**Sisteme şi Gestiunea Bazelor de Date**

**Proiect**

**Geamănu Elena**

**Bucureşti**

**2023**

Cuprins

Contents

[1. Descrierea domeniului ales 3](#_Toc124124220)

[2. Schema conceptuală a bazei de date 4](#_Toc124124221)

[3. Crearea tabelelor si adaugarea datelor 5](#_Toc124124222)

[3.1. Creare tabel facultate 5](#_Toc124124223)

[3.2. Creare tabel angajat 6](#_Toc124124224)

[3.3. Creare tabel departament 7](#_Toc124124225)

[3.4. Creare tabel student 8](#_Toc124124226)

[3.5. Creare tabel inregistrare student 9](#_Toc124124227)

[4. Blocuri PL/SQL conţinând structuri de control variate 11](#_Toc124124228)

[5. Utilizarea cursorilor şi a excepţiilor în cadrul blocurilor PL/SQL 14](#_Toc124124229)

[6. Funcţii şi proceduri 18](#_Toc124124230)

[6.1. Funcţii 18](#_Toc124124231)

[6.2. Proceduri 24](#_Toc124124232)

# Descrierea domeniului ales

Modelul de date gestionează informaţii despre facultăţi din Bucureşti: ce departmente şi programe de studii au. Sunt incluse informaţii legate de angajaţii fiecărei facultăţi şi de studenţi.

Acest model de date este util în vederea repartizărilor studenţilor în facultăţi, repartizarea acestora pe departamente, ţinerea evidenţei studenţilor înscrişi la mai multe facultăţi şi departamente.

O facultate poate avea mai multe departamente. Fiecare facultate are angajaţii proprii. Prin angajat, înţeleg următoarele joburi posibile: profesor, secretară, agent de pază.

Un student poate fii înscris simultan la mai multe facultăţi, ceea ce implică că poate fii înscris simultan la mai multe departamente şi programe de studii. Modelul de date include toate înregistrările unui student la facultăţile din Bucureşti.

Aceasta bază de date este unică deoarece gestionează facultăţile din cadrul mai multor universităţi din Bucureşti.

Modelul de date respectă următoarele restricţii de funcţionare:

* O facultate trebuie să aibă minim un angajat.
* Un angajat lucrează doar la o singură facultate.
* Prin program de studii înteleg: licenţă, masterat, doctorat.
* Un departament există doar la o facultate
* Un student poate fi înscris la mai multe facultăţi si departamente

# Schema conceptuală a bazei de date

1-n

|  |
| --- |
| **Departament** |
| Cod\_departament (PK) |
| Nume\_departament |
| Program\_studii |
| Cod\_Facultate (FK) |

|  |
| --- |
| **Facultate** |
| Cod\_facultate (PK) |
| Nume\_facultate |
| Data\_înfiinţării |

1-n

Entitate de legatura (surogat)

1-n

|  |
| --- |
| **Angajat** |
| Cod\_angajat (PK) |
| Nume\_angajat |
| Prenume\_angajat |
| Functie\_angajat |
| Salariu\_angajat |
| Cod\_Facultate (FK) |

|  |
| --- |
| **Inregistrare\_student** |
| Cod\_inregistrare (PK) |
| An |
| Data\_inscrierii |
| Cod\_student (FK) |
| Cod\_departament (FK) |

1-n

|  |
| --- |
| **Student** |
| Cod\_student (PK) |
| Nume\_student |
| Prenume\_student |
| Vârstă\_student |
| Cod\_departament (FK) |

# Crearea tabelelor si adaugarea datelor

# Creare tabel facultate

create table facultate (

Cod\_facultate number(2) constraint pk\_fac primary key,

Nume\_facultate character(30),

Data\_infiintarii date);

***Inserare date***

insert into facultate values(20,'Filozofie', to\_date('16/07/1864', 'DD/MM/YYYY'));

insert into facultate values(21,'Fizica', to\_date('22/06/1962', 'DD/MM/YYYY'));

insert into facultate values(22,'Stiinte Politice', to\_date('03/12/1991', 'DD/MM/YYYY'));

insert into facultate values(23,'Drept', , to\_date('27/07/1859', 'DD/MM/YYYY'));

insert into facultate values(24,'Matematica Informatica', to\_date('17/06/1866', 'DD/MM/YYYY'));

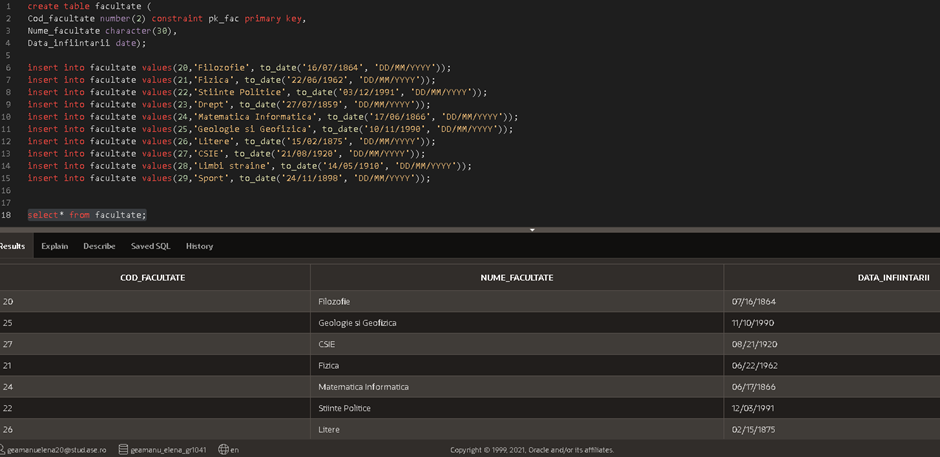
insert into facultate values(25,'Geologie si Geofizica', to\_date('10/11/1990', 'DD/MM/YYYY'));

insert into facultate values(26,'Litere', to\_date('15/02/1875', 'DD/MM/YYYY'));

insert into facultate values(27,'CSIE', to\_date('21/08/1920', 'DD/MM/YYYY'));

insert into facultate values(28,'Limbi straine', to\_date('14/05/1910', 'DD/MM/YYYY'));

insert into facultate values(29,'Sport', to\_date('24/11/1898', 'DD/MM/YYYY'));



# Creare tabel angajat

create table angajat\_ge (

Cod\_angajat number(3) constraint pk\_angge primary key,

Nume\_angajat character(25),

Prenume\_angajat character(25),

Functie\_angajat character(25),

Salariu\_angajat number(5),

Cod\_facultate number(2),

foreign key(Cod\_facultate) references facultate(Cod\_facultate));

***Inserare date***

insert into angajat\_ge values(100,'Popescu','Alexandru','Profesor',5000,21);

insert into angajat\_ge values(101,'Mirea','Andreea','Secretara',4500,28);

insert into angajat\_ge values(102,'Hentes','Stefan','Agent de Paza',6000,25);

insert into angajat\_ge values(103,'Mocanescu','Anca','Profesor',6000,26);

insert into angajat\_ge values(104,'Baesu','Paul','Profesor',3900,20);

insert into angajat\_ge values(105,'Dragan','Andrei','Agent de Paza',4600,23);

insert into angajat\_ge values(106,'Florea','Diana','Secretara',5100,27);

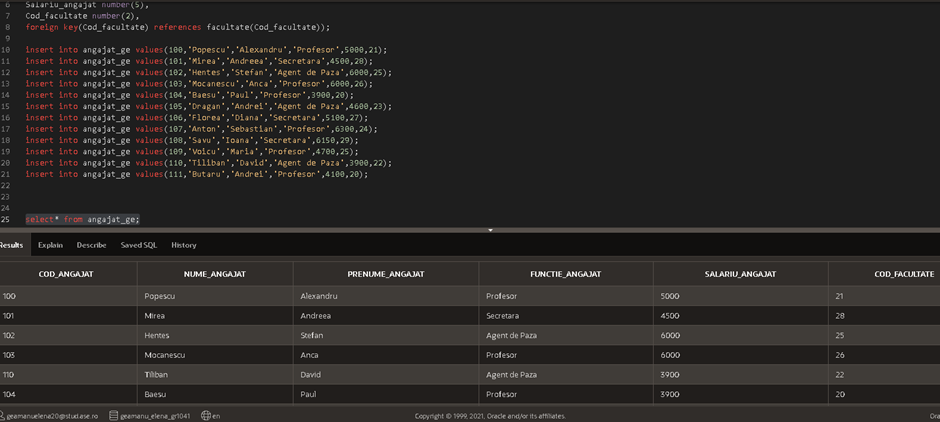
insert into angajat\_ge values(107,'Anton','Sebastian','Profesor',6300,24);

insert into angajat\_ge values(108,'Savu','Ioana','Secretara',6150,29);

insert into angajat\_ge values(109,'Voicu','Maria','Profesor',4700,25);

insert into angajat\_ge values(110,'Tiliban','David','Agent de Paza',3900,22);

insert into angajat\_ge values(111,'Butaru','Andrei','Profesor',4100,20);



# Creare tabel departament

create table departament (

Cod\_departament number(2) constraint pk\_dept primary key,

Nume\_departament character(25),

Program\_studii character(20),

Cod\_facultate number(2),

foreign key(Cod\_facultate) references facultate(Cod\_facultate));

***Inserare date***

insert into departament values(50,'Matematica','Licenta',24);

insert into departament values(51,'Informatica','Licenta',24);

insert into departament values(52,'Biostatistica','Master',24);

insert into departament values(53,'Drept Penal','Licenta',23);

insert into departament values(54,'Filosofie Teoretica','Licenta',20);

insert into departament values(55,'Filosofie Analitica','Doctorat',20);

insert into departament values(56,'Fizica Teoretica','Licenta',21);

insert into departament values(57,'Biofizica','Licenta',21);

insert into departament values(58,'Geologie','Licenta',25);

insert into departament values(59,'Inginerie Geologica','Master',25);

insert into departament values(60,'Relatii Internationale','Licenta',22);

insert into departament values(61,'Sociologie Aplicata','Master',22);

insert into departament values(62,'Studii Europene','Doctorat',26);

insert into departament values(63,'Relatii Publice','Licenta',26);

insert into departament values(64,'Cibernetica','Licenta',27);

insert into departament values(65,'Statistica','Master',27);

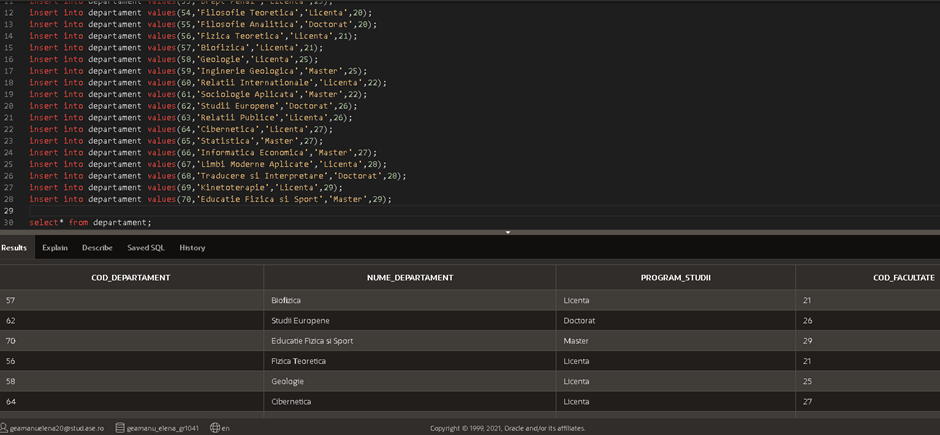
insert into departament values(66,'Informatica Economica','Master',27);

insert into departament values(67,'Limbi Moderne Aplicate','Licenta',28);

insert into departament values(68,'Traducere si Interpretare','Doctorat',28);

insert into departament values(69,'Kinetoterapie','Licenta',29);

insert into departament values(70,'Educatie Fizica si Sport','Master',29);



# Creare tabel student

create table student (

Cod\_student number(5) constraint pk\_student primary key,

Nume\_student character(20),

Prenume\_student character(20),

Varsta number(2) default 18,

Cod\_departament number(2),

foreign key(Cod\_departament) references departament(Cod\_departament));

***Inserare date***

insert into student values(200,'Vasilescu','Mihai',18,51);

insert into student values(201,'Magheran','Ioana',20,56);

insert into student values(202,'Hostinar','Radu',19,67);

insert into student values(203,'Petre','Andrei',23,63);

insert into student values(204,'Iancu','Elena',20,54);

insert into student values(205,'Costache','Adrian',18,68);

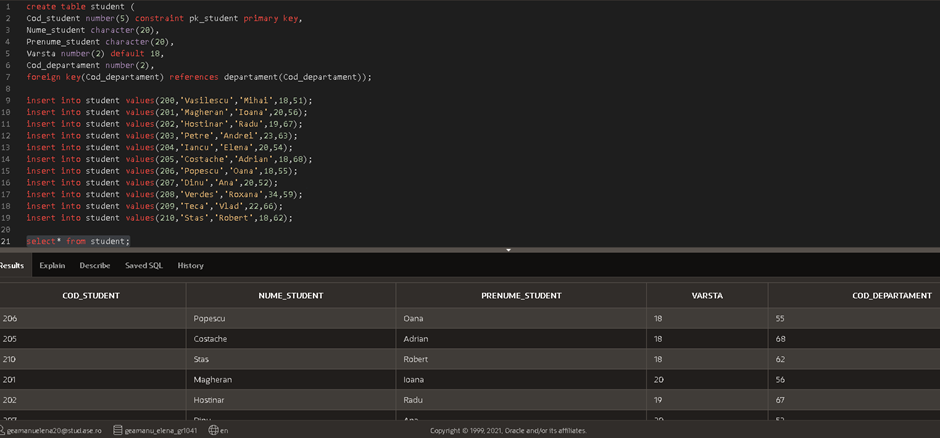
insert into student values(206,'Popescu','Oana',18,55);

insert into student values(207,'Dinu','Ana',20,52);

insert into student values(208,'Verdes','Roxana',34,59);

insert into student values(209,'Teca','Vlad',22,66);

insert into student values(210,'Stas','Robert',18,62);



# Creare tabel inregistrare student

create table inregistrare\_student (

Cod\_inregistare number(6) constraint pk\_inreg\_student primary key,

An number(1) default 1,

Data\_inscrierii date default sysdate,

Cod\_student number(5),

Cod\_departament number(2),

foreign key(Cod\_student) references student(Cod\_student),

foreign key(Cod\_departament) references departament(Cod\_departament));

***Inserare date***

insert into inregistrare\_student values(500,1,to\_date('03/07/2021', 'DD/MM/YYYY'),200,51);

insert into inregistrare\_student values(501,3,to\_date('11/07/2018', 'DD/MM/YYYY'),201,56);

insert into inregistrare\_student values(502,2, to\_date('05/07/2020', 'DD/MM/YYYY'),202,67);

insert into inregistrare\_student values(503,3,to\_date('09/07/2017', 'DD/MM/YYYY'),203,63);

insert into inregistrare\_student values(504,2 ,to\_date('03/07/2020', 'DD/MM/YYYY'),203,64);

insert into inregistrare\_student values(505,2,to\_date('05/07/2020', 'DD/MM/YYYY'),204,54);

insert into inregistrare\_student values(506,1,to\_date('06/07/2021', 'DD/MM/YYYY'),205,68);

insert into inregistrare\_student values(507,1,to\_date('05/07/2021', 'DD/MM/YYYY'),206,55);

insert into inregistrare\_student values(508,1,to\_date('05/07/2021', 'DD/MM/YYYY'),206,70);

insert into inregistrare\_student values(509,2,to\_date('04/07/2020', 'DD/MM/YYYY'),207,52);

insert into inregistrare\_student values(510,3,to\_date('19/07/2013', 'DD/MM/YYYY'),208,59);

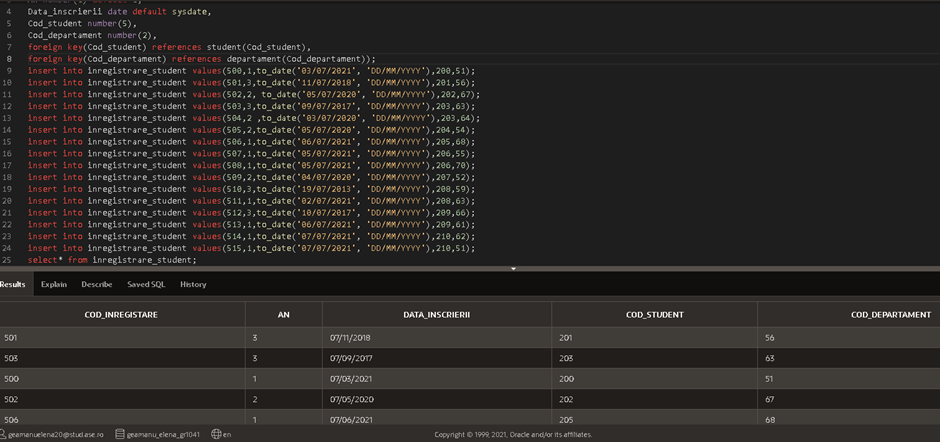
insert into inregistrare\_student values(511,1,to\_date('02/07/2021', 'DD/MM/YYYY'),208,63);

insert into inregistrare\_student values(512,3,to\_date('10/07/2017', 'DD/MM/YYYY'),209,66);

insert into inregistrare\_student values(513,1,to\_date('06/07/2021', 'DD/MM/YYYY'),209,61);

insert into inregistrare\_student values(514,1,to\_date('07/07/2021', 'DD/MM/YYYY'),210,62);

insert into inregistrare\_student values(515,1,to\_date('07/07/2021', 'DD/MM/YYYY'),210,51);



# Blocuri PL/SQL conţinând structuri de control variate

***Exemplul 1***

Sa se afiseze functia si salariul angajatului Popescu si sa i se mareasca salariul in functie de postul pe care il ocupa astfel: cu 10% daca este agent de paza, cu 15% daca este secretara si respectiv cu 20% daca este profesor.

declare

v\_salariu angajat\_ge.salariu\_angajat%type;

v\_nume angajat\_ge.nume\_angajat%type := 'Popescu';

v\_functie angajat\_ge.functie\_angajat%type;

begin

select salariu\_angajat, functie\_angajat into v\_salariu, v\_functie

from angajat\_ge

where nume\_angajat=v\_nume;

dbms\_output.put\_line ('Popescu are functia de '||v\_functie||'iar salariul initial este de: '||v\_salariu||' lei');

case

when v\_functie='Agent de Paza' then v\_salariu:=v\_salariu+v\_salariu/10;

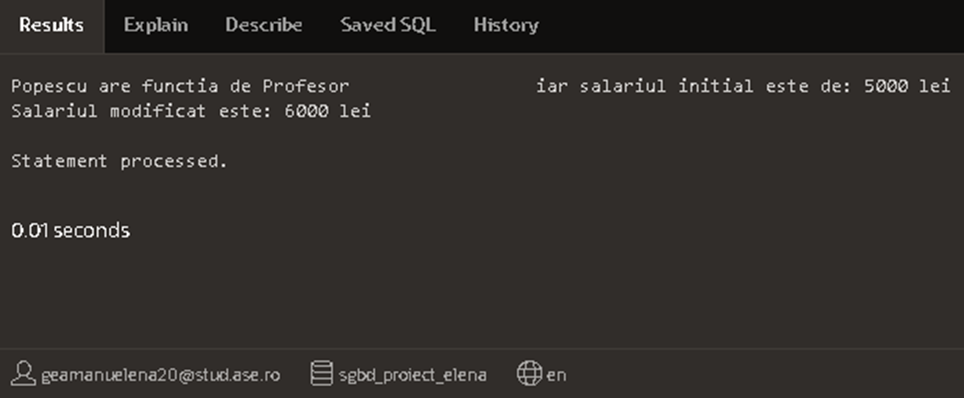
when v\_functie='Secretara' then v\_salariu:=v\_salariu+v\_salariu\*3/20;

else v\_salariu:=v\_salariu+v\_salariu/5;

end case;

dbms\_output.put\_line ('Salariul modificat este: '||v\_salariu||' lei');

end;



***Exemplul 2***

Sa se afiseze in ordine departamentele cu codurile in intervalul 50-55 care au ca program de studii Licenta.

declare

v\_studii departament.program\_studii%type;

v\_nume\_dep departament.nume\_departament%type;

i number(4):=50;

begin

loop

select nume\_departament, program\_studii into v\_nume\_dep, v\_studii from departament where cod\_departament=i;

if v\_studii='Licenta' then

dbms\_output.put\_line(i||' '||v\_nume\_dep);

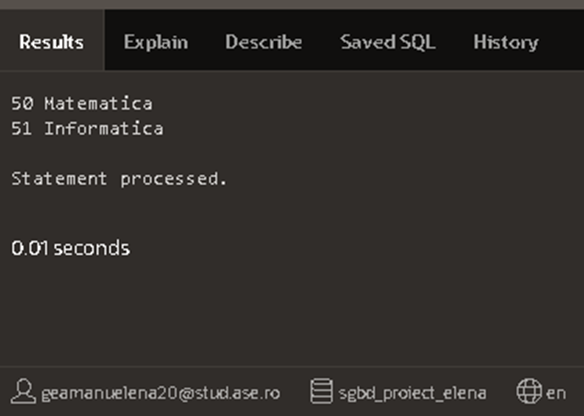
end if;

i:=i+1;

exit when v\_studii!='Licenta' or i>55;

end loop;

end;



***Exemplul 3***

Sa se afiseze la ce program de studii este inscris studentul Costache si cati ani dureaza acest program de studii.

declare

v\_durata number(4);

v\_nume student.nume\_student%type := 'Costache';

v\_studii departament.program\_studii%type;

begin

select program\_studii into v\_studii

from departament d,student s

where d.cod\_departament=s.cod\_departament

and nume\_student=v\_nume;

if v\_studii='Licenta' then

v\_durata:=4;

elsif v\_studii='Master' then

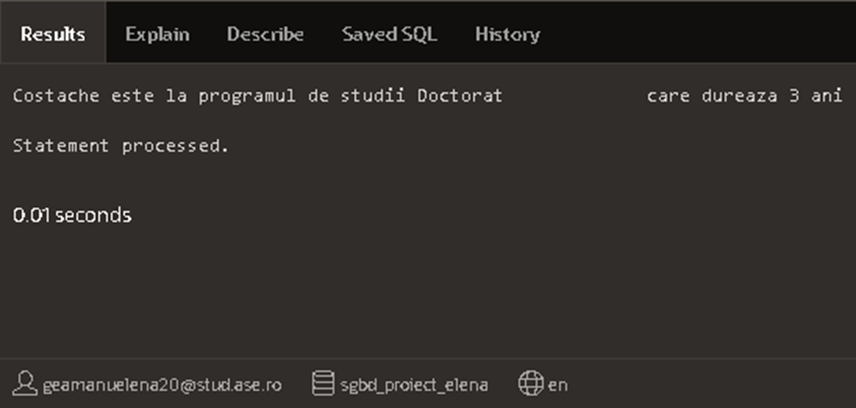
v\_durata:=2;

else v\_durata:=3;

end if;

dbms\_output.put\_line ('Costache este la programul de studii '||v\_studii||' care dureaza '||v\_durata||' ani');

end;



5. Utilizarea cursorilor şi a excepţiilor în cadrul blocurilor PL/SQL

***Exemplul 1***

Sa se afiseze studenţii cu vârsta mai mare de 23 de ani de la Facultatea de Informatică. Este tratată excepţia no\_data\_found:

declare

v\_nume student.nume\_student%type;

begin

select nume\_student into v\_nume

from student

where varsta>23 and cod\_departament=51;

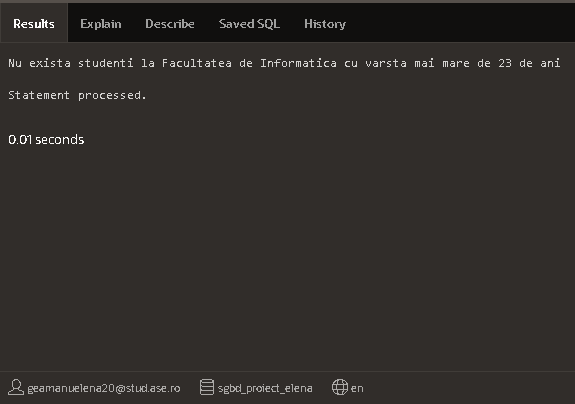
dbms\_output.put\_line(v\_nume||' are varsta mai mare decat 23 si este la Facultatea de Informatica');

exception

when no\_data\_found then

dbms\_output.put\_line('Nu exista studenti la Facultatea de Informatica cu varsta mai mare de 23 de ani');

end;



***Exemplul 2***

Afişează numele studentului cu vârsta sub 19 ani. Este tratată excepţia too\_many\_rows:

declare

v\_nume student.nume\_student%type;

begin

select nume\_student into v\_nume

from student

where varsta<19;

dbms\_output.put\_line(v\_nume||' are varsta mai mica decat 19');

exception

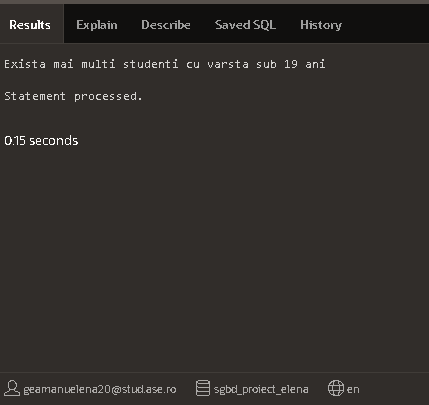
when too\_many\_rows then

dbms\_output.put\_line('Exista mai multi studenti cu varsta sub 19 ani');

when others then

dbms\_output.put\_line('Alta eroare');

end;



***Exemplul 3***

Să se afişeze toate facultăţile şi data înfiinţări acestora. Cursor explicit:

declare

cursor c is select nume\_facultate, data\_infiintarii from facultate;

v\_nume facultate.nume\_facultate%type;

v\_data facultate.data\_infiintarii%type;

begin

open c;

loop

fetch c into v\_nume, v\_data;

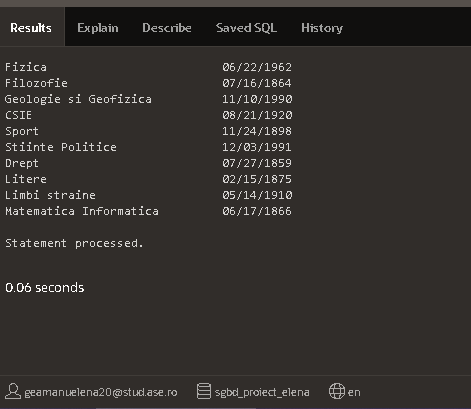
exit when c%notfound;

dbms\_output.put\_line(v\_nume||' '||v\_data);

end loop;

close c;

end;



***Exemplul 4***

Să se afişeze numele, prenumele şi funcţiile angajaţilor din facultate.

declare

cursor c is select nume\_angajat,prenume\_angajat,functie\_angajat from angajat;

v\_nume angajat.nume\_angajat%type;

v\_prenume angajat.prenume\_angajat%type;

v\_functie angajat.functie\_angajat%type;

begin

open c;

loop

fetch c into v\_nume, v\_prenume, v\_functie;

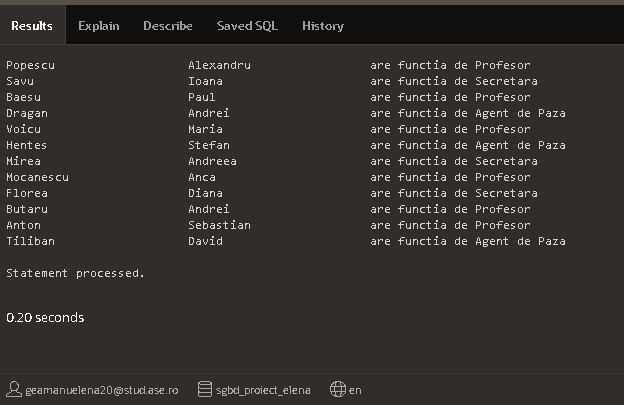
exit when c%notfound;

dbms\_output.put\_line(v\_nume||' '||v\_prenume||' are functia de '||v\_functie);

end loop;

close c;

end;



# 6. Funcţii şi proceduri

# 6.1. Funcţii

1.Creaţi funcţia afisare\_salariu care să returneze salariul unui angajat al cărui nume este specificat. Se tratează toate excepţiile ce pot fi generate.

create or replace function afisare\_salariu

(v\_nume angajat\_ge.nume\_angajat%type)

return number is

v\_sal angajat\_ge.salariu\_angajat%type;

begin

select salariu\_angajat into v\_sal

from angajat\_ge

where nume\_angajat=v\_nume;

return v\_sal;

exception

when no\_data\_found then

dbms\_output.put\_line('Nu exista angajati cu acest nume');

return null;

when too\_many\_rows then

dbms\_output.put\_line('Exista mai multi angajati cu acelasi nume');

return null;

end;

--primul apel al functiei

begin

dbms\_output.put\_line(afisare\_salariu('Popescu'));

end;

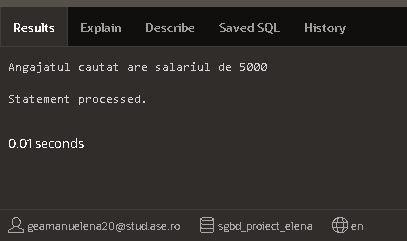
--al doilea apel al functiei

begin

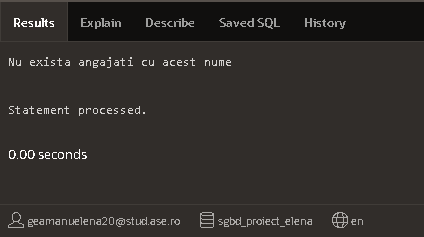
dbms\_output.put\_line(afisare\_salariu('Avram'));

end;

--primul apel al functiei



--al doilea apel al functiei



2. Creaţi funcţia varsta\_minima care să returneze numele studentului cu vârsta cea mai mică al carui cod de departament este specificat. Se tratează toate excepţiile ce pot fi generate.

create or replace function varsta\_minima

(v\_cod\_dep student.cod\_departament%type, min\_varsta student.varsta%type)

return varchar is

v\_nume student.nume\_student%type;

begin

select nume\_student into v\_nume

from student

where cod\_departament=v\_cod\_dep

and varsta=min\_varsta;

return v\_nume;

exception

when no\_data\_found then

dbms\_output.put\_line('Nu exista codul departamentului dat');

return null;

when too\_many\_rows then

dbms\_output.put\_line('Exista mai multi studenti cu aceeasi varsta minima');

return null;

end;

--primul apel al functiei

declare

min\_varsta student.varsta%type;

begin

select min(varsta) into min\_varsta

from student

where cod\_departament='56';

dbms\_output.put\_line('Numele studentului cu varsta minima de la departamentul dat este '||varsta\_minima('56',min\_varsta));

end;

--al doilea apel al functiei

declare

min\_varsta student.varsta%type;

begin

select min(varsta) into min\_varsta

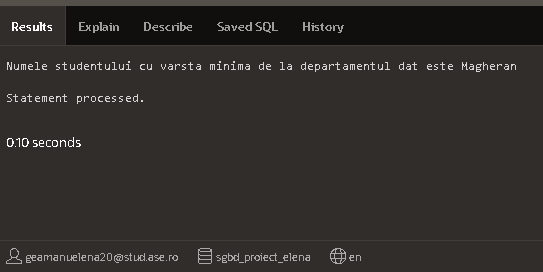
from student

where cod\_departament='11';

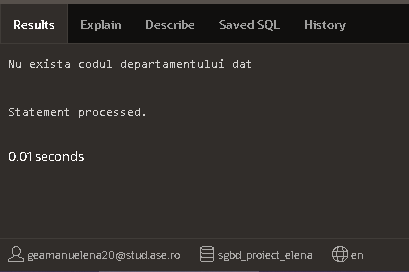
dbms\_output.put\_line(varsta\_minima('11',min\_varsta));

end;

--primul apel al functiei



--al doilea apel al functiei



3. Creaţi funcţia salariu\_maxim care să returneze numele angajatului cu cel mai mare salariu al cărui funcţie este specificată. Se tratează toate excepţiile ce pot fi generate.

create or replace function salariu\_maxim

(v\_functie angajat\_ge.functie\_angajat%type, max\_sal angajat\_ge.salariu\_angajat%type)

return varchar is

v\_nume angajat\_ge.nume\_angajat%type;

begin

select nume\_angajat into v\_nume

from angajat\_ge

where functie\_angajat=v\_functie

and salariu\_angajat=max\_sal;

return v\_nume;

exception

when no\_data\_found then

dbms\_output.put\_line('Nu exista functia cautata');

return null;

when too\_many\_rows then

dbms\_output.put\_line('Exista mai multi angajati cu acelasi salariu maxim pentru functia data');

return null;

end;

--primul apel al functiei

declare

max\_sal angajat\_ge.salariu\_angajat%type;

begin

select max(salariu\_angajat) into max\_sal

from angajat\_ge

where functie\_angajat='Agent de Paza';

dbms\_output.put\_line('Numele angajatului cu salariul cel mai mare pentru functia data este '||salariu\_maxim ('Agent de Paza', max\_sal));

end;

--al doilea apel al functiei

declare

max\_sal angajat\_ge.salariu\_angajat%type;

begin

select max(salariu\_angajat) into max\_sal

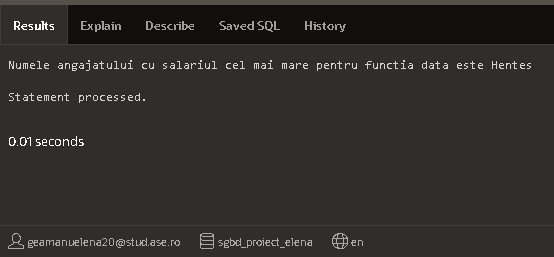
from angajat\_ge

where functie\_angajat='Asistent';

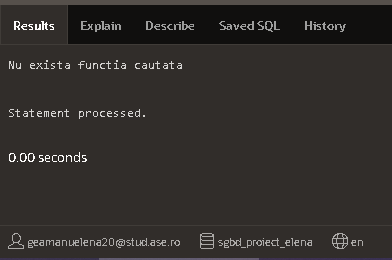
dbms\_output.put\_line(salariu\_maxim ('Asistent',max\_sal));

end;

--primul apel al functiei



--al doilea apel al functiei



# 6.2. Proceduri

1. Procedura date\_student primeşte ca parametru de tip IN codul unui student şi returnează prin parametrii de tip OUT numele şi vârsta acestuia.

create or replace procedure date\_student

(v\_cod in student.cod\_student%type,

v\_nume out student.nume\_student%type,

v\_varsta out student.varsta%type)

is

begin

select nume\_student, varsta into v\_nume, v\_varsta

from student

where cod\_student=v\_cod;

dbms\_output.put\_line(' studentul '||v\_nume||' are varsta de '||v\_varsta||' ani');

end;

declare

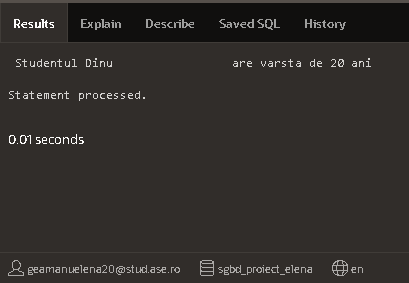
v\_nume student.nume\_student%type;

v\_varsta student.varsta%type;

begin

date\_student(207, v\_nume, v\_varsta);

end;



2. Procedura cauta\_facultate primeşte ca parametru de tip IN numele unui student şi returnează prin parametrii de tip OUT numele facultăţii la care este înscris.

create or replace procedure cauta\_facultate

(v\_nume in student.nume\_student%type,

v\_facult out facultate.nume\_facultate%type)

is

begin

select nume\_facultate into v\_facult

from student, departament, facultate

where nume\_student = v\_nume

and student.cod\_departament = departament.cod\_departament

and departament.cod\_facultate = facultate.cod\_facultate;

dbms\_output.put\_line('Studentul '||v\_nume||' este inscris la Facultatea de '||v\_facult);

end;

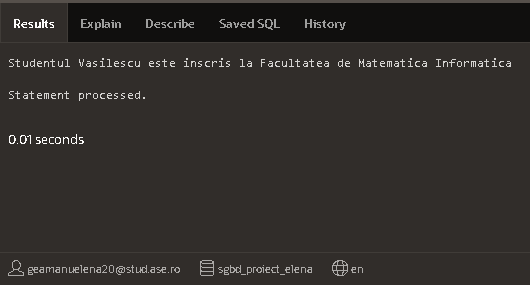
declare

v\_facult facultate.nume\_facultate%type;

begin

cauta\_facultate('Vasilescu', v\_facult);

end;



3. Procedura actualizeaza\_varsta primeşte ca parametru de tip IN numele unui student şi actualizează vârsta acestuia.

create or replace procedure actualizeaza\_varsta

(v\_nume in student.nume\_student%type)

is

begin

update student

set varsta = varsta + 1

where nume\_student = v\_nume;

end;

begin

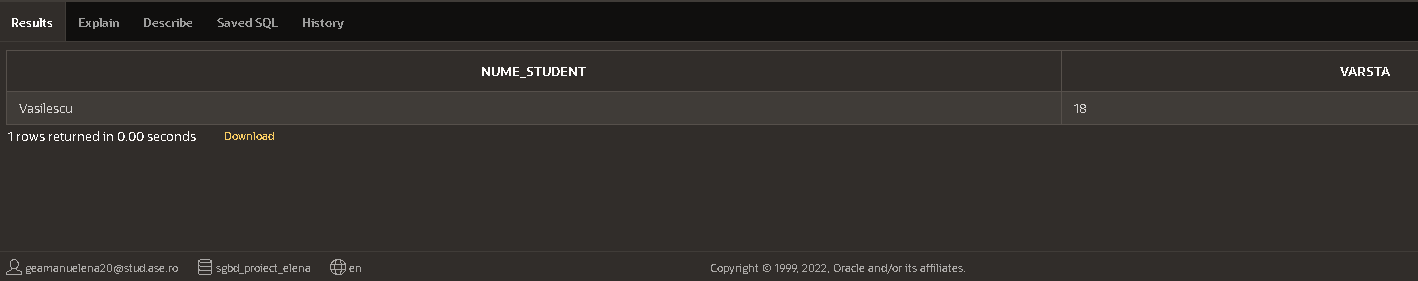
actualizeaza\_varsta('Vasilescu');

end;

select nume\_student,varsta

from student

where nume\_student='Vasilescu';

--inainte de a fi actualizata varsta

--dupa ce a fost actualizata varsta 