PROIECT BAZE DE DATE

Administrarea căminelor studențești

**Profesor coordonator:** Iuliana Botha

**Student:** Palade Gabriel

**Grupa:** 1053

**Seria:** D

# Prezentarea temei economice: Administrarea căminelor studențești

Baza de date are rolul de a ține evidența studenților repartizați pe o anumită perioadă. Astfel, oferă informații precum: datele studenților (nume, prenume, email, număr de telefon, localitate etc.), tipul și denumirea căminelor în care aceștia sunt cazați, plata regiei de cămin (data plății, suma încasată și suma datorată). Baza de date conține 6 tabele: ADMINISTRATIE, CAMIN, CAMERA, STUDENT, REGIE\_CAMIN ȘI ISTORIC\_REGIE.

ADMINISTRATIE este o tabelă individuală, cu atributul id\_admin cheie primară. Un administrator întreține mai multe clădiri de cămin. Tabela mai conține, ca atribute, datele personale ale unui administrator: nume\_admin, prenume\_admin, nr\_telefon, salariu.

CAMIN este o tabelă care stochează căminele, numărul de camere pe care îl deține fiecare și adresa la care se găsește. Este important să memorăm și capacitatea unei camere, întrucât un cămin are același număr de locuri per cameră în toată clădirea. Astfel, atributele tabelei sunt: nr\_camin (cheie primară), denumire\_camin, adresa, nr\_camere, capacitate\_camera, id\_admin (cheie externă - tabela ADMINISTRATIE).

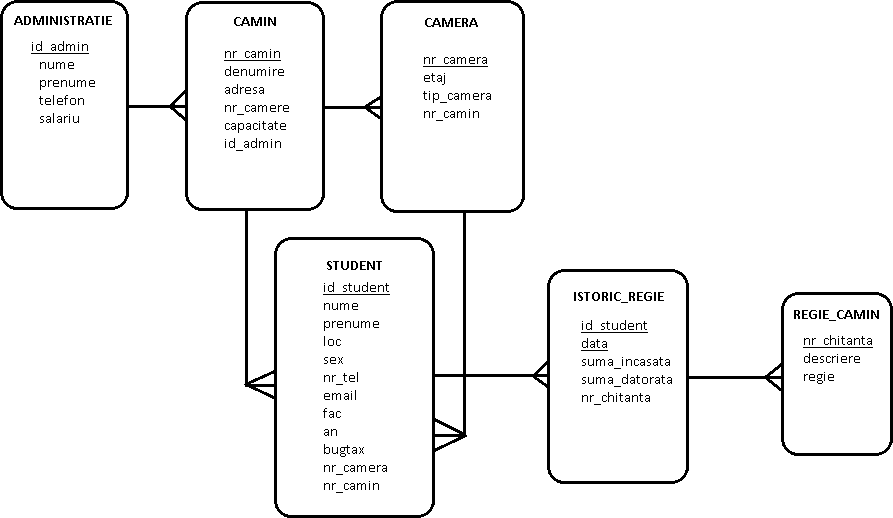
CAMERA este o tabelă care stochează câteva date despre camera în care sunt cazați studenții (2,3,4 sau 5 studenți - dar pot fi și mai mulți dacă este vorba de o academie militară). Astfel, atributele sunt: nr\_camera (cheie primară), etaj, tip\_camera, nr\_camin (cheie externă - tabela CAMIN).

STUDENT este o tabelă în care sunt înregistrați studenții. Un student este repartizat după media anului precedent (sau după media de admitere dacă este anul I). Denumirea facultății din care fac parte, precum și căminul, camera în care sunt repartizați sunt, de asemenea, date esențiale. Astfel, atributele tabelei sunt: id\_stud (cheie primară), nume\_stud, prenume\_stud, localitate, sex, telefon, email, den\_fac, an\_studiu, bugtax, nr\_camin (cheie externa - tabela CAMIN), nr\_camera (cheie externa - tabela CAMERA).

REGIE\_CAMIN este o tabelă individuală, cu atributul nr\_chitanta cheie primară. Este o tabelă simplă, ce reține regia căminului pe o anumită lună și descrierea ei (observație: în descriere se memorează, de exemplu, luna + id-ul caminului).

ISTORIC\_REGIE este o tabelă de legătură, având o cheie primară formată din două atribute (id\_student și data\_plata), care memorează situația plății regiei de către un anumit student. Unii studenți nu plătesc totul la timp și, în acest caz, se înregistrează o sumă datorată. Data plății este o informație esențială, întrucât se pot evidenția eventuale penalizări. Atributele tabelei sunt: id\_stud (cheie externa - tabela STUDENT), data\_plata, suma\_incasata, suma\_datorata, nr\_chitanta (cheie externa - tabela REGIE\_CAMIN).

# Schema conceptuală. Legături. Restricții



Legături 1:M

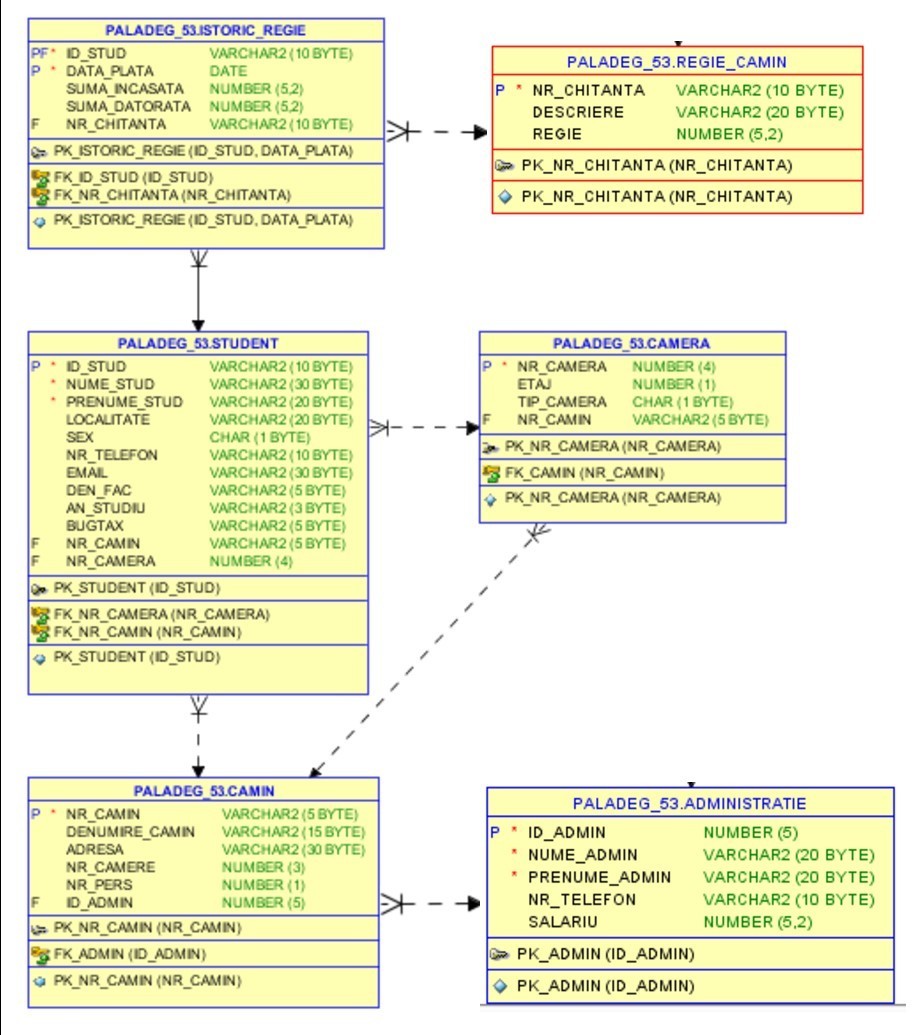
Un cămin este gestionat de către un singur administrator. Un administrator gestionează unul sau mai multe cămine.

Un student este cazat într-un singur cămin. Un cămin are mai mulți studenți. O cameră aparține unui singur cămin. Un cămin are mai multe camere.

Un student este cazat într-o singură cameră. O cameră are mai mulți studenți.

Legături M:M

Un student plătește mai multe regii. O regie este plătită de mai mulți studenți. De aceea, pentru a reține un istoric al regiilor de cămin, se crează tabela intermediară ISTORIC\_REGIE.



**Restricții**

Tabela ADMINISTRATIE

id\_admin PRIMARY KEY;

câmpul nume\_admin trebuie să fie diferit de NULL; câmpul prenume\_admin trebuie să fie diferit de NULL.

Tabela CAMIN

nr\_camin PRIMARY KEY; id\_admin FOREIGN KEY.

Tabela CAMERA

nr\_camera PRIMARY KEY; nr\_camin FOREIGN KEY.

Tabela STUDENT

id\_stud PRIMARY KEY;

câmpul nume\_stud trebuie să fie diferit de NULL; câmpul prenume\_stud trebuie să fie diferit de NULL; câmpul email trebuie să fie sub forma: %@%.%;

câmpul bugtax trebuie să ia una din valorile: Buget, Taxa;

nr\_camin FOREIGN KEY; nr\_camera FOREIGN KEY.

Tabela REGIE\_CAMIN

nr\_chitanta PRIMARY KEY

Tabela ISTORIC\_REGIE

id\_stud, data\_plata PRIMARY KEY; id\_stud FOREIGN KEY; nr\_chitanta FOREIGN KEY.

CREATE

CREATE TABLE ADMINISTRATIE(

id\_admin NUMBER(5) CONSTRAINT PK\_admin PRIMARY KEY,

nume\_admin VARCHAR2(20) CONSTRAINT NN\_nume\_admin CHECK (nume\_admin is not null),

prenume\_admin VARCHAR2(20) CONSTRAINT NN\_prenume\_admin CHECK (prenume\_admin is not null),

nr\_telefon VARCHAR2(10), salariu NUMBER (5,2)

);

CREATE TABLE CAMIN (

nr\_camin VARCHAR2(5) CONSTRAINT PK\_nr\_camin PRIMARY KEY, denumire\_camin VARCHAR2(15),

adresa VARCHAR2(45), nr\_camere NUMBER(3), nr\_pers NUMBER(1),

id\_admin NUMBER(5), CONSTRAINT FK\_admin FOREIGN KEY (id\_admin) REFERENCES administratie (id\_admin)

);

CREATE TABLE CAMERA(

nr\_camera NUMBER(4) CONSTRAINT PK\_nr\_camera PRIMARY KEY, etaj NUMBER(1),

tip\_camera CHAR(1),

nr\_camin VARCHAR2(5), CONSTRAINT FK\_camin FOREIGN KEY (nr\_camin) REFERENCES camin (nr\_camin)

);

CREATE TABLE REGIE\_CAMIN(

nr\_chitanta VARCHAR2(10) CONSTRAINT PK\_nr\_chitanta PRIMARY KEY, descriere VARCHAR2(100),

regie NUMBER(5,2)

);

CREATE TABLE STUDENT(

id\_stud VARCHAR2(10) CONSTRAINT PK\_STUDENT PRIMARY KEY,

nume\_stud VARCHAR2(30) CONSTRAINT NN\_nume\_st CHECK (nume\_stud is not null) ,

prenume\_stud VARCHAR2(20) CONSTRAINT NN\_prenume\_st CHECK (prenume\_stud is not null),

localitate VARCHAR2(20), sex CHAR(1),

nr\_telefon VARCHAR2(10),

email VARCHAR2(30) CONSTRAINT CK\_email CHECK (email like '%@%.%'), den\_fac VARCHAR2(5),

an\_studiu VARCHAR2(3),

bugtax VARCHAR2(5) CONSTRAINT CK\_bugtax CHECK(bugtax in ('Buget','Taxa')),

nr\_camin VARCHAR2(5), CONSTRAINT FK\_nr\_camin FOREIGN KEY(nr\_camin) REFERENCES camin(nr\_camin),

nr\_camera NUMBER(4), CONSTRAINT FK\_nr\_camera FOREIGN KEY(nr\_camera) REFERENCES camera(nr\_camera)

);

CREATE TABLE ISTORIC\_REGIE(

id\_stud VARCHAR2(10),

data\_plata DATE, suma\_incasata NUMBER(5,2), suma\_datorata NUMBER(5,2), nr\_chitanta VARCHAR(10),

CONSTRAINT PK\_istoric\_regie PRIMARY KEY (id\_stud, data\_plata),

CONSTRAINT FK\_nr\_chitanta FOREIGN KEY (nr\_chitanta) REFERENCES regie\_camin (nr\_chitanta),

CONSTRAINT FK\_id\_stud FOREIGN KEY (id\_stud) REFERENCES student (id\_stud)

);

## ALTER

ALTER TABLE STUDENT ADD CONSTRAINT UNIQUEKEY\_telefon UNIQUE(nr\_telefon);

ALTER TABLE STUDENT ADD medie\_repartizare NUMBER(3,2); ALTER TABLE CAMIN MODIFY adresa VARCHAR2(46);

ALTER TABLE CAMIN MODIFY DENUMIRE\_CAMIN VARCHAR2(50); ALTER TABLE ISTORIC\_REGIE MODIFY SUMA\_INCASATA NUMBER(7,2); ALTER TABLE ISTORIC\_REGIE MODIFY SUMA\_DATORATA NUMBER(7,2); ALTER TABLE ADMINISTRATIE MODIFY SALARIU NUMBER (7,2);

ALTER TABLE REGIE\_CAMIN MODIFY REGIE NUMBER (7,2);

## DROP

ALTER TABLE STUDENT DROP COLUMN sex; ALTER TABLE STUDENT ADD sex CHAR; ALTER TABLE STUDENT DROP COLUMN email;

ALTER TABLE ADMINISTRATIE DROP CONSTRAINT NN\_prenume\_adm CASCADE CONSTRAINT;

## INSERT

### ADMINISTRATIE

INSERT INTO ADMINISTRATIE(id\_admin,nume\_admin, prenume\_admin, nr\_telefon, salariu)

VALUES (1000, 'Costache','Georgel','0732541455', 1200);

INSERT INTO ADMINISTRATIE(id\_admin,nume\_admin, prenume\_admin, nr\_telefon, salariu) VALUES (1001, 'Neagu','Ionut','0732564455', 1500);

INSERT INTO ADMINISTRATIE(id\_admin,nume\_admin, prenume\_admin, nr\_telefon, salariu) VALUES (1002, 'Constantin','George','0732241455', 1100);

INSERT INTO ADMINISTRATIE(id\_admin,nume\_admin, prenume\_admin, nr\_telefon, salariu) VALUES (1003, 'Oprea','Andrei','0738946755', 2500);

INSERT INTO ADMINISTRATIE(id\_admin,nume\_admin, prenume\_admin, nr\_telefon, salariu) VALUES (1004, 'Farcas','Mihai','0732853455', 1800);

INSERT INTO ADMINISTRATIE(id\_admin,nume\_admin, prenume\_admin, nr\_telefon, salariu) VALUES (1005, 'Miron','Gabriel','0732241335', 1300);

INSERT INTO ADMINISTRATIE(id\_admin,nume\_admin, prenume\_admin, nr\_telefon, salariu) VALUES (1006, 'Costache','Ionut','0737689455', 1700);

### CAMIN

INSERT INTO CAMIN(nr\_camin, denumire\_camin, adresa, nr\_camere, nr\_pers, id\_admin) VALUES ('A1BV', 'Belvedere Vechi', 'Str. Cristian Pascal nr.27', 60, 3, 1000);

INSERT INTO CAMIN(nr\_camin, denumire\_camin, adresa, nr\_camere, nr\_pers, id\_admin) VALUES ('A2BV', 'Belvedere Vechi A2', 'Str. Cristian Pascal nr.27', 61, 3, 1000);

INSERT INTO CAMIN(nr\_camin, denumire\_camin, adresa, nr\_camere, nr\_pers, id\_admin) VALUES ('A3BV','Belvedere Vechi A3', 'Str. Cristian Pascal nr.27', 100, 3, 1002);

INSERT INTO CAMIN(nr\_camin, denumire\_camin, adresa, nr\_camere, nr\_pers, id\_admin) VALUES ('A4BV', 'Belvedere Vechi A4', 'Str. Cristian Pascal nr.27', 100, 3, 1002);

INSERT INTO CAMIN(nr\_camin, denumire\_camin, adresa, nr\_camere, nr\_pers, id\_admin) VALUES ('A6BN', 'Belvedere Nou A6','Str. Chibzuinței nr. 2', 160, 4, 1003);

INSERT INTO CAMIN(nr\_camin, denumire\_camin, adresa, nr\_camere, nr\_pers, id\_admin) VALUES ('A7BN', 'Belvedere Nou A7','Str. Chibzuinței nr. 2',180, 4, 1004);

INSERT INTO CAMIN(nr\_camin, denumire\_camin, adresa, nr\_camere, nr\_pers , id\_admin) VALUES ('A8BN', 'Belvedere Nou A8', 'Str. Chibzuinței nr. 2', 180, 4, 1005);

### CAMERA

INSERT INTO CAMERA(nr\_camera, etaj, tip\_camera, nr\_camin) VALUES (026, 0, 'M', 'A3BV');

INSERT INTO CAMERA(nr\_camera, etaj, tip\_camera, nr\_camin) VALUES (103, 1, 'F', 'A3BV'); INSERT INTO CAMERA (nr\_camera, etaj, tip\_camera, nr\_camin) VALUES (202, 2, 'M', 'A3BV'); INSERT INTO CAMERA(nr\_camera, etaj, tip\_camera, nr\_camin) VALUES (305, 3, 'F', 'A1BV'); INSERT INTO CAMERA(nr\_camera, etaj, tip\_camera, nr\_camin) VALUES (421, 4, 'M', 'A2BV');

INSERT INTO CAMERA(nr\_camera, etaj, tip\_camera, nr\_camin) VALUES (017, 0, 'F', 'A6BN'); INSERT INTO CAMERA(nr\_camera, etaj, tip\_camera, nr\_camin) VALUES (124, 1, 'F', 'A6BN');

### STUDENT

INSERT INTO STUDENT(id\_stud, nume\_stud, prenume\_stud, nr\_telefon, den\_fac, an\_studiu, bugtax, nr\_camin, nr\_camera, medie\_repartizare, sex)VALUES ('ICT2341', 'Ioanovici', 'Teodora','0732260599', 'CSIE', 'II','Buget','A3BV', 103, 8.33, 'F');

INSERT INTO STUDENT(id\_stud, nume\_stud, prenume\_stud, nr\_telefon, den\_fac, an\_studiu, bugtax, nr\_camin, nr\_camera, medie\_repartizare, sex)VALUES ('IPI8974', 'Ivasisin', 'Ionela', '0742564592', 'CSIE', 'III', 'Taxa', 'A3BV', 103, 8.10, 'F');

INSERT INTO STUDENT(id\_stud, nume\_stud, prenume\_stud, nr\_telefon, den\_fac, an\_studiu, bugtax, nr\_camin, nr\_camera, medie\_repartizare, sex)VALUES ('FD7830', 'Florete', 'Maria', '0738126580', 'CSIE', 'I', 'Buget', 'A3BV', 017, 9.40, 'F');

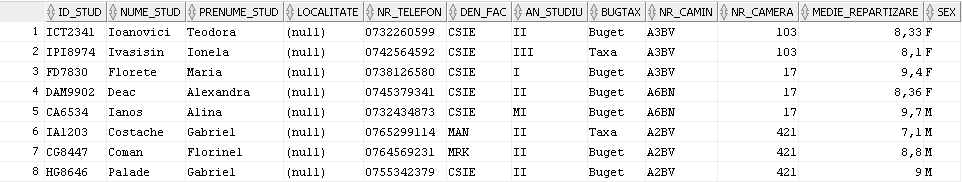
INSERT INTO STUDENT(id\_stud, nume\_stud, prenume\_stud, nr\_telefon, den\_fac, an\_studiu, bugtax, nr\_camin, nr\_camera, medie\_repartizare, sex)VALUES ('DAM9902', 'Deac', 'Alexandra','0745379341', 'CSIE', 'II', 'Buget', 'A6BN', 017, 8.36, 'F');

INSERT INTO STUDENT(id\_stud, nume\_stud, prenume\_stud, nr\_telefon, den\_fac, an\_studiu, bugtax, nr\_camin, nr\_camera, medie\_repartizare, sex)VALUES ('CA6534', 'Ianos', 'Alina','0732434873', 'CSIE', 'II', 'Buget', 'A6BN', 017, 9.70, 'M');

INSERT INTO STUDENT(id\_stud, nume\_stud, prenume\_stud, nr\_telefon, den\_fac, an\_studiu, bugtax, nr\_camin, nr\_camera, medie\_repartizare, sex)VALUES ('IA1203', 'Costache', 'Gabriel','0765299114', 'MAN', 'II', 'Taxa', 'A2BV', 421, 7.10, 'M');

INSERT INTO STUDENT(id\_stud, nume\_stud, prenume\_stud, nr\_telefon, den\_fac, an\_studiu, bugtax, nr\_camin, nr\_camera, medie\_repartizare, sex)VALUES ('CG8447', 'Coman', 'Florinel','0764569231','MRK', 'II', 'Buget', 'A2BV', 421, 8.80, 'M');

INSERT INTO STUDENT(id\_stud, nume\_stud, prenume\_stud, nr\_telefon, den\_fac, an\_studiu, bugtax, nr\_camin, nr\_camera, medie\_repartizare, sex)VALUES ('HG8646', 'Palade', 'Gabriel','0755342379','CSIE', 'II', 'Buget', 'A2BV', 421, 9, 'M');



### REGIE\_CAMIN

INSERT INTO REGIE\_CAMIN(nr\_chitanta, descriere, regie) VALUES ('100290', 'OCT\_BEL\_19',

250);

INSERT INTO REGIE\_CAMIN(nr\_chitanta, descriere, regie) VALUES ('100291', 'OCT\_MOXA\_19', 300);

INSERT INTO REGIE\_CAMIN(nr\_chitanta, descriere, regie) VALUES ('100292', 'OCT\_VITAN\_19', 200);

INSERT INTO REGIE\_CAMIN(nr\_chitanta, descriere, regie) VALUES ('100293', 'OCT\_AGR\_19', 200);

INSERT INTO REGIE\_CAMIN(nr\_chitanta, descriere, regie) VALUES ('100294', 'OCT\_TEI\_19', 200);

INSERT INTO REGIE\_CAMIN(nr\_chitanta, descriere, regie) VALUES ('100295', 'NOI\_BEL\_19', 120);

INSERT INTO REGIE\_CAMIN(nr\_chitanta, descriere, regie) VALUES ('100296', 'NOI\_MOXA\_19', 150);

### ISTORIC\_REGIE

INSERT INTO ISTORIC\_REGIE(id\_stud, data\_plata, suma\_incasata, suma\_datorata, nr\_chitanta) VALUES ('ICT2341', TO\_DATE('27-09-2019', 'DD-MM-YYYY'), 250, 0, '100290');

INSERT INTO ISTORIC\_REGIE(id\_stud, data\_plata, suma\_incasata, suma\_datorata, nr\_chitanta) VALUES ('IPI8974', TO\_DATE('03-10-2019', 'DD-MM-YYYY'),250,0,'100290');

INSERT INTO ISTORIC\_REGIE(id\_stud, data\_plata, suma\_incasata, suma\_datorata, nr\_chitanta) VALUES ('FD7830', TO\_DATE('01-10-2019', 'DD-MM-YYYY'), 250, 0, '100290');

INSERT INTO ISTORIC\_REGIE(id\_stud, data\_plata, suma\_incasata, suma\_datorata, nr\_chitanta) VALUES ('DAM9902', TO\_DATE('01-10-2019', 'DD-MM-YYYY'), 250, 0, '100290');

INSERT INTO ISTORIC\_REGIE(id\_stud, data\_plata, suma\_incasata, suma\_datorata, nr\_chitanta) VALUES ('CA6534', TO\_DATE('06-10-2019', 'DD-MM-YYYY'), 250, 0, '100290');

INSERT INTO ISTORIC\_REGIE(id\_stud, data\_plata, suma\_incasata, suma\_datorata, nr\_chitanta) VALUES ('IA1203', TO\_DATE('01-10-2019', 'DD-MM-YYYY'), 250, 0, '100290');

INSERT INTO ISTORIC\_REGIE(id\_stud, data\_plata, suma\_incasata, suma\_datorata, nr\_chitanta) VALUES ('CG8447', TO\_DATE('25-09-2019', 'DD-MM-YYYY'), 300, 0, '100291');

### UPDATE

UPDATE CAMIN

SET denumire\_camin = 'Belvedere Vechi A2' WHERE nr\_camin = 'A2BV';

UPDATE CAMIN

SET adresa = 'Str. Chibzuintei nr.2'

WHERE denumire\_camin LIKE 'Belvedere Nou A\_';

UPDATE REGIE\_CAMIN

SET regie = (SELECT regie FROM REGIE\_CAMIN WHERE nr\_chitanta='100296') WHERE descriere = 'OCT\_VITAN\_19';

UPDATE ADMINISTRATIE

SET salariu = salariu + 100 WHERE salariu < 2000; DELETE

CREATE TABLE SOMERI

AS SELECT \* FROM ADMINISTRATIE

WHERE id\_admin NOT IN (SELECT id\_admin FROM CAMIN);

DELETE FROM ADMINISTRATIE

WHERE id\_admin NOT IN (SELECT id\_admin FROM CAMIN);

v.1

--1) Numarul administratorilor cu salariul mai mare sau egal cu 1500 lei

--1

DECLARE

cursor admin is select nume\_admin, prenume\_admin, id\_admin from administratie where salariu>=1500;

nume administratie.nume\_admin%type;

prenume administratie.prenume\_admin%type;

id administratie.id\_admin%type;

BEGIN

dbms\_output.put\_line('Lista cu administratorii cu salariul mai mare de 1500 lei');

open admin;

loop

fetch admin into nume, prenume, id;

exit when admin%notfound;

dbms\_output.put\_line('Nume: '||nume||' '||'Prenume: '||prenume||' '||'ID: '||id);

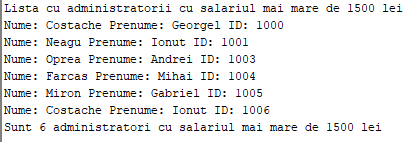
end loop;

dbms\_output.put\_line('Sunt '||admin%rowcount||' administratori cu salariul mai mare de 1500 lei');

close admin;

end;

/



--2) Schimba numarul de persoana dintr-o camera

--2

BEGIN

UPDATE camin

SET nr\_pers=3

WHERE nr\_camere=101;

IF SQL%NOTFOUND THEN

DBMS\_OUTPUT.PUT\_LINE('Numarul camerei e incorect');

END IF;

END;

/

2.PNG

--3) Numarul de camine administrate de fiecare

--3

DECLARE

CURSOR admin IS SELECT a.id\_admin, a.nume\_admin, a.prenume\_admin, COUNT(c.id\_admin) numar

FROM administratie a, camin c

WHERE a.id\_admin=c.id\_admin

GROUP BY a.id\_admin, a.nume\_admin, a.prenume\_admin

ORDER BY numar DESC;

nr admin%ROWTYPE;

BEGIN

DBMS\_OUTPUT.PUT\_LINE('Numarul de camine pentru fiecare administrator:');

IF NOT admin%ISOPEN THEN

OPEN admin;

END IF;

LOOP

FETCH admin INTO nr;

EXIT WHEN admin%NOTFOUND;

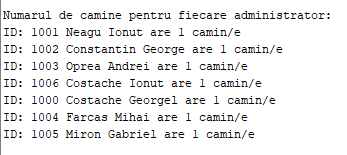
DBMS\_OUTPUT.PUT\_LINE('ID: '||nr.id\_admin||' '||nr.nume\_admin||' '||nr.prenume\_admin||' are '||nr.numar||' camin/e');

END LOOP;

CLOSE admin;

END;

/



--4) Numarul studentilor de la taxa

--4

declare

cursor bt is select nume\_stud, prenume\_stud,nr\_telefon

from student where lower(bugtax)='taxa';

nume student.nume\_stud%type;

prenume student.prenume\_stud%type;

nr\_telefon student.nr\_telefon%type;

begin

DBMS\_OUTPUT.PUT\_LINE('Studentii de la taxa');

OPEN bt;

LOOP

FETCH bt INTO nume, prenume, nr\_telefon;

EXIT WHEN bt%NOTFOUND;

DBMS\_OUTPUT.PUT\_LINE(nume||' '||prenume||' e la taxa');

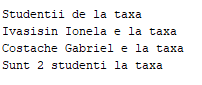
END LOOP;

dbms\_output.put\_line('Sunt '||bt%rowcount||' studenti la taxa');

CLOSE bt;

END;

/



--5) Lista de datorii la regie

--5

DECLARE

CURSOR admin IS SELECT s.nume\_stud, s.prenume\_stud, ir.suma\_datorata

FROM student s, istoric\_regie ir

WHERE s.id\_stud=ir.id\_stud

GROUP BY s.nume\_stud, s.prenume\_stud, ir.suma\_datorata;

nr admin%ROWTYPE;

BEGIN

DBMS\_OUTPUT.PUT\_LINE('Datorii la regie:');

IF NOT admin%ISOPEN THEN

OPEN admin;

END IF;

LOOP

FETCH admin INTO nr;

EXIT WHEN admin%NOTFOUND;

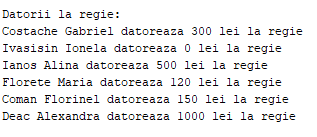
DBMS\_OUTPUT.PUT\_LINE(nr.nume\_stud||' '||nr.prenume\_stud||' datoreaza '||nr.suma\_datorata||' lei la regie');

END LOOP;

CLOSE admin;

END;

**/**

****

**v.2**

SET SERVEROUTPUT ON

--1

DECLARE

cursor c is select nume\_admin,prenume\_admin,salariu from administratie where nume\_admin='Costache';

nr c%ROWTYPE;

BEGIN

open c;

loop

FETCH c INTO nr;

exit when c%notfound;

DBMS\_OUTPUT.PUT\_LINE(nr.nume\_admin||' '||nr.prenume\_admin||' are salariul de '||nr.salariu||' lei');

end loop;

close c;

EXCEPTION

when no\_data\_found then

dbms\_output.put\_line('Numele administratorului nu exista');

end;

/

6.PNG

--2

DECLARE

v\_numec VARCHAR2(20);

v\_id administratie.id\_admin%type:='&id';

BEGIN

select denumire\_camin into v\_numec from camin where id\_admin=v\_id;

dbms\_output.put\_line('Denumirea caminului e: '||v\_numec);

exception

when no\_data\_found then

DBMS\_OUTPUT.PUT\_LINE('Administratorul nu exista sau nu are camin');

when too\_many\_rows then

DBMS\_OUTPUT.PUT\_LINE('Administratorul are mai multe camine');

end;

/

7.PNG

--3

DECLARE

er exception;

BEGIN

update administratie

set salariu=ROUND(salariu\*1.25)

where salariu<=1300;

if SQL%NOTFOUND then

raise er;

end if;

EXCEPTION

when er then

DBMS\_OUTPUT.PUT\_LINE('Toti au salarii mai mari de 1300 lei');

end;

/

8.PNG

--4

DECLARE

v\_id\_stud student.id\_stud%type:='&id';

restanta istoric\_regie.suma\_datorata%type;

ex exception;

BEGIN

select suma\_datorata into restanta from istoric\_regie where id\_stud=v\_id\_stud;

if (restanta)>0 then

DBMS\_OUTPUT.PUT\_LINE('Suma datorata: '||restanta);

update istoric\_regie set suma\_datorata=0 where id\_stud=v\_id\_stud;

else

raise ex;

end if;

exception

when ex then

DBMS\_OUTPUT.PUT\_LINE('A platit caminul');

end;

/

1.PNG

**v.3**

--PROCEDURI

--1 MODIF SAL

CREATE OR REPLACE

PROCEDURE MODIF\_SAL

(V\_ID\_ADMIN IN ADMINISTRATIE.ID\_ADMIN%type, PROC IN number)

IS

V\_SAL ADMINISTRATIE.SALARIU%type;

BEGIN

SELECT SALARIU INTO V\_SAL FROM ADMINISTRATIE WHERE ID\_ADMIN=V\_ID\_ADMIN;

DBMS\_OUTPUT.PUT\_LINE('Administratorul are salariul de '||V\_SAL);

UPDATE ADMINISTRATIE SET SALARIU=SALARIU\*(1+PROC/100) WHERE ID\_ADMIN=V\_ID\_ADMIN;

SELECT SALARIU INTO V\_SAL FROM ADMINISTRATIE WHERE ID\_ADMIN=V\_ID\_ADMIN;

DBMS\_OUTPUT.PUT\_LINE('Noul salariu este de '||V\_SAL);

END;

/

BEGIN

MODIF\_SAL(1001,15);

END;

/

2.PNG

--2 DATE DESPRE STUDENTI

CREATE OR REPLACE PROCEDURE C\_STUD

(V\_ID\_STUD IN STUDENT.ID\_STUD%type, V\_NUME OUT STUDENT.NUME\_STUD%type, V\_PRENUME OUT STUDENT.PRENUME\_STUD%type, V\_NR\_CAMIN OUT STUDENT.NR\_CAMIN%type, V\_NR\_CAMERA OUT STUDENT.NR\_CAMERA%type)

IS

BEGIN

SELECT NUME\_STUD, PRENUME\_STUD, NR\_CAMIN, NR\_CAMERA INTO V\_NUME, V\_PRENUME, V\_NR\_CAMIN, V\_NR\_CAMERA FROM STUDENT WHERE ID\_STUD=V\_ID\_STUD;

DBMS\_OUTPUT.PUT\_LINE(' Studentul '||V\_NUME||' are camera '||V\_NR\_CAMERA);

END;

/

DECLARE

NUME STUDENT.NUME\_STUD%type;

PRENUME STUDENT.PRENUME\_STUD%type;

NR\_CAMIN STUDENT.NR\_CAMIN%type;

NR\_CAMERA STUDENT.NR\_CAMERA%type;

BEGIN

C\_STUD('ICT2341', NUME, PRENUME, NR\_CAMIN, NR\_CAMERA);

END;

/

3.PNG

--3 REGIA

CREATE OR REPLACE PROCEDURE S\_REGIE

(V\_SUMA\_INCASATA OUT ISTORIC\_REGIE.SUMA\_INCASATA%type, V\_SUMA\_DATORATA OUT ISTORIC\_REGIE.SUMA\_DATORATA%type)

IS

BEGIN

SELECT SUM(SUMA\_INCASATA), SUM(SUMA\_DATORATA) INTO V\_SUMA\_INCASATA, V\_SUMA\_DATORATA FROM ISTORIC\_REGIE;

DBMS\_OUTPUT.PUT\_LINE('Totalul incasat este de: '||V\_SUMA\_INCASATA||' si totalul datorat este de: '||V\_SUMA\_DATORATA);

END;

/

DECLARE

SUMA\_INC ISTORIC\_REGIE.SUMA\_INCASATA%type;

SUMA\_DAT ISTORIC\_REGIE.SUMA\_DATORATA%type;

BEGIN

S\_REGIE(SUMA\_INC, SUMA\_DAT);

END;

/

4.PNG

--FUNCTII

--1 SALARIUL ADMIN

CREATE OR REPLACE FUNCTION MSAL

(V\_ID\_ADMIN IN ADMINISTRATIE.ID\_ADMIN%type)

RETURN BOOLEAN

IS

V\_SAL ADMINISTRATIE.SALARIU%type;

MSAL ADMINISTRATIE.SALARIU%type;

BEGIN

SELECT SUM(SALARIU)/COUNT(ID\_ADMIN) INTO MSAL FROM ADMINISTRATIE;

SELECT SALARIU INTO V\_SAL FROM ADMINISTRATIE WHERE ID\_ADMIN=V\_ID\_ADMIN;

IF V\_SAL > MSAL THEN

RETURN TRUE;

ELSE

RETURN FALSE;

END IF;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RETURN NULL;

END;

/

BEGIN

IF (MSAL(1000) IS NULL) THEN

DBMS\_OUTPUT.PUT\_LINE('Administratorul nu exista');

ELSIF (MSAL(1000)) THEN

DBMS\_OUTPUT.PUT\_LINE('Administratorul are salariul mai mare decat media');

ELSE

DBMS\_OUTPUT.PUT\_LINE('Administratorul are salariul mai mic decat media');

END IF;

END;

/

5.PNG

--2 LOCURI CAMIN

CREATE OR REPLACE FUNCTION MCAMERE

(V\_NR\_CAMIN IN CAMIN.NR\_CAMIN%type)

RETURN BOOLEAN

IS

V\_NR\_CAMERE CAMIN.NR\_CAMERE%type;

MCAMERE CAMIN.NR\_CAMERE%type;

BEGIN

SELECT SUM(NR\_CAMERE)/COUNT(NR\_CAMIN) INTO MCAMERE FROM CAMIN;

SELECT NR\_CAMERE INTO V\_NR\_CAMERE FROM CAMIN WHERE NR\_CAMIN=V\_NR\_CAMIN;

IF V\_NR\_CAMERE > MCAMERE THEN

RETURN TRUE;

ELSE

RETURN FALSE;

END IF;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RETURN NULL;

END;

/

BEGIN

IF (MCAMERE('A1BV') IS NULL) THEN

DBMS\_OUTPUT.PUT\_LINE('Caminul nu exista');

ELSIF (MCAMERE('A1BV')) THEN

DBMS\_OUTPUT.PUT\_LINE('Caminul are mai multe camere decat media');

ELSE

DBMS\_OUTPUT.PUT\_LINE('Caminul are mai putine camere decat media');

END IF;

END;

/

6.PNG

--3 MEDIE STUDENT

CREATE OR REPLACE FUNCTION STX

(V\_ID\_STUD IN STUDENT.ID\_STUD%type)

RETURN BOOLEAN

IS

MEDIE STUDENT.MEDIE\_REPARTIZARE%TYPE;

MM STUDENT.MEDIE\_REPARTIZARE%TYPE :=8.34;

BEGIN

SELECT MEDIE\_REPARTIZARE INTO MEDIE FROM STUDENT WHERE ID\_STUD=V\_ID\_STUD;

IF MEDIE > MM THEN

RETURN TRUE;

ELSE

RETURN FALSE;

END IF;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RETURN NULL;

END;

/

BEGIN

IF (STX('ICT2341') IS NULL) THEN

DBMS\_OUTPUT.PUT\_LINE('Studentul nu exista');

ELSIF (STX('ICT2341')) THEN

DBMS\_OUTPUT.PUT\_LINE('Studentul are media mai mare decat cea de intrare');

ELSE

DBMS\_OUTPUT.PUT\_LINE('Studentul are media mai mica decat cea de intrare');

END IF;

END;

/

7.PNG

--4 TIP CAMERA

CREATE OR REPLACE FUNCTION CT

(V\_NR\_CAMERA IN CAMERA.NR\_CAMERA%type)

RETURN BOOLEAN

IS

TC CAMERA.TIP\_CAMERA%TYPE;

BEGIN

SELECT TIP\_CAMERA INTO TC FROM CAMERA WHERE NR\_CAMERA=V\_NR\_CAMERA;

IF TC='F' THEN

RETURN TRUE;

ELSE

RETURN FALSE;

END IF;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RETURN NULL;

END;

/

BEGIN

IF (CT(103) IS NULL) THEN

DBMS\_OUTPUT.PUT\_LINE('Camera nu exista');

ELSIF (CT(103)) THEN

DBMS\_OUTPUT.PUT\_LINE('Camera de fete');

ELSE

DBMS\_OUTPUT.PUT\_LINE('Camera de baieti');

END IF;

END;

1.PNG

**v.4**

CREATE OR REPLACE PACKAGE admsal AS

PROCEDURE mics\_sal(p\_id\_admin IN administratie.id\_admin%TYPE, proc IN NUMBER);

FUNCTION nr\_stud (p\_camin IN student.nr\_camin%TYPE) RETURN NUMBER;

PROCEDURE red\_sal\_nrstud (p\_camin IN student.nr\_camin%TYPE);

END;

/

CREATE OR REPLACE PACKAGE BODY admsal AS

PROCEDURE mics\_sal(p\_id\_admin IN administratie.id\_admin%TYPE, proc IN NUMBER)

IS

    v\_sal administratie.salariu%TYPE;

BEGIN

    SELECT salariu INTO v\_sal FROM administratie WHERE id\_admin = p\_id\_admin;

    DBMS\_OUTPUT.PUT\_LINE('Salariul initial era ' || v\_sal);

    UPDATE administratie

    SET salariu = salariu\*(1-proc/100)

    WHERE id\_admin = p\_id\_admin;

    SELECT salariu INTO v\_sal FROM administratie WHERE id\_admin = p\_id\_admin;

    DBMS\_OUTPUT.PUT\_LINE('Salariul este acum ' || v\_sal);

END;

FUNCTION nr\_stud (p\_camin IN student.nr\_camin%TYPE) RETURN NUMBER

IS

nr NUMBER;

BEGIN

    SELECT COUNT(id\_stud) INTO nr FROM student WHERE nr\_camin = p\_camin;

    RETURN nr;

END;

FUNCTION bugtax (p\_camin IN student.nr\_camin%TYPE) RETURN NUMBER

IS

nr NUMBER;

BEGIN

    SELECT COUNT(id\_stud) INTO nr FROM student WHERE nr\_camin = p\_camin AND lower(bugtax)='taxa';

    RETURN nr;

END;

PROCEDURE red\_sal\_nrstud (p\_camin IN student.nr\_camin%TYPE)

IS

high EXCEPTION;

nr NUMBER;

CURSOR c IS SELECT id\_admin FROM administratie WHERE id\_admin=(SELECT id\_admin FROM camin WHERE nr\_camin=p\_camin);

BEGIN

    nr := nr\_stud(p\_camin);

    IF nr > 50 THEN RAISE high;

    ELSE

        FOR r IN c

        LOOP

            mics\_sal(r.id\_admin,5);

        END LOOP;

    END IF;

EXCEPTION

    WHEN high THEN DBMS\_OUTPUT.PUT\_LINE('Prea multi studenti ca sa ii micsoreze salariul');

END;

END;

/

ACCEPT camin PROMPT 'Introduceti caminul cautat:';

DECLARE

    camin student.nr\_camin%TYPE := '&camin';

    id administratie.id\_admin%TYPE;

    CURSOR c IS SELECT \* FROM administratie WHERE id\_admin=(SELECT id\_admin FROM camin WHERE nr\_camin=camin);

BEGIN

    admsal.red\_sal\_nrstud(camin);

    DBMS\_OUTPUT.PUT\_LINE(chr(10));

    FOR r IN c

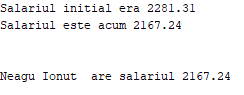
    LOOP

        DBMS\_OUTPUT.PUT\_LINE(r.nume\_admin || ' ' || r.prenume\_admin || ' ' || ' are salariul ' || r.salariu);

    END LOOP;

END;

/

****

**v.5**

--1

create or replace trigger marire\_salariu

before update of salariu on administratie

for each row

when(new.salariu>old.salariu)

begin

if :new.salariu>3500

then raise\_application\_error(-20000,'Nu se poate mari salariul');

end if;

end;

/

update administratie set salariu=salariu\*10 where id\_admin=1000;

/

1.PNG

--2

create or replace trigger reg

before update of suma\_datorata on istoric\_regie

for each row

when(new.suma\_datorata<old.suma\_datorata)

begin

if :new.suma\_datorata<0

then raise\_application\_error(-20000,'Suma incorecta');

end if;

end;

/

update istoric\_regie set suma\_datorata=-1 where id\_stud='IPI8974';

/

2.PNG

--3

create or replace trigger tel

before update of nr\_telefon on administratie

for each row

when(new.id\_admin=old.id\_admin)

begin

if length(:new.nr\_telefon)!=10

then raise\_application\_error(-20000,'Numar de telefon incorect');

end if;

end;

/

update administratie set nr\_telefon='0755' where id\_admin=1000;

/

3.PNG