

Gestiunea unei biblioteci

Cuprins

1. *Descrierea bazei de date*
2. *ERD*
3. *Diagrama conceptuala*
4. *Realizarea tabelor în Oracle*
5. *Adaugarea informatiilor în tabele*
6. *Subprogram stocat + colectie*
7. *Subprogram stocat + cursor*
8. *Subprogram stocat functie 3 tabele*
9. *Subprogram stocat procedura 5 tabele*
10. *Trigger LMD comanda*
11. *Trigger LMD linie*
12. *Trigger LDD*
13. *Pachet obiecte definite*
14. *Pachet date complexe*

1. Descrierea bazei de date

Directorul Bibliotecii Naționale din București dorește să realizeze o bază de date pentru a ține evidența împrumuturilor și a personalului angajat. Astfel, acesta consideră că această trecere va fi benefică pentru bibliotecă, va ușura munca angajaților și va reduce consumul de timp, hârtie și energie.

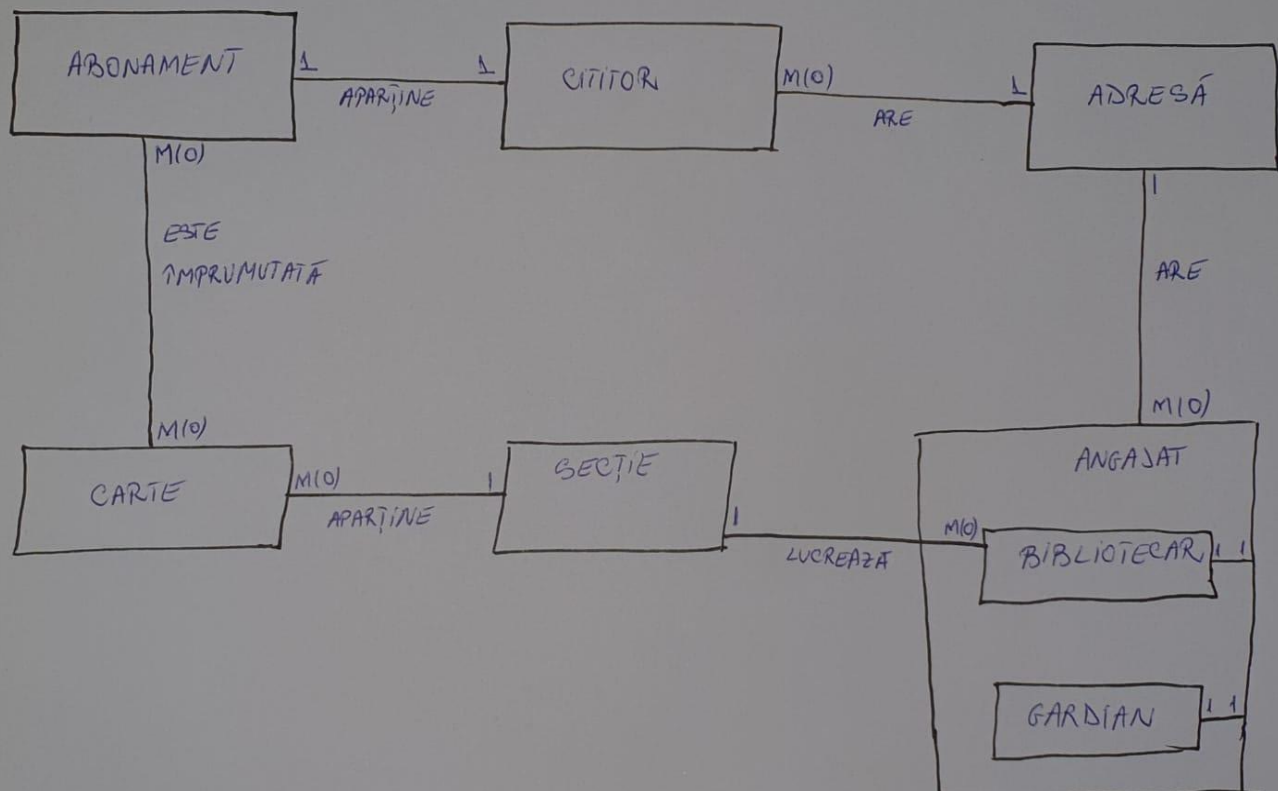
Orice cititor își poate face un singur abonament la bibliotecă, cu ajutorul căruia va împrumuta una sau mai multe cărți, de la ce secție dorește, oricând are nevoie.

Cărțile sunt organizate pe secții, cum ar fi beletristica, matematica sau limbi străine, secții situate în săli diferite, iar în fiecare sală lucrează unul sau mai mulți bibliotecari.

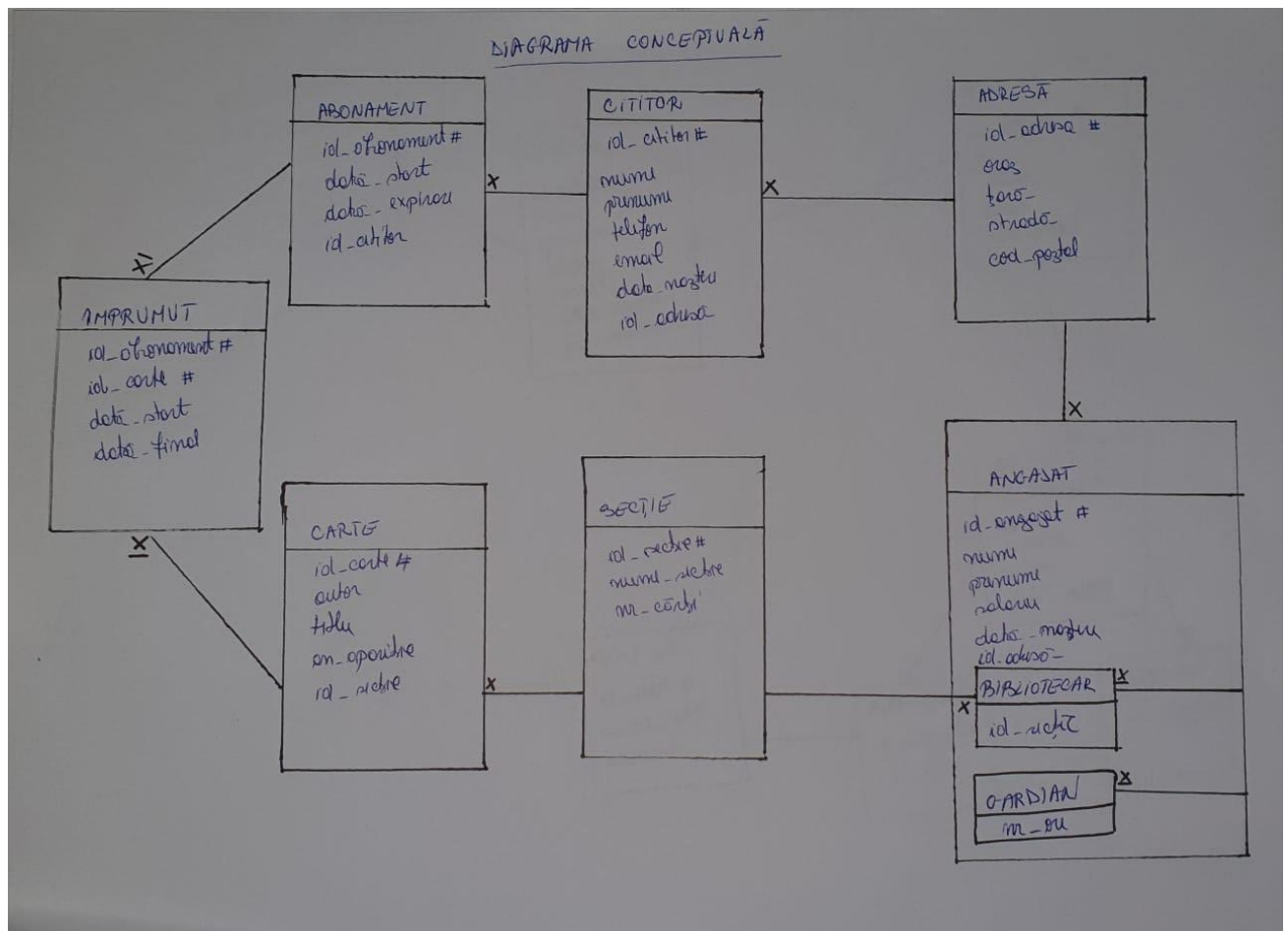
Atât pentru angajați, cât și pentru cititori se va reține adresa. Împrumuturile au o dată de start și una de final, iar astfel cititorii vor avea un istoric al cărților împrumutate. Angajații vor înregistra împrumuturile zilnic, iar astfel baza de date va fi actualizată în permanență.

2. ERD

DIAGRAMA ENTITATE - RELATIE



3. Diagrama Conceptuala



4. Realizarea tabelelor în Oracle

```

create table adresa (
  id_adresa int NOT NULL primary key,
  oras varchar2(40) NOT NULL,
  tara varchar2(40) NOT NULL,
  strada varchar2(60),
  cod_postal varchar2(6)
);
  
```

```
create table cititor (  
    id_cititor int NOT NULL primary key,  
    nume varchar2(40) NOT NULL,  
    prenume varchar2(40) NOT NULL,  
    telefon varchar2(10) NOT NULL,  
    email varchar2(30),  
    data_nastere date default sysdate,  
    id_adresa int,  
    CONSTRAINT FK_CititorAdresa FOREIGN KEY (id_adresa)  
REFERENCES adresa(id_adresa) on delete cascade  
);
```

```
create table abonament (  
    id_abonament int NOT NULL primary key,  
    data_start date default sysdate NOT NULL,  
    data_expirare date default sysdate NOT NULL,  
    id_cititor int,  
    CONSTRAINT FK_AbonamentCititor FOREIGN KEY (id_cititor)  
REFERENCES cititor(id_cititor) on delete cascade  
);
```

```
create table sectie (  
    id_sectie int NOT NULL primary key,  
    nume_sectie varchar2(40),  
    nr_carti int  
);
```

```
create table carte (  
    id_carte int NOT NULL primary key,  
    autor varchar2(40),  
    titlu varchar2(40),  
    an_aparitiei varchar2(4),  
    id_sectie int,  
    CONSTRAINT FK_CarteSectie FOREIGN KEY (id_sectie) REFERENCES  
sectie(id_sectie) on delete cascade  
);
```

```
create table imprumut (  
    id_abonament int NOT NULL,  
    id_carte int NOT NULL,  
    data_start date default sysdate,  
    data_final date default sysdate,  
    CONSTRAINT FK_ImprumutAbonament FOREIGN KEY (id_abonament)  
REFERENCES abonament(id_abonament) on delete cascade,  
    CONSTRAINT FK_ImprumutCarte FOREIGN KEY (id_carte)  
REFERENCES carte(id_carte) on delete cascade,  
    primary key (id_abonament, id_carte, data_start)  
);
```

```
create table angajat (  
    id_angajat int NOT NULL primary key,  
    nume varchar2(40) NOT NULL,  
    prenume varchar2(40) NOT NULL,  
    salariu number(10),  
    data_nastere date default sysdate,  
    id_adresa int,  
    CONSTRAINT FK_AngajatAdresa FOREIGN KEY (id_adresa)  
REFERENCES adresa(id_adresa) on delete cascade  
);
```

```
create table bibliotecar(  
    id_angajat int primary key,  
    id_sectie,  
    CONSTRAINT FK_BibliotecarAngajat FOREIGN KEY (id_angajat)  
REFERENCES angajat(id_angajat) ON DELETE CASCADE,  
    CONSTRAINT FK_BibliotecarSectie FOREIGN KEY (id_sectie)  
REFERENCES sectie(id_sectie) ON DELETE CASCADE  
);
```

```
create table gardian(  
    id_angajat int primary key,  
    nr_ore number(10),  
    CONSTRAINT FK_GardianAngajat FOREIGN KEY (id_angajat)  
REFERENCES angajat(id_angajat) ON DELETE CASCADE );
```

Live SQL

Feedback Help barbu.mary54@yahoo.com

SQL Worksheet Clear Find Actions Save Run

```
39 an_apartie varchar2(4),
40 id_sectie int,
41 CONSTRAINT FK_CarteSectie FOREIGN KEY (id_sectie) REFERENCES sectie(id_sectie) on delete cascade
42 );
43
44 create table imnrumit (
```

Table created.

Table created.

Table created.

Table created.

Table created.

Table created.

Table created.

5. Adaugarea informatiilor în tabele

--adresa--

```
insert into adresa
values (1, 'Bucuresti', 'Romania', 'Splaiul Independentei', '123456');
```

```
insert into adresa
values (2, 'Pitesti', 'Romania', 'Raiului', '234567');
```

```
insert into adresa
values (3, 'Bucuresti', 'Romania', 'Unicornilor', '345678');
```

```
insert into adresa
values (4, 'Craiova', 'Romania', 'Primaverii', '456789');
```

```
insert into adresa
values (5, 'Buzau', 'Romania', 'Libertatii', '124567');
```

--citittor--

```
insert into cititor
values (1, 'Popescu', 'Ion', '0711111111', 'popescu@gmail.com', '12-dec-1980',
3);
```

```
insert into cititor  
values (2, 'Nicolescu', 'Alicia', '0722222222', 'nicolescu@gmail.com',  
'30-dec-1990', 2);
```

```
insert into cititor  
values (3, 'Ionescu', 'Aurel', '0711111111', 'ionescu@gmail.com', '07-jul-2001',  
1);
```

```
insert into cititor  
values (4, 'Udrea', 'Cristina', '0744444444', 'udrea@gmail.com', '05-jun-1997',  
5);
```

```
insert into cititor  
values (5, 'Dinu', 'Andreea', '0755555555', 'dinu@gmail.com', '24-mar-1999',  
4);
```

```
insert into cititor  
values (6, 'Cirstea', 'Gabriel', '0766666666', 'cirstea@gmail.com',  
'13-jan-2003', 3);
```

```
insert into cititor  
values (7, 'Cirstea', 'Ioana', '0777777777', 'cirstea@gmail.com', '15-jan-2003',  
4);
```

```
insert into cititor  
values (8, 'Oprea', 'Carmen', '0788888888', 'oprea@gmail.com', '09-jun-2001',  
1);
```

--abonament--

```
insert into abonament  
values (1, '05-apr-2018', '05-apr-2020', 1);
```

```
insert into abonament  
values (2, '12-dec-2019', '12-dec-2021', 2);
```

```
insert into abonament  
values (3, '13-may-2020', '13-may-2022', 3);
```


insert into abonament
values (4, '21-nov-2017', '21-nov-2019', 4);

insert into abonament
values (5, '05-oct-2018', '05-oct-2020', 5);

insert into abonament
values (6, '17-apr-2016', '17-apr-2018', 6);

insert into abonament
values (7, '20-apr-2017', '20-apr-2019', 7);

--sectie--

insert into sectie
values (1, 'Beletristica', 10000);

insert into sectie
values (2, 'Matematica', 2000);

insert into sectie
values (3, 'poezii', 7540);

insert into sectie
values (4, ' Stiintele naturii', 1230);

insert into sectie
values (5, 'Copii', 3480);

insert into sectie
values (6, 'Informatica', 21340);

--carte--

insert into carte
values (1, 'Mihai Eminescu', 'Poezii', '1881', 3);

insert into carte
values (2, 'Camil Petrescu', 'Ultima noapte de dragoste', '1918', 1);

insert into carte

values (3, 'Fratii Grimm', 'Alba-ca-Zapada', '1812', 5);

insert into carte

values (4, 'Ion Luca Caragiale', 'O scrisoare pierduta', '1879', 1);

insert into carte

values (5, 'Marius Perianu', 'Culegere Matematica', '2015', 2);

insert into carte

values (6, 'Tudor Sorin', 'Algoritmi', '2004', 6);

--imprumut--

insert into imprumut

values (1, 1, '19-dec-2020', '3-jan-2021');

insert into imprumut

values (2, 2, '20-dec-2020', '4-jan-2021');

insert into imprumut

values (3, 3, '21-dec-2020', '5-jan-2021');

insert into imprumut

values (4, 4, '22-dec-2020', '6-jan-2021');

insert into imprumut

values (5, 5, '23-dec-2020', '7-jan-2021');

insert into imprumut

values (1, 5, '24-dec-2020', '8-jan-2021');

insert into imprumut

values (2, 4, '10-dec-2020', '24-dec-2021');

insert into imprumut

values (1, 2, '11-dec-2020', '26-dec-2021');

```
insert into imprumut  
values (2, 3, '12-dec-2020', '27-dec-2021');
```

```
insert into imprumut  
values (4, 5, '13-dec-2020', '28-dec-2021');
```

```
insert into imprumut  
values (1, 3, '13-dec-2019', '28-dec-2020');
```

```
insert into imprumut  
values (1, 4, '13-jan-2018', '28-jan-2019');
```

```
insert into imprumut  
values (7, 1, '12-jan-2015', '27-jan-2015');
```

```
insert into imprumut  
values (7, 2, '12-jan-2015', '27-jan-2015');
```

```
--angajat--
```

```
insert into angajat  
values (1, 'Parnescu', 'Tamarel', 1000, '2-dec-1998', 3);
```

```
insert into angajat  
values (2, 'Preda', 'Larisa', 1200, '12-dec-1990', 1);
```

```
insert into angajat  
values (3, 'Celmare', 'Diana', 2000, '9-feb-2001', 2);
```

```
insert into angajat  
values (4, 'Istrare', 'Maria', 1500, '15-dec-1983', 4);
```

```
insert into angajat  
values (5, 'Calin', 'George', 1340, '10-mar-1980', 5);
```

```
--bibliotecar--
```

```
insert into bibliotecar  
values (1, 3);
```

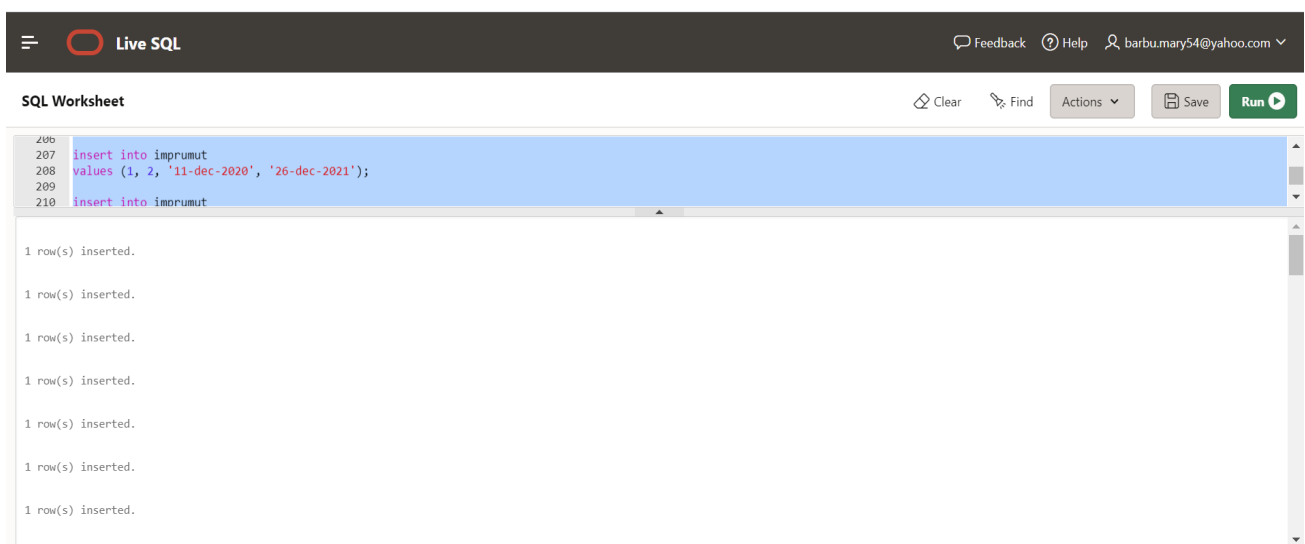
```
insert into bibliotecar  
values (2,1);
```

```
insert into bibliotecar  
values (3, 2);
```

```
--gardian--
```

```
insert into gardian  
values (4, 30);
```

```
insert into gardian  
values (5, 48);
```



```
206  
207 insert into imprumut  
208 values (1, 2, '11-dec-2020', '26-dec-2021');  
209  
210 insert into imprumut
```

1 row(s) inserted.
1 row(s) inserted.
1 row(s) inserted.
1 row(s) inserted.
1 row(s) inserted.
1 row(s) inserted.
1 row(s) inserted.

6. Definiti un subprogram stocat care primeste id-ul unui cititor si retine intr-o colectie toate cartile imprumutate de acesta in ultimul an, iar apoi le afiseaza. Apelati subprogramul.

```
CREATE OR REPLACE PROCEDURE f1_biblioteca  
(v_id_cititor cititor.id_cititor%TYPE )  
IS nume cititor.nume%type;  
BEGIN  
    SELECT nume INTO nume  
    FROM cititor  
    WHERE id_cititor = v_id_cititor;
```

```
DECLARE
    TYPE carti IS TABLE OF INT INDEX BY PLS_INTEGER;
    x carti;
    titlu_carte carte.titlu%type;
BEGIN
    select c.id_carte
    BULK COLLECT INTO x
    from carte c, imprumut i, abonament a
    where c.id_carte = i.id_carte and i.id_abonament = a.id_abonament and
a.id_cititor = v_id_cititor
    and i.data_start > sysdate - 365;

    IF x.first IS NULL THEN DBMS_OUTPUT.PUT_LINE('Nu exista
imprumuturi! ');
    ELSE
        DBMS_OUTPUT.PUT_LINE('Cartile imprumutate de cititorul ' ||
v_id_cititor || ': ');
        FOR i IN x.first..x.last LOOP
            SELECT titlu
            INTO titlu_carte
            from carte
            where carte.id_carte = x(i);
            DBMS_OUTPUT.PUT_LINE(titlu_carte);
        END LOOP;
    END IF;
END;
EXCEPTION
    WHEN NO_DATA_FOUND THEN RAISE_APPLICATION_ERROR(-20000,
'Nu exista cititori cu id-ul dat');
    WHEN OTHERS THEN RAISE_APPLICATION_ERROR(-20002,'Alta
eroare!');
END f1_biblioteca;
/
```

```
BEGIN
f1_biblioteca(1);
--f1_biblioteca(6);
--f1_biblioteca(8);
END;
/
```

6.1. Am apelat subprogramul pentru cititorul cu id-ul 1 și au fost afisate toate cărțile împrumutate de acesta în ultimul an.



```
290 END;
291 END;
292 EXCEPTION
293 WHEN NO_DATA_FOUND THEN RAISE_APPLICATION_ERROR(-20000, 'Nu exista cititori cu id-ul dat');
294 WHEN OTHERS THEN RAISE_APPLICATION_ERROR(-20002, 'Alta eroare!');
295 END f1_biblioteca;
296 /
297
298
299 BEGIN
300 f1_biblioteca(1);
301 --f1_biblioteca(6);
302 --f1_biblioteca(8);
303 END;
304 /
305
```

Statement processed.
Cartile imprumutate de cititorul 1:
Poezii
Ultima noapte de dragoste
Culegere Matematica

6.2. Am apelat subprogramul pentru un cititor care nu a imprumutat nicio carte.



```
290 END;
291 END;
292 EXCEPTION
293 WHEN NO_DATA_FOUND THEN RAISE_APPLICATION_ERROR(-20000, 'Nu exista cititori cu id-ul dat');
294 WHEN OTHERS THEN RAISE_APPLICATION_ERROR(-20002, 'Alta eroare!');
295 END f1_biblioteca;
296 /
297
298
299 BEGIN
300 --f1_biblioteca(1);
301 f1_biblioteca(6);
302 --f1_biblioteca(8);
303 END;
304 /
305
```

Statement processed.
Nu exista imprumuturi!

6.3. Am apelat subprogramul pentru un cititor care nu exista.



```
290 END;
291 END;
292 EXCEPTION
293 WHEN NO_DATA_FOUND THEN RAISE_APPLICATION_ERROR(-20000, 'Nu exista cititori cu id-ul dat');
294 WHEN OTHERS THEN RAISE_APPLICATION_ERROR(-20002, 'Alta eroare!');
295 END f1_biblioteca;
296 /
297
298
299 BEGIN
300 --f1_biblioteca(1);
301 --f1_biblioteca(6);
302 f1_biblioteca(8);
303 END;
304 /
305
```

ORA-20000: Nu exista cititori cu id-ul dat
ORA-06512: at "SQL_PZREDAVYTHSLJYMLF3POBLZH3.F1_BIBLIOTECA", line 32
ORA-06512: at line 4
ORA-06512: at "SYS.DBMS_SQL", line 1721

7. Pentru o sectie transmisa prin id(nume - care va fi afisat o singura data) obtineti lista cartilor care se gasesc in sectia respectiva.

```
CREATE OR REPLACE PROCEDURE f2_biblioteca
(v_id_sectie sectie.id_sectie%TYPE )
IS nume sectie.nume_sectie%type;
BEGIN
    SELECT nume_sectie INTO nume
    FROM sectie
    WHERE id_sectie = v_id_sectie;
    DBMS_OUTPUT.PUT_LINE('Sectia ' || nume || ': ');
    DECLARE
        v_nr number(4) :=0;
        CURSOR cu IS
            SELECT titlu, autor
            FROM carte c, sectie s
            WHERE c.id_sectie = s.id_sectie
            AND s.id_sectie = v_id_sectie;
    BEGIN
        FOR j in cu LOOP
            v_nr := v_nr + 1;
            DBMS_OUTPUT.PUT_LINE('Cartea ' || j.titlu || ' scrisa de ' || j.autor);
        END LOOP;
        IF v_nr = 0 THEN DBMS_OUTPUT.PUT_LINE('Nu exista carti!');
        END IF;
    END;
EXCEPTION
    WHEN NO_DATA_FOUND THEN RAISE_APPLICATION_ERROR(-20000,
    'Nu exista sectii cu id-ul dat');
    WHEN OTHERS THEN RAISE_APPLICATION_ERROR(-20002,'Alta
eroare!');
END f2_biblioteca;
/
BEGIN
f2_biblioteca(1);
--f2_biblioteca(4);
--f2_biblioteca(9);
END;
/
```

7.1. Am apelat subprogramul pentru o secție existentă.

Live SQL

Feedback Help barbu.mary54@yahoo.com

SQL Worksheet

Clear Find Actions Save Run

```
282 IF v_nm = 0 THEN DBMS_OUTPUT.PUT_LINE( 'Nu exista carti! ');
283 END IF;
284
285 EXCEPTION
286 WHEN NO_DATA_FOUND THEN RAISE_APPLICATION_ERROR(-20000, 'Nu exista sectii cu id-ul dat');
287 WHEN OTHERS THEN RAISE_APPLICATION_ERROR(-20002, 'Alta eroare!');
288 END f2_biblioteca;
289 /
290
291 BEGIN
292 f2_biblioteca(1);
293 --f2_biblioteca(4);
294 --f2_biblioteca(9);
295 END;
296 /
```

Statement processed.
Sectia Beletristica:
Cartea Ultima noapte de dragoste scrisa de Camil Petrescu
Cartea O scrisoare pierduta scrisa de Ion Luca Caragiale

7.2. Am apelat subprogramul pentru o secție care exista, dar în care nu exista cărți.

Live SQL

Feedback Help barbu.mary54@yahoo.com

SQL Worksheet

Clear Find Actions Save Run

```
282 IF v_nm = 0 THEN DBMS_OUTPUT.PUT_LINE( 'Nu exista carti! ');
283 END IF;
284
285 EXCEPTION
286 WHEN NO_DATA_FOUND THEN RAISE_APPLICATION_ERROR(-20000, 'Nu exista sectii cu id-ul dat');
287 WHEN OTHERS THEN RAISE_APPLICATION_ERROR(-20002, 'Alta eroare!');
288 END f2_biblioteca;
289 /
290
291 BEGIN
292 --f2_biblioteca(1);
293 f2_biblioteca(4);
294 --f2_biblioteca(9);
295 END;
296 /
```

Statement processed.
Sectia Stiintele naturii:
Nu exista carti!

7.3. Am apelat subprogramul pentru o secție care nu exista.

Live SQL

Feedback Help barbu.mary54@yahoo.com

SQL Worksheet

Clear Find Actions Save Run

```
282 IF v_nm = 0 THEN DBMS_OUTPUT.PUT_LINE( 'Nu exista carti! ');
283 END IF;
284
285 EXCEPTION
286 WHEN NO_DATA_FOUND THEN RAISE_APPLICATION_ERROR(-20000, 'Nu exista sectii cu id-ul dat');
287 WHEN OTHERS THEN RAISE_APPLICATION_ERROR(-20002, 'Alta eroare!');
288 END f2_biblioteca;
289 /
290
291 BEGIN
292 --f2_biblioteca(1);
293 --f2_biblioteca(4);
294 f2_biblioteca(9);
295 END;
296 /
```

ORA-20000: Nu exista sectii cu id-ul dat
ORA-06512: at line 4
ORA-06512: at "SYS.DBMS_SQL", line 1721

8. Definiti un subprogram prin care sa obtineti orasul in care locuieste un cititor daca se cunoaste id-ul abonamentului sau.

```
CREATE OR REPLACE FUNCTION f3_biblioteca
(v_id_abonament abonament.id_abonament%TYPE DEFAULT 'Bell')
RETURN VARCHAR2 IS
oras adresa.oras%type;
BEGIN
SELECT oras INTO oras
FROM adresa ad, cititor c, abonament ab
WHERE ab.id_abonament = v_id_abonament AND ad.id_adresa =
c.id_adresa AND c.id_cititor = ab.id_cititor;
RETURN oras;
EXCEPTION
WHEN NO_DATA_FOUND THEN RAISE_APPLICATION_ERROR(-20000,
'Nu exista abonamente cu id-ul dat');
WHEN OTHERS THEN RAISE_APPLICATION_ERROR(-20002,'Alta
eroare!');
END f3_biblioteca;
/

BEGIN
DBMS_OUTPUT.PUT_LINE('Orasul este '|| f3_biblioteca(2));
--DBMS_OUTPUT.PUT_LINE('Orasul este '|| f3_biblioteca(9));
END;
/
```

8.1. Am apelat subprogramul pentru un cititor care exista.



The screenshot shows a web-based SQL editor interface titled "Live SQL". The main area is labeled "SQL Worksheet" and contains the following SQL code:

```
270 RETURN oras;
271 EXCEPTION
272 WHEN NO_DATA_FOUND THEN RAISE_APPLICATION_ERROR(-20000, 'Nu exista abonamente cu id-ul dat');
273 WHEN OTHERS THEN RAISE_APPLICATION_ERROR(-20002,'Alta eroare!');
274 END f3_biblioteca;
275 /
276
277 BEGIN
278 DBMS_OUTPUT.PUT_LINE('Orasul este '|| f3_biblioteca(2));
279 --DBMS_OUTPUT.PUT_LINE('Orasul este '|| f3_biblioteca(9));
280 END;
281 /
282
283
284
```

Below the code editor, the output area shows the message: "Statement processed. Orasul este Pitesti". The interface includes a top navigation bar with "Live SQL", "Feedback", "Help", and a user profile. The code editor has buttons for "Clear", "Find", "Actions", "Save", and "Run".

8.2. Am apelat subprogramul pentru un cititor care nu exista.

```

270 RETURN ora;
271 EXCEPTION
272 WHEN NO_DATA_FOUND THEN RAISE_APPLICATION_ERROR(-20000, 'Nu exista abonamente cu id-ul dat');
273 WHEN OTHERS THEN RAISE_APPLICATION_ERROR(-20002, 'Alta eroare!');
274 END f3_biblioteca;
275 /
276
277 BEGIN
278 --DBMS_OUTPUT.PUT_LINE('Orasul este '|| f3_biblioteca(2));
279 DBMS_OUTPUT.PUT_LINE('Orasul este '|| f3_biblioteca(9));
280 END;
281 /
282
283
284

```

ORA-20000: Nu exista abonamente cu id-ul dat ORA-06512: at "SQL_SVGHAUGFJAKGXENCLMRYZQQV.F3_BIBLIOTECA", line 11
ORA-06512: at line 3
ORA-06512: at "SYS.DBMS_SQL", line 1721

9. Sa se defineasca un subprogram prin care se afiseaza numele sectiei de la care a imprumutat cele mai multe carti un cititor al carui nume se da ca parametru.

```

CREATE OR REPLACE PROCEDURE f4_biblioteca
(v_nume cititor.nume%TYPE)
IS
v_id cititor.id_cititor%TYPE;
BEGIN
    SELECT id_cititor into v_id
    FROM cititor
    WHERE v_nume = nume;
    DECLARE
        nume_s sectie.nume_sectie%TYPE;
    BEGIN
        SELECT nume_sectie INTO nume_s
        FROM sectie
        WHERE id_sectie in (
            SELECT id_sectie
            FROM (
                SELECT aux.id_sectie
                FROM (
                    SELECT COUNT(data_start) nr, c.id_sectie id_sectie
                    FROM imprumut i, carte c

```

```

WHERE i.id_abonament = (
    SELECT id_abonament
    FROM abonament a, cititor ci
    WHERE a.id_cititor = ci.id_cititor
    AND ci.nume = v_nume
)

AND i.id_carte = c.id_carte
GROUP BY c.id_sectie
) aux
WHERE aux.nr IN (
    SELECT MAX (nr)
    FROM (
        SELECT COUNT(data_start) nr, c.id_sectie
        id_sectie
        FROM imprumut i, carte c
        WHERE i.id_abonament = (
            SELECT id_abonament
            FROM abonament a, cititor ci
            WHERE a.id_cititor = ci.id_cititor
            AND ci.nume = v_nume
        )

        AND i.id_carte = c.id_carte
        GROUP BY c.id_sectie
    )
)

);

DBMS_OUTPUT.PUT_LINE('Sectia este '|| nume_s);
EXCEPTION
    WHEN NO_DATA_FOUND THEN
RAISE_APPLICATION_ERROR(-20000, 'Nu exista imprumuturi facute de
acest cititor!');
    WHEN TOO_MANY_ROWS THEN
RAISE_APPLICATION_ERROR(-20001, 'Exista mai multe sectii cu acelasi
numar de carti!');
    WHEN OTHERS THEN RAISE_APPLICATION_ERROR(-20002,'Alta
eroare!');

```

```
END;  
EXCEPTION  
WHEN NO_DATA_FOUND THEN RAISE_APPLICATION_ERROR(-20000,  
'Nu exista cititori cu numele dat!');  
WHEN TOO_MANY_ROWS THEN RAISE_APPLICATION_ERROR(-20001,  
'Exista mai multi cititori cu numele dat!');  
END f4_biblioteca;  
/  
  
BEGIN  
f4_biblioteca('Dinu');  
--f4_biblioteca('Radu');  
--f4_biblioteca('Cirstea');  
--f4_biblioteca('Udrea');  
--f4_biblioteca('Oprea');  
END;  
/  
  
9.1. Apelare pentru un nume existent.
```

The screenshot shows a web-based SQL editor interface. At the top, there's a dark header with a hamburger menu, a "Live SQL" logo, and links for Feedback, Help, and a user profile (barbu.mary54@yahoo.com). Below the header, the main area is titled "SQL Worksheet" and contains a code editor with a PL/SQL block. The code is as follows:

```
318 WHEN TOO_MANY_ROWS THEN RAISE_APPLICATION_ERROR(-20001, 'Exista mai multi cititori cu numele dat!');  
319 END f4_biblioteca;  
320 /  
321  
322  
323  
324 BEGIN  
325 f4_biblioteca('Dinu');  
326 --f4_biblioteca('Radu');  
327 --f4_biblioteca('Cirstea');  
328 --f4_biblioteca('Udrea');  
329 --f4_biblioteca('Oprea');  
330 END;  
331 /  
332  
333  
334  
335  
336
```

Below the code editor, there's a results pane showing the output of the execution:

```
Statement processed.  
Sectia este Matematica
```

9.2. Apelare pentru un cititor care nu exista.



```
318 WHEN TOO_MANY_ROWS THEN RAISE_APPLICATION_ERROR(-20001, 'Exista mai multi cititori cu numele dat!');
319 END f4_biblioteca;
320 /
321
322
323
324 BEGIN
325 --f4_biblioteca('Dinu');
326 f4_biblioteca('Radu');
327 --f4_biblioteca('Cirstea');
328 --f4_biblioteca('Udrea');
329 --f4_biblioteca('Oprea');
330 END;
331 /
332
333
334
335
336
```

ORA-20000: Nu exista cititori cu numele dat! ORA-06512: at "SQL_SVGMAUGFJAKGXENCLMRYZQQV.F4_BIBLIOTECA", line 49
ORA-06512: at line 3
ORA-06512: at "SYS.DBMS_SQL", line 1721

9.3. Apelare pentru un cititor cu nume ambiguu.



```
318 WHEN TOO_MANY_ROWS THEN RAISE_APPLICATION_ERROR(-20001, 'Exista mai multi cititori cu numele dat!');
319 END f4_biblioteca;
320 /
321
322
323
324 BEGIN
325 --f4_biblioteca('Dinu');
326 --f4_biblioteca('Radu');
327 f4_biblioteca('Cirstea');
328 --f4_biblioteca('Udrea');
329 --f4_biblioteca('Oprea');
330 END;
331 /
332
333
334
335
336
```

ORA-20001: Exista mai multi cititori cu numele dat! ORA-06512: at "SQL_SVGMAUGFJAKGXENCLMRYZQQV.F4_BIBLIOTECA", line 50
ORA-06512: at line 4
ORA-06512: at "SYS.DBMS_SQL", line 1721

9.4. Apelare pentru un cititor care a imprumutat un numar egal de cărți de la mai multe secții.



```
318 WHEN TOO_MANY_ROWS THEN RAISE_APPLICATION_ERROR(-20001, 'Exista mai multi cititori cu numele dat!');
319 END f4_biblioteca;
320 /
321
322
323
324 BEGIN
325 --f4_biblioteca('Dinu');
326 --f4_biblioteca('Radu');
327 --f4_biblioteca('Cirstea');
328 f4_biblioteca('Udrea');
329 --f4_biblioteca('Oprea');
330 END;
331 /
332
333
334
335
336
```

ORA-20001: Exista mai multe sectii de acelasi numar de cartii ORA-06512: at "SQL_SVGMAUGFJAKGXENCLMRYZQQV.F4_BIBLIOTECA", line 45
ORA-06512: at line 5
ORA-06512: at "SYS.DBMS_SQL", line 1721

9.5. Apelare pentru un cititor care nu a realizat imprumuturi.

```

318 WHEN TOO_MANY_ROWS THEN RAISE_APPLICATION_ERROR(-20001, 'Exista mai multi cititori cu numele dat!');
319 END f4_biblioteca;
320 /
321
322
323
324 BEGIN
325 --f4_biblioteca('Dinu');
326 --f4_biblioteca('Radu');
327 --f4_biblioteca('Cirstea');
328 --f4_biblioteca('Udrea');
329 f4_biblioteca('Oprea');
330 END;
331 /
332
333
334
335
336

```

ORA-20000: Nu exista imprumuturi facute de acest cititor! ORA-06512: at "SQL_SVGPAUGFJAKGXENCLWRYZQV.F4_BIBLIOTECA", line 44
ORA-06512: at line 6
ORA-06512: at "SYS.DBMS_SQL", line 1721

10. Definiti un trigger care sa permita inserarea si stergerea in tabelul angajat doar in zilele lucratoare(sambata si duminica nu se fac angajari sau concedieri), intre orele 08-18. Declansati trigger ul.

```

CREATE OR REPLACE TRIGGER trig1_biblioteca
BEFORE INSERT OR DELETE ON angajat
BEGIN
    IF ((TO_CHAR(SYSDATE,'D') = 1) OR (TO_CHAR(SYSDATE,'D') = 7)) OR
    (TO_CHAR(SYSDATE,'HH24') NOT BETWEEN 8 AND 18)
    THEN RAISE_APPLICATION_ERROR(-20001,'Nu se fac angajari sau
concedieri!');
END IF;
END;
/
ALTER TRIGGER trig1_biblioteca ENABLE;

```

```

insert into angajat
values (6, 'Dragnea', 'Alex', 2000, '15-mar-1970', 4);

```

10.1. Inserare în zilele lucratoare:

```

265
266
267 CREATE OR REPLACE TRIGGER trigi_biblioteca
268 BEFORE INSERT OR DELETE ON angajat
269 BEGIN
270     IF ((TO_CHAR(SYSDATE,'D') = 1) OR (TO_CHAR(SYSDATE,'D') = 7)) OR (TO_CHAR(SYSDATE,'HH24') NOT BETWEEN 8 AND 18)
271     THEN RAISE_APPLICATION_ERROR(-20001,'Nu se fac angajari sau concedieri!');
272 END IF;
273 END;
274 /
275 ALTER TRIGGER trigi_biblioteca ENABLE;
276
277 insert into angajat
278 values (6, 'Dragnea', 'Alex', 2000, '15-mar-1970', 4);
279
280
281

```

Trigger created.

Trigger altered.

1 row(s) inserted.

10.2. Inserare în weekend:

```

265
266 --10. Definiti un trigger care sa permita inserarea si stergerea in tabelul angajat doar in zilele lucratoare( sambata si duminica nu se fac angajari sau concedieri), intre orele 08-18. Declansati trigger ul.
267
268 CREATE OR REPLACE TRIGGER trigi_biblioteca
269 BEFORE INSERT OR DELETE ON angajat
270 BEGIN
271     IF ((TO_CHAR(SYSDATE,'D') = 1) OR (TO_CHAR(SYSDATE,'D') = 7)) OR (TO_CHAR(SYSDATE,'HH24') NOT BETWEEN 8 AND 18)
272     THEN RAISE_APPLICATION_ERROR(-20001,'Nu se fac angajari sau concedieri!');
273 END IF;
274 END;
275 /
276 ALTER TRIGGER trigi_biblioteca ENABLE;
277
278 insert into angajat
279 values (6, 'Dragnea', 'Alex', 2000, '15-mar-1970', 4);
280
281

```

Trigger created.

Trigger altered.

ORA-20001: Nu se fac angajari sau concedieri! ORA-00512: at "SQL_CHECKYUHC7ZNGE3ITRCVAVV.TRIGI_BIBLIOTECA", line 3
ORA-00512: at "SYS.DBMS_SQL", line 1721

11. Definiti un trigger care va actualiza automat campul nr_carti dintr-o sectie atunci cand se adauga sau se sterge o carte in sectia respectiva. Declansati trigger-ul.

```

CREATE OR REPLACE PROCEDURE modific_nr_carti_biblioteca
(v_id_sectie sectie.id_sectie%TYPE,
aux int ) AS
BEGIN
UPDATE sectie
SET nr_carti = NVL (nr_carti, 0) + aux
WHERE id_sectie = v_id_sectie;
END;
/

```

```
CREATE OR REPLACE TRIGGER trig2_biblioteca
AFTER DELETE OR INSERT ON carte
FOR EACH ROW
BEGIN
  IF DELETING THEN
    modific_nr_carti_biblioteca (:OLD.id_sectie, -1);
  ELSE
    modific_nr_carti_biblioteca (:NEW.id_sectie, 1);
  END IF;
END;
/
```

```
ALTER TRIGGER trig2_biblioteca ENABLE;
```

```
select * from sectie;
```

```
insert into carte
```

```
values (8, 'Alexandru Sorin', 'Culegere ecuatii', '2000', 2);
```

```
select * from sectie;
```

11.1. Am creat și declansat trigger-ul.



The screenshot shows a web-based SQL editor titled "Live SQL". The interface includes a top navigation bar with a menu icon, the text "Live SQL", and links for "Feedback", "Help", and a user profile "barbu.mary54@yahoo.com". Below the navigation bar is a toolbar with buttons for "Clear", "Find", "Actions", "Save", and a green "Run" button. The main area is labeled "SQL Worksheet" and contains a text editor with SQL code. The code is as follows:

```
285 END IF;
286 END;
287 /
288
289 ALTER TRIGGER trig2_biblioteca ENABLE;
290
291
292 select * from sectie;
293
294 insert into carte
295 values (8, 'Alexandru Sorin', 'Culegere ecuatii', '2000', 2);
296
297 select * from sectie;
298
299
300
```

Below the code editor, there is a results pane showing the output of the executed SQL statements:

```
Procedure created.

Trigger created.

Trigger altered.
```


11.2. Numarul de cărți din secții înainte de a insera inca o carte:

Live SQL

SQL Worksheet

```

287 /
288
289
290 ALTER TRIGGER trig2_biblioteca ENABLE;
291
292 select * from sectie;
293
294 insert into carte
295 values (8, 'Alexandru sorin', 'Culegere ecuatii', '2000', 2);
296
297 select * from sectie;
298

```

ID_SECTIE	NUME_SECTIE	NR_CARTI
1	Beletristica	10000
2	Matematica	2000
3	poezii	7540
4	Stiintele naturii	1230
5	Copii	3480

11.3. Numarul de cărți dupa ce am inserat o carte:

Live SQL

SQL Worksheet

```

287 /
288
289
290 ALTER TRIGGER trig2_biblioteca ENABLE;
291
292 select * from sectie;
293
294 insert into carte
295 values (8, 'Alexandru sorin', 'Culegere ecuatii', '2000', 2);
296
297 select * from sectie;
298
299
300

```

1 row(s) inserted.

ID_SECTIE	NUME_SECTIE	NR_CARTI
1	Beletristica	10000
2	Matematica	2001
3	poezii	7540
4	Stiintele naturii	1230

12. Creati tabelul info_biblioteca cu urmatoarele campuri : eveniment(eventimentul sistem), nume_obiect(numele obiectului) si data (data producerii evenimentului), iar apoi definiti un trigger care sa introduca date in acest tabel dupa ce utilizatorul a folosit o comanda LDD.

```

CREATE TABLE info_biblioteca
(eventiment VARCHAR2(20),
nume_obiect VARCHAR2(30),
data DATE);

```

```
CREATE OR REPLACE TRIGGER trig3_biblioteca
AFTER CREATE OR DROP OR ALTER ON SCHEMA
BEGIN
INSERT INTO info_biblioteca
VALUES (SYS.SYSEVENT,
SYS.DICTIONARY_OBJ_NAME, SYSDATE);
END;
/
CREATE INDEX ind_biblioteca ON cititor(ume);
DROP INDEX ind_biblioteca;
SELECT * FROM info_biblioteca;
DROP TRIGGER trig3_biblioteca;
```

12.1. Am creat trigger-ul.

Live SQL

SQL Worksheet

```
266 CREATE TABLE info_biblioteca
267 (eventiment VARCHAR2(20),
268 nume_obiect VARCHAR2(30),
269 data DATE);
270
271
272 CREATE OR REPLACE TRIGGER trig3_biblioteca
273 AFTER CREATE OR DROP OR ALTER ON SCHEMA
274 BEGIN
275 INSERT INTO info_biblioteca
276 VALUES (SYS.SYSEVENT,
277 SYS.DICTIONARY_OBJ_NAME, SYSDATE);
278 END;
279 /
280 CREATE INDEX ind_biblioteca ON cititor(ume);
281 DROP INDEX ind_biblioteca;
282 SELECT * FROM info_biblioteca;
283 DROP TRIGGER trig3_biblioteca;
```

Table created.

Trigger created.

12.2. Inainte de orice comanda LDD tabelul info_biblioteca era gol.

Live SQL

SQL Worksheet

```
269 nume_obiect VARCHAR2(30),
270 data DATE);
271
272 CREATE OR REPLACE TRIGGER trig3_biblioteca
273 AFTER CREATE OR DROP OR ALTER ON SCHEMA
274 BEGIN
275 INSERT INTO info_biblioteca
276 VALUES (SYS.SYSEVENT,
277 SYS.DICTIONARY_OBJ_NAME, SYSDATE);
278 END;
279 /
280 CREATE INDEX ind_biblioteca ON cititor(ume);
281 DROP INDEX ind_biblioteca;
282 SELECT * FROM info_biblioteca;
283 DROP TRIGGER trig3_biblioteca;
```

no data found

12.3. Am rulat de doua ori cele doua comenzi LDD și am afisat datele din tabelul info_biblioteca.

The screenshot shows a 'Live SQL' interface with a 'SQL Worksheet' tab. The SQL commands entered are:

```

279 /
280 CREATE INDEX ind_biblioteca ON cititor(ume);
281 DROP INDEX ind_biblioteca;
282 SELECT * FROM info_biblioteca;
283 DROP TRIGGER trig_biblioteca;
284
285
286

```

The results section shows two messages: 'Index created.' and 'Index dropped.' Below these is a table with the following data:

EVENIMENT	NUME_OBIECT	DATA
CREATE	IND_BIBLIOTECA	31-DEC-20
DROP	IND_BIBLIOTECA	31-DEC-20
CREATE	IND_BIBLIOTECA	31-DEC-20
DROP	IND_BIBLIOTECA	31-DEC-20

At the bottom, there is a 'Download CSV' link.

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

```

CREATE OR REPLACE PACKAGE pachet1_biblioteca AS
    PROCEDURE f1_biblioteca (v_id_cititor cititor.id_cititor%TYPE );
    PROCEDURE f2_biblioteca (v_id_sectie sectie.id_sectie%TYPE );
    FUNCTION f3_biblioteca (v_id_abonament
abonament.id_abonament%TYPE DEFAULT 'Bell') RETURN VARCHAR2;
    PROCEDURE f4_biblioteca (v_ume cititor.ume%TYPE);
    PROCEDURE modific_nr_carti_biblioteca (v_id_sectie
sectie.id_sectie%TYPE, aux int );
END pachet1_biblioteca;
/

```

```

CREATE OR REPLACE PACKAGE BODY pachet1_biblioteca AS
    PROCEDURE f1_biblioteca
    (v_id_cititor cititor.id_cititor%TYPE )
    IS ume cititor.ume%type;
    BEGIN
    SELECT ume INTO ume
    FROM cititor
    WHERE id_cititor = v_id_cititor;
    DECLARE
        TYPE carti IS TABLE OF INT INDEX BY PLS_INTEGER;

```

```
x carti;
titlu_carte carte.titlu%type;
BEGIN
  select c.id_carte
  BULK COLLECT INTO x
  from carte c, imprumut i, abonament a
  where c.id_carte = i.id_carte and i.id_abonament = a.id_abonament and
a.id_cititor = v_id_cititor
  and i.data_start > sysdate - 365;

  IF x.first IS NULL THEN DBMS_OUTPUT.PUT_LINE('Nu exista
imprumuturi! ');
  ELSE
    DBMS_OUTPUT.PUT_LINE('Cartile imprumutate de cititorul ' ||
v_id_cititor || ': ');
    FOR i IN x.first..x.last LOOP
      SELECT titlu
      INTO titlu_carte
      from carte
      where carte.id_carte = x(i);
      DBMS_OUTPUT.PUT_LINE(titlu_carte);
    END LOOP;
  END IF;
END;
EXCEPTION
  WHEN NO_DATA_FOUND THEN RAISE_APPLICATION_ERROR(-20000,
'Nu exista cititori cu id-ul dat');
  WHEN OTHERS THEN RAISE_APPLICATION_ERROR(-20002,'Alta
eroare!');

END f1_biblioteca;
```

```
PROCEDURE f2_biblioteca
(v_id_sectie sectie.id_sectie%TYPE )
IS nume_sectie.nume_sectie%type;
BEGIN
  SELECT nume_sectie INTO nume
  FROM sectie
```

```
WHERE id_sectie = v_id_sectie;
DBMS_OUTPUT.PUT_LINE('Sectia ' || nume || ': ');
DECLARE
    v_nr number(4) :=0;
    CURSOR cu IS
    SELECT titlu, autor
    FROM carte c, sectie s
    WHERE c.id_sectie = s.id_sectie
    AND s.id_sectie = v_id_sectie;
BEGIN
    FOR j in cu LOOP
        v_nr := v_nr + 1;
        DBMS_OUTPUT.PUT_LINE('Cartea ' || j.titlu || ' scrisa de ' || j.autor);
    END LOOP;
    IF v_nr = 0 THEN DBMS_OUTPUT.PUT_LINE('Nu exista carti!');
    END IF;
END;
EXCEPTION
    WHEN NO_DATA_FOUND THEN RAISE_APPLICATION_ERROR(-20000,
    'Nu exista sectii cu id-ul dat');
    WHEN OTHERS THEN RAISE_APPLICATION_ERROR(-20002,'Alta
eroare!');
END f2_biblioteca;
```

```
FUNCTION f3_biblioteca
(v_id_abonament abonament.id_abonament%TYPE DEFAULT 'Bell')
RETURN VARCHAR2 IS
oras adresa.oras%type;
BEGIN
    SELECT oras INTO oras
    FROM adresa ad, cititor c, abonament ab
    WHERE ab.id_abonament = v_id_abonament AND ad.id_adresa =
c.id_adresa AND c.id_cititor = ab.id_cititor;
    RETURN oras;
EXCEPTION
    WHEN NO_DATA_FOUND THEN RAISE_APPLICATION_ERROR(-20000,
    'Nu exista abonamente cu id-ul dat');
    WHEN OTHERS THEN RAISE_APPLICATION_ERROR(-20002,'Alta
eroare!');
```

[illegible]

```
AND ci.num =
v_num)

AND i.id_carte = c.id_carte
GROUP BY c.id_sectie
)
)
)

);

DBMS_OUTPUT.PUT_LINE('Sectia este '|| num_s);
EXCEPTION
WHEN NO_DATA_FOUND THEN
RAISE_APPLICATION_ERROR(-20000, 'Nu exista imprumuturi facute de
acest cititor!');
WHEN TOO_MANY_ROWS THEN
RAISE_APPLICATION_ERROR(-20001, 'Exista mai multe sectii cu acelasi
numar de carti!');
WHEN OTHERS THEN RAISE_APPLICATION_ERROR(-20002,'Alta
eroare!');
END;
EXCEPTION
WHEN NO_DATA_FOUND THEN RAISE_APPLICATION_ERROR(-20000,
'Nu exista cititori cu numele dat!');
WHEN TOO_MANY_ROWS THEN RAISE_APPLICATION_ERROR(-20001,
'Exista mai multi cititori cu numele dat!');
END f4_biblioteca;

PROCEDURE modific_nr_carti_biblioteca
(v_id_sectie sectie.id_sectie%TYPE,
aux int ) AS
BEGIN
UPDATE sectie
SET nr_carti = NVL (nr_carti, 0) + aux
WHERE id_sectie = v_id_sectie;
END;
```

```
END pachet1_biblioteca;
/
```

```
BEGIN
  pachet1_biblioteca.f1_biblioteca(1);
  pachet1_biblioteca.f2_biblioteca(2);
  --DBMS_OUTPUT.PUT_LINE('Orasul este '||
  pachet1_biblioteca.f3_biblioteca(9));
  DBMS_OUTPUT.PUT_LINE('Orasul este '||
  pachet1_biblioteca.f3_biblioteca(2));
  pachet1_biblioteca.f4_biblioteca('Dinu');
  --pachet1_biblioteca.f4_biblioteca('Radu');
  --pachet1_biblioteca.f4_biblioteca('Cirstea');
END;
/
```

Am creat pachetul :

The screenshot shows a Live SQL interface with a dark header bar containing a menu icon, the text "Live SQL", and links for Feedback, Help, and a user profile. Below the header is a "SQL Worksheet" section with a toolbar (Clear, Find, Actions, Save, Run). The main area displays SQL code for creating a package and its body. The package body is created successfully, as indicated by the output messages.

```

266 CREATE OR REPLACE PACKAGE pachet1_biblioteca AS
267   PROCEDURE f1_biblioteca (v_id_cititor cititor.id_cititor%TYPE );
268   PROCEDURE f2_biblioteca (v_id_sectie sectie.id_sectie%TYPE );
269   FUNCTION f3_biblioteca (v_id_abonament abonament.id_abonament%TYPE DEFAULT 'Bell') RETURN VARCHAR2;
270   PROCEDURE f4_biblioteca (v_nume cititor.nume%TYPE);
271   PROCEDURE modific_nr_carti_biblioteca (v_id_sectie sectie.id_sectie%TYPE, aux int );
272 END pachet1_biblioteca;
273 /
274
275 CREATE OR REPLACE PACKAGE BODY pachet1_biblioteca AS
276   PROCEDURE f1_biblioteca
277     (v_id_cititor cititor.id_cititor%TYPE )
278     IS nume cititor.nume%TYPE;
279   BEGIN
280     SELECT nume INTO nume
281     FROM cititor
  
```

Package created.

Package Body created.

Și am rulat pentru mai multe cazuri:

The screenshot shows the same Live SQL interface, but now the SQL code is being executed. The output shows the results of the package functions, including the list of books borrowed by a reader, the section of the book, and the city.

```

426 DBMS_OUTPUT.PUT_LINE(' ');
427 pachet1_biblioteca.f2_biblioteca(2);
428 DBMS_OUTPUT.PUT_LINE(' ');
429 --DBMS_OUTPUT.PUT_LINE('Orasul este '|| pachet1_biblioteca.f3_biblioteca(9));
430 DBMS_OUTPUT.PUT_LINE('Orasul este '|| pachet1_biblioteca.f3_biblioteca(2));
431 DBMS_OUTPUT.PUT_LINE(' ');
432 pachet1_biblioteca.f4_biblioteca('Dinu');
433 --pachet1_biblioteca.f4_biblioteca('Radu');
434 --pachet1_biblioteca.f4_biblioteca('Cirstea');
435 END;
436 /
  
```

Statement processed.

Cartile imprumutate de cititorul 1:

Poezii

Ultima noapte de dragoste

Culegere Matematica

Sectia Matematica:

Cartea Culegere Matematica scrisa de Marius Perianu

Orasul este Pitesti

Sectia este Matematica


```
Live SQL
Feedback Help barbu.mary54@yahoo.com
SQL Worksheet
Clear Find Actions Save Run
422
423 BEGIN
424 -- pacheti_biblioteca.f1_biblioteca(1);
425 -- DBMS_OUTPUT.PUT_LINE(' ');
426 -- pacheti_biblioteca.f2_biblioteca(2);
427 -- DBMS_OUTPUT.PUT_LINE(' ');
428 -- DBMS_OUTPUT.PUT_LINE('Orasul este ' || pacheti_biblioteca.f3_biblioteca(9));
429 -- DBMS_OUTPUT.PUT_LINE('Orasul este ' || pacheti_biblioteca.f3_biblioteca(2));
430 -- DBMS_OUTPUT.PUT_LINE(' ');
431 -- pacheti_biblioteca.f4_biblioteca('Dinu');
432 -- pacheti_biblioteca.f4_biblioteca('Radu');
433 -- pacheti_biblioteca.f4_biblioteca('Cirstea');
434 END;
435 /
436
ORA-20000: Nu exista abonamente cu id-ul dat ORA-06512: at "SQL_SVGMAUGFJAKGXENCLMRYZQQV.PACHET1_BIBLIOTECA", line 79
ORA-06512: at line 6
ORA-06512: at "SYS.DBMS_SQL", line 1721
```

```
Live SQL
Feedback Help barbu.mary54@yahoo.com
SQL Worksheet
Clear Find Actions Save Run
421 /
422
423 BEGIN
424 -- pacheti_biblioteca.f1_biblioteca(1);
425 -- DBMS_OUTPUT.PUT_LINE(' ');
426 -- pacheti_biblioteca.f2_biblioteca(2);
427 -- DBMS_OUTPUT.PUT_LINE(' ');
428 -- DBMS_OUTPUT.PUT_LINE('Orasul este ' || pacheti_biblioteca.f3_biblioteca(9));
429 -- DBMS_OUTPUT.PUT_LINE('Orasul este ' || pacheti_biblioteca.f3_biblioteca(2));
430 -- DBMS_OUTPUT.PUT_LINE(' ');
431 -- pacheti_biblioteca.f4_biblioteca('Dinu');
432 -- pacheti_biblioteca.f4_biblioteca('Radu');
433 -- pacheti_biblioteca.f4_biblioteca('Cirstea');
434 END;
435 /
436
ORA-20000: Nu exista cititori cu numele dat! ORA-06512: at "SQL_SVGMAUGFJAKGXENCLMRYZQQV.PACHET1_BIBLIOTECA", line 131
ORA-06512: at line 10
ORA-06512: at "SYS.DBMS_SQL", line 1721
```

```
Live SQL
Feedback Help barbu.mary54@yahoo.com
SQL Worksheet
Clear Find Actions Save Run
421 /
422
423 BEGIN
424 -- pacheti_biblioteca.f1_biblioteca(1);
425 -- DBMS_OUTPUT.PUT_LINE(' ');
426 -- pacheti_biblioteca.f2_biblioteca(2);
427 -- DBMS_OUTPUT.PUT_LINE(' ');
428 -- DBMS_OUTPUT.PUT_LINE('Orasul este ' || pacheti_biblioteca.f3_biblioteca(9));
429 -- DBMS_OUTPUT.PUT_LINE('Orasul este ' || pacheti_biblioteca.f3_biblioteca(2));
430 -- DBMS_OUTPUT.PUT_LINE(' ');
431 -- pacheti_biblioteca.f4_biblioteca('Dinu');
432 -- pacheti_biblioteca.f4_biblioteca('Radu');
433 -- pacheti_biblioteca.f4_biblioteca('Cirstea');
434 END;
435 /
436
ORA-20001: Exista mai multi cititori cu numele dat! ORA-06512: at "SQL_SVGMAUGFJAKGXENCLMRYZQQV.PACHET1_BIBLIOTECA", line 132
ORA-06512: at line 11
ORA-06512: at "SYS.DBMS_SQL", line 1721
```

14. Creati un pachet care sa contina o procedura ce returneaza numele, prenumele si email-ul cititorilor care si-au facut abonament inainte de o data specificata ca si parametru.

```
CREATE OR REPLACE PACKAGE pachet2_biblioteca AS
  PROCEDURE abonamente
    (v_data abonament.data_start%TYPE);
END pachet2_biblioteca;
/

CREATE OR REPLACE PACKAGE BODY pachet2_biblioteca AS
  PROCEDURE abonamente
    (v_data abonament.data_start%TYPE)
  AS
  BEGIN
    DECLARE
      TYPE cititor_record IS RECORD
        (nume cititor.nume%TYPE,
         prenume cititor.prenume%TYPE,
         email cititor.email%TYPE);
      v_cititor cititor_record;

      BEGIN
        DECLARE
          TYPE tablou_cititori IS TABLE OF NUMBER
            INDEX BY BINARY_INTEGER;
          t tablou_cititori;
        BEGIN
          SELECT c.id_cititor
          BULK COLLECT INTO t
          FROM cititor c, abonament a
          WHERE c.id_cititor = a.id_cititor
          AND a.data_start < v_data;

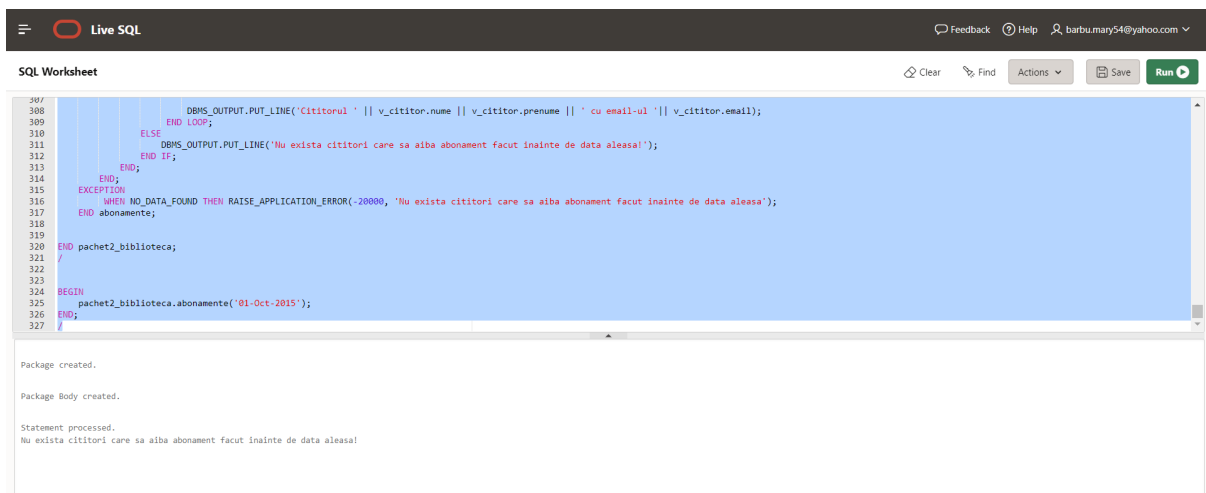
          IF t.COUNT > 0 THEN
            FOR i in t.first..t.last LOOP
              SELECT nume, prenume, email
              INTO v_cititor
              FROM cititor
              WHERE t(i) = id_cititor;

              DBMS_OUTPUT.PUT_LINE('Cititorul ' || v_cititor.nume ||
v_cititor.prenume || ' cu email-ul ' || v_cititor.email);
            END LOOP;
          ELSE
```

```
        DBMS_OUTPUT.PUT_LINE('Nu exista cititori care sa aiba  
abonament facut inainte de data aleasa!');  
    END IF;  
    END;  
    END;  
    EXCEPTION  
        WHEN NO_DATA_FOUND THEN  
        RAISE_APPLICATION_ERROR(-20000, 'Nu exista cititori care sa aiba  
abonament facut inainte de data aleasa');  
    END abonamente;  
  
END pachet2_biblioteca;  
/  
  
BEGIN  
    pachet2_biblioteca.abonamente('01-Oct-2019');  
END;  
/  

```

14.1. Am apelat procedura pentru o data inainte de care nu exista abonamente create.



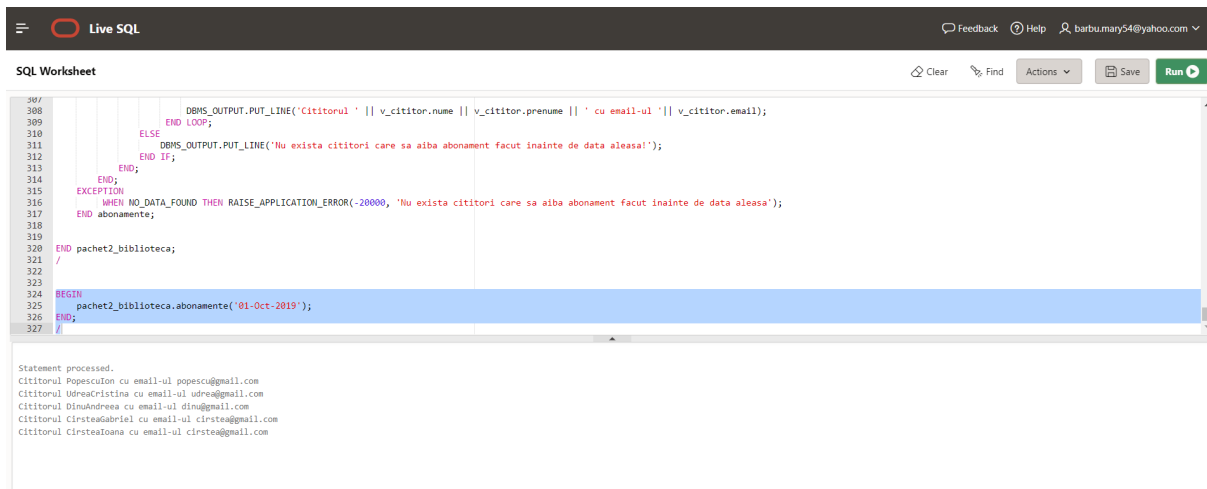
```
Live SQL
```

SQL Worksheet

```
307  
308         DBMS_OUTPUT.PUT_LINE('Cititorul ' || v_cititor.nume || v_cititor.prenume || ' cu email-ul ' || v_cititor.email);  
309     END LOOP;  
310     ELSE  
311         DBMS_OUTPUT.PUT_LINE('Nu exista cititori care sa aiba abonament facut inainte de data aleasa!');  
312     END IF;  
313 END;  
314 END;  
315 EXCEPTION  
316     WHEN NO_DATA_FOUND THEN RAISE_APPLICATION_ERROR(-20000, 'Nu exista cititori care sa aiba abonament facut inainte de data aleasa');  
317 END abonamente;  
318  
319 END pachet2_biblioteca;  
320 /  
321  
322  
323 BEGIN  
324     pachet2_biblioteca.abonamente('01-Oct-2015');  
325 END;  
326 /  
327
```

Package created.
Package Body created.
Statement processed.
Nu exista cititori care sa aiba abonament facut inainte de data aleasa!

14.2. Am apelat procedura pentru o data care returneaza mai multe valori.



The screenshot shows a web-based SQL editor titled "Live SQL". The interface includes a top navigation bar with a menu icon, the "Live SQL" logo, and links for "Feedback", "Help", and a user profile "barbu.mary54@yahoo.com". Below the navigation bar is a toolbar with "Clear", "Find", "Actions", "Save", and a "Run" button. The main area is labeled "SQL Worksheet" and contains an SQL script. The script is a PL/SQL block that defines a procedure to insert subscription data. It includes a loop to process multiple rows, an exception handler for data not found, and a final call to the procedure. The script is as follows:

```
307/
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
DBMS_OUTPUT.PUT_LINE('Cititorul ' || v_cititor.nume || v_cititor.prenume || ' cu email-ul ' || v_cititor.email);
END LOOP;
ELSE
DBMS_OUTPUT.PUT_LINE('Nu exista cititori care sa aiba abonament facut inainte de data aleasa');
END IF;
END;
EXCEPTION
WHEN NO_DATA_FOUND THEN RAISE_APPLICATION_ERROR(-20000, 'Nu exista cititori care sa aiba abonament facut inainte de data aleasa');
END abonamente;
END pachet2_biblioteca;
/
BEGIN
pachet2_biblioteca.abonamente('01-Oct-2019');
END;
```

Below the script, the execution results are displayed:

Statement processed.
Cititorul Popescu Ion cu email-ul popescu@gmail.com
Cititorul UdreaCristina cu email-ul udrea@gmail.com
Cititorul DinuAndreea cu email-ul dinu@gmail.com
Cititorul CirsteaGabriel cu email-ul cirstea@gmail.com
Cititorul CirsteaIoana cu email-ul cirstea@gmail.com

Proiect realizat de

Barbu Mariana
GRUPA 233

Sistemul de gestiune al bazelor de date
Prof. Gabriela Mihai