf
    from jaf;

    insert into jaf
    values (v_id_jaf, v_id_banca, v_data_jaf);

    dbms_output.put_line('Jaful a fost inregistrat.');
exception
    when TOO_MANY_ROWS then
        dbms_output.put_line('Operatie esuata! Exista doua banci cu
                             aceeasi denumire.');
    when others then
        dbms_output.put_line('Operatie esuata! Eroarea este: ' ||
                             sqlerrm || '.');
end adaugare_jaf;

```

```

---b) Adaugarea unui hot la un anumit jaf
procedure adaugare_hot(v_nume_hot hot_de_banci.nume_hot%type,
                      v_prenume_hot hot_de_banci.prenume_hot%type,
                      v_denumire_banca banca.denumire_banca%type,
                      v_data_jaf jaf.data_jaf%type default
                      sysdate) is
    v_id_banca banca.id_banca%type;
    v_id_jaf jaf.id_jaf%type;
    v_id_hot hot_de_banci.id_hot%type;
    v_exista_hot number;
begin
    select count(*)
    into v_exista_hot
    from hot_de_banci
    where nume_hot = v_nume_hot
    and prenume_hot = v_prenume_hot;

    select id_banca
    into v_id_banca
    from banca
    where denumire_banca = v_denumire_banca;

    select id_jaf
    into v_id_jaf
    from jaf
    where id_banca = v_id_banca
    and data_jaf = v_data_jaf;

    if v_exista_hot = 0 then
        select nvl(max(id_hot),0) + 1
        into v_id_hot
        from hot_de_banci;

        insert into hot_de_banci (id_hot, nume_hot, prenume_hot)
        values (v_id_hot, v_nume_hot, v_prenume_hot);
    end if;

    insert into hot_jaf (id_jaf, id_hot)
    values (v_id_jaf, v_id_hot);

    dbms_output.put_line('Hotul a fost inregistrat.');
exception
    when NO_DATA_FOUND then
        dbms_output.put_line('Operatie esuata! Verificati
                             corectitudinea denumirii bancii si a
                             datii jafului.');
    when TOO_MANY_ROWS then
        dbms_output.put_line('Operatie esuata! Exista doua banci cu
                             aceeasi denumire.');
    when others then
        dbms_output.put_line('Operatie esuata! Eroarea este: ' ||
                             sqlerrm || '.');
end adaugare_hot;

```

```

---c) Adaugarea unui martor la un anumit jaf
procedure adaugare_martor
    (v_nume_martor martor.nume_martor%type,
     v_prenume_martor martor.prenume_martor%type,
     v_denumire_banca banca.denumire_banca%type,
     v_data_jaf jaf.data_jaf%type default sysdate) is
    v_id_banca banca.id_banca%type;
    v_id_jaf jaf.id_jaf%type;
    v_id_martor martor.id_martor%type;
    v_exista_martor number;
begin
    select count(*)
    into v_exista_martor
    from martor
    where nume_martor = v_nume_martor
    and prenume_martor = v_prenume_martor;

    select id_banca
    into v_id_banca
    from banca
    where denumire_banca = v_denumire_banca;

    select id_jaf
    into v_id_jaf
    from jaf
    where id_banca = v_id_banca
    and data_jaf = v_data_jaf;

    if v_exista_martor = 0 then
        select nvl(max(id_martor),0) + 1
        into v_id_martor
        from martor;

        insert into martor (id_martor, nume_martor, prenume_martor)
        values (v_id_martor, v_nume_martor, v_prenume_martor);
    end if;

    insert into martor_jaf (id_jaf, id_martor)
    values (v_id_jaf, v_id_martor);

    dbms_output.put_line('Martorul a fost inregistrat.');
exception
    when NO_DATA_FOUND then
        dbms_output.put_line('Operatie esuata! Verificati
                            corectitudinea denumirii bancii si a
                            datii jafului.');
    when TOO_MANY_ROWS then
        dbms_output.put_line('Operatie esuata! Exista doua banci cu
                            aceeasi denumire.');
    when others then
        dbms_output.put_line('Operatie esuata! Eroarea este: ' ||
                            sqlerrm || '.');
end adaugare_martor;

```

```

---d) Arestarea unui hot
procedure arestare_hot_politist
    (v_nume_hot hot_de_banci.nume_hot%type,
     v_prenume_hot hot_de_banci.prenume_hot%type,
     v_nume_politist politist.nume_politist%type,
     v_prenume_politist politist.prenume_politist%type,
     v_data_arestarii arestare_hot.data_arestarii%type default
     sysdate) is
    v_id_politist politist.id_politist%type;
    v_id_hot hot_de_banci.id_hot%type;
begin
    select id_politist
    into v_id_politist
    from politist
    where nume_politist = v_nume_politist
    and prenume_politist = v_prenume_politist;

    select id_hot
    into v_id_hot
    from hot_de_banci
    where nume_hot = v_nume_hot
    and prenume_hot = v_prenume_hot;

    insert into arestare_hot
    values (v_id_hot, v_id_politist, v_data_arestarii);

    dbms_output.put_line('Arestitarea a fost inregistrata.');
exception
    when NO_DATA_FOUND then
        dbms_output.put_line('Operatie esuata! Verificati
                           corectitudinea numelor hotului si
                           politistului.');
    when TOO_MANY_ROWS then
        dbms_output.put_line('Operatie esuata! Există mai multe
                           persoane cu același nume.');
    when others then
        dbms_output.put_line('Operatie esuata! Eroarea este: ' ||
                           sqlerrm || '.');
end arestare_hot_politist;

---e) Adaugarea unei sentinte
procedure adaugare_sentinta
    (v_nume_hot hot_de_banci.nume_hot%type,
     v_prenume_hot hot_de_banci.prenume_hot%type,
     v_nume_judecator judecator.nume_judecator%type,
     v_prenume_judecator judecator.prenume_judecator%type,
     v_luni_sentinta sentinta.luni_de_inchisoare%type,
     v_denumire_inchisoare inchisoare.denumire_inchisoare%type,
     v_data_acordarii judecator_sentinta_hot.data_acordarii%type
     default sysdate) is
    v_id_judecator judecator.id_judecator%type;
    v_id_hot hot_de_banci.id_hot%type;
    v_id_sentinta sentinta.id_sentinta%type;
    v_id_inchisoare inchisoare.id_inchisoare%type;
    v_exista_sentinta number;
begin
    select id_inchisoare
    into v_id_inchisoare

```

```

from inchisoare
where denumire_inchisoare = v_denumire_inchisoare;

select count(*)
into v_exista_sentinta
from sentinta
where luni_de_inchisoare = v_luni_sentinta
and id_inchisoare = v_id_inchisoare;

select id_judecator
into v_id_judecator
from judecator
where nume_judecator = v_nume_judecator
and prenume_judecator = v_prenume_judecator;

select id_hot
into v_id_hot
from hot_de_banci
where nume_hot = v_nume_hot
and prenume_hot = v_prenume_hot;

if v_exista_sentinta = 0 then
    select nvl(max(id_sentinta), 0) + 1
    into v_id_sentinta
    from sentinta;

    insert into sentinta
    values (v_id_sentinta, v_luni_sentinta, v_id_inchisoare);
else
    select id_sentinta
    from v_id_sentinta
    where luni_de_inchisoare = v_luni_sentinta
    and id_inchisoare = v_id_inchisoare;
end if;

insert into judecator_sentinta_hot
values (v_id_judecator, v_id_sentinta, v_id_hot,
        v_data_acordarii);

dbms_output.put_line('Sentinta a fost inregistrata.');
exception
    when NO_DATA_FOUND then
        dbms_output.put_line('Operatie esuata! Verificati
                            corectitudinea denumirii inchisorii
                            si numelor hotului si
                            judecatorului.');
    when TOO_MANY_ROWS then
        dbms_output.put_line('Operatie esuata! Exista mai multe
                            inchisori/persoane cu acelasi
                            nume.');
    when others then
        dbms_output.put_line('Operatie esuata! Eroarea este: ' ||
                            sqlerrm || '.');
end adaugare_sentinta;

---f) Verificarea existentei unui anumit client la o anumita banca
function exists_client(v_nume_client client.nume_client%type,
                       v_prenume_client client.prenume_client%type,

```

```

        v_denumire_banca banca.denumire_banca%type)
return number is
    v_exista_client number;
begin
    select count(*)
    into v_exista_client
    from client
    join banca using (id_banca)
    where nume_client = v_nume_client
    and prenume_client = v_prenume_client
    and denumire_banca = v_denumire_banca;

    if v_exista_client = 0 then
        return 0;
    else
        return 1;
    end if;
exception
    when others then
        return -1;
end exista_client;

---g) Verificarea existentei a cel putin unui martor la un anumit jaf
function exista_martori_la_jaf
    (v_denumire_banca banca.denumire_banca%type,
     v_data_jaf jaf.data_jaf%type default sysdate)
return number is
    v_exista_martori number;
begin
    select count(*)
    into v_exista_martori
    from martor
    join martor_jaf using (id_martor)
    join jaf using (id_jaf)
    join banca using (id_banca)
    where denumire_banca = v_denumire_banca
    and data_jaf = v_data_jaf;

    if v_exista_martori = 0 then
        return 0;
    else
        return 1;
    end if;
exception
    when others then
        return -1;
end exista_martori_la_jaf;

---h) Verificarea existentei unui anumit martor la un anumit jaf
function exista_martor(v_nume_martor martor.nume_martor%type,
                       v_prenume_martor martor.prenume_martor%type,
                       v_denumire_banca banca.denumire_banca%type,
                       v_data_jaf jaf.data_jaf%type default
sysdate)
return number is
    v_exista_martor number;
begin
    select count(*)

```

```

        into v_exista_martor
        from martor
        join martor_jaf using (id_martor)
        join jaf using (id_jaf)
        join banca using (id_banca)
        where nume_martor = v_numemartor
        and prenume_martor = v_prenume_martor
        and denumire_banca = v_denumire_banca
        and data_jaf = v_data_jaf;

        if v_exista_martor = 0 then
            return 0;
        else
            return 1;
        end if;
    exception
        when others then
            return -1;
    end exists_martor;

---i) Afisarea orasului si a tarii unde se afla o anumita banca
function locatie_banca(v_denumire_banca banca.denumire_banca%type)
return varchar2 is
    v_id_banca banca.id_banca%type;
    v_oras oras.denumire_oras%type;
    v_tara tara.denumire_tara%type;
begin
    select denumire_oras, denumire_tara
    into v_oras, v_tara
    from banca b, oras o, tara t
    where b.id_oras = o.id_oras
    and b.id_tara = t.id_tara
    and denumire_banca = v_denumire_banca;

    return v_oras || ', ' || v_tara;
exception
    when others then
        dbms_output.put_line('Operatie esuata! Eroarea este: ' ||
                             sqlerrm || '.');
end locatie_banca;

---j) Afisarea orasului si a tarii unde se afla o anumita inchisoare
function locatie_inchisoare(v_denumire_inchisoare
inchisoare.denumire_inchisoare%type)
return varchar2 is
    v_id_inchisoare inchisoare.id_inchisoare%type;
    v_oras oras.denumire_oras%type;
    v_tara tara.denumire_tara%type;
begin
    select denumire_oras, denumire_tara
    into v_oras, v_tara
    from inchisoare i, oras o, tara t
    where i.id_oras = o.id_oras
    and i.id_tara = t.id_tara
    and denumire_inchisoare = v_denumire_inchisoare;

    return v_oras || ', ' || v_tara;
exception

```

```

        when others then
            dbms_output.put_line('Operatie esuata! Eroarea este: ' ||
                                sqlerrm || '.');
    end locatie_inchisoare;

---k) Afisarea sentintelor (lunilor de inchisoare) savarsite de / pe
care le va savarsi un anumit hot
procedure sentinte_hot(v_numar_hot hot_de_banci.nume_hot%type,
                       v_prenume_hot hot_de_banci.prenume_hot%type)
is
    v_sentinte_hot sentinte := sentinte();
    v_numar_sentinte number;
begin
    select count(*)
    into v_numar_sentinte
    from sentinta
    join judecator_sentinta_hot using (id_sentinta)
    join hot_de_banci using (id_hot)
    where numar_hot = v_numar_hot
    and prenume_hot = v_prenume_hot
    group by numar_hot, prenume_hot;

    if v_numar_sentinte = 0 then
        dbms_output.put_line('Hotul ' || v_prenume_hot || ' ' ||
                            v_numar_hot || ' nu are inca nicio
                            sentinta.');
    else
        select luni_de_inchisoare
        bulk collect into v_sentinte_hot
        from sentinta
        join judecator_sentinta_hot using (id_sentinta)
        join hot_de_banci using (id_hot)
        where numar_hot = v_numar_hot
        and prenume_hot = v_prenume_hot
        order by data_acordarii;

        for i in v_sentinte_hot.first..v_sentinte_hot.last loop
            dbms_output.put_line('Sentinta ' || i || ':' || |
                                v_sentinte_hot(i) || ' luni');
        end loop;
    end if;
exception
    when others then
        dbms_output.put_line('Operatie esuata! Eroarea este: ' ||
                                sqlerrm || '.');
end sentinte_hot;

---l) Afisarea hotilor care nu sunt inca prinsi
procedure hoti_neprinsi is
    v_id_hoti_ids := ids();
    v_numar_hoti_neprinsi numar := numar();
    v_numar_hot hot_de_banci.nume_hot%type;
    v_prenume_hot hot_de_banci.prenume_hot%type;
    v_nr_arestari number;
    v_nr_sentinte number;
    counter number := 1;
begin
    select id_hot

```

```

bulk collect into v_id_hoti
from hot_de_banci;

for i in v_id_hoti.first..v_id_hoti.last loop
    select count(*), nume_hot, prenume_hot
    into v_nr_sentinte, v_numelhot, v_prenume_hot
    from sentinta
    join judecator_sentinta_hot using (id_sentinta)
    right join hot_de_banci using (id_hot)
    where id_hot = v_id_hoti(i)
    group by nume_hot, prenume_hot;

    select count(*)
    into v_nr_arestari
    from arestare_hot
    right join hot_de_banci using (id_hot)
    where id_hot = v_id_hoti(i)
    group by id_hot;

    if v_nr_arestari < v_nr_sentinte then
        v_numelhot_neprinsi(counter) := v_numelhot || ' ' ||
                                         v_prenume_hot;
        counter := counter + 1;
    end if;
end loop;

if v_numelhot_neprinsi.count <> 0 then
    for i in
v_numelhot_neprinsi.first..v_numelhot_neprinsi.last loop
        dbms_output.put_line('Hot neprinsi ' || i || ': ' ||
                             v_numelhot_neprinsi(i));
    end loop;
else
    dbms_output.put_line('Nu exista niciun hot neprinsi.');
end if;
exception
    when NO_DATA_FOUND then
        dbms_output.put_line('Nu exista niciun hot inregistrat in
                             baza de date.');
    when others then
        dbms_output.put_line('Operatie esuata! Eroarea este: ' ||
                             sqlerrm || '.');
end hoti_neprinsi;

---m) Afisarea sectiei la care lucreaza un anumit politist
function sectie_politist
    (v_numelpolitist politist.numelpolitist%type,
     v_prenume_politist politist.prenume_politist%type)
return varchar2 is
    v_id_sectie sectie_politie.id_sectie%type;
    v_denumire_sectie sectie_politie.denumire_sectie%type;
begin
    select id_sectie
    into v_id_sectie
    from politist
    where nume_politist = v_numelpolitist
    and prenume_politist = v_prenume_politist;

```

```

        select denumire_sectie
        into v_denumire_sectie
        from sectie_politie
        where id_sectie = v_id_sectie;

        return v_denumire_sectie;
exception
    when NO_DATA_FOUND then
        return 'Sectia politistului' || v_nume_politist || ' ' ||
               v_prenume_politist || 'nu este inregistrata in baza
               de date.';
    when TOO_MANY_ROWS then
        return 'Exista mai multi politisti cu acelasi nume.';
    when others then
        return 'Operatie esuata! Eroarea este: ' || sqlerrm || '.';
end sectie_politist;

---n) Afisarea politistilor care lucreaza la o anumita sectie
procedure politisti_din_sectie
    (v_denumire_sectie sectie_politie.denumire_sectie%type) is
    v_id_sectie sectie_politie.id_sectie%type;
    v_nume_politisti nume();
begin
    select id_sectie
    into v_id_sectie
    from sectie_politie
    where denumire_sectie = v_denumire_sectie;

    select nume_politist || ' ' || prenume_politist
    bulk collect into v_nume_politisti
    from politist
    where id_sectie = v_id_sectie;

    if v_nume_politisti.count <> 0 then
        for i in v_nume_politisti.first..v_nume_politisti.last loop
            dbms_output.put_line('Politist' || i || ': ' ||
                                 v_nume_politisti(i));
        end loop;
    else
        dbms_output.put_line('Nu exista politisti care lucreaza in
                             aceasta sectie.');
    end if;

exception
    when NO_DATA_FOUND then
        dbms_output.put_line('Nu exista nicio sectie de politie cu
                             acest nume.');
    when others then
        dbms_output.put_line('Operatie esuata! Eroarea este: ' ||
                             sqlerrm || '.');
end politisti_din_sectie;

---o) Afisarea tuturor tribunalelor in care a fost judecat un anumit hot
procedure tribunale_hot
    (v_nume_hot hot_de_banci.nume_hot%type,
     v_prenume_hot hot_de_banci.prenume_hot%type) is
    v_id_hot hot_de_banci.id_hot%type;
    v_nume_tribunale nume();

```

```

        v_nr_tribunale number;
begin
    select id_hot
    into v_id_hot
    from hot_de_banci
    where nume_hot = v_nume_hot
    and prenume_hot = v_prenume_hot;

    select count(*)
    into v_nr_tribunale
    from judecator_sentinta_hot
    join judecator using (id_judecator)
    where id_hot = v_id_hot;

    if v_nr_tribunale <> 0 then
        select denumire_tribunal
        bulk collect into v_nume_tribunale
        from judecator_sentinta_hot
        join judecator using (id_judecator)
        where id_hot = v_id_hot;

        for i in v_nume_tribunale.first..v_nume_tribunale.last loop
            dbms_output.put_line('Tribunal ' || i || ':' || v_nume_tribunale(i));
        end loop;
    else
        dbms_output.put_line('Hotul ' || v_nume_hot || ' ' ||
                             v_prenume_hot || ' nu a fost judecat
                             pana in acest moment.');
    end if;

exception
    when NO_DATA_FOUND then
        dbms_output.put_line('Nu exista niciun hot cu numele
                             dat.');
    when others then
        dbms_output.put_line('Operatie esuata! Eroarea este: ' ||
                             sqlerrm || '.');
end tribunale_hot;

--p) Afisarea tuturor politistilor care au prins, de-a lungul timpului,
un anumit hot
procedure politisti_hot
    (v_nume_hot hot_de_banci.nume_hot%type,
     v_prenume_hot hot_de_banci.prenume_hot%type) is
    v_id_hot hot_de_banci.id_hot%type;
    v_id_politisti ids := ids();
    v_nume_politisti nume := nume();
    v_nr_politisti number;
begin
    select id_hot
    into v_id_hot
    from hot_de_banci
    where nume_hot = v_nume_hot
    and prenume_hot = v_prenume_hot;

    select count(*)
    into v_nr_politisti

```

```

from arestare_hot
where id_hot = v_id_hot
group by id_hot;

if v_nr_politisti <> 0 then
    select nume_politist || ' ' || prenume_politist
    bulk collect into v_nume_politisti
    from arestare_hot
    join politist using (id_politist)
    where id_hot = v_id_hot;

    for i in v_nume_politisti.first..v_nume_politisti.last loop
        dbms_output.put_line('Politist' || i || ': ' ||
                             v_nume_politisti(i));
    end loop;
else
    dbms_output.put_line('Hotul ' || v_nume_hot || ' ' ||
                         v_prenume_hot || ' nu a fost prins
                         pana in acest moment.');
end if;

exception
when NO_DATA_FOUND then
    dbms_output.put_line('Nu exista niciun hot cu numele
                         dat.');
when others then
    dbms_output.put_line('Operatie esuata! Eroarea este: ' ||
                         sqlerrm || '.');
end politisti_hot;

---q) Afisarea tuturor judecatorilor care au judecat, de-a lungul
timpului, un anumit hot
procedure judecatori_hot
    (v_nume_hot hot_de_banci.nume_hot%type,
     v_prenume_hot hot_de_banci.prenume_hot%type) is
    v_id_hot hot_de_banci.id_hot%type;
    v_id_judecatori_ids := ids();
    v_nume_judecatori nume := nume();
    v_nr_judecatori number;
begin
    select id_hot
    into v_id_hot
    from hot_de_banci
    where nume_hot = v_nume_hot
    and prenume_hot = v_prenume_hot;

    select count(*)
    into v_nr_judecatori
    from judecator_sentinta_hot
    where id_hot = v_id_hot
    group by id_hot;

    if v_nr_judecatori <> 0 then
        select nume_judecator || ' ' || prenume_judecator
        bulk collect into v_nume_judecatori
        from judecator_sentinta_hot
        join judecator using (id_judecator)
        where id_hot = v_id_hot;

```

```

        for i in v_nume_judecatori.first..v_nume_judecatori.last
            loop
                dbms_output.put_line('Judecator ' || i || ':' || 
                    v_nume_judecatori(i));
            end loop;
        else
            dbms_output.put_line('Hotul ' || v_nume_hot || ' ' ||
                v_prenume_hot || ' nu a fost judecat
                pana in acest moment.');
        end if;

exception
    when NO_DATA_FOUND then
        dbms_output.put_line('Nu exista niciun hot cu numele
            dat.');
    when others then
        dbms_output.put_line('Operatie esuata! Eroarea este: ' ||
            sqlerrm || '.');
end judecatori_hot;

---r) Afisarea numarului de jafuri la care a participat un anumit hot
function numar_jafuri_hot
    (v_nume_hot hot_de_banci.nume_hot%type,
     v_prenume_hot hot_de_banci.prenume_hot%type)
return number is
    v_nr_jafuri number;
begin
    select count(*)
    into v_nr_jafuri
    from hot_jaf
    join hot_de_banci using (id_hot)
    where nume_hot = v_nume_hot
    and prenume_hot = v_prenume_hot
    group by id_hot;

    return v_nr_jafuri;
exception
    when NO_DATA_FOUND then
        dbms_output.put_line('Nu exista niciun hot cu numele
            dat.');
    when others then
        dbms_output.put_line('Operatie esuata! Eroarea este: ' ||
            sqlerrm || '.');
end numar_jafuri_hot;

---s) Afisarea tuturor bancilor pe care le-a jefuit un anumit hot
procedure banci_jefuite_hot
    (v_nume_hot hot_de_banci.nume_hot%type,
     v_prenume_hot hot_de_banci.prenume_hot%type) is
    v_id_hot hot_de_banci.id_hot%type;
    v_id_banci ids := ids();
    v_nume_banci nume := nume();
    v_nr_banci number;
begin
    select id_hot
    into v_id_hot
    from hot_de_banci

```

```

where nume_hot = v_nume_hot
and prenume_hot = v_prenume_hot;

select count(distinct id_banca)
into v_nr_banci
from hot_jaf
join hot_de_banci using (id_hot)
join jaf using (id_jaf)
where id_hot = v_id_hot
group by id_hot;

if v_nr_banci <> 0 then
    select distinct denumire_banca
    bulk collect into v_nume_banci
    from hot_jaf
    join hot_de_banci using (id_hot)
    join jaf using (id_jaf)
    join banca using (id_banca)
    where id_hot = v_id_hot;

    for i in v_nume_banci.first..v_nume_banci.last loop
        dbms_output.put_line('Banca ' || i || ':' || v_nume_banci(i));
    end loop;
else
    dbms_output.put_line('Hotul ' || v_nume_hot || ' ' ||
                          v_prenume_hot || ' nu a jefuit nicio
                           banca pana in acest moment.');
end if;

exception
    when NO_DATA_FOUND then
        dbms_output.put_line('Nu exista niciun hot cu numele
                           dat.');
    when others then
        dbms_output.put_line('Operatie esuata! Eroarea este: ' ||
                           sqlerrm || '.');
end banci_jefuite_hot;

---t) Afisarea tuturor oraselor si tarilor in care exista banchi jefuite
de un anumit hot
procedure orase_tari_jefuite_hot
    (v_nume_hot hot_de_banci.nume_hot%type,
     v_prenume_hot hot_de_banci.prenume_hot%type) is
    v_id_hot hot_de_banci.id_hot%type;
    v_id_banci ids := ids();
    v_nume_orase_tari nume := nume();
    v_nr_banci number;
begin
    select id_hot
    into v_id_hot
    from hot_de_banci
    where nume_hot = v_nume_hot
    and prenume_hot = v_prenume_hot;

    select count(distinct id_banca)
    into v_nr_banci
    from hot_jaf

```

```

join hot_de_banci using (id_hot)
join jaf using (id_jaf)
where id_hot = v_id_hot
group by id_hot;

if v_nr_banci <> 0 then
    select distinct denumire_oras || ', ' || denumire_tara
    bulk collect into v_nume_orase_tari
    from hot_jaf hj, hot_de_banci h, jaf j,
         banca b, oras o, tara t
    where hj.id_hot = h.id_hot
    and j.id_jaf = hj.id_jaf
    and j.id_banca = b.id_banca
    and b.id_oras = o.id_oras
    and o.id_tara = t.id_tara
    and h.id_hot = v_id_hot;

    for i in v_nume_orase_tari.first..v_nume_orase_tari.last
        loop
            dbms_output.put_line('Locatie banca ' || i || ': ' ||
                                v_nume_orase_tari(i));
        end loop;
    else
        dbms_output.put_line('Hotul ' || v_nume_hot || ' ' ||
                            v_prenume_hot || ' nu a jefuit nicio
                            banca pana in acest moment.');
    end if;

exception
    when NO_DATA_FOUND then
        dbms_output.put_line('Nu exista niciun hot cu numele
                            dat.');
    when others then
        dbms_output.put_line('Operatie esuata! Eroarea este: ' ||
                            sqlerrm || '.');
end orase_tari_jefuite_hot;

---u) Afisarea detaliilor martorilor la un anumit jaf
procedure martori(v_denumire_banca banca.denumire_banca%type,
                   v_data_jaf jaf.data_jaf%type default sysdate) is
    v_id_jaf jaf.id_jaf%type;
    v_nr_martori number;
    v_nume_martori nume := nume();
    v_varsta_martori ids := ids();
begin
    select id_jaf
    into v_id_jaf
    from jaf
    join banca using (id_banca)
    where denumire_banca = v_denumire_banca
    and data_jaf = v_data_jaf;

    select count(*)
    into v_nr_martori
    from martor_jaf
    where id_jaf = v_id_jaf
    group by id_jaf;

```

```

if v_nr_martori <> 0 then
    select nume_martor || ' ' || prenume_martor, varsta_martor
    bulk collect into v_numemartori, v_varstamartori
    from martor_jaf
    join martor using (id_martor)
    where id_jaf = v_id_jaf;

    for i in v_numemartori.first..v_numemartori.last loop
        dbms_output.put_line('Martor ' || i || ': ' ||
            v_numemartori(i) || ', ' ||
            v_varstamartori(i) || ' ani');
    end loop;
else
    dbms_output.put_line('Jaful banchii ' || v_denumire_banca ||
        ' din data de ' || v_data_jaf || ' nu
        are niciun martor.');
end if;
exception
when NO_DATA_FOUND then
    dbms_output.put_line('Nu exista niciun jaf la aceasta
        banca, in data respectiva.');
when others then
    dbms_output.put_line('Operatie esuata! Eroarea este: ' ||
        sqlerrm || '.');
end martori;

---v) Afisarea detaliilor clientilor unei anumite banci
procedure clienti(v_denumire_banca banca.denumire_banca%type) is
    v_nr_clienti number;
    v_numemclienti nume := nume();
    v_varstamclienti ids := ids();
begin
    select count(*)
    into v_nr_clienti
    from client
    join banca using (id_banca)
    where denumire_banca = v_denumire_banca
    group by id_banca;

    if v_nr_clienti <> 0 then
        select nume_client || ' ' || prenume_client, varsta_client
        bulk collect into v_numemclienti, v_varstamclienti
        from client
        join banca using (id_banca)
        where denumire_banca = v_denumire_banca;

        for i in v_numemclienti.first..v_numemclienti.last loop
            dbms_output.put_line('Client ' || i || ': ' ||
                v_numemclienti(i) || ', ' ||
                v_varstamclienti(i) || ' ani');
        end loop;
    else
        dbms_output.put_line('Banca ' || v_denumire_banca || ' nu
            are niciun client.');
    end if;
exception
when NO_DATA_FOUND then
    dbms_output.put_line('Nu exista nicio banca cu aceasta

```

```

                denumire.') ;
when others then
    dbms_output.put_line('Operatie esuata! Eroarea este: ' ||
                           sqlerrm || '.');
end clienti;

end utilities;
/

--apelare:
---a)
----OK! (exista banca)
begin
    utilities.adaugare_jaf('BCR');
end;
/

select denumire_banca
from jaf
join banca using (id_banca)
where denumire_banca = 'BCR';

----OK! (nu exista banca)
begin
    utilities.adaugare_jaf('BRD');
end;
/

select denumire_banca
from jaf
join banca using (id_banca)
where denumire_banca = 'BRD';

----EROARE!
begin
    insert into banca values (505, 'BCR', 2, 'RO');
    utilities.adaugare_jaf('BCR');
end;
/

rollback;

---b)
begin
    utilities.adaugare_hot('Nume', 'Prenume', 'Caixabank', '07-APR-
                           02');
end;
/

select nume_hot, prenume_hot, denumire_banca
from hot_jaf
join jaf using (id_jaf)
join banca using (id_banca)
join hot_de_banci using (id_hot)
where nume_hot = 'Nume'
and denumire_banca = 'Caixabank';

rollback;

```

```

---c)
begin
    utilities.adaugare_martor('Nume', 'Prenume', 'Caixabank', '07-APR-
                                02');
end;
/

select nume_martor, prenume_martor, denumire_banca
from martor_jaf
join jaf using (id_jaf)
join banca using (id_banca)
join martor using (id_martor)
where nume_martor = 'Nume'
and denumire_banca = 'Caixabank';

rollback;

---d)
begin
    utilities.arestare_hot_politist('Derichs', 'Roelof', 'Voiculescu',
                                    'Alina');
end;
/

select nume_hot, prenume_hot, nume_politist, prenume_politist
from arestare_hot
join hot_de_banci using (id_hot)
join politist using (id_politist)
where nume_hot = 'Derichs'
and nume_politist = 'Voiculescu';

rollback;

---e)
begin
    utilities.adaugare_sentinta('Spitznagel', 'Tania', 'Villanova',
                                'Renata', 500, 'Inchisoarea St.
                                Gilles');
end;
/

select nume_hot, prenume_hot, nume_judecator, prenume_judecator,
luni_de_inchisoare
from judecator_sentinta_hot
join judecator using (id_judecator)
join hot_de_banci using (id_hot)
join sentinta using (id_sentinta)
where nume_hot = 'Spitznagel'
and nume_judecator = 'Villanova';

---f)
select utilities.exista_client('Gabrielli', 'Jaimie', 'BCR') as "Exista
clientul dat?"
from dual;

select nume_client, prenume_client, denumire_banca
from client

```

```

join banca using (id_banca)
where nume_client = 'Gabrielli';

---g)
select utilities.exista_martori_la_jaf('Caixabank', '07-APR-02') as
          "Exista martori?"
from dual;

select count(*) as "Numar martori"
from martor
join martor_jaf using (id_martor)
join jaf using (id_jaf)
join banca using (id_banca)
where denumire_banca = 'Caixabank'
and data_jaf = '07-APR-02';

---h)
select utilities.exista_martor('Norman', 'Valerie', 'Caixabank',
                                '07-APR-02') as "Exista martorul dat?"
from dual;

select nume_martor, prenume_martor, denumire_banca, data_jaf
from martor
join martor_jaf using (id_martor)
join jaf using (id_jaf)
join banca using (id_banca)
where nume_martor = 'Norman'
and denumire_banca = 'Caixabank'
and data_jaf = '07-APR-02';

---i)
select utilities.locatie_banca('Caixabank') as "Locatie banca"
from dual;

select denumire_oras, denumire_tara
from banca b, oras o, tara t
where b.id_oras = o.id_oras
and b.id_tara = t.id_tara
and denumire_banca = 'Caixabank';

---j)
select utilities.locatie_inchisoare('Penitenciarul Bucuresti-Jilava')
                           as "Locatie inchisoare"
from dual;

select denumire_oras, denumire_tara
from inchisoare i, oras o, tara t
where i.id_oras = o.id_oras
and i.id_tara = t.id_tara
and denumire_inchisoare = 'Penitenciarul Bucuresti-Jilava';

---k)
begin
    utilities.sentinte_hot('Armando', 'Anika');
end;
/

```

select luni_de_inchisoare, nume_hot, prenume_hot

```

from sentinta
join judecator_sentinta_hot using (id_sentinta)
join hot_de_banci using (id_hot)
where nume_hot = 'Armando'
and prenume_hot = 'Anika'
order by data_acordarii;

---l)
begin
    utilities.hoti_neprinsi;
end;
/

---m)
select utilities.sectie_politist('Voiculescu', 'Alina')
from dual;

---n)
begin
    utilities.politisti_din_sectie('Sectia 2 Politie');
end;
/

---o)
begin
    utilities.tribunale_hot('Jerome', 'Veronika');
end;
/

---p)
begin
    utilities.politisti_hot('Armando', 'Anika');
end;
/

---q)
begin
    utilities.judecatori_hot('Armando', 'Anika');
end;
/

---r)
select utilities.numar_jafuri_hot('Armando', 'Anika') "Numar jafuri"
from dual;

---s)
begin
    utilities.banci_jefuite_hot('Armando', 'Anika');
end;
/

select count(distinct id_banca) "Numar banci jefuite"
from hot_jaf
join hot_de_banci using (id_hot)
join jaf using (id_jaf)
where nume_hot = 'Armando'
group by id_hot;

```

```

---t)
begin
    utilities.orase_tari_jefuite_hot('Armando', 'Anika');
end;
/

---u)
begin
    utilities.martori('Commerzbank', '15-JUL-20');
end;
/

---v)
begin
    utilities.clienti('Commerzbank');
end;
/

```

SQL Worksheet | History

Worksheet | Query Builder

```

        for i in v_nume_clienti.first..v_nume_clienti.last loop
            dbms_output.put_line('Client ' || i || ':' || v_nume_clienti(i) || ', ' || v_varsta_clienti(i) || ' ani');
        end loop;
    else
        dbms_output.put_line('Banca' || v_denumire_banca || ' nu are niciun client.');
    end if;
exception
    when NO_DATA_FOUND then
        dbms_output.put_line('Nu exista nicio banca cu aceasta denumire.');
    when others then
        dbms_output.put_line('Operatie esuata! Eroarea este: ' || sqlerrm || '.');
end clienti;

end utilities;
/

```

Script Output x | Task completed in 0.491 seconds

Package UTILITIES compiled

Package Body UTILITIES compiled

a) Adăugarea unui jaf

The screenshot shows the Oracle SQL Developer interface with the following components:

- Worksheet Tab:** Contains the PL/SQL code for adding a new bank account. The code includes exception handling for NO_DATA_FOUND and others, and a call to the utilities package's adaugare_jaf procedure.
- Script Output Tab:** Shows the message "PL/SQL procedure successfully completed." indicating the script ran successfully.
- Dbms Output Tab:** Shows the message "Jaful a fost inregistrat." (The account was registered).

```
else
    dbms_output.put_line('Banca' || v_denumire_banca || ' nu are niciun client.');
end if;
exception
when NO_DATA_FOUND then
    dbms_output.put_line('Nu exista nicio banca cu aceasta denumire.');
when others then
    dbms_output.put_line('Operatie esuata! Eroarea este: ' || sqlerrm || '.');
end clienti;

end utilities;
/

--apelare:
---a)
-----OK! (există banca)
begin
    utilities.adaugare_jaf('BCR');
end;
/
```

PL/SQL procedure successfully completed.

Jaful a fost inregistrat.

SQL Worksheet History

Worksheet Query Builder

```
--OK! (exista banca)
begin
    utilities.adaugare_jaf('BCR');
end;
/
select denumire_banca
from jaf
join banca using (id_banca)
where denumire_banca = 'BCR';

--OK! (nu exista banca)
begin
    utilities.adaugare_jaf('BRD');
end;
/
```

Script Output x | Query Result x

Task completed in 0.1 seconds

```
PL/SQL procedure successfully completed.
```

Dbms Output x

Buffer Size: 20000

DB final project 1st year x

```
Jaful a fost inregistrat.
```

```
Jaful a fost inregistrat.
```

SQL Worksheet History

Worksheet Query Builder

```
where denumire_banca = 'BCR';

-----OK! (nu exista banca)
begin
    utilities.adaugare_jaf('BRD');
end;
/

select denumire_banca
from jaf
join banca using (id_banca)
where denumire_banca = 'BRD';

-----EROARE!
begin
    insert into banca values (505, 'BCR', 2, 'RO');
    utilities.adaugare_jaf('BCR');
end;
/
```

Script Output x | Query Result x
Task completed in 0.134 seconds

PL/SQL procedure successfully completed.

Dbms Output x
+ | Buffer Size: 20000 |

DB final project 1st year x
Jaful a fost inregistrat.
Jaful a fost inregistrat.
Operatie esuata! Exista doua banchi cu aceeasi denumire.

b) Adăugarea unui hot la un anumit jaf

The screenshot shows the Oracle SQL Developer interface. The top window is a 'Worksheet' tab where PL/SQL code is written. The code performs a join between 'jaf' and 'banca' tables, filters for 'BRD' bank, inserts a new bank record ('BCR') with its details, and then rolls back the transaction. It also includes a section for adding a guest ('hot') with name 'Nume', prename 'Prenume', and birthdate '07-APR-02'. The bottom windows show the 'Script Output' and 'Dbms Output' tabs. The 'Script Output' tab shows the message 'PL/SQL procedure successfully completed.'. The 'Dbms Output' tab shows messages indicating the successful insertion of the bank record and the failure of the guest insertion due to a unique constraint violation.

```
from jaf
join banca using (id_banca)
where denumire_banca = 'BRD';

-----EROARE!
begin
    insert into banca values (505, 'BCR', 2, 'RO');
    utilities.adaugare_jaf('BCR');
end;
/

rollback;

---b)
begin
    utilities.adaugare_hot('Nume', 'Prenume', 'Caixabank', '07-APR-02');
end;
/
```

PL/SQL procedure successfully completed.

DB final project 1st year x
Jaful a fost inregistrat.

Operatie esuata! Există două banchi cu aceeași denumire.

Hotul a fost inregistrat.

c) Adăugarea unui martor la un anumit jaf

The screenshot shows the Oracle SQL Developer interface. The top window is a 'Worksheet' tab containing PL/SQL code. The code creates a procedure named 'utilities.adaugare_martor' that takes four parameters: 'Nume', 'Prenume', 'Banca', and 'Data'. It performs a complex multi-table join to find the account ID ('id_hot') for a given name and bank, then inserts a new record into the 'hot_jaf' table with the found ID and the provided witness details. A 'rollback' statement is included at the end. The bottom windows show the 'Script Output' and 'Dbms Output' panes, both indicating successful execution.

```
end;
/
select nume_hot, prenume_hot, denumire_banca
from hot_jaf
join jaf using (id_jaf)
join banca using (id_banca)
join hot_de_banci using (id_hot)
where nume_hot = 'Nume'
and denumire_banca = 'Caixabank';

rollback;

---C)
begin
    utilities.adaugare_martor('Nume', 'Prenume', 'Caixabank', '07-APR-02');
end;
/
```

PL/SQL procedure successfully completed.

Dbms Output
Buffer Size: 20000
DB final project 1st year
Martorul a fost inregistrat.

d) Areșarea unui hot

The screenshot shows the Oracle SQL Developer interface. The main area displays a PL/SQL script:

```
SQL Worksheet History
Worksheet Query Builder

end;
/
select nume_martor, prenume_martor, denumire_banca
from martor_jaf
join jaf using (id_jaf)
join banca using (id_banca)
join martor using (id_martor)
where nume_martor = 'Nume'
and denumire_banca = 'Caixabank';

rollback;

---d)
begin
    utilities.arestare_hot_politist('Derichs', 'Roelof', 'Voiculescu', 'Alina');
end;
/
PL/SQL procedure successfully completed.
```

The script performs a select query to find a criminal named 'Nume' from the 'Caixabank'. It then rolls back the transaction. A comment block starts with '---d)' followed by a begin block that calls the 'utilities.arestare_hot_politist' procedure with four parameters: 'Derichs', 'Roelof', 'Voiculescu', and 'Alina'. The script ends with an end; block and a final slash. Below the script, a message indicates the procedure was successfully completed.

At the bottom, the 'Dbms Output' window shows the message: "Arestarea a fost înregistrată."

e) Adăugarea unei sentințe

The screenshot shows the Oracle SQL Developer interface. In the top navigation bar, 'Worksheet' is selected. The main area contains the following PL/SQL code:

```

end;
/
select nume_hot, prenume_hot, nume_politist, prenume_politist
from arestare_hot
join hot_de_banci using (id_hot)
join politist using (id_politist)
where nume_hot = 'Derichs'
and nume_politist = 'Voiculescu';

rollback;

---e)
begin
    utilities.adaugare_sentinta('Spitznagel', 'Tania', 'Villanova', 'Renata', 500, 'Inchisoarea St. Gilles');
end;
/

```

Below the code, the status bar indicates: 'Script Output x | Query Result x | Task completed in 0.055 seconds'. The status bar also shows 'PL/SQL procedure successfully completed.'

The screenshot shows the Oracle SQL Developer interface. In the top navigation bar, 'Dbms Output' is selected. The main area displays the message: 'Sentinta a fost inregistrata.' (The sentence was registered).

f) Verificarea existenței unui anumit client la o anumită bancă

The screenshot shows the Oracle SQL Developer interface. In the top navigation bar, 'Worksheet' is selected. The main area contains the following PL/SQL code:

```

end;
/
select nume_hot, prenume_hot, nume_judecator, prenume_judecator, luni_de_inchisoare
from judecator_sentinta_hot
join judecator using (id_judecator)
join hot_de_banci using (id_hot)
join sentinta using (id_sentinta)
where nume_hot = 'Spitznagel'
and nume_judecator = 'Villanova';

rollback;

---f)
select utilities.exista_client('Gabrielli', 'Jaimie', 'BCR') as "Exista clientul dat?"
from dual;

```

Below the code, the status bar indicates: 'Script Output x | Query Result x | SQL | All Rows Fetched: 1 in 0.02 seconds'. The status bar also shows the result of the query: 'Exista clientul dat? 1'.

g) Verificarea existenței a cel puțin unui martor la un anumit jaf

```

SQL Worksheet History
Worksheet Query Builder
where nume_hot = 'Spitznagel'
and nume_judecator = 'Villanova';

rollback;

---f)
select utilities.exista_client('Gabrielli', 'Jaimie', 'BCR') as "Exista clientul dat?"
from dual;

---g)
select nume_client, prenume_client, denumire_banca
from client
join banca using (id_banca)
where nume_client = 'Gabrielli';

---g)
select utilities.exista_martori_la_jaf('Caixabank', '07-APR-02') as "Exista martori?"
from dual;

```

Script Output | Query Result

SQL | All Rows Fetched: 1 in 0.091 seconds

Exista martori?
1

h) Verificarea existenței unui anumit martor la un anumit jaf

```

SQL Worksheet History
Worksheet Query Builder
---g)
select utilities.exista_martori_la_jaf('Caixabank', '07-APR-02') as "Exista martori?"
from dual;

---g)
select count(*) as "Numar martori"
from martor
join martor_jaf using (id_martor)
join jaf using (id_jaf)
join banca using (id_banca)
where denumire_banca = 'Caixabank'
and data_jaf = '07-APR-02';

---h)
select utilities.exista_martor('Norman', 'Valerie', 'Caixabank', '07-APR-02') as "Exista martorul dat?"
from dual;

```

Script Output | Query Result

SQL | All Rows Fetched: 1 in 0.033 seconds

Exista martorul dat?
1

i) Afişarea oraşului şi a ţării unde se află o anumită bancă

The screenshot shows the Oracle SQL Developer interface. The top menu bar has 'SQL Worksheet' and 'History'. Below it is a toolbar with various icons. The main area is divided into two panes: 'Worksheet' and 'Query Builder'. The 'Worksheet' pane contains the following SQL code:

```
--h)
select utilities.exista_martor('Norman', 'Valerie', 'Caixabank', '07-APR-02') as "Exista martorul dat?"

--i)
select nume_martor, prenume_martor, denumire_banca, data_jaf
from martor
join martor_jaf using (id_martor)
join jaf using (id_jaf)
join banca using (id_banca)
where nume_martor = 'Norman'
and denumire_banca = 'Caixabank'
and data_jaf = '07-APR-02';

---i)
select utilities.locatie_banca('Caixabank') as "Locatie banca"
from dual;
```

The bottom pane shows the 'Script Output' tab with the results of the query:

Locatie banca
1 Madrid, Spania

Statistics at the bottom of the output pane: All Rows Fetched: 1 in 0.021 seconds.

j) Afişarea oraşului şi a ţării unde se află o anumită închisoare

The screenshot shows the Oracle SQL Developer interface. The top menu bar has 'SQL Worksheet' and 'History'. Below it is a toolbar with various icons. The main area is divided into two panes: 'Worksheet' and 'Query Builder'. The 'Worksheet' pane contains the following SQL code:

```
and denumire_banca = 'Caixabank'
and data_jaf = '07-APR-02';

---i)
select utilities.locatie_banca('Caixabank') as "Locatie banca"
from dual;

--i)
select denumire_oras, denumire_tara
from banca b, oras o, tara t
where b.id_oras = o.id_oras
and b.id_tara = t.id_tara
and denumire_banca = 'Caixabank';

---j)
select utilities.locatie_inchisoare('Penitenciarul Bucuresti-Jilava') as "Locatie inchisoare"
from dual;
```

The bottom pane shows the 'Script Output' tab with the results of the query:

Locatie inchisoare
1 Bucuresti, Romania

Statistics at the bottom of the output pane: All Rows Fetched: 1 in 0.012 seconds.

k) Afişarea sentinţelor (lunilor de închisoare) săvârşite de / pe care le va săvârşi un anumit hot

The screenshot shows the Oracle SQL Developer interface. The top window is a 'Worksheet' containing PL/SQL code. The code includes a query to find the location of an inmate ('Penitenciarul Bucuresti-Jilava') and a procedure call to 'utilities.sentinte_hot' for two specific individuals ('Armando' and 'Anika'). The code is divided into sections labeled '---j)', '---k)', and a block starting with 'begin'. The bottom left panel shows the 'Script Output' tab with the message 'PL/SQL procedure successfully completed.' The bottom right panel shows the 'Dbms Output' tab with the results of the 'sentinte_hot' procedure, listing four sentences with their respective durations: 'Sentinta 1: 80 luni', 'Sentinta 2: 200 luni', 'Sentinta 3: 135 luni', and 'Sentinta 4: 200 luni'.

```
--j)
select utilities.locatie_inchisoare('Penitenciarul Bucuresti-Jilava') as "Locatie inchisoare"
from dual;

select denumire_oras, denumire_tara
from inchisoare i, oras o, tara t
where i.id_oras = o.id_oras
and i.id_tara = t.id_tara
and denumire_inchisoare = 'Penitenciarul Bucuresti-Jilava';

---k)
begin
    utilities.sentinte_hot('Armando', 'Anika');
end;
/

```

PL/SQL procedure successfully completed.

Dbms Output

DB final project 1st year

Sentinta 1: 80 luni
Sentinta 2: 200 luni
Sentinta 3: 135 luni
Sentinta 4: 200 luni

I) Afisarea hotilor care nu sunt incă prinși

The screenshot shows the Oracle SQL Developer interface. The main window displays a PL/SQL script:

```
end;
/
select luni_de_inchisoare, nume_hot, prenume_hot
from sentinta
join judecator_sentinta_hot using (id_sentinta)
join hot_de_banci using (id_hot)
where nume_hot = 'Armando'
and prenume_hot = 'Anika'
order by data_acordarii;

---1)
begin
    utilities.hotii_neprinsi;
end;
/
```

The code is highlighted with syntax coloring. The 'begin' keyword is selected. Below the code, the message "PL/SQL procedure successfully completed." is displayed.

In the bottom panel, the 'Dbms Output' tab shows the result:

```
Nu există niciun hot neprins.
```

m) Afişarea secției la care lucrează un anumit polițist

The screenshot shows the Oracle SQL Developer interface. The top part is the 'Worksheet' tab, which contains the following PL/SQL code:

```
join judecator_sentinta_hot using (id_sentinta)
join hot_de_banci using (id_hot)
where nume_hot = 'Armando'
and prenume_hot = 'Anika'
order by data_acordarii;

---l)
begin
    utilities.hoti_neprinsi;
end;
/

---m)
select utilities.sectie_politist('Voiculescu', 'Alina')
```

The bottom part is the 'Query Result' tab, which displays the output of the last query:

UTILITIES.SECTIE_POLITIST('VOICULESCU','ALINA')
1 Sectia 2 Politie

Information bar at the bottom of the result tab: All Rows Fetched: 1 in 0.063 seconds

n) Afişarea poliştilor care lucrează la o anumită secţie

The screenshot shows the Oracle SQL Developer interface. The top menu bar has 'SQL Worksheet' selected. The main workspace contains the following PL/SQL code:

```
--l)
begin
    utilities.hoti_neprinsi;
end;
/
--m)
select utilities.sectie_polist('Voiculescu', 'Alina')
from dual;

--n)
begin
    utilities.politisti_din_sectie('Sectia 2 Politie');
end;
/

```

Below the code, the status bar indicates: 'Task completed in 0.058 seconds'.

PL/SQL procedure successfully completed.

The screenshot shows the 'Dbms Output' window in Oracle SQL Developer. It displays the results of the executed PL/SQL code:

```
Politist 1: Voiculescu Alina
Politist 2: Dumitrescu Dragos
```

o) Afisarea tuturor tribunalelor in care a fost judecat un anumit hot

The screenshot shows the Oracle SQL Developer interface. The top window is a 'Worksheet' tab containing a PL/SQL script. The script includes several sections labeled with dashed numbers (---m, ---n, ---o) and contains calls to utility procedures like 'sectie_politist', 'politisti_din_sectie', and 'tribunale_hot'. The 'begin' and 'end;' statements for section o are highlighted in blue. Below the worksheet is a 'Script Output' tab showing the message 'PL/SQL procedure successfully completed.' The bottom window is a 'Dbms Output' tab displaying the results: 'Tribunal 1: Tribunalul Madrid' and 'Tribunal 2: Tribunalul Tokyo'.

```
--m)
select utilities.sectie_politist('Voiculescu', 'Alina')
from dual;

--n)
begin
    utilities.politisti_din_sectie('Sectia 2 Politie');
end;
/

--o)
begin
    utilities.tribunale_hot('Jerome', 'Veronika');
end;
/

```

PL/SQL procedure successfully completed.

```
Tribunal 1: Tribunalul Madrid
Tribunal 2: Tribunalul Tokyo
```

p) Afişarea tuturor poliştilor care au prins, de-a lungul timpului, un anumit hot

The screenshot shows the Oracle SQL Developer interface with three main panes:

- SQL Worksheet**: Contains the PL/SQL code for listing police officers who have caught a specific suspect over time. The code uses nested loops and procedures from the 'utilities' package.
- Script Output**: Shows the message "PL/SQL procedure successfully completed." indicating the script ran without errors.
- Dbms Output**: Displays the names of four police officers: Gimenez Alejandro, Herrmann Loreley, Piette Janne, and Voiculescu Alina.

```
begin
    utilities.politisti_din_sectie('Sectia 2 Politie');
end;
/
---o)
begin
    utilities.tribunale_hot('Jerome', 'Veronika');
end;
/
---p)
begin
    utilities.politisti_hot('Armando', 'Anika');
end;
/
```

Script Output: Task completed in 0.072 seconds

PL/SQL procedure successfully completed.

Dbms Output: Buffer Size: 20000

DB final project 1st year

Politist 1: Gimenez Alejandro
Politist 2: Herrmann Loreley
Politist 3: Piette Janne
Politist 4: Voiculescu Alina

q) Afişarea tuturor judecătorilor care au judecat, de-a lungul timpului, un anumit hot

SQL Worksheet History

Worksheet Query Builder

```
begin
    utilities.tribunale_hot('Jerome', 'Veronika');
end;
/
---p)
begin
    utilities.politisti_hot('Armando', 'Anika');
end;
/
---q)
begin
    utilities.judecatori_hot('Armando', 'Anika');
end;
/
```

Script Output x Query Result x

Task completed in 0.058 seconds

PL/SQL procedure successfully completed.

Dbms Output x

Buffer Size: 20000

DB final project 1st year x

```
Judecator 1: Gordon Gilbert
Judecator 2: Vincent Horace
Judecator 3: Lehmann Emmerich
Judecator 4: Takayuki Kawakami
```

r) Afisarea numarului de jafuri la care a participat un anumit hot

The screenshot shows the Oracle SQL Developer interface. The top part is the SQL Worksheet window, which contains a PL/SQL block. The block includes three parts labeled p, q, and r. Part p calls a procedure to add a record to a table named 'politisti_hot'. Part q adds another record to the same table. Part r selects the count of records from the table for specific names ('Armando' and 'Anika'). The result of part r is displayed in the Query Result window, showing a single row with the value 4. The bottom part is the Dbms Output window, which displays the names of four judges: Gordon Gilbert, Vincent Horace, Lehmann Emmerich, and Takayuki Kawakami.

```
-->p)
begin
    utilities.politisti_hot('Armando', 'Anika');
end;
/
-->q)
begin
    utilities.judecatori_hot('Armando', 'Anika');
end;
/
-->r)
select utilities.numar_jafuri_hot('Armando', 'Anika') "Numar jafuri"
from dual;
```

Script Output x | Query Result x
SQL | All Rows Fetched: 1 in 0.058 seconds

Numar jafuri
1 4

Dbms Output x
Buffer Size: 20000 |
DB final project 1st year x

```
Judecator 1: Gordon Gilbert
Judecator 2: Vincent Horace
Judecator 3: Lehmann Emmerich
Judecator 4: Takayuki Kawakami
```

s) Afisarea tuturor bancilor pe care le-a jefuit un anumit hot

The screenshot shows the Oracle SQL Developer interface. The top window is a 'Worksheet' tab where PL/SQL code is written. The code includes comments (---q), begin/end blocks, and a select statement. The bottom window is a 'Script Output' tab showing the message 'PL/SQL procedure successfully completed.' The bottom-most window is a 'Dbms Output' tab displaying the results of the procedure, which are the names of three banks: Banca 1: Banque CPH, Banca 2: Caixabank, and Banca 3: Mizuho Bank.

```
--q)
begin
    utilities.judecatori_hot('Armando', 'Anika');
end;
/
---r)
select utilities.numar_jafuri_hot('Armando', 'Anika') "Numar jafuri"
from dual;

---s)
begin
    utilities.banci_jefuite_hot('Armando', 'Anika');
end;
/

```

PL/SQL procedure successfully completed.

DB final project 1st year

Banca 1: Banque CPH
Banca 2: Caixabank
Banca 3: Mizuho Bank

- t) Afişarea tuturor oraşelor şi ţărilor în care există bănci jefuite de un anumit hot

SQL Worksheet History

Worksheet Query Builder

```
utilities.banci_jefuite_hot('Armando', 'Anika'),  
end;  
/  
  
select count(distinct id_banca) "Numar banchi jefuite"  
from hot_jaf  
join hot_de_banci using (id_hot)  
join jaf using (id_jaf)  
where nume_hot = 'Armando'  
group by id_hot;  
  
---t)  
begin  
    utilities.orase_tari_jefuite_hot('Armando', 'Anika');  
end;  
/
```

Script Output x | Task completed in 0.048 seconds

PL/SQL procedure successfully completed.

Dbms Output x

+ Buffer Size: 20000 |

DB final project 1st year x

```
Locatie banca 1: Saint-Gilles, Belgia  
Locatie banca 2: Osaka, Japonia  
Locatie banca 3: Madrid, Spania
```

u) Afisarea detaliilor martorilor la un anumit jaf

The screenshot shows the Oracle SQL Developer interface. At the top, there's a toolbar with various icons. Below it is a tab bar with 'Worksheet' and 'Query Builder' selected. The main workspace contains the following PL/SQL code:

```
join hot_de_banci using (id_hot)
join jaf using (id_jaf)
where nume_hot = 'Armando'
group by id_hot;

---t)
begin
    utilities.orase_tari_jefuite_hot('Armando', 'Anika');
end;
/

---u)
begin
    utilities.martori('Commerzbank', '15-JUL-20');
end;
/
```

Below the code, there's a status bar with tabs for 'Script Output' and 'Query Result'. The 'Query Result' tab is active, showing the message: 'Task completed in 0.049 seconds'. The output pane at the bottom displays the message: 'PL/SQL procedure successfully completed.'.

At the bottom, there's another window titled 'Dbms Output' with a buffer size of 20000. It shows the results of the procedure execution:

```
Martor 1: Wu Viktoria, 34 ani
Martor 2: Mendez Erik, 49 ani
```

v) Afisarea detaliilor clientilor unei anumite banci

The screenshot shows the Oracle SQL Developer interface. The top window is a 'Worksheet' tab containing PL/SQL code. The code defines three procedures: 'orase_tari_jefuite_hot', 'martori', and 'clienti'. The 'clienti' procedure is highlighted with a blue selection bar. The bottom window is the 'Script Output' tab, which displays the message 'PL/SQL procedure successfully completed.' The 'Dbms Output' tab shows the result of the 'clienti' procedure, which is 'Client 1: Baaiman Nadine, 47 ani'.

```
begin
    utilities.orase_tari_jefuite_hot('Armando', 'Anika');
end;
/
---u)
begin
    utilities.martori('Commerzbank', '15-JUL-20');
end;
/
---v)
begin
    utilities.clienti('Commerzbank');
end;
/

```

PL/SQL procedure successfully completed.

Dbms Output

DB final project 1st year

Client 1: Baaiman Nadine, 47 ani