

GESTIONAREA LIBRARIILOR

Anul II, semestrul I, 2021-2022

Predescu Denisa, grupa 242

1. Baza de date gestioneaza datele unor librarii. Aceasta retine informatii referitoare la carti, categorii, scriitori, seriile din care sunt compuse cartile, angajati, bonurile emise, librarii si locatia acestora, precum si stocul de cartile ce se gasesti la fiecare librerie.

Cartile sunt incadrate in diverse categorii si pot fi scrise de unul sau mai multi scriitori, precizandu-se ca si acestia din urma pot avea mai multe carti scrise. O carte poate fi inclusa in cel mult o serie, dar nu este obligatoriu sa faca parte din una.

Angajati sunt de doua tipuri: vanzatori si ingrijitori. Vanzatorii sunt cei care vand cartile, iar bonurile emise sunt si ele retinute. Se precizeaza ca un angajat poate sa vanda de mai multe ori aceeasi carte, in aceeasi zi, aceasta cat timp cartea respective se gaseste la libraria de la care se doreste cumpararea (daca se gaseste pe stoc). Stocul este actualizat dupa fiecare vanzare.

Baza de date contine urmatoarele tabele:

SERIE (cod_serie#, nume)

ARE (cod_scriitor#, cod_carte#, cod_categorie#)

SCRIITOR (cod_scriitor#, nume, prenume, nationalitate, sex, data_nastere, data_deces)

CARTE (cod_carte#, denumire, numar_pagini, pret, an_aparitie, numar_volum, cod_serie)

CATEGORIE (cod_categorie#, gen)

BON (cod_bon #, cod_carte, cod_angajat, data_bon, ora_bon)

ANGAJAT (cod_angajat#, nume, prenume, salariu, telefon, sex, tip_angajat, cod_librarie)

VANZATOR (cod_angajat#, nume, prenume, salariu, telefon, sex, tip_angajat, cod_librarie)

INGRIJITOR (cod_angajat#, nume, prenume, salariu, telefon, sex, tip_angajat, cod_librarie)

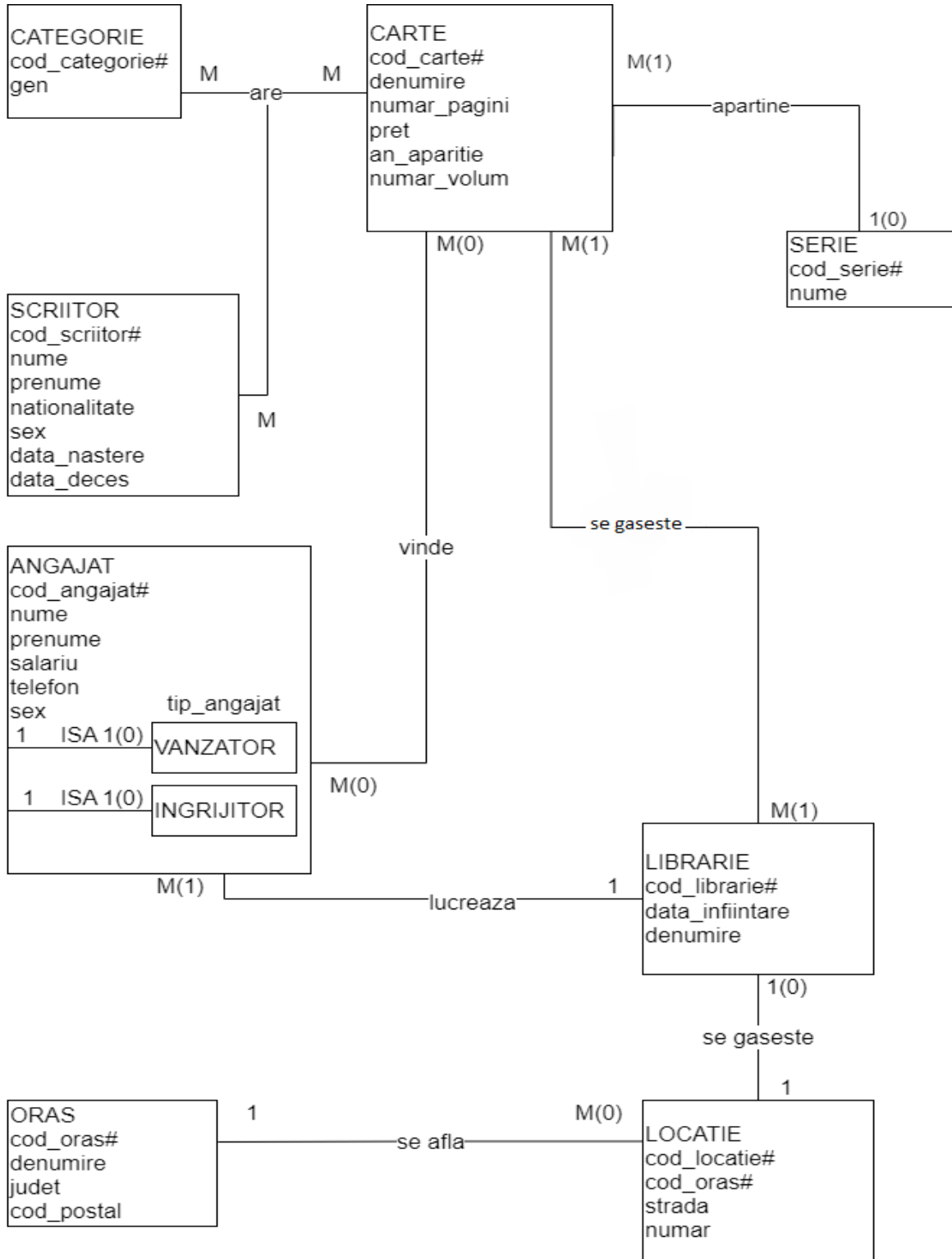
LIBRARIE (cod_librarie#, data_infiintare, denumire, cod_locatie, cod_oras)

LOCATIE (cod_locatie#, cod_oras#, strada, numar)

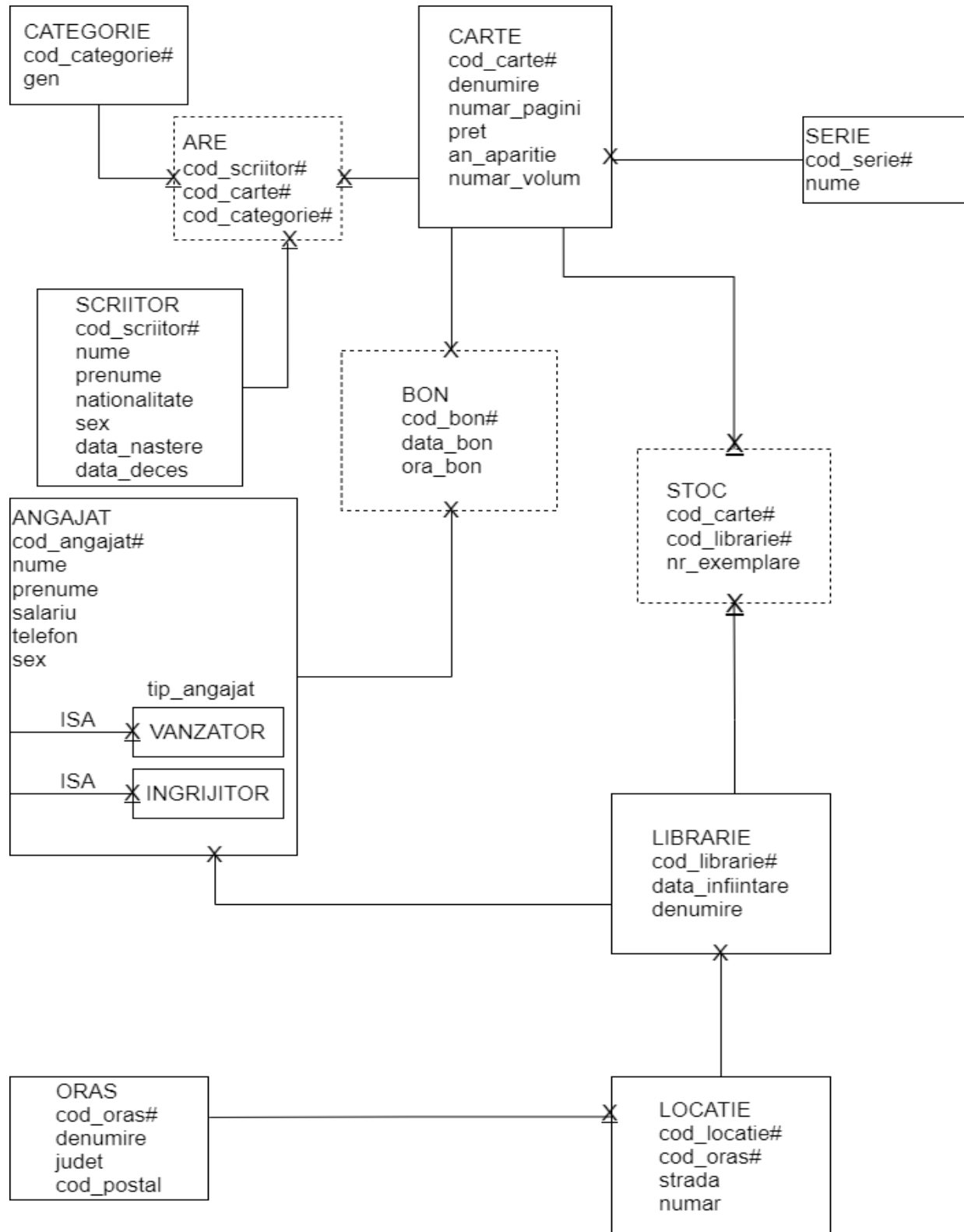
ORAS (cod_oras#, denumire, judet, cod_postal)

STOC(cod_carte#, cod_librarie#, nr_exemplare)

2. ERD



3. Diagrama conceptuala



4. si 5.

Privitor la tabelul asociativ BON rezultat in urma unei relatii de tip 2 dintre ANGAJAT si CARTE: acesta primeste o cheie artificiala din cauza faptului ca identificarea unei inregistrari s-ar fi facut prin prea multe atribute in absenta cheii artificiale. Aceasta deoarece o carte se poate vinde de acelasi vanzator in aceeasi zi, dar nu si in acelasi moment din zi (minut). Asadar cheia primara ar fi fost compusa din 4 atribute, ceea ce ar fi insemnat lipsa de eficienta.

--Scriitori

Script Output x Query Result x Query Result 1 x Query Result 2 x Query Result 3 x Query Result 4 x Query Result 5

SQL | All Rows Fetched: 7 in 0.002 seconds

	COD_SCRITOR	NUME	PRENUME	NATIONALITATE	SEX	DATA_NASTERE	DATA_DECES
1	100	Oke	Janette	Canadiana	f	18-FEB-35	(null)
2	101	Dickens	Charles	Britanica	m	07-FEB-12	09-JUN-70
3	102	Witemeyer	Karen	Americana	f	(null)	(null)
4	103	Drumes	Mihail	Romana	m	26-NOV-01	07-FEB-82
5	104	Bunn	T. Davis	Americana	m	01-JAN-52	(null)
6	105	Rowling	J.K.	Britanica	f	31-JUL-65	(null)
7	106	Dumas	Alexandre	Franceza	m	24-JUL-02	05-DEC-70

--SERII

SQL | All Rows Fetched: 5 in 0.015 seconds

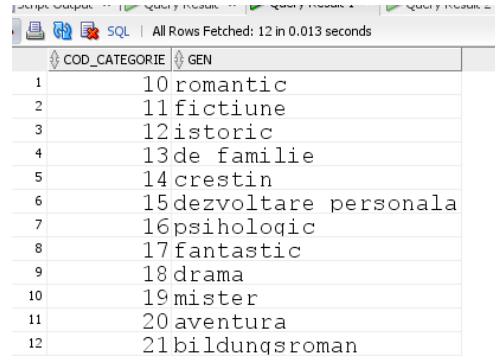
	COD_SERIE	NUME
1	300	Anotimpurile inimii
2	310	Cantecul Acadiei
3	320	Faptele credintei
4	330	Harry Potter
5	340	Cei trei muschetari

--CARTI

SQL | All Rows Fetched: 25 in 0.030 seconds

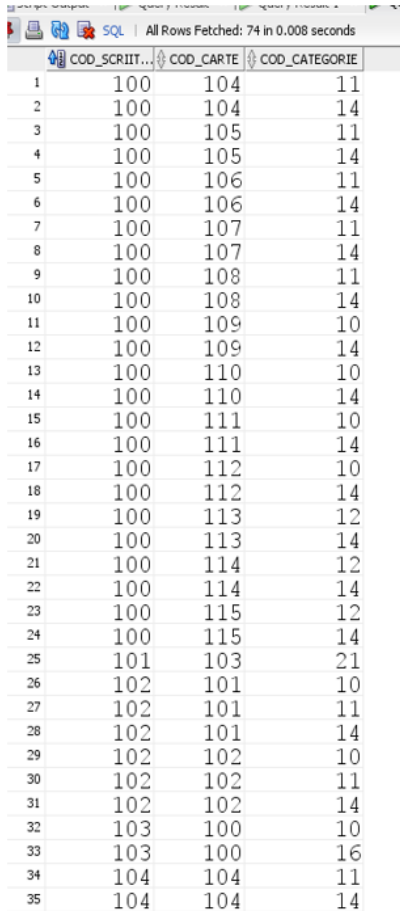
	COD_CA...	DENUMIRE	NUMAR_PAGINI	PRET	AN_APARITIE	NUMAR_VOLUM	COD_SERIE
1	100	Invitatie la vals	336	30.6	1936	(null)	(null)
2	101	Cu capul in nori	315	31	2010	(null)	(null)
3	102	Mireasa pe masura	295	28.5	2010	(null)	(null)
4	103	Marile sperante	544	30	1861	(null)	(null)
5	104	Mostenirea	294	27	2001	3	310
6	105	Binecuvantatul tarm	286	28.5	1936	2	310
7	106	Locul de intalnire	318	27	1999	1	310
8	107	Limanul mult dorit	324	27	2000	5	310
9	108	Un far calauzitor	256	27	2002	4	310
10	109	A fost odata intr-o vara	224	21.85	1981	1	300
11	110	Promisiunea unei noi primaveri	222	21.85	1989	4	300
12	111	Vanturi tomatice	220	21.85	1987	2	300
13	112	Iarna nu tine o vesnicie	216	21.85	1988	3	300
14	113	Sotia Centurionului	321	35	2010	1	320
15	114	Flacara ascunsa	388	35	2011	2	320
16	115	Drumul spre Damasc	432	35	2011	3	320
17	116	Harry Potter si camera secretelor	400	48.48	1998	2	330
18	117	Harry Potter si pocalul de foc	728	58	2000	4	330
19	118	Harry Potter si printul semisange	650	64.64	2005	6	330
20	119	Harry Potter si talismanele mortii	784	64.64	2007	7	330
21	120	Harry Potter si ordinul phoenix	990	64.64	2003	5	330
22	121	Harry Potter si piatra filosofala	532	42	1997	1	330
23	122	Harry Potter si prizonierul din Azkaban	464	48.48	1999	3	330
24	123	Cei trei muschetari	224	21	1844	1	340
25	124	Dupa 20 de ani	622	33	1845	2	340

--CATEGORII

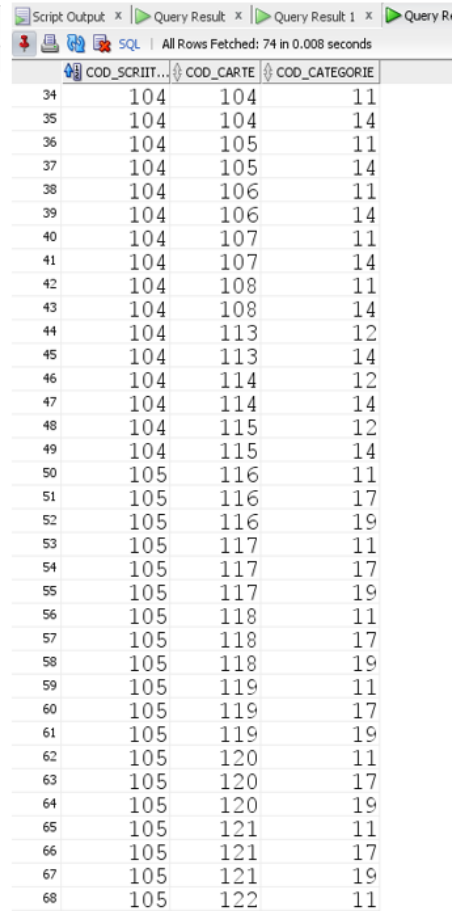


	COD_CATEGORIE	GEN
1	10	romantic
2	11	fictiune
3	12	istoric
4	13	de familie
5	14	crestin
6	15	dezvoltare personala
7	16	psihologic
8	17	fantastic
9	18	drama
10	19	mister
11	20	aventura
12	21	bildungsroman

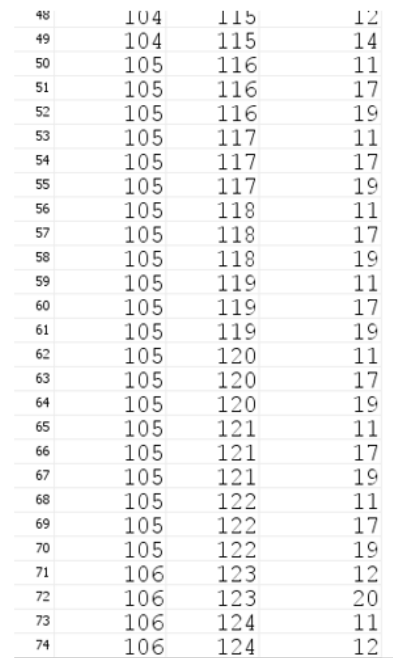
--ARE



	COD_SCRIIT...	COD_CARTE	COD_CATEGORIE
1	100	104	11
2	100	104	14
3	100	105	11
4	100	105	14
5	100	106	11
6	100	106	14
7	100	107	11
8	100	107	14
9	100	108	11
10	100	108	14
11	100	109	10
12	100	109	14
13	100	110	10
14	100	110	14
15	100	111	10
16	100	111	14
17	100	112	10
18	100	112	14
19	100	113	12
20	100	113	14
21	100	114	12
22	100	114	14
23	100	115	12
24	100	115	14
25	101	103	21
26	102	101	10
27	102	101	11
28	102	101	14
29	102	102	10
30	102	102	11
31	102	102	14
32	103	100	10
33	103	100	16
34	104	104	11
35	104	104	14



	COD_SCRIIT...	COD_CARTE	COD_CATEGORIE
34	104	104	11
35	104	104	14
36	104	105	11
37	104	105	14
38	104	106	11
39	104	106	14
40	104	107	11
41	104	107	14
42	104	108	11
43	104	108	14
44	104	113	12
45	104	113	14
46	104	114	12
47	104	114	14
48	104	115	12
49	104	115	14
50	105	116	11
51	105	116	17
52	105	116	19
53	105	117	11
54	105	117	17
55	105	117	19
56	105	118	11
57	105	118	17
58	105	118	19
59	105	119	11
60	105	119	17
61	105	119	19
62	105	120	11
63	105	120	17
64	105	120	19
65	105	121	11
66	105	121	17
67	105	121	19
68	105	122	11



48	104	115	12
49	104	115	14
50	105	116	11
51	105	116	17
52	105	116	19
53	105	117	11
54	105	117	17
55	105	117	19
56	105	118	11
57	105	118	17
58	105	118	19
59	105	119	11
60	105	119	17
61	105	119	19
62	105	120	11
63	105	120	17
64	105	120	19
65	105	121	11
66	105	121	17
67	105	121	19
68	105	122	11
69	105	122	17
70	105	122	19
71	106	123	12
72	106	123	20
73	106	124	11
74	106	124	12

--ORASE

All Rows Fetched: 6 in 0.015 seconds

	COD_ORAS	DENUMIRE	JUDET	COD_POSTAL
1	1000	Bucuresti	(null)	012581
2	1010	Oradea	Bihor	410001
3	1020	Cluj-Napoca	Cluj	010292
4	1030	Iasi	Iasi	700028
5	1040	Drobeta Turnul Severin	Mehedinti	220036
6	1050	Voluntari	Ilfov	077191

--LOCATII

All Rows Fetched: 7 in 0.015 seconds

	COD_LOCATIE	COD_ORAS	STRADA	NUMAR
1	90	1000	Strada Doamnei	20
2	100	1020	Strada Fierului	2
3	110	1020	Strada Republicii	35
4	120	1010	Strada Avram Iancu	10
5	130	1030	Strada Vasile Lupu	148
6	140	1040	Strada Gheorghe Sincai	17
7	150	1000	Strada Baltaretului	7

--LIBRARII

All Rows Fetched: 5 in 0.026 seconds

	COD_LIBRARIE	DATA_INFIINTARE	DENUMIRE	COD_LOCATIE	COD_ORAS
1	40	10-OCT-15	Libraria Iulius	90	1000
2	50	06-NOV-20	Libraria Iulius	100	1020
3	60	04-JUN-17	Libraria Iulius	130	1030
4	70	10-AUG-10	Libraria Iulius	120	1010
5	80	16-MAY-08	Libraria Iulius	110	1020

--ANGAJATI

All Rows Fetched: 12 in 0.009 seconds

	COD_ANGA...	NUME	PRENUME	SALARIU	TELEFON	SEX	TIP_ANGAJAT	COD_LIBRARIE
1	400	Popescu	Maria	1275	0761234567	f	vanzator	40
2	401	Popa	Marin	2450	0761236666	m	vanzator	60
3	402	Ionescu	Adiel	2725	0769999967	m	vanzator	70
4	403	Marinescu	Claudiu	1600	0761234000	m	vanzator	80
5	404	Ionescu	Mircea	1550	0731114567	m	ingrijitor	80
6	405	Georgescu	Aurica	1500	0724444413	f	ingrijitor	70
7	406	Popa	Marinela	1450	0760004567	f	ingrijitor	60
8	407	Noiescu	Amalia	1970	0761233357	f	ingrijitor	40
9	408	Iona	Adrian	1300	0733333387	m	vanzator	40
10	409	Popovici	Adina	1750	0727677606	f	vanzator	50
11	410	Mihaita	Claudia	1000	0729090909	f	vanzator	50
12	411	Vasilescu	Lenuta	1600	0761114597	f	ingrijitor	50

--BONURI

	COD_B...	DATA_BON	COD_CARTE	COD_ANGAJAT
1	100	20-APR-21	100	400
2	101	20-JUL-20	115	408
3	102	20-JUL-20	114	408
4	103	20-JUL-20	113	408
5	104	04-OCT-19	103	401
6	105	25-MAR-21	104	409
7	106	25-MAR-21	102	410
8	107	26-FEB-21	104	409
9	108	13-OCT-20	116	410
10	109	29-AUG-18	123	410
11	110	15-SEP-19	105	402
12	111	15-AUG-20	115	400

--STOC

All Rows Fetched: 40 in 0.01 seconds							
	COD_CARTE	COD_LIBRARIE	NR_EXEMPLARE				
1	100	40	2	21	111	80	3
2	115	40	3	22	112	80	5
3	114	40	1	23	103	50	6
4	113	40	4	24	114	50	7
5	103	80	3	25	115	50	10
6	104	50	6	26	116	70	12
7	102	50	2	27	117	70	11
8	116	50	4	28	118	70	2
9	123	50	4	29	119	70	7
10	105	70	3	30	120	70	8
11	100	60	2	31	121	70	4
12	101	80	1	32	122	70	6
13	102	40	2	33	123	60	8
14	103	60	3	34	124	60	9
15	105	50	4	35	109	40	4
16	106	50	4	36	109	60	5
17	107	50	5	37	123	70	9
18	108	50	4	38	124	70	6
19	109	80	3	39	104	80	6
20	110	80	2	40	104	60	5
21	111	80	3				

6. Cerinta:

Pentru a determina ce carti atrag cel mai mult cumparatorii, se doreste sa se determine care sunt genurile de carti ce au fost cautate cel mai des in anul curent.

In rezolvarea problemei am folosit un tablou indexat si un tablou imbricat.

create or replace procedure categorii_cautate

is

type tab_ind is table of number index by PLS_INTEGER;

nr_aparitii tab_ind; --retin numarul de aparitii al fiecarui cod

type tab_imb is table of are.cod_categorie%type;

v_coduri tab_imb := tab_imb(); --retin codul categoriilor cautate

nr_max_aparitii number := 0;

ok number;

indice number;

v_gen categorii.gen%type;

begin

for vanzari in (select max(cod_carte) cod, count(*) numar

from bonuri

where extract(year from data_bon) = extract(year from sysdate)

group by cod_carte

) loop

for categ_cautate in (select distinct cod_categorie

 --am nevoie de distinct pentru cazul in care cartea respectiva

 --este scrisa de mai multi scriitori => in tabelul are va fi

--trece la cartea si categoria de n numar de ori , unde n
--este numarul de scriitori.

from are

where cod_carte = vanzari.cod

) loop

ok := 0;

indice := v_coduri.first;

--ma plimb prin codurile categoriilor selectate deja si, daca gasesc codul curent, cresc numarul de
-- aparitii; observatie: o categorie poate fi reprezentativa pentru mai multe carti vandute

while indice <= v_coduri.last and ok = 0 loop

if v_coduri(indice) = categ_cautate.cod_categorie then

nr_aparitii(indice) := nr_aparitii(indice) + vanzari.numar;

ok := 1;

if nr_aparitii(indice) > nr_max_aparitii then

nr_max_aparitii := nr_aparitii(indice);

end if;

end if;

indice := indice + 1;

end loop;

if ok = 0 then --inseamna ca aceasta categorie nu a fost inca trecuta in colectie

v_coduri.extend;

```

        v_coduri(v_coduri.last) := categ_cautate.cod_categorie;

        nr_aparitii(v_coduri.last) := vanzari.numar;

    end if;

end loop;

end loop;

if nr_aparitii.count = 0 --orice carte se incadreaza in cel putin o categorie, deci, daca
    --tabloul ramane gol, inseamna ca nu a existat nicio carte selectata initial
then

    dbms_output.put_line('Nu au fost cumparate carti in anul curent.');
```

else

```

    for i in nr_aparitii.first..nr_aparitii.last loop

        if nr_aparitii(i) = nr_max_aparitii then

            select gen into v_gen

            from categorii

            where cod_categorie = v_coduri(i);

            dbms_output.put_line('Este cautat genul ' || v_gen);

        end if;

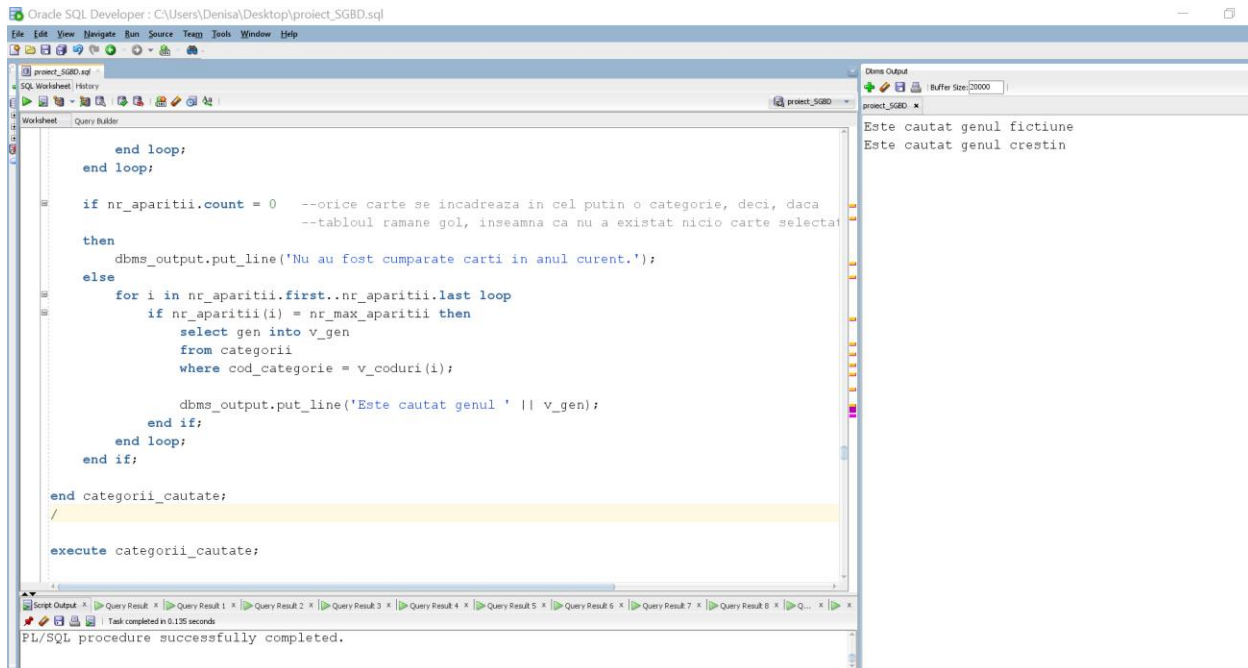
    end loop;

end if;

end categorii_cautate;

/
```

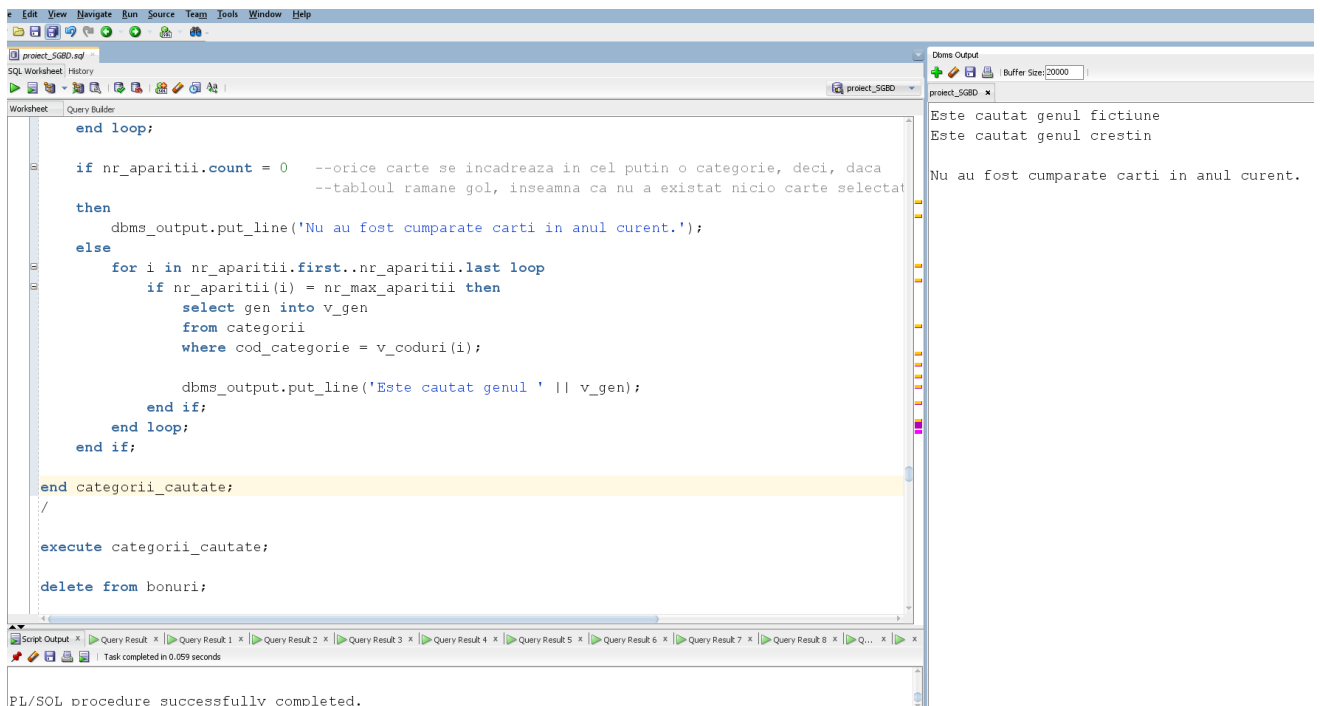
execute categorii_cautate;



Sterg datele din tabelul bonuri pentru a prezenta cazul in care nu au fost cumparate carti.

delete from bonuri;

rollback;



7. Cerinta:

Pentru un nume de scriitor dat, determinati cat costa fiecare serie pe care a scris-o in cazul in care cumparatorul ar dori sa cumpere toate volumele. Se doreste sa se afiseze pentru fiecare serie pretul acesteia, precum si un mesaj sugestiv in cazul in care scriitorul dat nu a scris nicio serie de carti. Se trateaza exceptia daca numele dat nu apare in baza de date.

Am folosit un cursor explicit.

```
create or replace procedure pret_serie(ume_scriitor in scriitori.ume%type default 'none')
```

```
is
```

```
cursor suma_serie (id scriitori.cod_scriitor%type) is
```

```
select sum(pret) suma, cod_serie --det. suma unei serii si codul acesteia
```

```
from carti
```

```
where cod_serie is not null and cod_carte in ( select distinct cod_carte
```

```
from are
```

```
where cod_scriitor = id
```

```
) --det. cartile scrise de acel scriitor
```

```
group by cod_serie;
```

```
v_serie serii.ume%type;
```

```
v_cod_serie serii.cod_serie%type;
```

```
v_suma number;
```

```
v_cod_scriitor scriitori.cod_scriitor%type;
```

```
begin
```

```
select cod_scriitor into v_cod_scriitor
```

```
from scriitori
```

```
where upper(ume) = upper(ume_scriitor);
```

--Daca numele dat nu se afla in baza de date, se va declansa eroarea NO_DATA_FOUND

open suma_serie(v_cod_scriitor);

loop

fetch suma_serie into v_suma, v_cod_serie;

exit when suma_serie%NOTFOUND;

select nume into v_serie --det. numele seriei cu acel cod

from serii

where cod_serie = v_cod_serie;

dbms_output.put_line('Seria "'||v_serie||'" costa '||v_suma|| ' lei.');

end loop;

if suma_serie%rowcount = 0 then

dbms_output.put_line('Nu exista in librerie nicio serie scrisa de acest scriitor.');

end if;

close suma_serie;

exception

when NO_DATA_FOUND then

RAISE_APPLICATION_ERROR(-20000,'Nu exista niciun scriitor cu acest nume.');

```

        when others then

            RAISE_APPLICATION_ERROR (-20001,'Alta eroare');

end pret_serie;

/

begin

--  pret_serie('Witemeyer');

--      Nu exista in librerie nicio serie scrisa de acest scriitor.

--  pret_serie('Oke');

--      Seria "Cantecul Acadiei" costa 136.5 lei.

--      Seria "Anotimpurile inimii" costa 87.4 lei.

--      Seria "Faptele credintei" costa 105 lei.

    pret_serie;

    --Eroarea are codul = -20000 si mesajul = ORA-20000: Nu exista niciun scriitor cu acest
    nume.

exception

    when others

        then dbms_output.put_line('Eroarea are codul = '||SQLCODE || ' si mesajul = ' ||
SQLERRM);

end;

/

```

```
end if;

close suma_serie;

exception
  when NO_DATA_FOUND then
    RAISE_APPLICATION_ERROR(-20000,'Nu exista niciun scriitor cu acest nume.');
```

```
end pret_serie;
/

begin

  pret_serie('Witemeyer');

  -- Nu exista in librerie nicio serie scrisa de acest scriitor.
  -- pret_serie('Oke');
  -- Seria "Cantecul Acadie" costa 136.5 lei.
  -- Seria "Anotimpurile inimii" costa 87.4 lei.
  -- Seria "Faptele credintei" costa 105 lei.
  -- pret_serie;
  -- Eroarea are codul = -20000 si mesajul = ORA-20000: Nu exista niciun scriitor cu acest nume.

  exception
    when others
    then dbms_output.put_line('Eroarea are codul = '||SQLCODE || ' si mesajul = ' || SQLERRM);
end;
```

Script Output x Query Result x

Task completed in 0.438 seconds

PL/SQL procedure successfully completed.

Dbms Output

project_SGBD x

Nu exista in librerie nicio serie scrisa de acest scriitor.

```
end if;

close suma_serie;

exception
  when NO_DATA_FOUND then
    RAISE_APPLICATION_ERROR(-20000,'Nu exista niciun scriitor cu acest nume.');
```

```
end pret_serie;
/

begin

  -- pret_serie('Witemeyer');
  -- Nu exista in librerie nicio serie scrisa de acest scriitor.
  pret_serie('Oke');
  -- Seria "Cantecul Acadie" costa 136.5 lei.
  -- Seria "Anotimpurile inimii" costa 87.4 lei.
  -- Seria "Faptele credintei" costa 105 lei.
  -- pret_serie;
  -- Eroarea are codul = -20000 si mesajul = ORA-20000: Nu exista niciun scriitor cu acest nume.

  exception
    when others
    then dbms_output.put_line('Eroarea are codul = '||SQLCODE || ' si mesajul = ' || SQLERRM);
end;
```

Script Output x Query Result x

Task completed in 0.101 seconds

PL/SQL procedure successfully completed.

Dbms Output

project_SGBD x

Nu exista in librerie nicio serie scrisa de acest scriitor.

Seria "Cantecul Acadie" costa 136.5 lei.

Seria "Anotimpurile inimii" costa 87.4 lei.

Seria "Faptele credintei" costa 105 lei.

The screenshot shows the Oracle SQL Developer interface. The main window displays a PL/SQL procedure named `pret_serie`. The procedure logic includes a `begin` block with several `pret_serie` calls for different book titles and their prices. It also includes an `exception` block that handles `NO_DATA_FOUND` and other errors, raising application errors with specific codes and messages. The output window on the right shows the execution results, including the error message for the first call and the prices for the subsequent calls. The status bar at the bottom indicates that the PL/SQL procedure was successfully completed.

```

end if;

close suma_serie;

exception
  when NO_DATA_FOUND then
    RAISE_APPLICATION_ERROR(-20000,'Nu exista niciun scriitor cu
  when others then
    RAISE_APPLICATION_ERROR (-20001,'Alta eroare');
end pret_serie;
/

begin
  -- pret_serie('Witemeyer');
  -- Nu exista in librerie nicio serie scrisa de acest scriitor.
  pret_serie('Oke');
  -- Seria "Cantecul Acadie" costa 136.5 lei.
  -- Seria "Anotimpurile inimii" costa 87.4 lei.
  -- Seria "Faptele credintei" costa 105 lei.
  pret_serie;
  --Eroarea are codul = -20000 si mesajul = ORA-20000: Nu exista ni

  exception
    when others
      then dbms_output.put_line('Eroarea are codul = '||SQLCODE ||

end;

```

Output:

```

Nu exista in librerie nicio serie scrisa de acest scriitor.

Seria "Cantecul Acadie" costa 136.5 lei.
Seria "Anotimpurile inimii" costa 87.4 lei.
Seria "Faptele credintei" costa 105 lei.

Eroarea are codul = -20000 si mesajul = ORA-20000: Nu exista niciun scriitor cu acest nume.

```

PL/SQL procedure successfully completed.

8. Cerinta:

Intr-o libraria de la o locatie data prin nume vine un client care doreste sa cumpere o carte anume, data tot prin titlu. Sa se precizeze fiecare caz posibil, anume cartea sa existe in libraria respectiva si sa se creeze un nou bon (nu conteaza ce vanzator o vinde), dar si cazul in care cartea nu exista si sa se modifice numarul de exemplare disponibile pe stoc.

Singura exceptie care apare este cea declansata de mine `NO_DATA_FOUND`. Intra pe aceasta exceptie in cazul in care nu se gaseste cartea aceea in stocul acelei librarii, deci nu se poate cumpara cartea.

create or replace function client_cumpara_carte(

carte_ceruta carti.denumire%type default 'no_name',

librarie_de_unde_se_solicita locatiei.strada%type default 'no_adres'

)

return carti.pret%type is

carte_vanduta stoc.cod_carte%type;

librarie_care_vinde stoc.cod_librarie%type;

pret_cumparare carti.pret%type;


```

begin

    update stoc

    set nr_exemplare = nr_exemplare - 1

    where (cod_carte, cod_librarie) =

        (

            select cod_carte, st.cod_librarie -- daca cartea dorita se gaseste la libraria

                                           -- mentionata, selectul va intoarce i linie

            from locatii loc join librarii lib on (loc.cod_locatie=lib.cod_locatie)

            join ( select cod_carte, cod_librarie

                    --selectez toate codurile librariile in care se  gaseste cartea dorita

                    from carti join stoc using (cod_carte)

                    where upper(denumire) = upper(carte_ceruta)

                    ) st on(st.cod_librarie = lib.cod_librarie) --joinul se face pe cod librarie

            where upper(strada) = upper (librarie_de_unde_se_solicita) --selectez libraria dorita

        )

    returning cod_carte, cod_librarie into carte_vanduta, librerie_care_vinde;

--stim ca UPDATE nu intoarce NO_DATA_FOUND asadar verific

--eu cate linii s-au modificat(maxim 1 in cazul nostru) si tratez exceptia dorita

    if sql%rowcount = 0 then

        raise NO_DATA_FOUND;

    end if;

```

insert into bonuri

select SEQ_CUMP.nextval, sysdate, carte_vanduta, (

select cod_angajat

from (--aranjez vanzatorii care lucreaza la acea librerie

--in mod random pentru ca fiecare sa aiba sansa sa vanda clientului

select cod_angajat

from angajati

where cod_librarie = librerie_care_vinde and tip_angajat = 'vanzator'

ORDER BY DBMS_RANDOM.RANDOM

)

where rownum = 1

)

from dual;

--daca se ajunge pana aici inseamna ca o carte a fost vanduta, deci urmeaza sa selectez pretul

--pentru a-l transmite clientului

select pret

into pret_cumparare

from carti

where cod_carte = carte_vanduta;

return pret_cumparare;

```

exception

when NO_DATA_FOUND then

    RAISE_APPLICATION_ERROR(-20000, 'Ne pare rau, nu avem cartea '|| carte_ceruta ||

        ' pe stoc la locatia ' || librerie_de_unde_se_solicita);

when others then

    RAISE_APPLICATION_ERROR(-20001, SQLERRM);

end client_cumpara_carte;

/

declare

--  carte carti.denumire%type := 'Invitatie la vals'; --cod_carte: 100

--  strada locatii.strada%type := 'Strada Doamnei';  --cod_librarie: 40

    carte carti.denumire%type := 'Invitatie la vals';

    strada locatii.strada%type := 'Strada Fierului';

begin

    dbms_output.put_line('Clientul a platit '||client_cumpara_carte(carte, strada)||' lei pe cartea

||carte|| '.');

exception

when others then

    dbms_output.put_line(SQLERRM);

end;

/

```

Datele dinainte de apelare:

	COD_B...	DATA_BON	COD_CARTE	COD_ANGAJAT
1	100	20-APR-21	100	400
2	101	20-JUL-20	115	408
3	102	20-JUL-20	114	408
4	103	20-JUL-20	113	408
5	104	04-OCT-19	103	401
6	105	25-MAR-21	104	409
7	106	25-MAR-21	102	410
8	107	26-FEB-21	104	409
9	108	13-OCT-20	116	410
10	109	29-AUG-18	123	410
11	110	15-SEP-19	105	402
12	111	15-AUG-20	115	400

```
select * from stoc;
```

	COD_CARTE	COD_LIBRARIE	NR_EXEMPLARE
1	100	40	2
2	115	40	3
3	114	40	1
4	113	40	4
5	103	80	3
6	104	50	6
7	102	50	1

Dupa apelare:

The screenshot shows the SQL Developer interface with a PL/SQL procedure named `client_cumpara_carte` being executed. The procedure takes a card code and a library name as input. The output window shows the result of the procedure call.

```
return pret_cumpara;
```

```
exception
```

```
  when NO_DATA_FOUND then
```

```
    RAISE_APPLICATION_ERROR(-20000, 'Ne pare rau, nu avem cartea ' || carte_ceruta ||
```

```
      ' pe stoc la locatia ' || librerie_de_unde_se_solicita);
```

```
  when others then
```

```
    RAISE_APPLICATION_ERROR(-20001, SQLERRM);
```

```
end client_cumpara_carte;
```

```
/
```

```
declare
```

```
  carte carti.denumire%type := 'Invitatie la vals';
```

```
  strada locatii.strada%type := 'Strada Doamnei';
```

```
--  carte carti.denumire%type := 'Mireasa pe masura';
```

```
--  strada locatii.strada%type := 'Strada Fierului';
```

```
begin
```

```
  dbms_output.put_line('Clientul a platit ' || client_cumpara_carte(carte, strada) || ' lei pe cartea ' || carte || '.');
```

```
exception
```

```
  when others then
```

```
    dbms_output.put_line(SQLERRM);
```

```
end;
```

Clientul a platit 30.6 lei pe cartea Invitatie la vals.

PL/SQL procedure successfully completed.

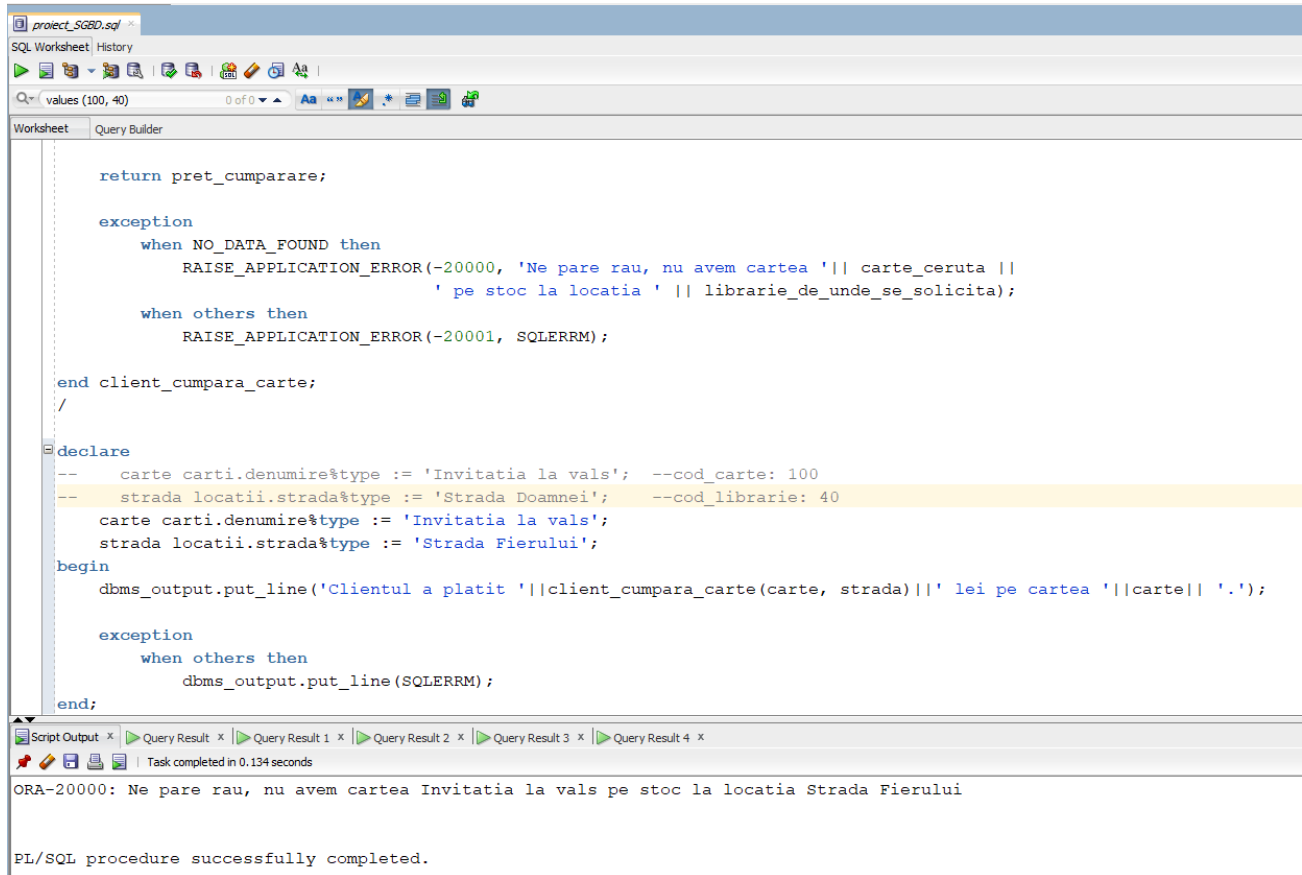
```
select * from bonuri;
```

	COD...	DATA_BON	COD_CARTE	COD_ANGAJAT
1	190	27-DEC-21	100	408
2	110	15-SEP-19	105	402
3	109	29-AUG-18	123	410
4	108	13-OCT-20	116	410
5	107	26-FEB-21	104	409
6	106	25-MAR-21	102	410

```
select * from stoc;
```

	COD_CARTE	COD_LIBRARIE	NR_EXEMPLARE
1	100	40	1
2	115	40	3
3	114	40	1
4	113	40	4
5	103	80	3
6	104	50	6
7	102	50	1

Exceptie:



```
return pret_cumparare;

exception
    when NO_DATA_FOUND then
        RAISE_APPLICATION_ERROR(-20000, 'Ne pare rau, nu avem cartea ' || carte_ceruta ||
            ' pe stoc la locatia ' || librerie_de_unde_se_solicita);
    when others then
        RAISE_APPLICATION_ERROR(-20001, SQLERRM);

end client_cumpara_carte;
/

declare
--   carte carti.denumire%type := 'Invitatie la vals'; --cod_carte: 100
--   strada locatii.strada%type := 'Strada Doamnei'; --cod_librarie: 40
    carte carti.denumire%type := 'Invitatie la vals';
    strada locatii.strada%type := 'Strada Fierului';
begin
    dbms_output.put_line('Clientul a platit ' || client_cumpara_carte(carte, strada) || ' lei pe cartea ' || carte || '.');

    exception
        when others then
            dbms_output.put_line(SQLERRM);
end;
```

Script Output x | Query Result x | Query Result 1 x | Query Result 2 x | Query Result 3 x | Query Result 4 x

Task completed in 0.134 seconds

ORA-20000: Ne pare rau, nu avem cartea Invitatie la vals pe stoc la locatia Strada Fierului

PL/SQL procedure successfully completed.

9. Cerinta:

Determinati care este bestsellerul dintr-un oras dat de la tastatura. Se mentioneaza ca bestsellerul trebuie sa reprezinte o singura cartea (cea care a fost vanduta de numarul maxim de ori) si se va afisa impreuna cu numarul de exemplare date. Daca sunt mai multe carti vandute de numar maxim de ori, se va afisa un mesaj sugestiv. La fel se va proceda si in cazul in care in orasul respectiv nu s-a vandut inca nicio carte.

S-au apelat tabelele bonuri, carti, angajati, librarii, orase intr-o singura comanda SQL.

create or replace procedure bestseller(ume_oras orase.denumire%type default 'none')

is

v_numar number;

v_ume carti.denumire%type := 'niciunul';

```

cursor vanzari_carte(parametru orase.denumire%type)
is
select count(bon.cod_carte) nr, max(car.denumire) den
           --determin pentru fiecare carte vanduta de cate ori a fost vanduta
from bonuri bon join carti car on (bon.cod_carte=car.cod_carte)
where cod_angajat in (
           --determin codul vanzatorilor care sunt angajati intr-o librerie din orasul dat
           select cod_angajat
           from angajati
           where lower(tip_angajat) = 'vanzator'
           and cod_librarie in (
                   select cod_librarie --codurile librariilor din acel oras
                   from librarii
                   where cod_oras = (
                           select cod_oras --codul orasului dat
                           from orase
                           where upper(denumire) = upper(parametru)
                           )
                   )
           )
group by bon.cod_carte --le grupez in functie de cod_carte
order by count(bon.cod_carte) desc;

begin

for i in vanzari_carte(nume_oras) loop
    if vanzari_carte%rowcount = 1 then --pentru prima linie intoarsa de cursor
        v_numar := i.nr;
    end if;
end loop;

```

```

v_nume := i.den;

--daca cursorul intoarce cel putin o linie, inseamna ca v_nume isi va schimba valoarea
end if;

if vanzari_carte%rowcount = 2 then
    if v_numar = i.nr then
        raise TOO_MANY_ROWS;
        --inseamna ca sunt cel putin 2 carti care au fost vandute de numar maxim de ori
    end if;
end if;

end loop;

if v_nume = 'niciunul' then          --in cazul in care cursorul nu intoarce nimic
    raise NO_DATA_FOUND;
end if;

dbms_output.put_line('Bestsellerul din orasul ' || nume_oras||
    ' este "'||v_nume||
    '" si a fost vandut de '||v_numar||
    ' ori.');
```

```

exception
    when NO_DATA_FOUND then
        RAISE_APPLICATION_ERROR (-20000,'Nu exista carti vandute in orasul ' ||
nume_oras);
    when TOO_MANY_ROWS then
        RAISE_APPLICATION_ERROR (-20001,'Sunt mai multe carti vandute de numar
maxim de ori in orasul '||nume_oras);
    when others then
        RAISE_APPLICATION_ERROR (-20002,'Alta eroare');
```

```
end bestseller;
```

```
/
```

```
declare
```

```
    oras orase.denumire%type := '&oras';
```

```
begin
```

```
    bestseller(oras);
```

```
exception
```

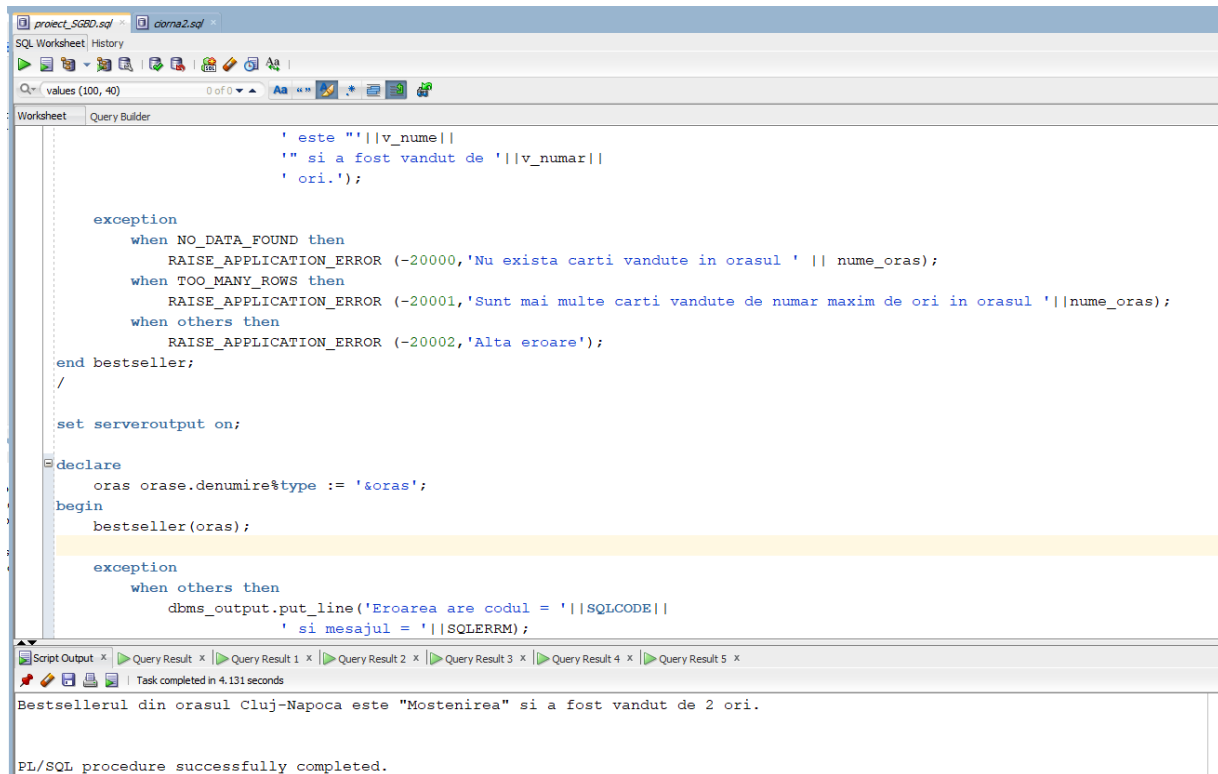
```
    when others then
```

```
        dbms_output.put_line('Eroarea are codul = '||SQLCODE||
```

```
                                ' si mesajul = '||SQLERRM);
```

```
end;
```

```
/
```



```
project_SQBD.sql  oras2.sql
SQL Worksheet: History
values (100, 40)  0 of 0
Worksheet  Query Builder

        ' este '||v_nume||
        ' si a fost vandut de '||v_numar||
        ' ori. ');

exception
    when NO_DATA_FOUND then
        RAISE_APPLICATION_ERROR (-20000,'Nu exista carti vandute in orasul ' || nume_oras);
    when TOO_MANY_ROWS then
        RAISE_APPLICATION_ERROR (-20001,'Sunt mai multe carti vandute de numar maxim de ori in orasul '||nume_oras);
    when others then
        RAISE_APPLICATION_ERROR (-20002,'Alta eroare');
end bestseller;
/

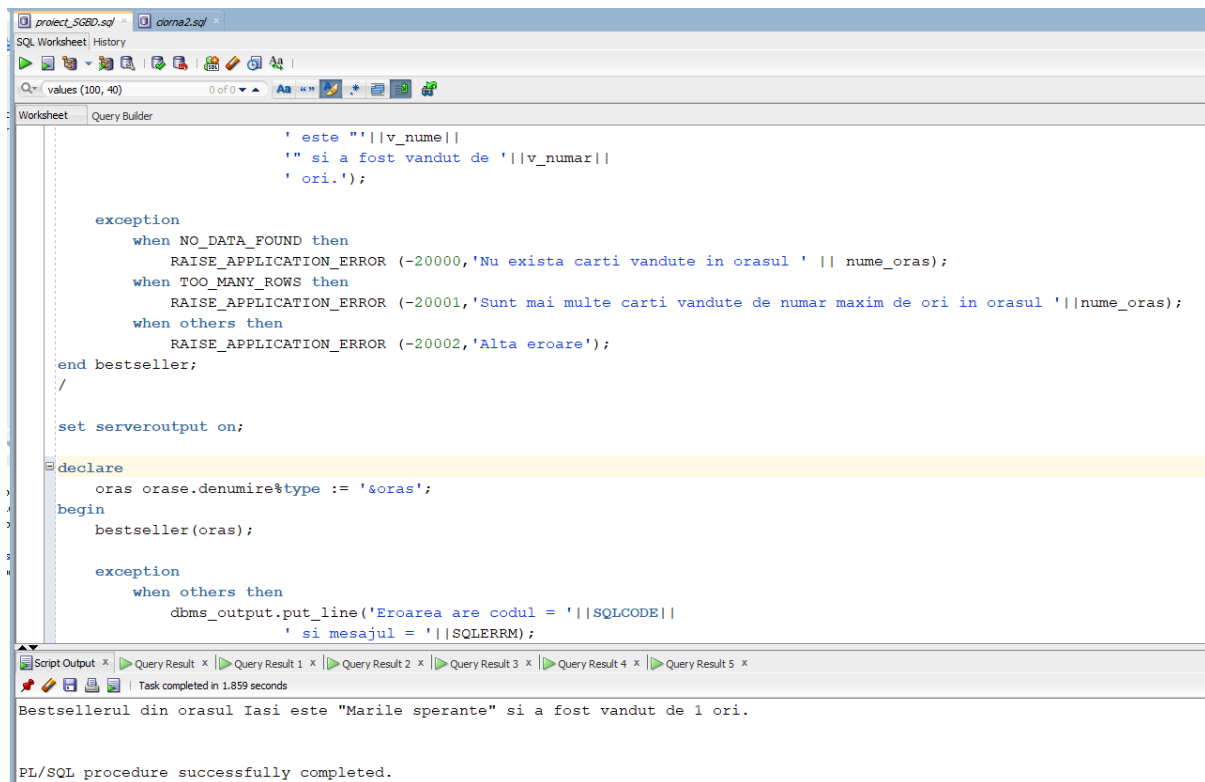
set serveroutput on;

declare
    oras orase.denumire%type := '&oras';
begin
    bestseller(oras);

exception
    when others then
        dbms_output.put_line('Eroarea are codul = '||SQLCODE||
                                ' si mesajul = '||SQLERRM);

Bestsellerul din orasul Cluj-Napoca este "Mostenirea" si a fost vandut de 2 ori.

PL/SQL procedure successfully completed.
```

The screenshot shows the SQL Developer interface with a PL/SQL procedure named `bestseller` being executed. The procedure has an exception block for `NO_DATA_FOUND` and other errors. The execution output shows the procedure completed successfully, and the message "Bestsellerul din orasul Iasi este 'Marile sperante' si a fost vandut de 1 ori." is displayed.

```
        ' este '||v_nume||
        ' si a fost vandut de '||v_numar||
        ' ori.');
```

```
exception
  when NO_DATA_FOUND then
    RAISE_APPLICATION_ERROR (-20000,'Nu exista carti vandute in orasul ' || nume_oras);
  when TOO_MANY_ROWS then
    RAISE_APPLICATION_ERROR (-20001,'Sunt mai multe carti vandute de numar maxim de ori in orasul '||nume_oras);
  when others then
    RAISE_APPLICATION_ERROR (-20002,'Alta eroare');
end bestseller;
/

set serveroutput on;

declare
  oras oras.denumire%type := '&oras';
begin
  bestseller(oras);

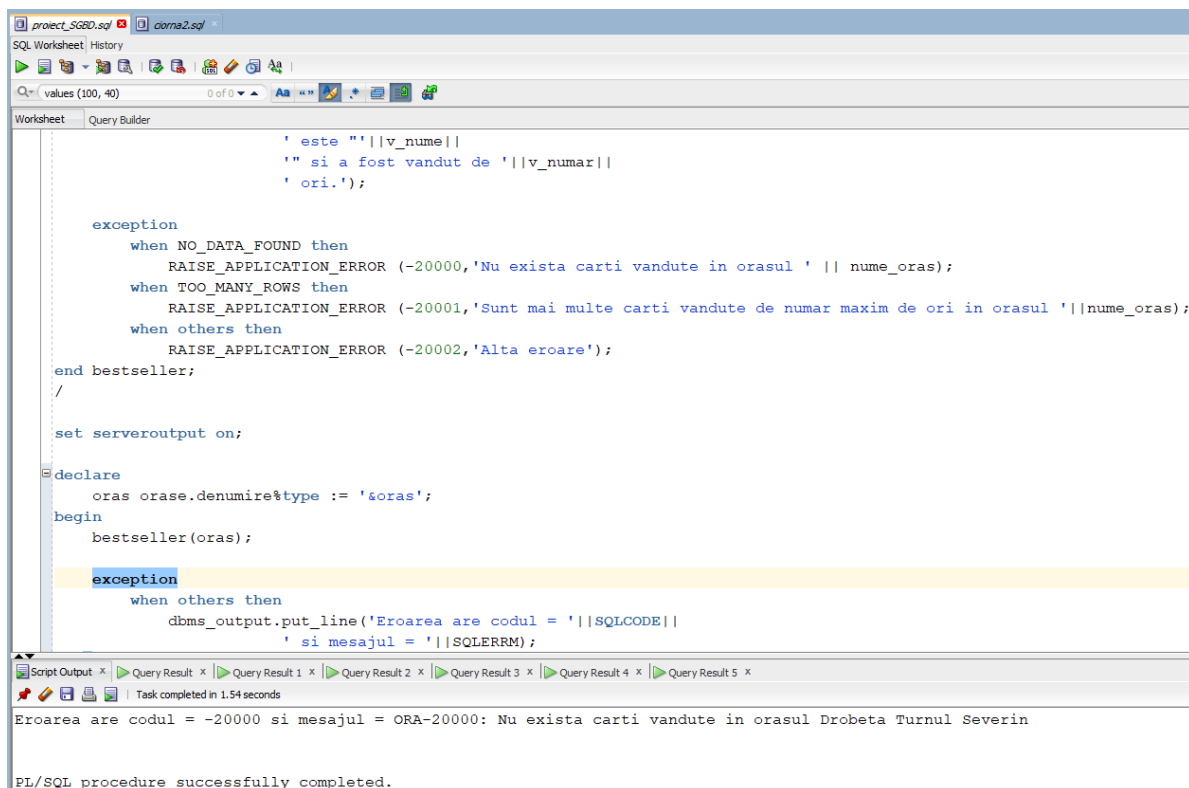
  exception
    when others then
      dbms_output.put_line('Eroarea are codul = '||SQLCODE||
        ' si mesajul = '||SQLERRM);
end;
/
```

Script Output x Query Result x Query Result 1 x Query Result 2 x Query Result 3 x Query Result 4 x Query Result 5 x
Task completed in 1.859 seconds

Bestsellerul din orasul Iasi este "Marile sperante" si a fost vandut de 1 ori.

PL/SQL procedure successfully completed.

Exceptia NO_DATA_FOUND:



The screenshot shows the same PL/SQL procedure being executed, but this time it triggers the `NO_DATA_FOUND` exception. The execution output shows the error message: "Eroarea are codul = -20000 si mesajul = ORA-20000: Nu exista carti vandute in orasul Drobeta Turnul Severin".

```
        ' este '||v_nume||
        ' si a fost vandut de '||v_numar||
        ' ori.');
```

```
exception
  when NO_DATA_FOUND then
    RAISE_APPLICATION_ERROR (-20000,'Nu exista carti vandute in orasul ' || nume_oras);
  when TOO_MANY_ROWS then
    RAISE_APPLICATION_ERROR (-20001,'Sunt mai multe carti vandute de numar maxim de ori in orasul '||nume_oras);
  when others then
    RAISE_APPLICATION_ERROR (-20002,'Alta eroare');
end bestseller;
/

set serveroutput on;

declare
  oras oras.denumire%type := '&oras';
begin
  bestseller(oras);

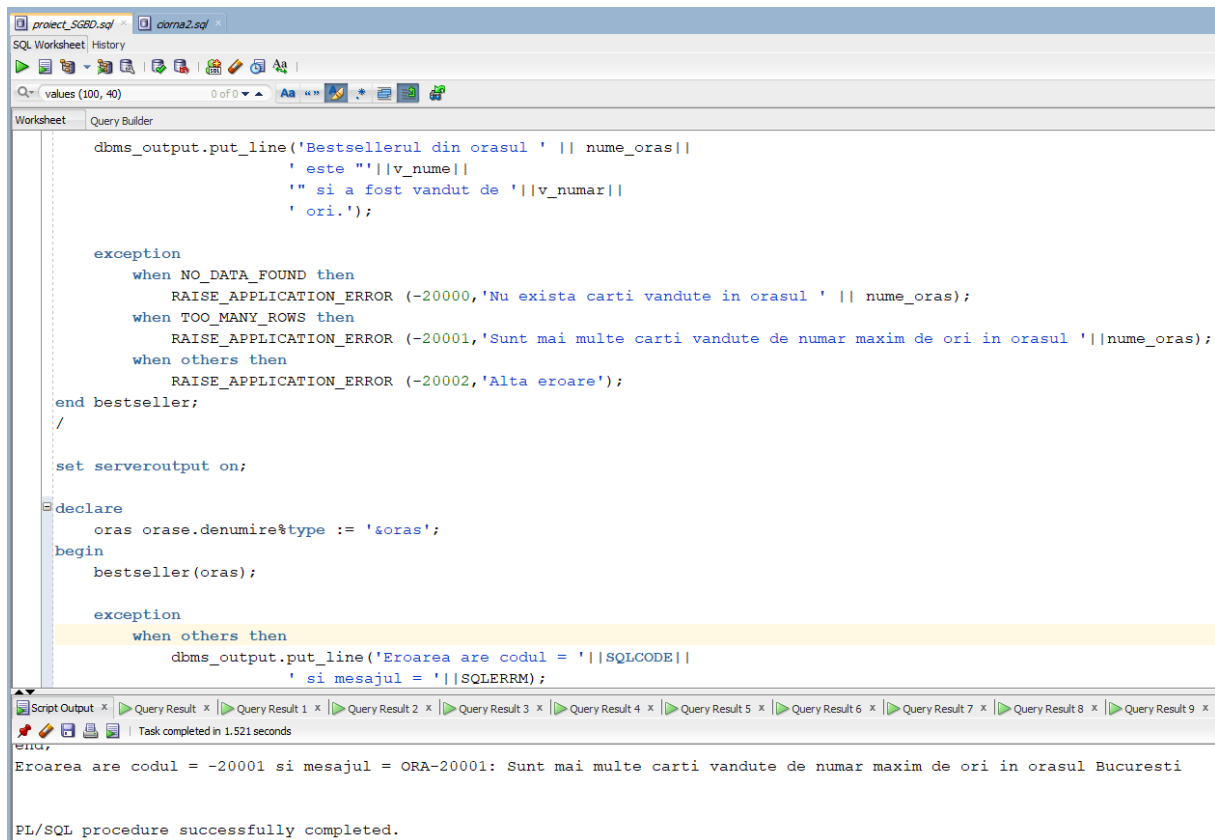
  exception
    when others then
      dbms_output.put_line('Eroarea are codul = '||SQLCODE||
        ' si mesajul = '||SQLERRM);
end;
/
```

Script Output x Query Result x Query Result 1 x Query Result 2 x Query Result 3 x Query Result 4 x Query Result 5 x
Task completed in 1.54 seconds

Eroarea are codul = -20000 si mesajul = ORA-20000: Nu exista carti vandute in orasul Drobeta Turnul Severin

PL/SQL procedure successfully completed.

Exceptia TOO_MANY_ROWS:



```
dbms_output.put_line('Bestsellerul din orasul ' || nume_oras ||
    ' este "' || v_nume ||
    '" si a fost vandut de ' || v_numar ||
    ' ori.');
```

```
exception
    when NO_DATA_FOUND then
        RAISE_APPLICATION_ERROR (-20000, 'Nu exista carti vandute in orasul ' || nume_oras);
    when TOO_MANY_ROWS then
        RAISE_APPLICATION_ERROR (-20001, 'Sunt mai multe carti vandute de numar maxim de ori in orasul ' || nume_oras);
    when others then
        RAISE_APPLICATION_ERROR (-20002, 'Alta eroare');
```

```
end bestseller;
/

set serveroutput on;

declare
    oras orase.denumire%type := '%oras';
begin
    bestseller(oras);

    exception
        when others then
            dbms_output.put_line('Eroarea are codul = ' || SQLCODE ||
                ' si mesajul = ' || SQLERRM);
end;
```

Script Output x Query Result 1 x Query Result 2 x Query Result 3 x Query Result 4 x Query Result 5 x Query Result 6 x Query Result 7 x Query Result 8 x Query Result 9 x

Task completed in 1.521 seconds

Eroarea are codul = -20001 si mesajul = ORA-20001: Sunt mai multe carti vandute de numar maxim de ori in orasul Bucuresti

PL/SQL procedure successfully completed.

10 si 11. Triggeri pe tabel mutating

Cerinta:

4 este pragul de exemplare gasite in stoc per carte (la nivelul tuturor librariilor) pentru care se aplica o reducere de 10% (deci se presupune ca o reducere se aplica atunci cand sunt ≤ 4 exemplare).

Cand un client cumpara o carte, in trigger se va specifica daca primeste reducere si se va actualiza pretul.

Trebuie sa avem grija si de stoc, sa se stearga din lista cartea daca nu mai sunt exemplare la acea librerie.

--Mod de modificare:

--Utilitatea triggerilor la nivel de comanda si de linie (before)

--se modifica pretul cartii din tabelul CARTI. (se face update in tabel)

--Actualizarea se face o singura data, atunci cand cineva cumpara a 4 a carte, apoi daca sunt

--mai putin de 4 carti, se specifica faptul ca reducerea a fost facuta (dar nu se face din nou
--update pentru ca asta ar insemna ca se aplica reducerea de doua ori)
--Este nevoie si de un trigger after update on stoc care sa stearga cartea din stocul libreriei
respectivei in cazul in care nu mai sunt exemplare.

CREATE OR REPLACE PACKAGE pachet

IS

type evidenta is table of carti.cod_carte%type;

retin_carti_aplic_reducere evidenta;

retin_carti_reducere_aplicata evidenta;

END;

/

create or replace trigger aplicare_reducere

BEFORE update of nr_exemplare on stoc

begin

--pastrez intr-un tablou imbricat lista codurilor cartilor care sunt fix 4 in stoc => asta insemna

-- ca daca cartea inserata se gaseste in lista, se va face update pe CARTI

select cod_carte

bulk collect into pachet.retin_carti_aplic_reducere

from stoc

group by cod_carte

having sum(nr_exemplare) = 4;

-- pastrez intr-un tablou imbricat lista codurilor cartilor care sunt mai putin de 4 in stoc => asta

-- insemna ca daca cartea inserata se gaseste in lista, ea profita de reducere, dar nu se mai

-- face update

select cod_carte

bulk collect into pachet.retin_carti_reducere_aplicata

from stoc

```
group by cod_carte
having sum(nr_exemplare) < 4;
end;
/
```

```
create or replace trigger cumparare_carte
BEFORE update of nr_exemplare on stoc
for each row
declare
```

```
nr number;
ok number := 0 ;
```

```
begin
```

```
nr := pachet.retin_carti_aplic_reducere.first;
while nr <= pachet.retin_carti_aplic_reducere.last and ok = 0 loop
  if pachet.retin_carti_aplic_reducere(nr) = :old.cod_carte then
    update carti
    set pret = pret - 0.1*pret
    where pachet.retin_carti_aplic_reducere(nr) = cod_carte;

    ok := 1;
  end if;
  nr := pachet.retin_carti_aplic_reducere.next(nr);
end loop;
```

```
if ok = 0 then
```

```
nr := pachet.retin_carti_reducere_aplicata.first;
while nr <= pachet.retin_carti_reducere_aplicata.last and ok = 0 loop
  if pachet.retin_carti_reducere_aplicata(nr) = :old.cod_carte then
```

```

        ok := 1;
    end if;

    nr := pachet.retin_carti_reducere_aplicata.next(nr);
end loop;
end if;

if ok = 1 then
    dbms_output.put_line('Se aplica reducere de 10% deoarece cumparati una dintre ultimele 4
exemplare din stoc');
end if;
end;
/

```

--trigger la nivel de comanda -- after pentru stergere din stoc
--nu pot folosi trigger la nivel de linie pentru ca ar recepta tabelul ca mutating
--dar in acest caz la fel de bine pot folosi un trigger la nivel de comanda

```

create or replace trigger nu_mai_exista_pe_stoc
AFTER update of nr_exemplare on stoc
begin
    delete from stoc
    where nr_exemplare = 0;
end;
/

```

```

declare
-- Cartea 'Mireasa pe masura' (cod 102) se gaseste in stoc in fix 4 exemplare
-- doua in libraria de la aceasta locatie, deci daca am vrea sa comparam de trei ori, nu ne va lasa
-- (celelalte 2 se gasesc la locatia 'Strada Doamnei')

```

```

--daca apelez functia o data, se aplica reducerea
--daca o apelez de doua ori, va sterge din stoc cartea unde codul bibliotecii este 50
--daca o apelez de 3 ori, apare eroare
--  carte carti.denumire%type := 'Mireasa pe masura'; --102
--  strada locatii.strada%type := 'Strada Fierului'; --cod: 50

-- cartea se gaseste in 5 exemplare --nu va primi reducere
  carte carti.denumire%type := 'Limanul mult dorit'; --cod: 107
  strada locatii.strada%type := 'Strada Fierului'; --cod: 50

```

begin

```

  dbms_output.put_line('Clientul a platit '||client_cumpara_carte(carte, strada)||' lei pe cartea
'||carte|| '.');

```

exception

when others then

```

  dbms_output.put_line(SQLERRM);

```

end;

/

Date inainte de apel:

```

select * from stoc;
select * from carti;
/

```

	COD_CARTE	DENUMIRE	NUMAR_PAGINI	PRET	AN_APARITIE	NUMAR_VOLUM	COD_SERIE
1	100	Invitatie la vals	336	30.6	1936	(null)	(null)
2	101	Cu capul in nori	315	31	2010	(null)	(null)
3	102	Mireasa pe masura	295	28.5	2010	(null)	(null)
4	103	Marile sperante	544	30	1861	(null)	(null)
5	104	Mostenirea	294	27	2001	3	310

```

select * from stoc;
select * from carti;
/

```

	COD_CARTE	COD_BIBLIOTECI	NR_EXEMPLARE
4	102	40	2
5	102	50	2
6	103	50	6
7	103	60	3

Primul apel:

```
create or replace trigger nu_mai_exista_pe_stoc
AFTER update of nr_exemplare on stoc
begin
    delete from stoc
    where nr_exemplare = 0;

end;
/

declare
-- Cartea 'Mireasa pe masura' (cod 102) se gaseste in stoc in fix 4 exemplare
-- doua in biblioteca de la aceasta locatie, deci daca am vrea sa comparam de trei ori, nu ne va lasa

--daca apelez functia o data, se aplica reducerea
--daca o apelez de doua ori, va sterge din stoc cartea unde codul bibliotecii este 50
--daca o apelez de 3 ori, apare eroare
carte carti.denumire&type := 'Mireasa pe masura'; --102
strada locatiei.strada&type := 'Strada Fierului'; --cod: 50

begin
    dbms_output.put_line('Clientul a platit '||client_cumpara_carte(carte, strada)||' lei pe cartea '||carte|| '.');

    exception
        when others then
            dbms_output.put_line(SQLERRM);
end;
```

Script Output x Query Result x Query Result 1 x Query Result 2 x

Task completed in 0.156 seconds

Se aplica reducere de 10% deoarece cumparati una dintre ultimele 4 exemplare din stoc
Clientul a platit 25.65 lei pe cartea Mireasa pe masura.

PL/SQL procedure successfully completed.

```
select * from bonuri;
select * from stoc;
select * from carti;
/

select * from bonuri;
select * from stoc;
select * from carti;
/

select * from bonuri;
select * from stoc;
select * from carti;
/
```

COD_BON	DATA_BON	COD_CARTE	COD_ANGAJAT
1	19427-DEC-21	102	410
2	19027-DEC-21	100	408
3	10020-APR-21	100	400
4	10625-MAR-21	100	410

COD_CA...	COD_LIBRARIE	NR_EXEMPLARE
4	102	40
5	102	50
6	103	50
7	103	60

COD_CARTE	DENUMIRE	NUMAR_PAGINI	PRET	AN_APARITIE	NUMAR_VOLUM	COD_SERIE
1	100Invitatie la vals	336	30.6	1936	(null)	(null)
2	101Cu capul in nori	315	31	2010	(null)	(null)
3	102Mireasa pe masura	255	25.65	2010	(null)	(null)
4	103Marile sperante	544	30	1861	(null)	(null)
5	104Mostenirea	294	27	2001	3	310
6	105Rinocerul din...	286	28.5	1996	2	310

La al doilea apel:

```
declare
-- Cartea 'Mireasa pe masura' (cod 102) se gaseste in stoc in fix 4 exemplare
-- doua in biblioteca de la aceasta locatie, deci daca am vrea sa comparam de trei ori, nu ne va lasa

--daca apelez functia o data, se aplica reducerea
--daca o apelez de doua ori, va sterge din stoc cartea unde codul bibliotecii este 50
--daca o apelez de 3 ori, apare eroare
carte carti.denumire&type := 'Mireasa pe masura'; --102
strada locatiei.strada&type := 'Strada Fierului'; --cod: 50

begin
    dbms_output.put_line('Clientul a platit '||client_cumpara_carte(carte, strada)||' lei pe cartea '||carte|| '.');

    exception
        when others then
            dbms_output.put_line(SQLERRM);
end;
```

Script Output x Query Result x Query Result 1 x Query Result 2 x Query Result 3 x Query Result 4 x Query Result 5 x

Task completed in 0.119 seconds

Se aplica reducere de 10% deoarece cumparati una dintre ultimele 4 exemplare din stoc
Clientul a platit 25.65 lei pe cartea Mireasa pe masura.

PL/SQL procedure successfully completed.

Se aplica reducere de 10% deoarece cumparati una dintre ultimele 4 exemplare din stoc
Clientul a platit 25.65 lei pe cartea Mireasa pe masura.

PL/SQL procedure successfully completed.

```

select * from bonuri;
select * from stoc;
select * from carti;
/

```

```

select * from bonuri;
select * from stoc;
select * from carti;
/

```

```

select * from bonuri;
select * from stoc;
select * from carti;
/

```

Script Output x Query Result x Query Result 1 x Query Result 2

SQL

All Rows Fetched: 15 in 0.002 seconds

	COD_BON	DATA_BON	COD_CARTE	COD_ANGAJAT
1	19527-DEC-21	102	410	
2	19427-DEC-21	102	410	
3	19027-DEC-21	100	408	

Script Output x Query Result x Query Result 1 x Query Result 2

SQL

All Rows Fetched: 26 in 0.002 seconds

	COD_CA...	COD_LIBRARIE	NR_EXEMPLARE
1	100	40	1
2	100	60	2
3	103	50	1
4	102	40	1
5	103	50	6
6	103	60	1
7	103	60	1

Script Output x Query Result x Query Result 1 x Query Result 2 x Query Result 3 x Query Result 4 x Query Result 5 x Query Result 6 x Query Result 7 x Query Result 8

SQL

All Rows Fetched: 25 in 0.002 seconds

	COD_CARTE	DENUMIRE	NUMAR_PAGINE	PRET	AN_LAPARASTE	NUMAR_VOLUM	COD_SERIE
1	100	Invitatie la vels	336	30.6	1936	(null)	(null)
2	101	Cu capul in mori	315	31	2010	(null)	(null)
3	102	Mireasa pe masura	295	25.65	2010	(null)	(null)
4	103	Marile sperante	544	30	1861	(null)	(null)
5	104	Marea noapte	964	37	2001	(null)	(null)

La al treilea apel:

proiect_SGBD.sql

SQL Worksheet History

Q= insert into car 1 of 25

Worksheet Query Builder

```

-- Cartea 'Mireasa pe masura' (cod 102) se gaseste in stoc in fix 4 exemplare
-- doua in biblioteca de la aceasta locatie, deci daca am vrea sa comparam de trei ori, nu ne va lasa

--daca apelez functia o data, se aplica reducerea
--daca o apelez de doua ori, va sterge din stoc cartea unde codul libreriei este 50
--daca o apelez de 3 ori, apare eroare
carte carti.denumire%type := 'Mireasa pe masura'; --102
strada locatii.strada%type := 'Strada Fierului'; --cod: 50

begin
    dbms_output.put_line('Clientul a platit '||client_cumpara_carte(carte, strada)||' lei pe cartea '||carte|| '.');

exception
    when others then
        dbms_output.put_line('GOLERRM: ');

```

Script Output x Query Result x Query Result 1 x Query Result 2 x Query Result 3 x Query Result 4 x Query Result 5 x Query Result 6 x Query Result 7 x Query Result 8

Task completed in 0.114 seconds

Se aplica reducere de 10% deoarece cumparati una dintre ultimele 4 exemplare din stoc
Clientul a platit 25.65 lei pe cartea Mireasa pe masura.

PL/SQL procedure successfully completed.

Se aplica reducere de 10% deoarece cumparati una dintre ultimele 4 exemplare din stoc
Clientul a platit 25.65 lei pe cartea Mireasa pe masura.

PL/SQL procedure successfully completed.

ORA-20000: Ne pare rau, nu avem cartea Mireasa pe masura pe stoc la locatia Strada Fierului

PL/SQL procedure successfully completed.

Cazul in care nu se primește reducere:

proiect_SGBD.sql

SQL Worksheet History

Q= insert into car 1 of 25

Worksheet Query Builder

```

delete from stoc
where nr_exemplare = 0;

end;
/

declare
-- Cartea 'Mireasa pe masura' (cod 102) se gaseste in stoc in fix 4 exemplare
-- doua in biblioteca de la aceasta locatie, deci daca am vrea sa comparam de trei ori, nu ne va lasa
-- (celelalte 2 se gasesc la locatia 'Strada Doamnei')

--daca apelez functia o data, se aplica reducerea
--daca o apelez de doua ori, va sterge din stoc cartea unde codul libreriei este 50
--daca o apelez de 3 ori, apare eroare
--
carte carti.denumire%type := 'Mireasa pe masura'; --102
strada locatii.strada%type := 'Strada Fierului'; --cod: 50

-- cartea se gaseste in 5 exemplare --nu va primi reducere
carte carti.denumire%type := 'Limanul mult dorit'; --cod: 107
strada locatii.strada%type := 'Strada Fierului'; --cod: 50

begin
    dbms_output.put_line('Clientul a platit '||client_cumpara_carte(carte, strada)||' lei pe cartea '||carte|| '.');

exception
    when others then
        dbms_output.put_line('GOLERRM: ');

```

Script Output x Query Result x Query Result 1 x Query Result 2

Task completed in 0.121 seconds

Clientul a platit 27 lei pe cartea Limanul mult dorit.

PL/SQL procedure successfully completed.

Înainte de apel:

```

select * from bonuri;
select * from stoc;
select * from carti;

```

Script Output

Query Result 3

Query Result 4

Query Result 5

SQL

All Rows Fetched: 16 in 0.002 seconds

	COD_BON	DATA_BON	COD_CARTE	COD_ANGAJAT
1	196	27-DEC-21	107	409
2	195	27-DEC-21	102	410
3	194	27-DEC-21	102	410
4	190	27-DEC-21	100	408
5	100	20-APR-21	100	400
6	106	25-MAR-21	102	410
7	105	25-MAR-21	104	409

```

select * from bonuri;
select * from stoc;
select * from carti;

```

Script Output

Query Result 1

Query Result 2

SQL

All Rows Fetched: 25 in 0.003 seconds

	COD_CARTE	DENUMIRE	NUMAR_PAGINI	PRET	AN_APARITIE	NUMAR_VOLUM	COD_SERIE
4	107	Marie's sperante	344	27	1992	1	(null)
5	104	Mostenirea	294	27	2001	3	310
6	105	Binecuvantul tarm	286	28.5	1936	2	310
7	106	Locul de intalnire	318	27	1999	1	310
8	107	Limanul mult dorit	324	27	2000	5	310
9	108	Un far calauzitor	256	27	2002	4	310

Dupa apel:

```
-- cartea se gaseste in 5 exemplare --nu va primi reducere
carte.carti.denumire$type := 'Limanul mult dorit'; --cod: 107
strada locatii.strada$type := 'Strada Fierului'; --cod: 50

begin
    dbms_output.put_line('Clientul a platit '||client_cumpara_carte(carte, strada)||' lei pe cartea '||carte|| '.');

    exception
        when others then
            dbms_output.put_line (SQLERRM);
end;
/
--rollback;

select * from bonuri;
select * from stoc;
select * from carti;
```

Script Output x Query Result 3 x Query Result 4 x Query Result 5 x

SQL All Rows Fetched: 25 in 0.003 seconds

	COD_CARTE	DENUMIRE	NUMAR_PAGINI	PRET	AN_APARITIE	NUMAR_VOLUM	COD_SERIE
7	106	Locul de intalnire	318	27	1999	1	310
8	107	Limanul mult dorit	324	27	2000	5	310
9	108	Un far calauzitor	256	27	2002	4	310
10	109	A fost odata intr-o vara	224	21.85	1981	1	300
11	110	Promisiunea unei noi primaveri	222	21.85	1989	4	300

```
select * from bonuri;
select * from stoc;
select * from carti;
```

Script Output x Query Result 3 x Query Result 4 x Query Result 5 x

SQL All Rows Fetched: 16 in 0.002 seconds

	COD_BON	DATA_BON	COD_CARTE	COD_ANGAJAT
1	196	27-DEC-21	107	409
2	195	27-DEC-21	102	410
3	194	27-DEC-21	102	410
4	190	27-DEC-21	100	408
5	100	20-APR-21	100	400
6	106	25-MAR-21	102	410
7	105	25-MAR-21	104	409

```
select * from bonuri;
select * from stoc;
select * from carti;
```

Script Output x Query Result 3 x Query Result 4 x Query Result 5 x

SQL All Rows Fetched: 39 in 0.003 seconds

	COD_CA...	COD_LIBRARIE	NR_EXEMPLARE
13	106	50	4
14	107	50	4
15	108	50	4
16	109	40	4

12. Cerinta:

Clientii nu au voie sa modifice baza de date din spatele aplicatiei, iar modificarile facute trebuie sa nu intervină cu programul deschis publicului. Asadar pentru schimbari referitoare la obiecte ale schemei sau ale bazei de date, administratorul trebuie sa le faca in intervale orare specifice.

Se stie ca in zilele lucratoare, librariile sunt deschise de la 8:00 la 20:00, sambata de la 10:00 la 19:00, iar duminica de la 12:00 la 16:00.

Triggerul creat va limita modificari la nivel de schema si are nevoie de o procedura pentru a putea insera in tabel modificarea dorita, precum si un mesaj corespunzator.

```
CREATE TABLE info
(utilizator VARCHAR2(30),
nume_bd VARCHAR2(50),
eveniment VARCHAR2(20),
nume_obiect VARCHAR2(30),
data DATE,
mesaj VARCHAR2(40));
```

```
create or replace procedure inserez_in_info(cod number)
```

```
is
```

```
    mesaj varchar2(40);
```

```
    pragma autonomous_transaction;
```

```
begin
```

```
    if cod = 1 then
```

```
        mesaj := 'eroare: in timpul programului';
```

```
    elsif cod = 2 then
```

```
        mesaj := 'eroare: not admin';
```

```
    else
```

```
        mesaj := 'succes';
```

```
    end if;
```

```
INSERT INTO info
```

```
VALUES (SYS.LOGIN_USER, SYS.DATABASE_NAME, SYS.SYSEVENT,
SYS.DICTIONARY_OBJ_NAME, SYSDATE, mesaj);
```

```
commit;
```

```
end inserez_in_info;
```

```
/
```

create or replace trigger LDD

before CREATE OR DROP OR ALTER ON SCHEMA

begin

if (to_char(sysdate, 'D') = 1 and (TO_CHAR(SYSDATE, 'HH24') BETWEEN 12 AND 16))

--pentru duminica

or

(to_char(sysdate, 'D') = 7 and (TO_CHAR(SYSDATE, 'HH24') BETWEEN 10 AND 19))

--pentru sambata

or

(to_char(sysdate, 'D') <> 1 and to_char(sysdate, 'D') <> 7 and (TO_CHAR(SYSDATE, 'HH24')
BETWEEN 8 AND 20))

--pentru zilele lucratoare

then

inserez_in_info(1);

RAISE_APPLICATION_ERROR(-20000, 'Nu sunt permise modificari ale schemei in
timpul programului cu publicul');

else

if sys.login_user <> 'DENISA' then --doar administratorul poate modifica

inserez_in_info(2);

RAISE_APPLICATION_ERROR(-20000, 'Doar administratorul poate modifica baza de
date');

end if;

end if;

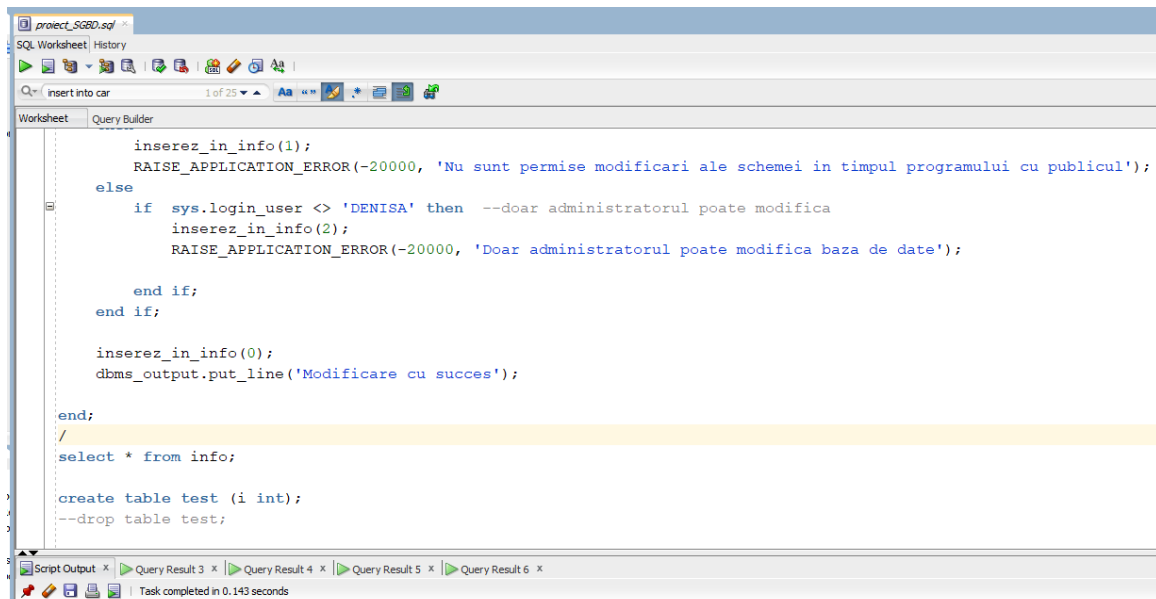
inserez_in_info(0);

dbms_output.put_line('Modificare cu succes');

end;

/

Cazul in care se doreste modificari in timpul programului cu publicul:



```
inserez_in_info(1);
RAISE_APPLICATION_ERROR(-20000, 'Nu sunt permise modificari ale schemei in timpul programului cu publicul');
else
    if sys.login_user <> 'DENISA' then --doar administratorul poate modifica
        inserez_in_info(2);
        RAISE_APPLICATION_ERROR(-20000, 'Doar administratorul poate modifica baza de date');
    end if;
end if;

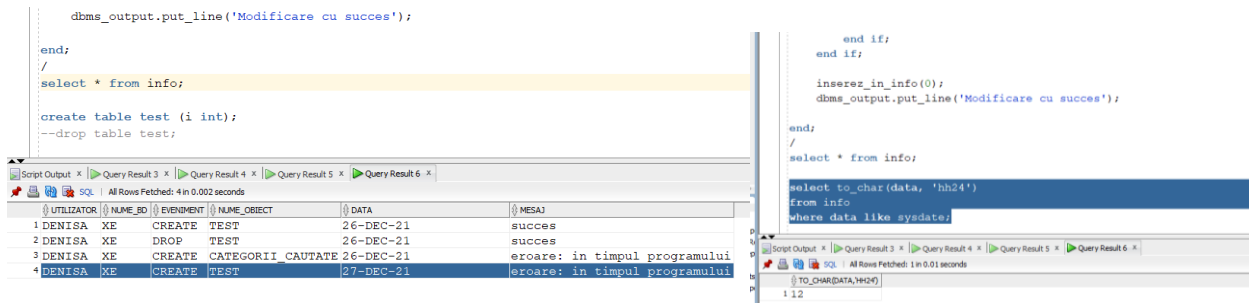
inserez_in_info(0);
dbms_output.put_line('Modificare cu succes');

end;
/
select * from info;

create table test (i int);
--drop table test;
```

Task completed in 0.143 seconds

```
Error starting at line : 1,081 in command -
create table test (i int)
Error report -
ORA-00604: error occurred at recursive SQL level 1
ORA-20000: Nu sunt permise modificari ale schemei in timpul programului cu publicul
ORA-06512: at line 9
00604. 00000 - "error occurred at recursive SQL level %s"
*Cause:      An error occurred while processing a recursive SQL statement
              (e.g. calling a PL/SQL procedure from within a PL/SQL block)
```



```
dbms_output.put_line('Modificare cu succes');

end;
/
select * from info;

create table test (i int);
--drop table test;
```

All Rows Fetched: 4 in 0.002 seconds

	UTILIZATOR	NAME_USER	EVENT	NAME_OBJECT	DATA	MESSAGE
1	DENISA	XE	CREATE	TEST	26-DEC-21	succes
2	DENISA	XE	DROP	TEST	26-DEC-21	succes
3	DENISA	XE	CREATE	CATEGORII_CAUTATE	26-DEC-21	eroare: in timpul programului
4	DENISA	XE	CREATE	TEST	27-DEC-21	eroare: in timpul programului

```
end if;
end if;

inserez_in_info(0);
dbms_output.put_line('Modificare cu succes');

end;
/
select * from info;

select to_char(data, 'hh24')
from info
where data like sysdate;
```

All Rows Fetched: 1 in 0.01 seconds

TO_CHAR(DATA, 'HH24')

1 12

Cazul in care se poate realiza modificarea:

Worksheet

Query Builder

```
(to_char(sysdate, 'D') <> 1 and to_char(sysdate, 'D') <> 7 and (to_char(sysdate, 'HH24') between 0 and 20)) then
    then
        inserez_in_info(1);
        RAISE_APPLICATION_ERROR(-20000, 'Nu sunt permise modificari ale schemei in timpul programului cu publicul');
    else
        if sys.login_user <> 'DENISA' then --doar administratorul poate modifica
            inserez_in_info(2);
            RAISE_APPLICATION_ERROR(-20000, 'Doar administratorul poate modifica baza de date');
        end if;
    end if;

    inserez_in_info(0);
    dbms_output.put_line('Modificare cu succes');

end;
/
select * from info;

select to_char(data, 'hh24')
from info
where data like sysdate;

create table test (i int);
drop table test;
```

Script Output

Query Result

Task completed in 0.085 seconds

Table TEST created.

Table TEST dropped.

Script Output

Query Result

All Rows Fetched: 6 in 0.003 seconds

	UTILIZATOR	NUME_RO	EVENTIMENT	NUME_OBIECT	DATA	MESAJ
1	DENISA	XE	CREATE	TEST	26-DEC-21	succes
2	DENISA	XE	DROP	TEST	26-DEC-21	succes
3	DENISA	XE	CREATE	CATEGORII_CAUTATE	26-DEC-21	eroare: in timpul programului
4	DENISA	XE	CREATE	TEST	27-DEC-21	eroare: in timpul programului
5	DENISA	XE	CREATE	TEST	28-DEC-21	succes
6	DENISA	XE	DROP	TEST	28-DEC-21	succes

Script Output

Query Result

All Rows Fetched: 2 in 0.01 seconds

	TO_CHAR(DATA, 'HH24')	TO_CHAR(DATA, 'D')
1	00	3
2	00	3

Cazul in care se doreste modificarea de o persoana care nu are dreptul:

The screenshot shows the SQL Developer interface with a PL/SQL script in the Query Builder. The script is designed to modify the STOC table, but it includes a check for the user's login name. If the user is not 'DENIS', it raises an application error. The script is executed, and the results pane shows the error report.

```
(to_char(sysdate, 'D') = 7 and (TO_CHAR(SYSDATE, 'HH24') BETWEEN 10 AND 19)) --pentru sambata
or
(to_char(sysdate, 'D') <> 1 and to_char(sysdate, 'D') <> 7 and (TO_CHAR(SYSDATE, 'HH24') BETWEEN 8 AND 20)) --pentru
then
    inserez_in_info(1);
    RAISE_APPLICATION_ERROR(-20000, 'Nu sunt permise modificari ale schemei in timpul programului cu publicul');
else
    if sys.login_user <> 'DENIS' then --doar administratorul poate modifica
        inserez_in_info(2);
        RAISE_APPLICATION_ERROR(-20000, 'Doar administratorul poate modifica baza de date');

    end if;
end if;

inserez_in_info(0);
dbms_output.put_line('Modificare cu succes');

end;
/
--Pentru cazul in care nu sunt administrator aka sys.login_user <> 'DENISA' nu pot face verificarea, dar
--in schimb pot scrimba stringul 'DENISA' in 'DENIS' si cum 'DENISA' <> 'DENIS' va intra pe acea conditie.
ALTER TABLE STOC
DISABLE ALL TRIGGERS;
```

Error starting at line : 1,075 in command -
ALTER TABLE STOC
DISABLE ALL TRIGGERS
Error report -
ORA-00604: error occurred at recursive SQL level 1
ORA-20000: Doar administratorul poate modifica baza de date
ORA-06512: at line 13
00604. 00000 - "error occurred at recursive SQL level %s"

end;
/
--Pentru cazul in care nu sunt administrator aka sys.login_user <> 'DENISA' nu pot f
--in schimb pot scrimba stringul 'DENISA' in 'DENIS' si cum 'DENISA' <> 'DENIS' va i
ALTER TABLE STOC
DISABLE ALL TRIGGERS;

UTILIZATOR	NUME_BD	EVENTIMENT	NUME_OBJECT	DATA	MESSAJ
DENISA	XE	CREATE	TEST	26-DEC-21	succes
DENISA	XE	DROP	TEST	26-DEC-21	succes
DENISA	XE	CREATE	CATEGORII_CAUTATE	26-DEC-21	eroare: in timpul prog
DENISA	XE	CREATE	TEST	27-DEC-21	eroare: in timpul prog
DENISA	XE	ALTER	STOC	28-DEC-21	eroare: not admin
DENISA	XE	CREATE	TEST	28-DEC-21	succes
DENISA	XE	DROP	TEST	28-DEC-21	succes

13.

create or replace package pachet13

is

--6.

procedure categorii_cautate;

--7.

```

procedure pret_serie(ume_scriitor in scriitori.ume%type default 'none');
--8.

function client_cumpara_carte(carte_ceruta carti.denumire%type default 'no_name',
                             librerie_de_unde_se_solicita locatii.strada%type default
'no_adress'
                             )

return carti.pret%type;
--9.

procedure bestseller(ume_oras orase.denumire%type default 'none');
end;
/

```

create or replace package body pachet13

is

--6

procedure categorii_cautate

is

```

type tab_ind is table of number index by PLS_INTEGER;
nr_aparitii tab_ind;      --retin numarul de aparitii al fiecarui cod
type tab_imb is table of are.cod_categorie%type;
v_coduri tab_imb := tab_imb(); --retin codul categoriilor cautate

nr_max_aparitii number := 0;
ok number;
indice number;
v_gen categorii.gen%type;
begin
for vanzari in ( select max(cod_carte) cod, count(*) numar
                from bonuri

```

```

where extract(year from data_bon) = extract(year from sysdate)
group by cod_carte
) loop
for categ_cautate in (select distinct cod_categorie
--am nevoie de distinct pentru cazul in care cartea
respectiva
--este scrisa de mai multi scriitori => in tabelul are va fi
--trecea cartea si categoria de n numar de ori , unde n
--este numarul de scriitori.

from are
where cod_carte = vanzari.cod
) loop
ok := 0;
indice := v_coduri.first;

--ma plimb prin codurile categoriilor selectate deja si, daca gasesc codul curent, cresc
numarul de aparitii;
--observatie: o categorie poate fi reprezentativa pentru mai multe carti vandute
while indice <= v_coduri.last and ok = 0 loop
if v_coduri(indice) = categ_cautate.cod_categorie then
nr_aparitii(indice) := nr_aparitii(indice) + vanzari.numar;
ok := 1;

if nr_aparitii(indice) > nr_max_aparitii then
nr_max_aparitii := nr_aparitii(indice);
end if;
end if;
indice := indice + 1;
end loop;

```



```

    if ok = 0 then --inseamna ca aceasta categorie nu a fost inca trecuta in colectie
        v_coduri.extend;
        v_coduri(v_coduri.last) := categ_cautate.cod_categorie;
        nr_aparitii(v_coduri.last) := vanzari.numar;
    end if;

    end loop;
end loop;

if nr_aparitii.count = 0 --orice carte se incadreaza in cel putin o categorie, deci, daca
    --tabloul ramane gol, inseamna ca nu a existat nicio carte selectata initial
then
    dbms_output.put_line('Nu au fost cumparate carti in anul curent.');
```

```

else
    for i in nr_aparitii.first..nr_aparitii.last loop
        if nr_aparitii(i) = nr_max_aparitii then
            select gen into v_gen
            from categorii
            where cod_categorie = v_coduri(i);

            dbms_output.put_line('Este cautat genul ' || v_gen);
        end if;
    end loop;
end if;

end categorii_cautate;
```

--7.

procedure pret_serie(ume_scriitor in scriitori.ume%type default 'none')

is

cursor suma_serie (id scriitori.cod_scriitor%type) is

select sum(pret) suma, cod_serie --det. suma unei serii si codul acesteia

from carti

where cod_serie is not null and cod_carte in (select distinct cod_carte

from are

where cod_scriitor = id

)--det. cartile scrise de acel scriitor

group by cod_serie;

v_serie serii.ume%type;

v_cod_serie serii.cod_serie%type;

v_suma number;

v_cod_scriitor scriitori.cod_scriitor%type;

begin

select cod_scriitor into v_cod_scriitor

from scriitori

where upper(ume) = upper(ume_scriitor);

--Daca numele dat nu se afla in baza de date, se va declansa eroarea NO_DATA_FOUND

open suma_serie(v_cod_scriitor);

loop

fetch suma_serie into v_suma, v_cod_serie;

exit when suma_serie%NOTFOUND;

```

select nume into v_serie --det. numele seriei cu acel cod
from serii
where cod_serie = v_cod_serie;

dbms_output.put_line('Seria "'||v_serie||'" costa '||v_suma|| ' lei.');
```

end loop;

```

if suma_serie%rowcount = 0 then
    dbms_output.put_line('Nu exista in librarii nicio serie scrisa de acest scriitor.');
```

end if;

```

close suma_serie;
```

exception

```

when NO_DATA_FOUND then
    RAISE_APPLICATION_ERROR(-20000,'Nu exista niciun scriitor cu acest nume.');
```

when others then

```

    RAISE_APPLICATION_ERROR (-20001,'Alta eroare');
```

end pret_serie;

--8.

```

function client_cumpara_carte(carte_ceruta carti.denumire%type default 'no_name',
                                librarii_de_unde_se_solicita locatie.strada%type default
'no_adress'
                                )
return carti.pret%type is

carte_vanduta stoc.cod_carte%type;
librarii_care_vinde stoc.cod_librarii%type;
```

```

    pret_cumparare carti.pret%type;
begin

    update stoc
    set nr_exemplare = nr_exemplare - 1
    where (cod_carte, cod_librarie) =
        (
            select cod_carte, st.cod_librarie -- daca cartea dorita se gaseste la libraria
                                           -- mentionata, selectul va intoarce i linie
            from locatii loc join librarii lib on (loc.cod_locatie=lib.cod_locatie)
            join ( select cod_carte, cod_librarie --selectez toate codurile librariile in care se
gaseste cartea dorita
                  from carti join stoc using (cod_carte)
                  where upper(denumire) = upper(carte_ceruta)
                  ) st on(st.cod_librarie = lib.cod_librarie) --joinul se face pe cod librerie
            where upper(strada) = upper (librarie_de_unde_se_solicita) --selectez libraria dorita
        )
    returning cod_carte, cod_librarie into carte_vanduta, librerie_care_vinde;

    if sql%rowcount = 0 then --stim ca UPDATE nu intoarce NO_DATA_FOUND asadar
verific
        raise NO_DATA_FOUND; --eu cate linii s-au modificat(maxim 1 in cazul nostru) si
tratez exceptia dorita
    end if;

    insert into bonuri
    select SEQ_CUMP.nextval, sysdate, carte_vanduta, (
        select cod_angajat
        from ( --aranjez vanzatorii care lucreaza la acea librerie

```

```

--in mod random pentru ca fiecare sa aiba sansa sa vanda clientului
select cod_angajat
from angajati
where cod_librarie = librarie_care_vinde and tip_angajat = 'vanzator'
ORDER BY DBMS_RANDOM.RANDOM
)
where rownum = 1
)
from dual;

--daca se ajunge pana aici inseamna ca o carte a fost vanduta, deci urmeaza sa selectez
pretul
--pentru a-l transmite clientului
select pret
into pret_cumparare
from carti
where cod_carte = carte_vanduta;

return pret_cumparare;

exception
when NO_DATA_FOUND then
    RAISE_APPLICATION_ERROR(-20000, 'Ne pare rau, nu avem cartea '|| carte_ceruta
||
    ' pe stoc la locatia ' || librarie_de_unde_se_solicita);
when others then
    RAISE_APPLICATION_ERROR(-20001, SQLERRM);

end client_cumpara_carte;

```

--9.

```
procedure bestseller(ume_oras orase.denumire%type default 'none')
```

```
is
```

```
    v_numar number;
```

```
    v_ume carti.denumire%type := 'niciunul';
```

```
    cursor vanzari_carte(parametru orase.denumire%type)
```

```
    is
```

```
    select count(bon.cod_carte) nr, max(car.denumire) den
```

```
        --determin pentru fiecare carte vanduta de cate ori a fost vanduta
```

```
    from bonuri bon join carti car on (bon.cod_carte=car.cod_carte)
```

```
    where cod_angajat in (
```

```
        select cod_angajat --determin codul vanzatorilor care sunt angajati intr-o librerie
    din orasul dat
```

```
        from angajati
```

```
        where lower(tip_angajat) = 'vanzator'
```

```
        and cod_librarie in (
```

```
            select cod_librarie --codurile librariilor din acel oras
```

```
            from librarii
```

```
            where cod_oras = (
```

```
                select cod_oras --codul orasului dat
```

```
                from orase
```

```
                where upper(denumire) = upper(parametru)
```

```
            )
```

```
        )
```

```
    )
```

```
group by bon.cod_carte --le grupez in functie de cod_carte
```

```
order by count(bon.cod_carte) desc;
```

begin

for i in vanzari_carte(ume_oras) loop

if vanzari_carte%rowcount = 1 then --pentru prima linie intoarsa de cursor

v_numar := i.nr;

v_ume := i.den; --daca cursorul intoarce cel putin o linie, inseamna ca v_ume isi va
schimba valoarea

end if;

if vanzari_carte%rowcount = 2 then

if v_numar = i.nr then

raise TOO_MANY_ROWS; --inseamna ca sunt cel putin 2 carti care au fost
vandute de numar maxim de ori

end if;

end if;

end loop;

if v_ume = 'niciunul' then --in cazul in care cursorul nu intoarce nimic

raise NO_DATA_FOUND;

end if;

dbms_output.put_line('Bestsellerul din orasul ' || ume_oras||

' este '"||v_ume||

" si a fost vandut de '"||v_numar||

' ori.');

exception

when NO_DATA_FOUND then

```

        RAISE_APPLICATION_ERROR (-20000,'Nu exista carti vandute in orasul ' ||
nume_oras);

        when TOO_MANY_ROWS then

            RAISE_APPLICATION_ERROR (-20001,'Sunt mai multe carti vandute de numar
maxim de ori in orasul '||nume_oras);

        when others then

            RAISE_APPLICATION_ERROR (-20002,'Alta eroare');

    end bestseller;

end;

/

```

14. Cerinta:

Un client vine la o librerie (data prin strada pe care se afla) si cere o serie de carti (data prin numele acesteia). Se verifica daca se gaseste seria respectiva pe stoc si in caz afirmativ, se ofera clientului o reducere de 10% la pretul total al seriei. Daca nu se gasesc toate cartile din serie pe stoc, clientul cumpara cartile care se gasesc, dar nu primeste reducere. Sa se afiseze si cartile cumparate de client, precum si pe cele pe care ar fi vrut sa le cumpere daca le gasea. La final se prezinta pretul.

Pentru a nu mai beneficia si de alte reduceri, triggerii creati la ex 10 si 11 sunt dezactivati.

```
ALTER TABLE STOC
```

```
DISABLE ALL TRIGGERS;
```

```
create or replace package pachet14
```

```
is
```

```
    type tab_imb is table of carti.cod_carte%type;
```

```
    TYPE informatii_return IS RECORD (
```

```
        pret_total number,
```

```
        lista_carti_gasite tab_imb := tab_imb(),
```



```
lista_carti_care_nu_se_gasesc tab_imb := tab_imb());
```

```
cursor lista_carti_din_serie(parametru serii.cod_serie%type)
```

```
is
```

```
    select cod_carte, pret
```

```
    from carti
```

```
    where cod_serie = parametru;
```

```
function det_librarie(ume_strada locatii.strada%type)
```

```
return librarii.cod_librarie%type;
```

```
function det_serie(ume_serie serii.ume%type)
```

```
return serii.cod_serie%type;
```

```
function informatii_cerere_serie(id_serie serii.cod_serie%type,
```

```
                                id_librarie librarii.cod_librarie%type)
```

```
return informatii_return;
```

```
procedure cerere(ume_serie serii.ume%type,
```

```
                ume_strada locatii.strada%type);
```

```
procedure vanzare(id_librarie librarii.cod_librarie%type,
```

```
                lista_carti tab_imb);
```

```
end;
```

```
/
```

```
create or replace package body pachet14
```

```
is
```

```

function det_librarie(nume_strada locatii.strada%type)
return librarii.cod_librarie%type
is
v_cod librarii.cod_librarie%type;
begin
    select cod_librarie
    into v_cod
    from librarii join locatii using(cod_locatie)
    where upper(strada) = upper(nume_strada);

    return v_cod;

exception
    when NO_DATA_FOUND then
        RAISE_APPLICATION_ERROR(-20000, 'Nu exista librarie pe aceasta strada');
end;

function det_serie(nume_serie serii.nume%type)
return serii.cod_serie%type
is
v_cod serii.cod_serie%type;
begin
    select cod_serie
    into v_cod
    from serii
    where upper(nume) = upper(nume_serie);

    return v_cod;

```

```
exception
    when NO_DATA_FOUND then
        RAISE_APPLICATION_ERROR(-20000, 'Nu exista serie cu acest nume');
end;
```

```
procedure vanzare(id_librarie librarii.cod_librarie%type,
                 lista_carti tab_imb)
is
    ang bonuri.cod_angajat%type;
    i number;
    numar_exemplare stoc.nr_exemplare%type;
begin
    select cod_angajat
    into ang
    from ( --aranjez vanzatorii care lucreaza la acea librarie
           --in mod random pentru ca fiecare sa aiba sansa sa vanda clientului
           select cod_angajat
           from angajati
           where cod_librarie = id_librarie and tip_angajat = 'vanzator'
           ORDER BY DBMS_RANDOM.RANDOM
         )
    where rownum = 1;

    i := lista_carti.first;
    while i <= lista_carti.last loop
        update stoc
```

```

set nr_exemplare = nr_exemplare - 1
where lista_carti(i) = cod_carte and id_librarie = cod_librarie
returning nr_exemplare into numar_exemplare;

if numar_exemplare = 0 then
    delete from stoc
    where lista_carti(i) = cod_carte and id_librarie = cod_librarie;
end if;

insert into bonuri
values (SEQ_CARTI.NEXTVAL, sysdate, lista_carti(i), ang);

i := lista_carti.NEXT(i);
end loop;
end;

function informatii_cerere_serie(id_serie serii.cod_serie%type,
                                id_librarie librarii.cod_librarie%type)
return informatii_return
is
    info informatii_return;    --(pret_total_comanda, lista_carti_cumparate,
lista_carti_care_nu_se_gasesc)
    exista number;
begin

    info.pret_total := 0;

    for i in lista_carti_din_serie(id_serie) --nu pot sa iau suma preturilor cartilor printr-un
singur select in
    loop
        --cazul in care nu se gaseste toata seria

```

```

select count(*)
into exista
from stoc
where i.cod_carte = cod_carte and id_librarie = cod_librarie;

if exista = 1 then --inseamna ca acea carte exista in biblioteca respectiva

    info.pret_total := info.pret_total + i.pret;
    info.lista_carti_gasite.EXTEND;
    info.lista_carti_gasite(info.lista_carti_gasite.last) := i.cod_carte;
else
    info.lista_carti_care_nu_se_gasesc.EXTEND;
    info.lista_carti_care_nu_se_gasesc(info.lista_carti_care_nu_se_gasesc.last) :=
i.cod_carte;
end if;
end loop;

return info;
end;

procedure cerere(ume_serie serii.ume%type,
                 ume_strada locatii.strada%type)
is
id_librarie stoc.cod_librarie%type;
retin_date informatii_return;
id_serie serii.cod_serie%type;
ume_carte carti.denumire%type;
begin

```

```

id_librarie := det_librarie(nume_strada);
id_serie := det_serie(nume_serie);

retin_date := informatii_cerere_serie(id_serie, id_librarie);

DBMS_OUTPUT.PUT_LINE('Pretul comenzii este de '|| retin_date.pret_total);

DBMS_OUTPUT.PUT_LINE('-----');
DBMS_OUTPUT.PUT_LINE('Se gasesc cartile: ');

for i in retin_date.lista_carti_gasite.first..retin_date.lista_carti_gasite.last loop
    select denumire
    into nume_carte
    from carti
    where cod_carte = retin_date.lista_carti_gasite(i);

    DBMS_OUTPUT.PUT_LINE(nume_carte);
end loop;

if retin_date.lista_carti_care_nu_se_gasesc.count = 0 then
    DBMS_OUTPUT.PUT_LINE('In librerie se gaseste toata seria. Se aplica reducere de
10%');
    retin_date.pret_total := retin_date.pret_total - 0.1*retin_date.pret_total;
else
    DBMS_OUTPUT.PUT_LINE('In librerie se gasesc doar '||
        retin_date.lista_carti_gasite.count ||
        ' carti din cele '||retin_date.lista_carti_care_nu_se_gasesc.count||
        ' ale seriei. Nu se aplica reducerea de 10%.');

```

```

        DBMS_OUTPUT.PUT_LINE('-----');
        DBMS_OUTPUT.PUT_LINE('Nu se gasesc cartile: ');
        for i in
retin_date.lista_carti_care_nu_se_gasesc.first..retin_date.lista_carti_care_nu_se_gasesc.last loop
            select denumire
            into nume_carte
            from carti
            where cod_carte = retin_date.lista_carti_care_nu_se_gasesc(i);

            DBMS_OUTPUT.PUT_LINE(nume_carte);
        end loop;
    end if;

    vanzare(id_librarie, retin_date.lista_carti_gasite);
    DBMS_OUTPUT.PUT_LINE('Asadar pretul final este de '||retin_date.pret_total);
end;
/

begin
--pachet14.cerere('Anotimpurile inimii', 'Strada Republicii'); --cod_librarie 80; cod_carti: 109,
110, 111, 112
--pachet14.cerere('Cantecul Acadiei', 'Strada Republicii');
--pachet14.cerere('Cantecul Acadiei', 'Strada Anonima');
pachet14.cerere('Anonima', 'Strada Republicii');
exception
    when others then
        DBMS_OUTPUT.PUT_LINE(SQLERRM);

```

end;

/

Date inainte de apelarea pachetul:

1576
1577
1578
1579

```
select * from stoc;  
select * from bonuri;
```

Script Output x Query Result x Query Result 1 x Query Result 2 x

SQL All Rows Fetched: 17 in 0.002 seconds

	COD_BON	DATA_BON	COD_CARTE	COD_ANGAJAT
1		200 03-JAN-22	109	403
2		196 27-DEC-21	107	409
3		195 27-DEC-21	102	410
4		194 27-DEC-21	102	410
5		190 27-DEC-21	100	408
6		100 20-APR-21	100	400
7		106 25-MAR-21	102	410
8		105 25-MAR-21	104	409
9		107 26-FEB-21	104	409
10		108 13-OCT-20	116	410

1577
1578
1579

```
select * from stoc;  
select * from bonuri;
```

Script Output x Query Result 1 x Query Result 2 x

SQL All Rows Fetched: 39 in 0.007 seconds

	COD_CARTE	COD_LIBRA...	NR_EXEMPLARE
1	111	80	3
2	104	80	6
3	112	80	5
4	101	80	1
5	109	80	1
6	110	80	1
7	103	80	3

Cazul in care se cumpara toata seria:

project_SGBD.sql

SQL Worksheet History

Worksheet Query Builder

```
1549 into nume_carte  
1550 from carti  
1551 where cod_carte = retin_date.lista_carti_care_nu_se_gasesc(i);  
1552  
1553 DBMS_OUTPUT.PUT_LINE(nume_carte);  
1554  
1555 end loop;  
1556 end if;  
1557  
1558 vanzare(id_librarie, retin_date.lista_carti_gasite);  
1559 DBMS_OUTPUT.PUT_LINE('Asadar pretul final este de '||retin_date.pret_total);  
1560 end;  
1561 end;  
1562 /  
1563  
1564  
1565 begin  
1566 pachet14.cerere('Anotimpurile inimii', 'Strada Republicii'); --cod_librarie 80; cod_carti: 109, 110, 111, 112  
1567 --pachet14.cerere('Cantecul Acadie', 'Strada Republicii');  
1568 --pachet14.cerere('Cantecul Acadie', 'Strada Anonima');  
1569 --pachet14.cerere('Anonima', 'Strada Republicii');  
1570 exception
```

Script Output x Query Result 1 x Query Result 2 x

Task completed in 0.057 seconds

Pretul comenzii este de 87.4

Se gasesc cartile:
A fost odata intr-o vara
Promisiunea unei noi primaveri
Vanturi tomatice
Iarna nu tine o vesnicie
In librerie se gaseste toata seria. Se aplica reducere de 10%
Asadar pretul final este de 78.66

Date dupa apel:

1576
1577
1578
1579

```
select * from stoc;  
select * from bonuri;
```

Script Output x Query Result 1 x Query Result 2 x Query Result 3 x

SQL All Rows Fetched: 21 in 0.002 seconds

	COD_BON	DATA_BON	COD_CARTE	COD_ANGAJAT
1		187 03-JAN-22	112	403
2		186 03-JAN-22	111	403
3		184 03-JAN-22	109	403
4		185 03-JAN-22	110	403
5		200 03-JAN-22	109	403
6		196 27-DEC-21	107	409
7		195 27-DEC-21	102	410

Script Output x Query Result 1 x Query Result 2 x

SQL All Rows Fetched: 37 in 0.001 seconds

	COD_CARTE	COD_LIBRA...	NR_EXEMPLARE
1	112	80	4
2	104	80	6
3	111	80	2
4	101	80	1
5	103	80	3

Cazul in care nu exista toata seria pe stoc:

```
Worksheet | Query Builder
1554
1555         end loop;
1556     end if;
1557
1558     vanzare(id_librarie, retin_date.lista_carti_gasite);
1559     DBMS_OUTPUT.PUT_LINE('Asadar pretul final este de '||retin_date.pret_total);
1560 end;
1561 end;
1562 /
1563
1564 begin
1565     --pachet14.cerere('Anotimpurile inimii', 'Strada Republicii'); --cod_librarie 80; cod_carti: 109, 110, 111, 112
1566     pachet14.cerere('Cantecul Acadie', 'Strada Republicii');
1567     --pachet14.cerere('Cantecul Acadie', 'Strada Anonima');
1568     --pachet14.cerere('Anonima', 'Strada Republicii');
1569 exception
1570     when others then
1571         DBMS_OUTPUT.PUT_LINE(SQLERRM);
1572 end;
1573 /
1574
```

Pretul comenzii este de 27

Se gasesc cartile:
Mostenirea
In librerie se gasesc doar 1 carti din cele 4 ale seriei. Nu se aplica reducerea de 10%.

Nu se gasesc cartile:
Binecuvantatul tarm
Locul de intalnire
Limanul mult dorit
Un far calauzitor
Asadar pretul final este de 27

Date dupa apel:

```
.563
.564 begin
.565     --pachet14.cerere('Anotimpurile inimii', 'Strada Republicii')
.566     pachet14.cerere('Cantecul Acadie', 'Strada Republicii')
.567     --pachet14.cerere('Cantecul Acadie', 'Strada Anonima');
.568     --pachet14.cerere('Anonima', 'Strada Republicii');
.569 exception
.570     when others then
.571         DBMS_OUTPUT.PUT_LINE(SQLERRM);
.572 end;
.573 /
.574
.575
.576 select * from stoc;
.577 select * from bonuri;
```

	COD_BON	DATA_BON	COD_CARTE	COD_ANGAJAT
1	188	03-JAN-22	104	403
2	184	03-JAN-22	109	403
3	186	03-JAN-22	111	403
4	187	03-JAN-22	112	403
5	185	03-JAN-22	110	403
6	200	03-JAN-22	108	403

```
1573 /
1574
1575
1576 select * from stoc;
1577 select * from bonuri;
```

	COD_CARTE	COD_LIBR...	NR_EXEMPLARE
1	112	80	4
2	104	80	5
3	111	80	2
4	101	80	1
5	103	80	3
6	122	70	6

Cazul in care nu exista strada:

```
Worksheet | Query Builder
1549         into nume_carte
1550         from carti
1551         where cod_carte = retin_date.lista_carti_care_nu_se_gasesc(i);
1552
1553         DBMS_OUTPUT.PUT_LINE(nume_carte);
1554
1555     end loop;
1556 end if;
1557
1558     vanzare(id_librarie, retin_date.lista_carti_gasite);
1559     DBMS_OUTPUT.PUT_LINE('Asadar pretul final este de '||retin_date.pret_total);
1560 end;
1561 end;
1562 /
1563
1564 begin
1565 --pachet14.cerere('Anotimpurile inimii', 'Strada Republicii'); --cod_librarie 80; cod_carti: 109, 110,
1566 --pachet14.cerere('Cantecul Acadie', 'Strada Republicii');
1567 pachet14.cerere('Cantecul Acadie', 'Strada Anonima');
1568 --pachet14.cerere('Anonima', 'Strada Republicii');
1569 exception
1570     when others then
1571         DBMS_OUTPUT.PUT_LINE(SQLERRM);
1572 end;
1573 /
1574
```

Script Output x Query Result 1 x Query Result 2 x Query Result 3 x Query Result 4 x Query Result 5 x Query Result 6 x

Task completed in 0.071 seconds

ORA-20000: Nu exista librerie pe aceasta strada

PL/SQL procedure successfully completed.

Cazul in care nu exista seria de carti:

```
proiect_SGBD.sql
SQL Worksheet | History
Worksheet | Query Builder
1546         DBMS_OUTPUT.PUT_LINE('Nu se gasesc cartile: ');
1547         for i in retin_date.lista_carti_care_nu_se_gasesc.first..retin_date.lista_carti_care_nu_se_gasesc.last
1548         select denumire
1549         into nume_carte
1550         from carti
1551         where cod_carte = retin_date.lista_carti_care_nu_se_gasesc(i);
1552
1553         DBMS_OUTPUT.PUT_LINE(nume_carte);
1554
1555     end loop;
1556 end if;
1557
1558     vanzare(id_librarie, retin_date.lista_carti_gasite);
1559     DBMS_OUTPUT.PUT_LINE('Asadar pretul final este de '||retin_date.pret_total);
1560 end;
1561 end;
1562 /
1563
1564 begin
1565 --pachet14.cerere('Anotimpurile inimii', 'Strada Republicii'); --cod_librarie 80; cod_carti: 109, 110, 1
1566 --pachet14.cerere('Cantecul Acadie', 'Strada Republicii');
1567 --pachet14.cerere('Cantecul Acadie', 'Strada Anonima');
1568 pachet14.cerere('Anonima', 'Strada Republicii');
1569 exception
1570     when others then
1571         DBMS_OUTPUT.PUT_LINE(SQLERRM);
1572 end;
1573 /
```

Script Output x Query Result 1 x Query Result 2 x Query Result 3 x Query Result 4 x Query Result 5 x Query Result 6 x

Task completed in 0.055 seconds

ORA-20000: Nu exista serie cu acest nume

PL/SQL procedure successfully completed.