

Proiect final - Sisteme de gestiune a bazelor de date

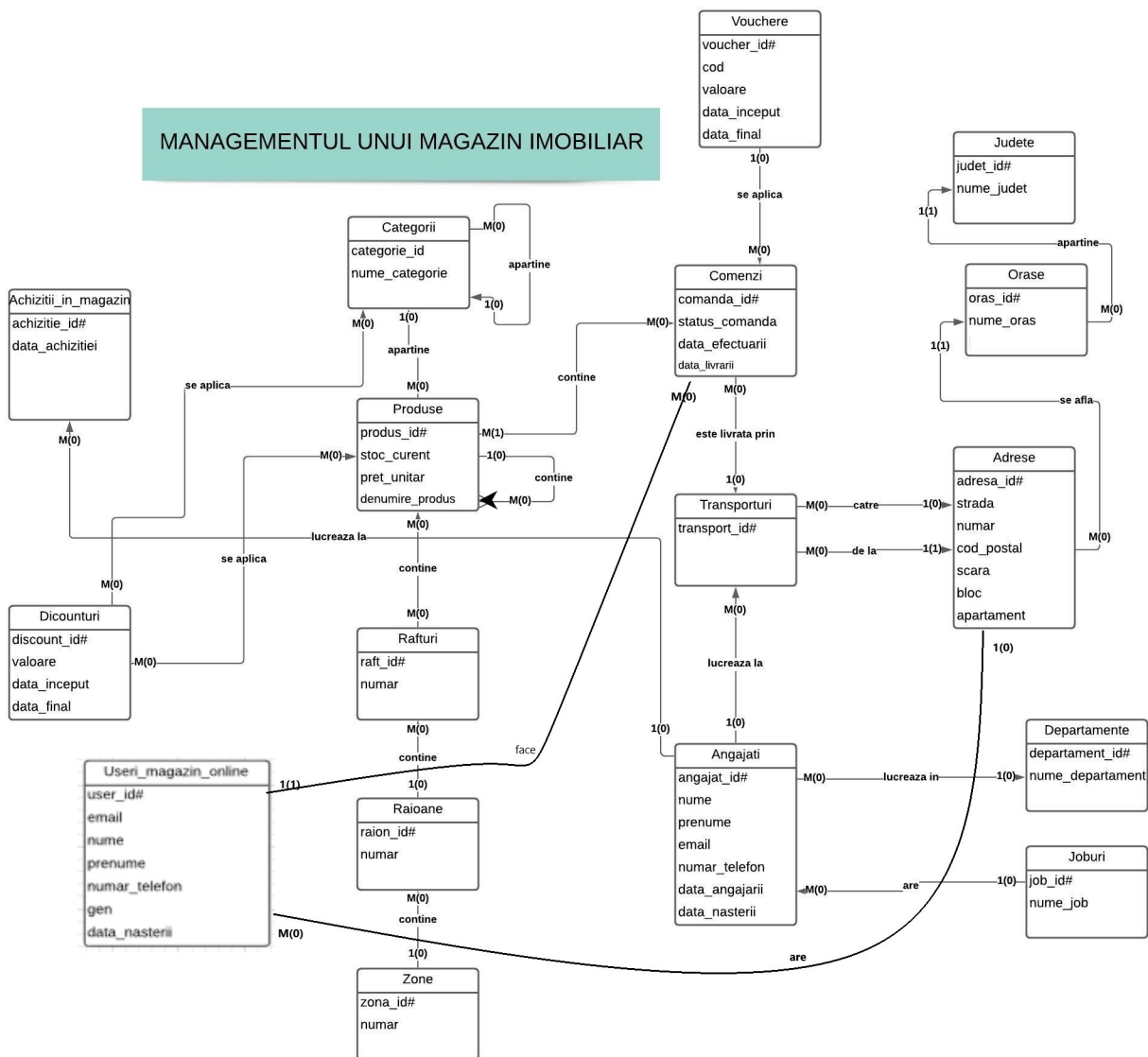
Stan Bianca-Mihaela, grupa 232

1 Prezentarea bazei de date

Baza de date implementeaza managementul unui magazin din Bucuresti care livreaza produse de mobilier in Romania. In contextul pandemiei, magazinul si-a imbunatatit platforma online. Prin intermediul bazei de date, aceasta integreaza vanzarea din magazin cu cea online pentru a oferi maxima satisfactie clientilor.

Conceptul provine dintr-o problema reala intampinata cu un magazin similar, care procesa comenzile chiar inainte sa le livreze, nu atunci cand acestea erau plasate. Astfel, stocul se putea goli, chiar daca un client plasase comanda cand acesta era plin. Aceasta baza de date incearca rezolvarea acestei probleme de logistica.

2 Diagrama Entitate-Relatie



Voi explica cateva dintre deciziile de design luate:

- Cand ma refer la un produs, ma refer la conceptul abstract de produs, nu la un obiect in sine. De aceea, tabela Produs are atributul "stoc_curent".
- Intre tabela USERI_MAGAZIN_ONLINE si ADRESE am o relatie de tip many to many. Un user poate adauga mai multe adrese in datele sale personale. Voi permite ca mai multi useri sa specifice aceeasi adresa pentru a lasa loc de erori umane ce vor fi rezolvate intre curier si client.
- Un voucher are o valoare fixa (100 de lei, 200 de lei, etc.) si poate fi aplicat pe o comanda sau in magazin.
- Un discount se poate aplica asupra unui produs sau asupra unei categorii. Nu se pot aplica mai multe discount-uri pe o categorie la un moment dat. In schimb, se pot adauga mai multe discount-uri pe un produs, cumulandu-se cu discount-ul de pe categoria sa (daca acesta exista) pe perioada suprapunerii discount-urilor.
exemplu:

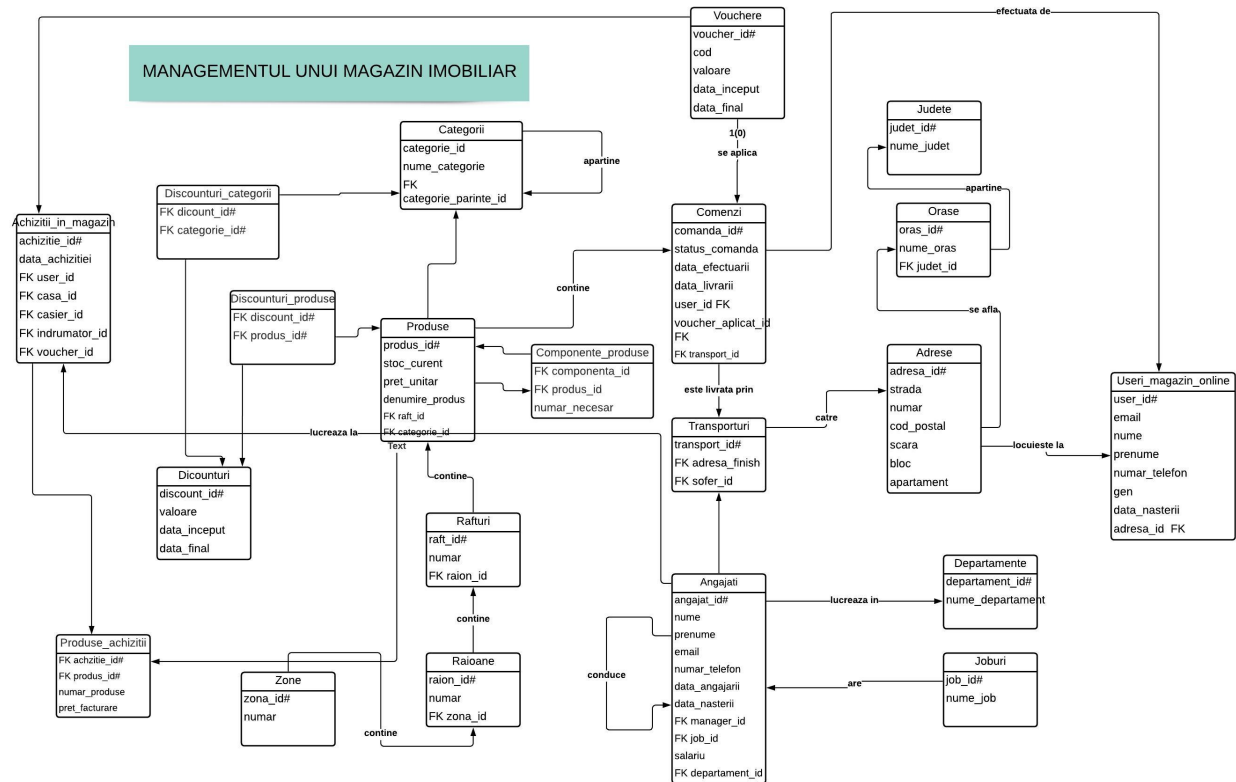
Produsul cu id 1 face parte din categoria cu id 2 si are pretul de 200 de lei. Pe produsul 1 aplic un discount de 10% in perioada 10.01.2021-30.01.2021. Deci pretul sau va fi de 180 de lei. Pe categoria 2 adaug un discount de 30 in perioada 12.01.2021-15.01.2021. => Pretul produsului in perioada 12-15 va fi de $\frac{180*70}{100}=126$ lei. Pentru perioada 15.01.2021-19.01.2021 mai adaug un discount asupra produsului 1, de 10%.

Deci pe data de 15.01.2021 pretul produsului va fi $\frac{126*90}{100}=113,4$ lei.

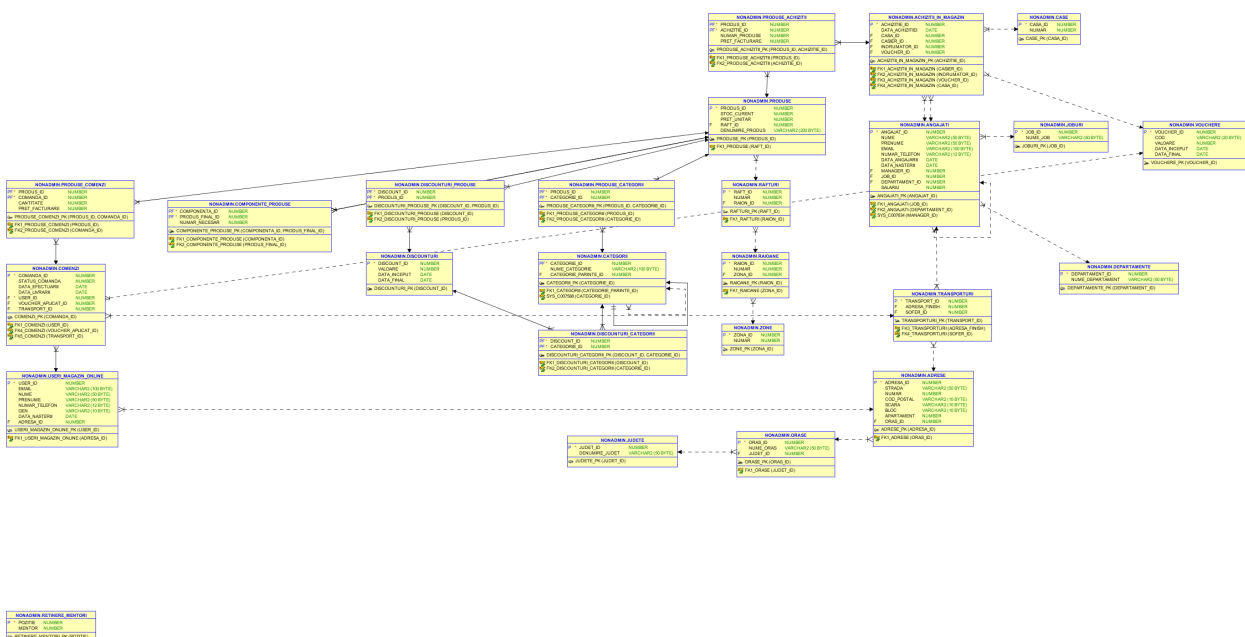
Iar in perioada 16-19 pretul va fi $\frac{180*90}{100}=162$ lei.

- La prima vedere, faptul ca pe o categorie se poate aplica maxim un discount la un moment dat pare ca ar determina o relatie one to many. In realitate, aceasta este o relatie many to many pentru ca de-a lungul timpului, pe o categorie se pot aplica mai multe discounturi. Constrangerea va trebui impusa de adminul bazei de date.
- Pentru tabela ZONE am 2 attribute: zona.id si numar. La prima vedere numar pare redundant, insa trebuie sa tinem cont ca in magazin zonele sunt asezate in functie de spatiu si locatie, astfel incat sa fie usor de accesat si parcurs. Astfel, daca am 3 zone(subsol - (id 1, numar -1), parter - (id 2, numar 0) si etaj 2 - (id 3, numar 2) si extind spatiul de depozitare si pe etajul 1, are sens ca numar sa fie 1, iar id sa fie 4. Un alt avantaj al acestei decizii este ca pot sa redenumesc oricand zonele (daca schimb intrarea, fac alta organizare, etc) fara a fi nevoie sa schimb in baza de date toate legaturile Zona-Raion. Analog pentru RAIOANE si RAFTURI.
- Pentru o comanda poate fi aplicat un singur voucher. In schimb, intr-o comanda un user poate achizitiona oricate vouchere, iar pentru asta am adaugat tabela COMENZI_VOUCHERE.
- Magazinul vinde obiecte de mobilier care au mai multe parti (o masa are 4 picioare si un blat). Fiecare dintre aceste componente este un produs care poate fi achizitionat individual. Astfel, am construit tabelul COMPONENTE_PRODUSE care, pentru fiecare produs compus, imi spune componentele sale si numarul acestora.
Aici va interveni o problema mai complexa, si anume, daca in stoc erau 5 picioare si se vand 2, trebuie updatat atat stocul piciorului, cat si cel al mesei (pentru o masa e nevoie de 4 picioare).
- Pentru tabelul COMENZI am atributul status_comanda: 0-in decurs de procesare, 1-trimisa la curier, 2-livrata, 3-anulata

3 Diagrama conceptuala



4 Implementare in Oracle



Am ales sa folosesc tipul de date number in toate cazurile in care avem nevoie sa lucrez cu numere pe tabele.
 "If a precision is not specified, the column stores values as given. If no scale is specified, the scale is zero." [1]

5 Adaugarea informatiilor

Am ales sa adaug informatii mai complexe cu ajutorul PLSQL si a pachetului dbms_random. Nu voi adauga aici programele pentru popularea fiecărei tabele, ci doar pe cele mai interesante.

Popularea tabelor a fost un proces complex, astfel incat in final:

- tabela PRODUSE are 1000 de linii
- tabela PRODUSE_COMENZI are 8600 de linii

5.1 COMENZI

```
1 declare
2 status_comanda number;
3 data_efectuarii date;
4 data_livrării date;
5 us_id number;
6 adr_id number;
7 sofer_id number;
8 type v_array is varray(100) of number;
9 soferi v_array:=v_array();
10 trans_id number;
11 voucher number;
12 valoare_random number;
13 begin
14     dbms_random.seed(val =>0);
15     —pun toti soferii in soferi
16     select angajat_id
17     bulk collect into soferi
18     from angajati
19     where job_id=7;
20
21     —voi introduce 130 de comenzi
22     for i in 1..130 loop
23
24         —iau cu random statusul comenzii
25         status_comanda:=round(dbms_random.value(high=>3, low=>0));
26         —si id-ul userului
27         us_id:=round(dbms_random.value(high=>100, low=>1));
28
29         data_efectuarii:=to_date(trunc(dbms_random.value(to_char(date '2000-01-01', 'J'),
30             to_char(date '2020-01-01', 'J'))), 'J');
31         data_livrării:=to_date(trunc(dbms_random.value(to_char(data_efectuarii, 'J'), (
32             to_char(add_months(data_efectuarii, 3), 'J'))), 'J');
33         voucher:=round(dbms_random.value(low=>1, high=>100));
34         valoare_random:=round(dbms_random.value(1, 1000));
35
36         —asa reduc sansele ca pe o comanda sa se aplice un voucher la 10%
37         if mod(valoare_random, 10)>0 then
38             voucher:=0;
39         end if;
40         —daca comanda a fost livrata
41         if status_comanda=2 then
42             —trebuie sa adaug si transportul in lista de transporturi
43             —deci imi trebuie adresa utilizatorului care a efectuat comanda
44             select adresa_id
45             into adr_id
46             from useri_magazin_online
47             where user_id=us_id;
48
49             —adaug trasnportul
50
51             trans_id:=seq1.nextval;
52             insert into transporturi
53             values (trans_id, adr_id, soferi(round(dbms_random.value(low=>1, high=>14))));
```

```

52      —adaug comanda, cu tot cu id-ul transportului
53      insert into comenzi(comanda_id, status_comanda, data_efectuarii, data_livrarii,
54          user_id, voucher_aplicat_id, transport_id)
55      values (i, —id
56      status_comanda, —status
57      data_efectuarii,
58      data_livrarii, —daca comanda nu a fost livrata nu data_livrarii e null
59      us_id,—id user
60      decode(voucher,0, null, voucher), —am luat o valoare random cu 10% sanse sa
        aplice un voucher
61          —daca valoarea din random e divizibila cu 10 voi alege un
        voucher random care sa se aplice pe comanda, altfel am null
62      trans_id);
63  else
64      —altfel, la transport_id am null
65      insert into comenzi(comanda_id, status_comanda, data_efectuarii, data_livrarii,
66          user_id, voucher_aplicat_id, transport_id)
67      values (i, —id
68      status_comanda, —status
69      data_efectuarii,
70      null, —daca comanda nu a fost livrata nu data_livrarii e null
71      us_id,—id user
72      decode(voucher,0, null, voucher), —am luat o valoare random cu 10% sanse sa
        aplice un voucher
73          —daca valoarea din random e divizibila cu 10 voi alege un
        voucher random care sa se aplice pe comanda, altfel am null
74      null);
75  end if;
76 end loop;
77 end;
/

```

populare comenzi.txt

5.2 COMPONENTE_PRODUSE

```

1  declare
2  numar_necesar number;
3  numar_subcomponente number;
4  componenta number;
5  componenta_necesara_stoc_curent number;
6  pret_componenta number;
7  index_nou_produs number;
8  nr number:=0;
9  begin
10     —voi genera 200 de produse care incapsuleaza alte produse
11     for i in 1..200 loop
12         —produsul i va avea numar_componente
13         numar_subcomponente:=round(dbms_random.value(low=>2, high=>6));
14         index_nou_produs:=800+i;
15
16         —inserez produsul incapsulator in tabela
17         insert into produse
18         values(index_nou_produs, null, 0, null, 'Produs '||index_nou_produs);
19         —voi adauga relatiile de incapsulare
20         for j in 1..numar_subcomponente loop
21             numar_necesar:= round(dbms_random.value(low=>1, high=>10));
22             componenta:=round(dbms_random.value(low=>1, high=>800));
23
24             if componenta<=100 or componenta>=161 then
25                 select count(*)
26                 into nr
27                 from componente_produse
28                 where componenta_id=componenta
29                 and produs_final_id=index_nou_produs;

```

```

30
31         if nr=0 then
32
33             insert into componente_produce
34             values (componenta,
35             index_nou_produus , numar_necesar);
36
37             —stocul curent din produsul incapsulator o sa fie min(stoc_curent ,
38             stoc_curent_componenta_necesara/numar_necesar)
39
40             select stoc_curent , pret_unitar
41             into componenta_necesara_stoc_curent , pret_componenta
42             from produse
43             where produs_id=componenta;
44
45             dbms_output.put_line(floor(componenta_necesara_stoc_curent/numar_necesar
46             ));
47
48             —
49             set stoc_curent=decode(stoc_curent , null , floor(
50             componenta_necesara_stoc_curent/numar_necesar),
51             —
52             decode(stoc_curent>floor(componenta_necesara_stoc_curent/numar_necesar
53             ), true , floor(componenta_necesara_stoc_curent/numar_necesar), stoc_curent)),
54
55
56             update produse
57             set stoc_curent= case
58                 when stoc_curent is null then floor(
59                     componenta_necesara_stoc_curent/numar_necesar)
60                 when stoc_curent>floor(
61                     componenta_necesara_stoc_curent/numar_necesar)
62                     then floor(componenta_necesara_stoc_curent/
63                     numar_necesar)
64                 else stoc_curent
65                 end ,
66             pret_unitar=pret_unitar+pret_componenta*numar_necesar
67             where produs_id=index_nou_produus;
68         else
69             nr:=0;
70         end if;
71     end if;
72 end loop;
73 end loop;
74 end;
75 /

```

populare componente_produce.txt

6 Colectie

In cazul achizitiilor in magazin, fiecare client poate primi un "indrumator"- Sales Consultant. Acesta ajuta clientii sa ajunga la o decizie finala in redecorarea locuintei lor. Vrem sa implementam un nou sistem de mentorat, astfel incat sa avem cate un mentor la 10 indrumatori. Cine devine mentor? Implementam un sistem de rotatie. Avand in vedere ca aceasta este prima luna in care intra in vigoare acest sistem, vor deveni mentori cei mai productivi indrumatori din ultima luna. Cat timp sunt mentori, acestia nu pot sa mai ofere consultanta clientilor. In fiecare luna, cel mai productiv indrumator va deveni mentor, iar cel mai vechi mentor va reveni la stadiul de indrumator.

Pe perioada mentoratului, mentorii au un salariu cu 10% mai mare. De mentionat ca numarul de mentori necesar variaza in functie de numarul de indrumatori din magazin.

Scrieti o functie care intoarce costul suplimentar necesar pentru implementarea acestui sistem in prima luna (suma maririlor de salarii).

Aceasta functie utilizeaza colectia de tip **tablou indexat**.

```

1 function exercitiul_6
2 return number
3 as
4 cei_mai_productivi_indrumatori indexed_by_table_of_info_mentor;
5 nr_necesar_mentori number;
6 productivitate_maxima number;
7 contor_productivi number:=1;
8 nr_indrumatori number;
9 indrumatori_cu_productivitate_maxima number;
10 suma_maririlor_de_salariu number:=0;
11 marire_salariu number;
12 begin
13
14     —aici nu poate sa imi dea eroare pentru ca am count
15     select count(*)
16     into nr_indrumatori
17     from angajati
18     where job_id=8;
19
20     —nici aici nu poate sa imi dea eroare pentru ca am bulk collect into
21     select e.indrumator_id, sum(f.pret_facturare*f.numar_produce), max(a.nume), max(a.
22         prenume)
23     bulk collect into cei_mai_productivi_indrumatori
24     from achizitii_in_magazin e, produse_achizitii f, angajati a
25     where e.achizitie_id=f.achizitie_id
26     and e.indrumator_id=a.angajat_id
27     and e.data_achizitiei>add_months(sysdate, -1)
28     and e.data_achizitiei<=sysdate
29     group by e.indrumator_id
30     order by sum(f.pret_facturare*f.numar_produce) desc;
31
32     nr_necesar_mentori:=floor(nr_indrumatori/10);
33     dbms_output.put_line('nr necesar mentori: ' || nr_necesar_mentori);
34     afisare_mentori();
35
36     —daca inca nu am pus numarul necesar de mentori sau daca inca nu am pus toti angajatii
37     cu productivitate maxima
38     while mentori.count<nr_necesar_mentori or cei_mai_productivi_indrumatori(
39         contor_productivi).suma=cei_mai_productivi_indrumatori(1).suma loop
40         —daca inca nu am pus toti mentorii
41         if mentori.count<=nr_necesar_mentori then
42             mentori(contor_mentori):=cei_mai_productivi_indrumatori(contor_productivi).
43                 id_angajat;
44
45             select round(salariu*1/10, 2)
46             into marire_salariu
47             from angajati
48             where angajat_id=mentori(contor_mentori);
49
50             update angajati
51             set salariu=round(salariu*11/10, 2)
52             where angajat_id=mentori(contor_mentori);
53
54             suma_maririlor_de_salariu:=suma_maririlor_de_salariu+marire_salariu;
55
56             dbms_output.put_line('Angajatul ' || cei_mai_productivi_indrumatori(
57                 contor_productivi).nume || ' ' || cei_mai_productivi_indrumatori(
58                 contor_productivi).prenume || ' a devenit mentor!');
59             contor_mentori:=contor_mentori+1;
60             if contor_mentori = nr_necesar_mentori+1 then
61                 contor_mentori:=1;
62             end if;
63             contor_productivi:=contor_productivi+1;
64
65         else
66             —daca deja am pus destui pun doar cat timp mai au productivitate maxima
67             if cei_mai_productivi_indrumatori(contor_productivi).suma =

```



```

62      cei_mai_productivi_indrumatori(1).suma then
63
64      --too_many_rows nu se poate pentru ca cheia primara e unica
65      update angajati
66      set salariu=round(salariu*9/10,2)
67      where angajat_id=mentori(contor_mentori);
68
69      mentori(contor_mentori):=cei_mai_productivi_indrumatori(contor_productivi).
70      id_angajat;
71
72      select salariu*round(1/10,2)
73      into marire_salariu
74      from angajati
75      where angajat_id=mentori(contor_mentori);
76
77      update angajati
78      set salariu=round(salariu*11/10, 2)
79      where angajat_id=mentori(contor_mentori);
80
81      suma_maririlor_de_salariu:=suma_maririlor_de_salariu+marire_salariu;
82
83      dbms_output.put_line('Angajatul ' || cei_mai_productivi_indrumatori(
84      contor_productivi).nume || ' ' || cei_mai_productivi_indrumatori(
85      contor_productivi).prenume || ' a devenit mentor!');
86
87      contor_mentori:=contor_mentori+1;
88      if contor_mentori = nr_necesar_mentori+1 then
89          contor_mentori:=1;
90      end if;
91      contor_productivi:=contor_productivi+1;
92      end if;
93
94      end if;
95
96      return suma_maririlor_de_salariu;
97
98      exception
99      when no_data_found then
100          raise_application_error(-20000, 'Angajatul nu a fost gasit!');
101      when others then
102          raise_application_error(-20001, 'Alta eroare! Codul erorii: ' || SQLCODE || ',
103          mesajul erorii: ' || SQLERRM);
104      end exercitiul_6;

```

exercitiul 8.txt

La rularea acestui program obținem:

```

28
29 begin
30 proiect.contor_mentori:=1;
31 proiect.mentori.delete;
32 dbms_output.put_line(proiect.exercitiul_6());
33 end;
34 /
35
36
27 begin

```

Script Output x Query Result x

Task completed in 13.349 seconds

nr necesar mentori: 7

Angajatul Harper Logan a devenit mentor!
 Angajatul Francis Harper a devenit mentor!
 Angajatul Joyce Mason a devenit mentor!
 Angajatul Khan Michael a devenit mentor!
 Angajatul Bailey Gabriel a devenit mentor!
 Angajatul Allen Christopher a devenit mentor!
 Angajatul Frost Ethan a devenit mentor!
 10758.56

PL/SQL procedure successfully completed.

7 Cursor

Pe pagina principala vrem sa afisam cele mai vandute produse in functie de categorie. Stim insa ca ordinea in care afisam produsele este foarte importanta, asa ca vom afisa cele mai populare categorii primele. Vom afisa 10 produse din fiecare categorie.

Mai mult de atat, vrem sa avem o prezentare personalizata pentru fiecare user. Astfel, pentru a nu fi redundanti, vrem sa afisam un produs pe prima pagina doar daca probabilitatea ca user-ul sa il cumpere este mare. Pentru asta, trebuie sa vedem care este numarul mediu de bucati cumparate din acel produs per user. Daca user-ul logat deja a cumparat mai mult decat media, consideram ca fie nu va mai cumpara mai mult (mobiliul nu este schimbat foarte des), fie este deja familiar cu produsul si il va cauta singur in cadrul site-ului.

Scrieti o functie care daca:

- primeste ca parametru id-ul userului afiseaza o recomandare pentru acesta
- daca nu primeste parametru afiseaza recomandari pentru un user neautentificat

Pentru implementarea acestui exercitiu am folosit un **ciclu cursor cu subcereri**.

```

1 create or replace procedure exercitiul_7 (user_conectat_id number default null)
2 is
3 nr_linii number:=1;
4 cantitate_user_conectat number:=0;
5 exista_user number:=0;
6 begin
7   --merg prin toate categoriile
8   if user_conectat_id is not null then
9     select count(*)
10    into exista_user
11   from useri_magazin_online
12  where user_id=user_conectat_id;
13
14   if exista_user=0 then
15     raise_application_error(-20006, 'Nu exista un user cu id-ul dat!');

```

```

16     end if;
17 end if;
18
19 for categorie in (select nume_categorie, categorie_id, level
20                  from categorii
21                  start with categorie_parinte_id is null
22                  connect by prior categorie_id=categorie_parinte_id) loop
23     dbms_output.put_line('
24
25         ');
26     dbms_output.put_line(rpad(' ', (categorie.level-1)*5, ' ') || categorie.level || '
27         ' || 'Categoria: ' || categorie.nume_categorie);
28     dbms_output.put_line('
29
30         ');
31     nr_linii:=1;
32     --prin toate produsele din acea categorie
33     for produs in ( select max(c.denumire_produs) denumire, sum(b.cantitate) numar, b.
34                    produs_id
35                    from produse_comenzi b, produse c
36                    where b.produc_id=c.produc_id
37                    and c.categorie_id=categorie.categorie_id
38                    group by b.produc_id
39                    order by sum(b.cantitate) desc
40                    ) loop
41
42         --afisez doar primele 10 cele mai importante
43         if nr_linii<=10 then
44             --ca sa evit eroarea no data found numar mai intai cate produse de acest
45             tip a comandat user-ul conectat
46             if user_conectat_id!=null then
47                 select count(*)
48                 into cantitate_user_conectat
49                 from comenzi a, produse_comenzi b
50                 where a.comanda_id=b.comanda_id
51                 and b.produc_id=produs.produc_id
52                 and a.user_id=user_conectat_id;
53
54                 if cantitate_user_conectat>0 then
55                     select sum(b.cantitate)
56                     into cantitate_user_conectat
57                     from comenzi a, produse_comenzi b
58                     where a.comanda_id=b.comanda_id
59                     and b.produc_id=produs.produc_id
60                     and a.user_id=user_conectat_id
61                     group by a.user_id;
62
63                     --daca a comandat produse de acest tip dar mai putin decat media
64                     , afisam
65                     if cantitate_user_conectat < proiect.bucati(produs.produc_id).
66                        fst/proiect.bucati(produs.produc_id).snd then
67                         dbms_output.put_line(rpad(' ', categorie.level*5, ' ') ||
68                             nr_linii || ' Produsul: ' || produs.denumire || ' numar
69                             cumparate: ' || produs.numar);
70                         nr_linii:=nr_linii+1;
71                     end if;
72
73                 else
74                     --daca nu a comandat deloc produse de acelasi tip afisez
75                     dbms_output.put_line(rpad(' ', (categorie.level+1)*5, ' ') ||
76                         nr_linii || ' Produsul: ' || produs.denumire || ' numar
77                         cumparate: ' || produs.numar);
78                     nr_linii:=nr_linii+1;
79                 end if;
80             else
81                 dbms_output.put_line(rpad(' ', (categorie.level+1)*5, ' ') ||
82                     nr_linii || ' Produsul: ' || produs.denumire || ' numar cumparate:
83                     ' || produs.numar);

```

```

69         end if;
70     end if;
71 end loop;
72 end loop;
73 end;

```

exercitiul 7.txt

La rularea acestui exercitiu am obtinut:

```

76 begin
77 exercitiul_7(1);
78 end;
79 /

```

Script Output x Explain Plan x

Task completed in 0.092 seconds

```

4. Produsul: Produs 50 numar cumarate: 211
5. Produsul: Produs 128 numar cumarate: 197
-----
1. Categoria: Categorie 26
-----
    1. Produsul: Produs 146 numar cumarate: 268
    2. Produsul: Produs 33 numar cumarate: 225
-----
2. Categoria: Categorie 24
-----
    1. Produsul: Produs 19 numar cumarate: 295
    2. Produsul: Produs 70 numar cumarate: 275
    3. Produsul: Produs 52 numar cumarate: 234
    4. Produsul: Produs 135 numar cumarate: 228
    5. Produsul: Produs 94 numar cumarate: 226
-----
3. Categoria: Categorie 15
-----
    1. Produsul: Produs 116 numar cumarate: 265
    2. Produsul: Produs 81 numar cumarate: 196
    3. Produsul: Produs 73 numar cumarate: 188
-----
3. Categoria: Categorie 16
-----
    1. Produsul: Produs 58 numar cumarate: 286
    2. Produsul: Produs 66 numar cumarate: 256
    3. Produsul: Produs 93 numar cumarate: 231
    4. Produsul: Produs 24 numar cumarate: 184
-----
2. Categoria: Categorie 25

```

Pot rula aceasta functie si fara aramtru si imi afiseaza recomandarile pentru un user neautentificat:

```
76 begin
```

```
77 exercitiul_7();
```

```
78 end;
```

```
79 /
```

```
80
```

Script Output x

Task completed in 3.218 seconds

1. Produsul: Produs 123 numar cumarate: 233

1. Produsul: Produs 105 numar cumarate: 213

1. Produsul: Produs 152 numar cumarate: 202

1. Produsul: Produs 160 numar cumarate: 195

2. Categoria: Categorie 2

1. Produsul: Produs 87 numar cumarate: 262

1. Produsul: Produs 55 numar cumarate: 255

1. Produsul: Produs 34 numar cumarate: 240

1. Produsul: Produs 101 numar cumarate: 238

1. Produsul: Produs 114 numar cumarate: 237

1. Produsul: Produs 26 numar cumarate: 230

1. Produsul: Produs 82 numar cumarate: 230

1. Produsul: Produs 129 numar cumarate: 226

2. Categoria: Categorie 8

Daca rulez pe un user invalid:

```
76 begin
```

```
77 exercitiul_7(-10);
```

```
78 end;
```

```
79 /
```

Script Output x

Task completed in 0.363 seconds

```
exercitiul_7(-10);
```

```
end;
```

Error report -

ORA-20006: Nu exista un user cu id-ul dat!

ORA-06512: at "NONADMIN.EXERCITIUL_7", line 15

ORA-06512: at line 2

8 Functie

Avand in vedere ca magazinul nostru vinde si componente iar unele produse sunt facute din mai mult de o componenta, scrieti un subprogram care ii returneaza clientului o lista cu componentele si locatiile in magazin pe care trebuie sa le caute pentru a avea produsul X (complet).

Aceasta functie returneaza un tip de date complex: un **tablou indexat** ce contine un tip definit de utilizator. Aceasta functie face un join intre 5 tabele: PRODUSE, COMPONENTE_PRODUSE, RAFTURI, RAIOANE si ZONE.

```

1 function exercitiul_8 (produs_cautat varchar2) return proiect.indexed_by_table_of_componente
2 as
3 produs number;
4 lista proiect.indexed_by_table_of_componente;
5 begin
6     produs:=get_produs_id(produs_cautat);
7
8     —nu avem cum sa avem exceptii fiindca avem bulk collect into
9     select b.componenta_id, a.denumire_produs, c.raft_id, d.raion_id, e.zona_id
10    bulk collect into lista
11   from produse a, componente_produce b, rafturi c, raioane d, zone e
12  where b.produs_final_id=produs
13     and a.produs_id=b.componenta_id
14     and a.raft_id=c.raft_id
15     and c.raion_id=d.raion_id
16     and d.zona_id=e.zona_id;
17
18
19     if lista.count=0 then
20         select produs_id, denumire_produs, b.numar, c.numar, d.numar
21        bulk collect into lista
22       from produse a, rafturi b, raioane c, zone d
23      where a.raft_id=b.raft_id
24         and b.raion_id=c.raion_id
25         and c.zona_id=d.zona_id
26         and a.produs_id=produs;
27     end if;
28     return lista;
29 end exercitiul_8;

```

gasire_produs.txt

Atunci cand caut un produs compus obtin (am modificat denumirea functiei, docul e acelasi):

```

52 declare
53     lista proiect.indexed_by_table_of_componente;
54     begin
55         lista:= gasire_produs('Produs 101');
56         dbms_output.put_line('Trebuie sa cautati ' || lista.count || ' produse:');
57         for i in nvl(lista.first,0)..nvl(lista.last,-1) loop
58             dbms_output.put_line('-produsul cu id ' || lista(i).id_produs || ' si denumirea ' || lista(i).denumire_produs
59                                 || ' la raftul ' || lista(i).raft_id || ' de la raionul ' || lista(i).raion_id || ' din zona ' || lista(i).zona_id);
60         end loop;
61     end;
62 /
63

```

Script Output x Query Result x Explain Plan x

Task completed in 0.04 seconds

PL/SQL procedure successfully completed.

Trebuie sa cautati 3 produse:

- produsul cu id 10 si denumirea Produs 10 la raftul 149 de la raionul 7 din zona 2
- produsul cu id 50 si denumirea Produs 50 la raftul 294 de la raionul 12 din zona 4
- produsul cu id 61 si denumirea Produs 61 la raftul 137 de la raionul 17 din zona 4

Atunci cand caut un produs care nu e compus obtin:

```

52 declare
53 lista proiect.indexed_by_table_of_componente;
54 begin
55     lista:= gasire_produs('Produs 10');
56     dbms_output.put_line('Trebuie sa cautati ' || lista.count || ' produse:');
57     for i in nvl(lista.first,0)..nvl(lista.last,-1) loop
58         dbms_output.put_line('-produsul cu id ' || lista(i).id_produs || ' si denumirea ' || lista(i).denumire_produs
59         || ' la raftul ' || lista(i).raft_id || ' de la raionul ' || lista(i).raion_id || ' din zona ' || lista(i).zona_id);
60     end loop;
61 end;
62 /
63

```

Script Output x Query Result x Explain Plan x

Task completed in 0.501 seconds

PL/SQL procedure successfully completed.

Trebuie sa cautati 1 produse:
 -produsul cu id 10 si denumirea Produs 10 la raftul 149 de la raionul 7 din zona 2

PL/SQL procedure successfully completed.

Atunci cand caut un produs care nu exista in magazin obtin:

```

52 declare
53 lista proiect.indexed_by_table_of_componente;
54 begin
55     lista:= gasire_produs('Produs -10');
56     dbms_output.put_line('Trebuie sa cautati ' || lista.count || ' produse:');
57     for i in nvl(lista.first,0)..nvl(lista.last,-1) loop
58         dbms_output.put_line('-produsul cu id ' || lista(i).id_produs || ' si denumirea ' || lista(i).denumire_produs
59         || ' la raftul ' || lista(i).raft_id || ' de la raionul ' || lista(i).raion_id || ' din zona ' || lista(i).zona_id);
60     end loop;
61 end;
62 /
63

```

Script Output x Query Result x Explain Plan x

Task completed in 0.078 seconds

dbms_output.put_line('-produsul cu id ' || lista(i).id_produs || ' si denumirea ' || lista(i).denumire_produs
 || ' la raftul ' || lista(i).raft_id || ' de la raionul ' || lista(i).raion_id || ' din zona ' || lista(i).zona_id);
 end loop;
 end;
 Error report -
 ORA-20004: Nu exista niciun produs cu aceasta denumire in magazin!
 ORA-06512: at "NONADMIN.GET_PRODUS_ID", line 13
 ORA-06512: at "NONADMIN.GASIRE_PRODUS", line 6
 ORA-06512: at line 4

9 Procedura

Scrieti o procedura de adaugare a unei comenzi. Subprogramul va primi un numar de produse, numele acestora, cantitatea din fiecare, id-ul userului si data efectuarii comenzii. Pentru a adauga o comanda avem nevoie de:

- un id pentru comanda
- statusul comenzii (default 0)
- data efectuarii (data de la tastatura)
- data_livrarii (null)
- id-ul userului
- id-ul voucherului, daca e cazul
- id-ul transportului (null)

Pentru a adauga produse in comanda avem nevoie de:

- id-ul produsului
- id-ul comenzii
- cantitatea
- pretul de facturare (pentru acesta avem nevoie sa cautam ce discounturi erau aplicate produsului la data efectuării comenzii si ce discount era aplicat pe categoria produsului la data efectuării comenzii)

Asadar, avem tabelele COMENZI, PRODUSE_COMENZI, PRODUSE, PRODUSE_CATEGORII, DISCOUNTURI_CATEGORII, DISCOUNTURI_PRODUSE.

```

1 create or replace procedure exercitiul_9 (prod nested_table1, id_user useri_magazin_online.
   user_id%type, id_voucher number)
2 as
3 id_comanda number;
4 pret_facturare number;
5 type v_array is varray(100) of number;
6 valori v_array;
7 valoare number;
8 stoc_produs number;
9 nr number;
10 stoc_initial number;
11 stoc_final number;
12 begin
13     id_comanda:=seq7.nextval;
14
15 —se adauga comanda
16     insert into comenzi
17     values(id_comanda, 0, sysdate, null, id_user, id_voucher, null);
18     dbms_output.put_line('S-a adaugat comanda cu id-ul '|| id_comanda);
19     for i in prod.first..prod.last loop
20 —pentru feicare produs.cerut
21         select pret_unitar, stoc_curent
22         into pret_facturare, stoc_produs
23         from produse
24         where produs_id=prod(i).produs_id;
25
26         if stoc_produs>=prod(i).numar_produce then
27
28             select a.valoare
29             bulk collect into valori
30             from discounturi a, discounturi_produce b
31             where a.discount_id=b.discount_id
32             and b.produs_id=prod(i).produs_id
33             and data_inceput<=sysdate
34             and sysdate>=data_final
35             order by data_inceput;
36
37 —se aplica discount-urile pe produs
38         for valoare in 1..valori.count loop
39             pret_facturare:=(100-valoare)*pret_facturare/100;
40         end loop;
41
42 —se cauta un discount in data comenzii pe categoria produsului
43         select count(*)
44         into nr
45         from discounturi_categorii a, discounturi b
46         where a.discount_id=b.discount_id
47         and b.data_inceput<=sysdate
48         and b.data_final>=sysdate
49         and a.categorie_id=(select categorie_id
50                             from produse
51                             where produs_id=prod(i).produs_id);
52         if nr>0 then
53             select b.valoare
54             into valoare
55             from discounturi_categorii a, discounturi b

```



```

56         where a.discount_id=b.discount_id
57         and b.data_inceput<=sysdate
58         and b.data_final>=sysdate
59         and a.categorie_id=(select categorie_id
60                             from produse
61                             where produs_id=prod(i).produs_id);
62
63         pret_facturare:=(100-valoare)*pret_facturare/100;
64
65     end if;
66     insert into produse_comenzi(produs_id, comanda_id, cantitate, pret_facturare)
67     values(prod(i).produs_id, id_comanda, prod(i).numar_produce, pret_facturare);
68     dbms_output.put_line('S-au comandat ' || prod(i).numar_produce || ' produse cu id
        -ul ' || prod(i).produs_id);
69
70     select stoc_curent
71     into stoc_initial
72     from produse
73     where produs_id=prod(i).produs_id;
74
75     update produse
76     set stoc_curent=stoc_curent-prod(i).numar_produce
77     where produs_id=prod(i).produs_id;
78
79     stoc_final:=stoc_initial-prod(i).numar_produce;
80
81     update produse
82     set stoc_curent=floor(stoc_final/(select numar_necesar
83                                     from componente_produce
84                                     where produs_final_id=produs_id
85                                     and componenta_id=prod(i).produs_id))
86     where produs_id in (select produs_final_id
87                       from componente_produce
88                       where produs_final_id=produs_id
89                       and componenta_id=prod(i).produs_id);
90
91     stoc_final:=stoc_initial-prod(i).numar_produce;
92     dbms_output.put_line('Stoc initial: ' || stoc_initial || ', stoc final : ' ||
        stoc_final);
93
94     else
95         dbms_output.put_line('Nu se pot achizitiona ' || prod(i).numar_produce || '
            exemplare din produsul cu id ' || prod(i).produs_id || ', doar ' ||
            stoc_product || ' sunt disponibile');
96     end if;
97 end loop;
98 exception
99     when no_data_found then
100         raise_application_error(-20004, 'Nu exista produse cu codul dat!');
101     when others then
102         raise_application_error(-20006, 'Alta eroare! Codul erorii: ' || SQLCODE || ',
            mesajul erorii: ' || SQLERRM);
103 end;

```

exercitiul 9.txt

Produsul 101 are nevoie de 2 bucati din produsul 61.


```

3  select * from produse
4  where produs_id=101;
5
6
7  declare
8  prod nested_table1:=nested_table1();
9  begin
10     prod.extend;
11     prod(1):=info_prod(61, 3);
12     exercitiul_9(prod, 3, null);
13 end;
14 /

```

Explain Plan x | Script Output x | Query Result x

SQL | All Rows Fetched: 1 in 0.003 seconds

	PRODUS_ID	STOC_CURENT	PRET_UNITAR	RAFT_ID	DENUMIRE_PRODUS
1	101	0	43472	(null)	Produs 101

Ramane un singur produs 61 in stoc:

```

3  select * from produse
4  where produs_id=61;
5
6
7  declare
8  prod nested_table1:=nested_table1();
9  begin
10     prod.extend;
11     prod(1):=info_prod(61, 3);
12     exercitiul_9(prod, 3, null);
13 end;
14 /

```

Explain Plan x | Script Output x | Query Result x

SQL | All Rows Fetched: 1 in 0.003 seconds

	PRODUS_ID	STOC_CURENT	PRET_UNITAR	RAFT_ID	DENUMIRE_PRODUS
1	61	1	3111	137	Produs 61

10 Trigger de tip LMD la nivel de comanda

La sfarsitul fiecarei luni, magazinul trece printr-un proces de inventariere. Deoarece acest proces se intampla regulat, angajatilor le este suficient un interval de 2 ore pentru a efectua inventarierea. Totusi, nu vrem ca

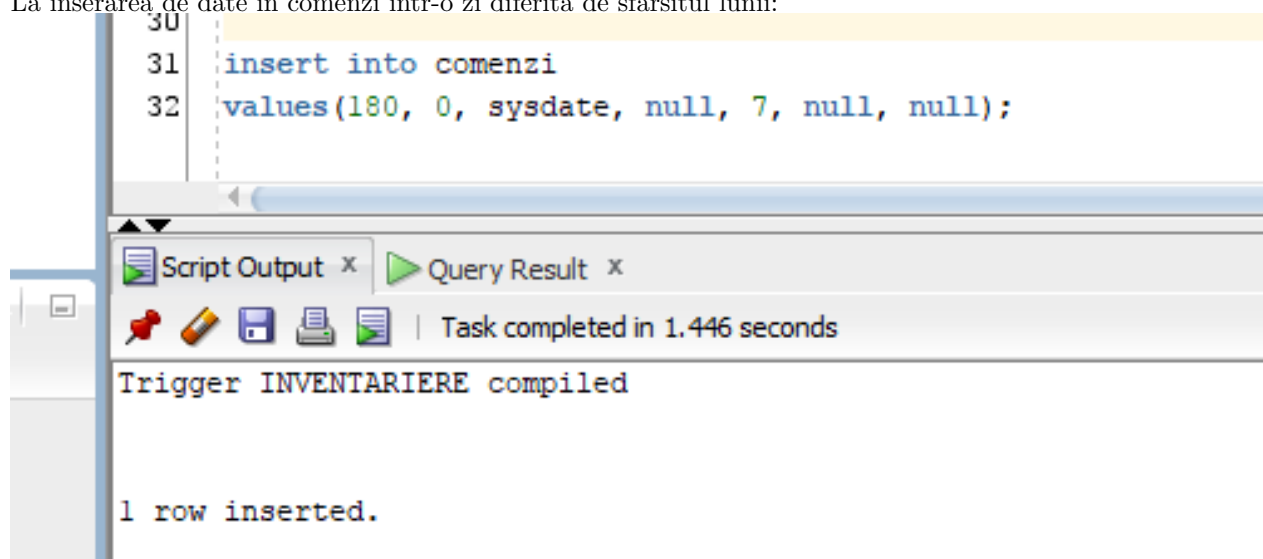
odata inventariat un produs, acesta sa mai primeasca o comanda si astfel sa avem date incorecte. Asadar, vom bloca introducerea, alterarea sau stergerea unei comenzi in intervalul orar 02:00 - 04:00 (atunci se fac cele mai putine comenzi). Scrieti un trigger care sa blocheze aceste modificari.

```

1 create or replace trigger inventariere
2 before insert or update or delete on comenzi
3 begin
4     if to_char(sysdate, 'dd/mm') = '31/01' or
5        to_char(sysdate, 'dd/mm') = '28/02' or
6        to_char(sysdate, 'dd/mm') = '31/03' or
7        to_char(sysdate, 'dd/mm') = '30/04' or
8        to_char(sysdate, 'dd/mm') = '31/05' or
9        to_char(sysdate, 'dd/mm') = '30/06' or
10       to_char(sysdate, 'dd/mm') = '31/07' or
11       to_char(sysdate, 'dd/mm') = '31/08' or
12       to_char(sysdate, 'dd/mm') = '30/09' or
13       to_char(sysdate, 'dd/mm') = '31/10' or
14       to_char(sysdate, 'dd/mm') = '30/11' or
15       to_char(sysdate, 'dd/mm') = '31/12'
16       and (to_char(sysdate, 'hh24') between 2 and 4)
17     then
18         raise_application_error(-20007, 'Magazinul se afla in proces de inventariere, va
19         rugam plasati comanda dupa ora 04:00. ');
20     end if;
21 end;
```

trigger inventariere.txt

La inserarea de date in comenzi intr-o zi diferita de sfarsitul lunii:



11 Trigger de tip LMD la nivel de linie

Definiti un trigger la nivel de linie care la adaugarea unui discount pe o categorie, verifica sa nu mai existe alt discount pe acea categorie cu care s-ar suprapune.

```

1 create or replace trigger exercitiul_11
2 before
3 insert or update on discounturi_categorii for each row
4 declare
5     nr number;
6     data_start date;
7     data_finish date;
8 begin
9     select data_inceput, data_final
```

```

10      into data_start, data_finish
11      from discounturi
12      where discount_id=:new.discount_id;
13
14      select count(*)
15      into nr
16      from discounturi_categorii a, discounturi b
17      where a.discount_id=b.discount_id
18      and a.categorie_id=:new.categorie_id
19      and data_start<=b.data_final
20      and data_finish>=b.data_inceput;
21
22      if nr>0 then
23          raise_application_error(-20008, 'Exista deja un discount pe aceasta categorie in
24              intervalul de timp al discount-ului');
25      end if;
26  end;
27 /

```

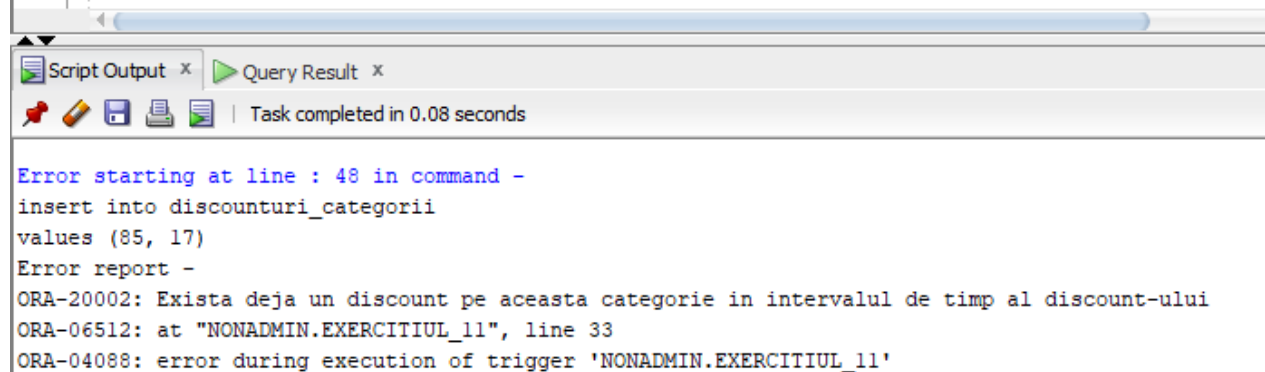
exercitiul 11.txt

In urma rularii pe un input incompatibil:

```

48      insert into discounturi_categorii
49      values (85, 17);
50 /
51

```



Script Output x Query Result x

Task completed in 0.08 seconds

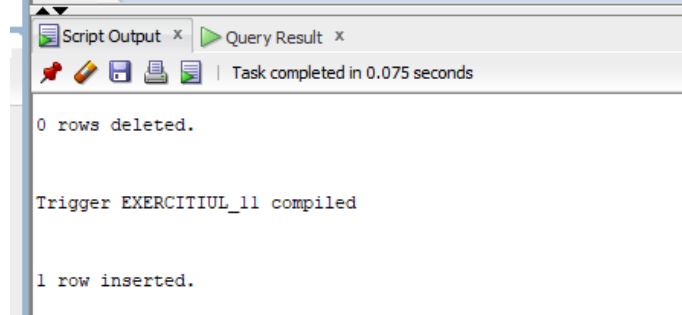
Error starting at line : 48 in command -
insert into discounturi_categorii
values (85, 17)
Error report -
ORA-20002: Exista deja un discount pe aceasta categorie in intervalul de timp al discount-ului
ORA-06512: at "NONADMIN.EXERCITIUL_11", line 33
ORA-04088: error during execution of trigger 'NONADMIN.EXERCITIUL_11'

Iar pe un input bun:

```

39      insert into discounturi_categorii
40      values (2, 17);
41 /

```



Script Output x Query Result x

Task completed in 0.075 seconds

0 rows deleted.

Trigger EXERCITIUL_11 compiled

1 row inserted.

12 Trigger de tip LDD

Creati un trigger care permite schimabri la nivelul bazei de date (adaugare, stergere sau alterare a tabelor) doar in timpul programului de lucru..

```

1 create or replace trigger exercitiul_12
2 before create or drop or alter on database
3 begin
4 if TO_CHAR(SYSDATE, 'D') = 1 or TO_CHAR(SYSDATE, 'D') = 7 or (TO_CHAR(SYSDATE, 'HH24') NOT
   BETWEEN 8 AND 20)
5 then
6 raise_application_error(-20009, 'Operatiile asupra bazei de date sunt permise doar in
   programul de lucru!');
7 end if;
8 end;
9 /

```

exercitiul 12.txt

La adaugarea unui tabel intr- zi de sambata am obtinut:

The screenshot shows the SQL Developer interface. The top part displays the SQL script being executed, which is the same trigger creation script as in the previous block. Below the script, the 'Script Output' tab is active, showing the execution status: 'Task completed in 0.128 seconds'. Below that, the 'Error report' tab is active, displaying the following error messages:

```

ORA-04088: error during execution of trigger 'NONADMIN.EXERCITIUL_12'
ORA-00604: error occurred at recursive SQL level 1
ORA-20001: Operatiile asupra bazei de date sunt permise doar in programul de lucru!
ORA-06512: at line 4
04088. 00000 - "error during execution of trigger '%s.%s'"
*Cause:      A runtime error occurred during execution of a trigger.
*Action:     Check the triggers which were involved in the operation.

```

13 Pachet cu obiecte

Aici doar am adagat toate obiectele create la punctele anterioare (tipuri de date, functii si proceduri) intr-un pachet.

14 Pachet complex

14.1 Recomandari bazate pe produs

Vrem sa imbunatatim recomandările site-ului nostru. Astfel, pentru fiecare produs X vrem sa identificam care sunt produsele Y (id-ul si denumirea, pretul la momentul apelarii functiei) care se vand intotdeauna cu produsul X. (deci X poate fi cumparat si fara Y, dar Y a fost cumparat impreuna cu X in proportie de minim 75%). Ordonati rezultatele pentru fiecare produs X in functie de procentajul de cumparare. Contruiti o functie care sa faca acest lucru, returnand un tabel imbricat de tabele imbricate cu datele cerute pentru fiecare produs.

```

1 create or replace function functie_6 return nested_table_of_nested_table
2 is
3 comenzile_in_care_apare_X nested_table_of_number;
4 comenzile_in_care_apare_Y nested_table_of_number;
5 intersectie nested_table_of_number;
6 recomandari_pentru_X nested_table:= nested_table();
7 tabel_final nested_table_of_nested_table:=nested_table_of_nested_table();
8 nr number;
9 valoare number;
10 begin

```

```

11 for X in (select produs_id
12           from produse
13           order by produs_id) loop
14   select comanda_id
15   bulk collect into comenzile_in_care_apare_X
16   from produse_comenzi
17   where produs_id=X.produs_id;
18
19   for Y in (select produs_id, denumire_produs, pret_unitar
20             from produse
21             where produs_id>X.produs_id
22             order by produs_id) loop
23     select comanda_id
24     bulk collect into comenzile_in_care_apare_Y
25     from produse_comenzi
26     where produs_id=Y.produs_id;
27
28     intersectie:=comenzile_in_care_apare_x multiset intersect
29                 comenzile_in_care_apare_y;
30
31     if intersectie.count>(comenzile_in_care_apare_Y.count)*3/4 then
32
33         for i in ( select a.valoare
34                   from discounturi a, discounturi_produse b
35                   where a.discount_id=b.discount_id
36                   and a.data_inceput<=sysdate
37                   and a.data_final>=sysdate
38                   and b.produs_id=Y.produs_id
39                   order by a.data_inceput) loop
40             Y.pret_unitar:=Y.pret_unitar * (100-i.valoare)/100;
41         end loop;
42
43         select count(*)
44         into nr
45         from discounturi a, discounturi_categorii b, produse_categorii c
46         where a.discount_id= b.discount_id
47         and b.categorie_id=c.categorie_id
48         and produs_id=Y.produs_id
49         and a.data_inceput<=sysdate
50         and a.data_final>=sysdate
51         order by a.data_inceput;
52
53         if nr>0 then
54             select a.valoare
55             into valoare
56             from discounturi a, discounturi_categorii b, produse_categorii c
57             where a.discount_id= b.discount_id
58             and b.categorie_id=c.categorie_id
59             and produs_id=Y.produs_id
60             and a.data_inceput<=sysdate
61             and a.data_final>=sysdate
62             order by a.data_inceput;
63
64             Y.pret_unitar:=Y.pret_unitar * (100-valoare)/100;
65         end if;
66
67         recomandari_pentru_X.extend;
68         recomandari_pentru_X(recomandari_pentru_X.count):=info_produs(Y.produs_id, Y
69                               .denumire_produs, Y.pret_unitar);
70     end if;
71 end loop;
72
73 tabel_final.extend;
74 tabel_final(tabel_final.count):=info_produs_produse(X.produs_id,
75               recomandari_pentru_X);
76 recomandari_pentru_x.delete;
77 end loop;

```

```

76     return tabel_final;
77 end;

```

exercitiul 6.txt

In urma rularii acestui exercitiu am obtinut:

```

58 declare
59   x nested_table_of_nested_table;
60   begin
61     x:=functie_6();
62     for i in x.first..x.last loop
63       if x(i).lista_produce.count>=1 then
64         dbms_output.put_line('-----');
65         dbms_output.put_line('Pentru produsul '|| x(i).id_produc || ' recomandările sunt: ');
66         dbms_output.put_line('-----');
67         for j in nvl(x(i).lista_produce.first, 0)..nvl(x(i).lista_produce.last, -1) loop
68           dbms_output.put_line(x(i).lista_produce(j).denumire_produc || ' cu id '|| x(i).lista_produce(j).id_produc || ' si pretul ' || x(i).lista_produce(j).pret);
69         end loop;
70       end if;
71     end loop;
72   end;
73 /

```

Script Output x

Task completed in 1.311 seconds

```

-----
Pentru produsul 62 recomandările sunt:
-----
Produs 63 cu id 63 si pretul 3235
Produs 74 cu id 74 si pretul 4236
Produs 89 cu id 89 si pretul 140
Produs 111 cu id 111 si pretul 86677
Produs 114 cu id 114 si pretul 69123
Produs 119 cu id 119 si pretul 66563
Produs 133 cu id 133 si pretul 84754
Produs 155 cu id 155 si pretul 51148
-----
Pentru produsul 63 recomandările sunt:
-----
Produs 70 cu id 70 si pretul 4564
Produs 110 cu id 110 si pretul 67267
Produs 127 cu id 127 si pretul 85945
-----
Pentru produsul 64 recomandările sunt:
-----
Produs 65 cu id 65 si pretul 128
Produs 95 cu id 95 si pretul 1513
Produs 155 cu id 155 si pretul 51148

```

14.2 Trigger la nivel de baza de date pentru retinere mentori

Ca sa pot sa retin mentorii de la o sesiune la alta definesc un nou tabel RETINERE_MENTORI(pozitie number, mentor number) In care voi retine id-ul fiecarui mentori si pozitia din colectia 'bucati'.

Am nevoie de 2 trigger la nivel de baza de date. Inainte de a inchide sesiunea trebuie sa retina datele tabel, iar dupa inceperea sesiunii refaca colectia 'mentori'.

```

1 create or replace trigger on_logon
2 after logon on schema
3 declare
4 begin
5   --retin valoarea contorului
6   select mentor
7   into proiect.contor_mentori
8   from retinere_mentori
9   where pozitie=0;
10  --si lista de mentori
11  select mentor
12  bulk collect into proiect.mentori
13  from retinere_mentori
14  where pozitie>0;
15 end;

```

trigger on logon.txt

```

1 create or replace trigger on_logoff
2 before logoff on database
3 declare
4   nr_mentori number;
5   nr_linii number;
6 begin

```



```

7      select count(*)
8      into nr_linii
9      from retinere_mentori;
10
11      nr_mentori:=proiect.mentori.count;
12
13      if nr_linii-1>=0 then
14          update retinere_mentori
15          set mentor=proiect.contor_mentori
16          where pozitie=0;
17      else
18          insert into retinere_mentori
19          values (0, nr_mentori);
20      end if;
21      dbms_output.put_line(proiect.mentori.first || ' ' || proiect.mentori.last);
22      for i in nvl(proiect.mentori.first,0)..nvl(proiect.mentori.last,-1) loop
23          if nr_linii-1>=i then
24              update retinere_mentori
25              set mentor=proiect.mentori(i)
26              where pozitie=i;
27          else
28              insert into retinere_mentori
29              values (i, proiect.mentori(i));
30          end if;
31      end loop;
32      delete from retinere_mentori
33      where pozitie>nr_mentori;
34
35      dbms_output.put_line('s-a rulat on_logoff');
36 end;

```

trigger on logoff.txt

14.3 Trigger care sterge angajatul din lista de mentori cand acesta este concediat

```

1 create or replace trigger update_mentori_cand_se_sterge_un_mentor
2 after delete on angajati
3 for each row
4 begin
5     for i in nvl(proiect.mentori.first, 0)..nvl(proiect.mentori.last, -1) loop
6         if proiect.mentori(i)=old.angajat_id then
7             for j in i..proiect.mentori.last-1 loop
8                 proiect.mentori(j):=proiect.mentori(j+1);
9             end loop;
10            proiect.mentori.delete(proiect.mentori.last);
11        end if;
12    end loop;
13
14 end;
15 /

```

trigee stergere mentor.txt

Inainte de stergerea mentorului lista de mentori era:

```

31 begin
32 proiect.afisare_mentori();
33 end;
34 /

```

Script Output x

Task completed in 0.032 seconds

Rollback complete.

Rollback complete.

110 116 104 150 124 157 156

PL/SQL procedure successfully completed.

Stererea mentorului:

```

41 delete from angajati
42 where angajat_id=124;

```

Script Output x

Task completed in 0.852 seconds

Trigger UPDATE_MENTORI_CAND_SE_STERGE_UN_MENTOR compiled

1 row deleted.

Dupa stergerea mentorului:

```

29
30 begin
31 proiect.afisare_mentori();
32 end;
33 /

```

Script Output x

Task completed in 0.72 seconds

110 116 104 150 157 156

PL/SQL procedure successfully completed.

14.4 Procedura care sa initializeze variabila bucati

Ca sa putem folosi variabila 'bucati' din pachetul 'proiect' ne trebuie o procedura care sa o initializeze:

```

1 create or replace procedure initializare_bucati
2 as
3 comenzi_de_la_acelasi_user_pe_acelasi_produc number;
4 begin
5     for i in (select * from produse) loop
6         proiect.bucati(i.produc_id).fst:=0;
7         proiect.bucati(i.produc_id).snd:=0;
8     end loop;
9
10    for i in (select * from produse_comenzi) loop
11        select decode((select count(*)
12                        from produse_comenzi c, comenzi d
13                        where c.comanda_id=d.comanda_id
14                        and c.produc_id=i.produc_id
15                        and d.user_id=b.user_id), 0, 1, 0)
16        into comenzi_de_la_acelasi_user_pe_acelasi_produc
17        from produse_comenzi a, comenzi b
18        where a.comanda_id=b.comanda_id
19        and a.comanda_id=i.comanda_id
20        and a.produc_id=i.produc_id;

```

```

21      dbms_output.put_line('Numar useri inainte: ' || proiect.bucati(i.produs_id).snd);
22      proiect.bucati(i.produs_id).snd:=proiect.bucati(i.produs_id).snd+
23      comenzi_de_la_acelasi_user_pe_acelasi_produs+1;
24      proiect.bucati(i.produs_id).fst:=proiect.bucati(i.produs_id).fst+i.cantitate;
25      dbms_output.put_line('Dupa: ' || proiect.bucati(i.produs_id).snd);
26      dbms_output.put_line(comenzi_de_la_acelasi_user_pe_acelasi_produs);
27  end loop;
28
29 end;

```

initializare_bucati.txt

14.5 Compound trigger pentru concedierea angajatilor

Inainte de a concedia un angajat, vrem sa analizam daca acesta este necesar magazinului, sau nu. Pentru asta, trebuie sa analizam numarul achizitiilor in magazin din utimele 6 luni. Un sofer poate efectua 2 transporturi pe zi, iar un angajat poate indruma 4 clienti pe zi. Cu aceste date, media devine 3 angajati/actiune(achizitie sau transport). Aflati care este numarul minim de soferi necesari la momentul curent si permiteti concedierea doar daca raman suficienti angajati in firma.

- un sofer poate efectua 2 transporturi pe zi
- un indrumator poate indruma in medie 4 clienti pe zi
- un casier poate scana 10 de comenzi pe zi
- avem nevoie de un si maxim 3 inventory control specialist
- avem nevoie de un sales asociative pentru fiecare 100 de comenzi (care nu au fost anulate)
- avem nevoie de un si maxim 3 visual merchandiser-i in magazin
- avem nevoie de 2 retail security officers pe fiecare zona
- avem nevoie de un customer service representative pentru fiecare 40 de indrumatori

Avand in vedere ca anagajtii sunt stersi de obicei dintr-o decizie umana si deci acesta nu este un proces automatizat, am dori sa descurajan comenzile care sterg din baza de date mai multi angajati deodata, mai ales daca acele stergeri nu sunt in avantajul magazinului (ex: stergem toti soferii). De aceea, vrem ca daca o comanda de concediere incalca limitele impuse pentru stabilitatea magazinui, aceasta comanda sa fie impiedicata din a rula.

O prima solutie la aceasta problema ar fi cu 3 triggeri:

```

1 create or replace trigger exercitiul_10_1
2 after delete on angajati
3 for each row
4 declare
5 begin
6     proiect.nr_intrari:=proiect.nr_intrari+1;
7     proiect.intrari(proiect.nr_intrari):=proiect.info_angajat(:old.angajat_id, :old.job_id);
8
9     dbms_output.put_line('trigger 1 ' || proiect.nr_intrari || ' ' || proiect.intrari.count)
10    ;
11 end exercitiul_10_1;
12 /
13 create or replace trigger exercitiul_10_2
14 after delete on angajati
15 declare
16 nr_comenzi_neanulate number;
17 nr_sales_associative number;
18 max_nr_achizitii_pe_zi number;

```

```

19 nr_casieri number;
20 nr_customer_service_representative number;
21 nr_visual_merchandiser number;
22 nr_inventory_control_specialist number;
23 nr_retail_security_officer number;
24 max_nr_transporturi_pe_zi number;
25 nr_indrumatori number;
26 nr_soferi number;
27 nr_zone number;
28 begin
29 —pentru id:1
30 —toate cererile sunt cu max si count deci nu am erori
31 select count(comanda_id)
32 into nr_comenzi_neanulate
33 from comenzi
34 where status_comanda!=3;
35
36 select count(*)
37 into nr_sales_associative
38 from angajati
39 where job_id=1;
40
41 —pentru id:2 si 8
42 select max(nr)
43 into max_nr_achizitii_pe_zi
44 from (select count(*) nr, data_achizitiei
45 from achizitii_in_magazin
46 where data_achizitiei>=add_months(sysdate, -6)
47 group by data_achizitiei);
48
49 select count(*)
50 into nr_casieri
51 from angajati
52 where job_id=2;
53
54 —pentru id:3
55 select count(*)
56 into nr_customer_service_representative
57 from angajati
58 where job_id=3;
59
60 —pentru id:4
61 select count(*)
62 into nr_visual_merchandiser
63 from angajati
64 where job_id=4;
65
66 —pentru id:5
67 select count(*)
68 into nr_inventory_control_specialist
69 from angajati
70 where job_id=5;
71
72 select count(*)
73 into nr_zone
74 from zone;
75
76 —pentru id:6
77 select count(*)
78 into nr_retail_security_officer
79 from angajati
80 where job_id=6;
81
82 —pentru id:7
83 select max(nvl(count(*),0))
84 into max_nr_transporturi_pe_zi
85 from transporturi a, comenzi b
86 where a.transport_id=b.transport_id

```

```

87     group by data_livrarii;
88
89 —pentru id:8
90     select count(*)
91     into nr_indrumatori
92     from angajati
93     where job_id=8;
94     dbms_output.put_line('trigger 2' || ' ' || proiect.nr_intrari || ' ' ||
95         nr_inventory_control_specialist);
96
97 for i in 1..proiect.nr_intrari loop
98     dbms_output.put_line('B ' || proiect.intrari(i).job_id || ' ' || proiect.intrari(i).
99         angajat_id);
100     if proiect.intrari(i).job_id=1 then
101         if nr_comenzi_neanulate>100*(nr_sales_associative) then
102             raise_application_error(-20004, 'Nu vor ramane destui Sales Associatives!');
103         else
104             nr_sales_associative:=nr_sales_associative-1;
105         end if;
106     elsif proiect.intrari(i).job_id=2 then
107         if max_nr_achizitii_pe_zi>10*(nr_casieri) then
108             raise_application_error(-20005, 'Nu vor ramane destui casieri!');
109         else
110             nr_casieri:=nr_casieri-1;
111         end if;
112     elsif proiect.intrari(i).job_id=3 then
113         if nr_customer_service_representative*10<(nr_indrumatori) then
114             raise_application_error(-20006, 'Nu vor ramane destui Customer Service
115                 Representatives!');
116         else
117             nr_indrumatori:=nr_indrumatori-1;
118         end if;
119     elsif proiect.intrari(i).job_id=4 then
120         if (nr_visual_merchandiser-1)<1 then
121             raise_application_error(-20007, 'Nu vor ramane destui Visual Merchandisers!');
122         else
123             nr_visual_merchandiser:=nr_visual_merchandiser-1;
124         end if;
125     elsif proiect.intrari(i).job_id=5 then
126         if (nr_inventory_control_specialist)<1 then
127             dbms_output.put_line('Nr inventory control specialists ' ||
128                 nr_inventory_control_specialist);
129             raise_application_error(-20008, 'Nu vor ramane destui Inventory Control
130                 Specialists!');
131         else
132             dbms_output.put_line('Inainte: ' || nr_inventory_control_specialist);
133             nr_inventory_control_specialist:=nr_inventory_control_specialist-1;
134             dbms_output.put_line('Dupa: ' || nr_inventory_control_specialist);
135         end if;
136     elsif proiect.intrari(i).job_id=6 then
137         if (nr_retail_security_officer)*2<nr_zone then
138             raise_application_error(-20009, 'Nu vor ramane destui Retail Security
139                 Officers!');
140         else
141             nr_retail_security_officer:=nr_retail_security_officer-1;
142         end if;
143     elsif proiect.intrari(i).job_id=7 then
144         if max_nr_transporturi_pe_zi>(nr_soferi)*2 then
145             raise_application_error(-20010, 'Nu vor ramane destui Soferi!');
146         else
147             nr_soferi:=nr_soferi-1;
148         end if;
149     elsif proiect.intrari(i).job_id=8 then
150         if max_nr_achizitii_pe_zi>(nr_indrumatori)*4 then
151             raise_application_error(-20011, 'Nu vor ramane destui Indrumatori!');
152         else
153             nr_indrumatori:=nr_indrumatori-1;

```

```

148         end if;
149     end if;
150 end loop;
151
152 end exercitiul_10_2;
153 /
154
155 create or replace trigger exercitiul_10_3
156 before delete on angajati
157 begin
158 proiect.intrari.delete;
159 proiect.nr_intrari:=0;
160
161 dbms_output.put_line('trigger 3 ' || proiect.nr_intrari || ' ' || proiect.intrari.count);
162 end;
163 /

```

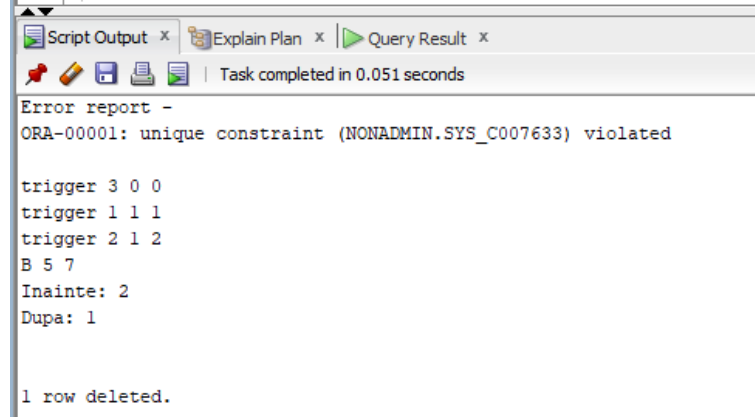
exercitiul 10.txt

Sunt 3 Inventory Control Specialists in firma. Incerc sa il sterg pe unul dintre ei:

```

188 delete from angajati
189 where angajat_id=7;
190 rollback;
191

```



Script Output x Explain Plan x Query Result x

Task completed in 0.051 seconds

Error report -

ORA-00001: unique constraint (NONADMIN.SYS_C007633) violated

trigger 3 0 0
trigger 1 1 1
trigger 2 1 2
B 5 7
Inainte: 2
Dupa: 1

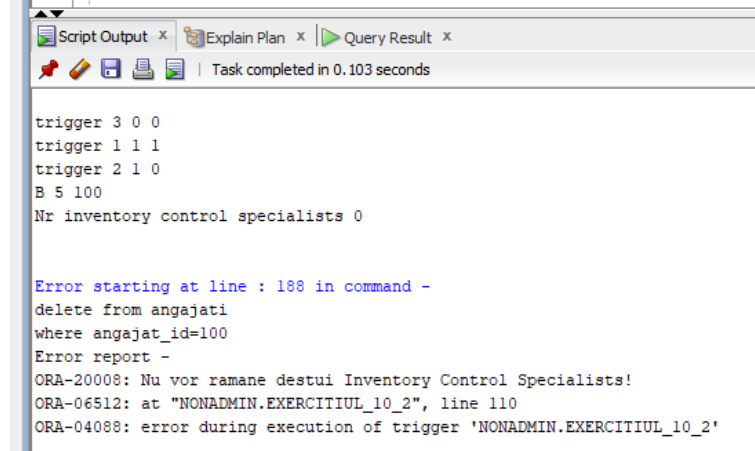
1 row deleted.

Exista doar un Inventory Control Specialist in firma. Incerc sa il sterg:

```

188 delete from angajati
189 where angajat_id=100;
190 rollback;
191

```



Script Output x Explain Plan x Query Result x

Task completed in 0.103 seconds

trigger 3 0 0
trigger 1 1 1
trigger 2 1 0
B 5 100
Nr inventory control specialists 0

Error starting at line : 188 in command -
delete from angajati
where angajat_id=100

Error report -

ORA-20008: Nu vor ramane destui Inventory Control Specialists!
ORA-06512: at "NONADMIN.EXERCITIUL_10_2", line 110
ORA-04088: error during execution of trigger 'NONADMIN.EXERCITIUL_10_2'

Putem totusi sa definim mult mai simplu un compound trigger:

```

1  —am dezactivat trigger-ul de la exercitiul 10
2
3  create or replace trigger exercitiul_10_compound
4  for delete on angajati
5  compound trigger
6
7      type info_angajat is record (angajat_id number, job_id number);
8      type indexed_by_table_of_info_angajat is table of info_angajat index by pls_integer;
9      nr_intrari number:=0;
10     intrari indexed_by_table_of_info_angajat;
11     nr_comenzi_neanulate number;
12     nr_sales_associative number;
13     max_nr_achizitii_pe_zi number;
14     nr_casieri number;
15     nr_customer_service_representative number;
16     nr_visual_merchandiser number;
17     nr_inventory_control_specialist number;
18     nr_retail_security_officer number;
19     max_nr_transporturi_pe_zi number;
20     nr_indrumatori number;
21     nr_soferi number;
22     nr_zone number;
23
24     before statement is
25     begin
26         null;
27     end before statement;
28
29     before each row is
30     begin
31         null;
32     end before each row;
33
34     after each row is
35     begin
36         nr_intrari:=nr_intrari+1;
37         intrari(nr_intrari):=info_angajat(:old.angajat_id, :old.job_id);
38     end after each row;
39
40     after statement is
41     begin
42         —pentru id:1
43         —toate cererile sunt cu max si count deci nu am erori
44         select count(comanda_id)
45         into nr_comenzi_neanulate
46         from comenzi
47         where status.comanda!=3;
48
49         select count(*)
50         into nr_sales_associative
51         from angajati
52         where job_id=1;
53
54         —pentru id:2 si 8
55         select max(nr)
56         into max_nr_achizitii_pe_zi
57         from (select count(*) nr, data_achizitiei
58              from achizitii_in_magazin
59              where data_achizitiei>=add_months(sysdate, -6)
60              group by data_achizitiei);
61
62         select count(*)
63         into nr_casieri
64         from angajati
65         where job_id=2;
66
67         —pentru id:3
68         select count(*)

```

```

69         into nr_customer_service_representative
70         from angajati
71         where job_id=3;
72
73     —pentru id:4
74     select count(*)
75     into nr_visual_merchandiser
76     from angajati
77     where job_id=4;
78
79     —pentru id:5
80     select count(*)
81     into nr_inventory_control_specialist
82     from angajati
83     where job_id=5;
84
85     select count(*)
86     into nr_zone
87     from zone;
88
89     —pentru id:6
90     select count(*)
91     into nr_retail_security_officer
92     from angajati
93     where job_id=6;
94
95     —pentru id:7
96     select max(nvl(count(*),0))
97     into max_nr_transporturi_pe_zi
98     from transporturi a, comenzi b
99     where a.transport_id=b.transport_id
100    group by data_livrarii;
101
102     —pentru id:8
103     select count(*)
104     into nr_indrumatori
105     from angajati
106     where job_id=8;
107
108     for i in 1..nr_intrari loop
109         if intrari(i).job_id=1 then
110             if nr_comenzi_neanulate>100*(nr_sales_associative) then
111                 raise_application_error(-20004, 'Nu vor ramane destui Sales
112                     Associatives!');
113             else
114                 nr_sales_associative:=nr_sales_associative-1;
115             end if;
116         elsif intrari(i).job_id=2 then
117             if max_nr_achizitii_pe_zi>10*(nr_casieri) then
118                 raise_application_error(-20005, 'Nu vor ramane destui casieri!');
119             else
120                 nr_casieri:=nr_casieri-1;
121             end if;
122         elsif intrari(i).job_id=3 then
123             if nr_customer_service_representative*10<(nr_indrumatori) then
124                 raise_application_error(-20006, 'Nu vor ramane destui Customer
125                     Service Representatives!');
126             else
127                 nr_indrumatori:=nr_indrumatori-1;
128             end if;
129         elsif intrari(i).job_id=4 then
130             if (nr_visual_merchandiser-1)<1 then
131                 raise_application_error(-20007, 'Nu vor ramane destui Visual
132                     Merchandisers!');
133             else
134                 nr_visual_merchandiser:=nr_visual_merchandiser-1;
135             end if;
136         elsif intrari(i).job_id=5 then

```



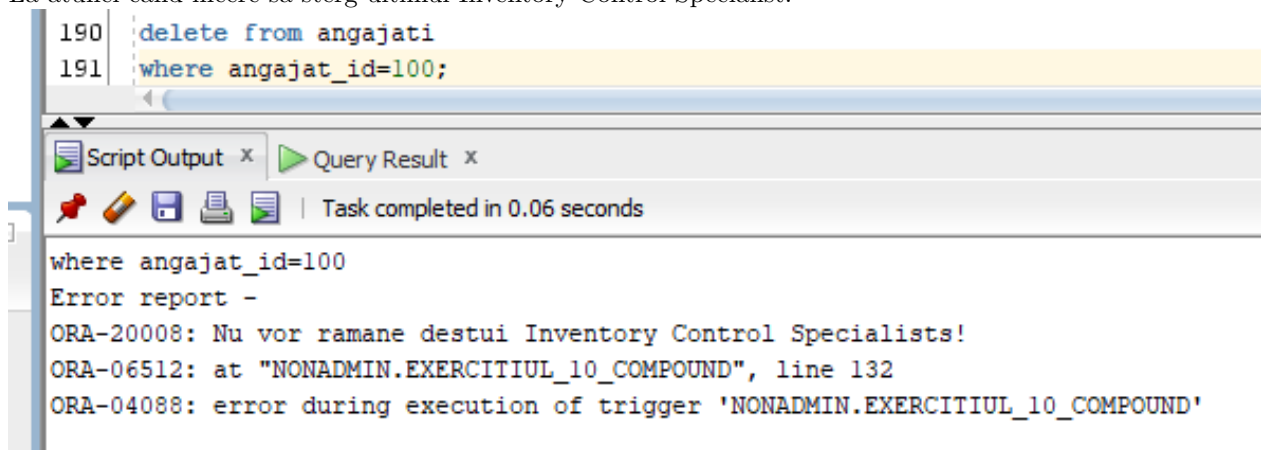
```

134         if (nr_inventory_control_specialist)<1 then
135             dbms_output.put_line('Nr inventory control specialists ' ||
                                   nr_inventory_control_specialist);
136             raise_application_error(-20008, 'Nu vor ramane destui Inventory
                                   Control Specialists!');
137         else
138             dbms_output.put_line('Inainte: ' || nr_inventory_control_specialist);
139             nr_inventory_control_specialist:=nr_inventory_control_specialist-1;
140             dbms_output.put_line('Dupa: ' || nr_inventory_control_specialist);
141         end if;
142     elsif intrari(i).job_id=6 then
143         if (nr_retail_security_officer)*2<nr_zone then
144             raise_application_error(-20009, 'Nu vor ramane destui Retail
                                   Security Officers!');
145         else
146             nr_retail_security_officer:=nr_retail_security_officer-1;
147         end if;
148     elsif intrari(i).job_id=7 then
149         if max_nr_transporturi_pe_zi>(nr_soferi)*2 then
150             raise_application_error(-20010, 'Nu vor ramane destui Soferi!');
151         else
152             nr_soferi:=nr_soferi-1;
153         end if;
154     elsif intrari(i).job_id=8 then
155         if max_nr_achizitii_pe_zi>(nr_indrumatori)*4 then
156             raise_application_error(-20011, 'Nu vor ramane destui Indrumatori!')
157             ;
158         else
159             nr_indrumatori:=nr_indrumatori-1;
160         end if;
161     end if;
162 end loop;
163 end after statement;
164 end exercitiul_10_compound;
165 /

```

trigger compound.txt

La atunci cand incerc sa sterg ultimul Inventory Control Specialist:



```

190 delete from angajati
191 where angajat_id=100;

```

Script Output x Query Result x

Task completed in 0.06 seconds

```

where angajat_id=100
Error report -
ORA-20008: Nu vor ramane destui Inventory Control Specialists!
ORA-06512: at "NONADMIN.EXERCITIUL_10_COMPOUND", line 132
ORA-04088: error during execution of trigger 'NONADMIN.EXERCITIUL_10_COMPOUND'

```

Atunc cand incerc sa sterg un angajat oarecare:

```
190 delete from angajati
191 where angajat_id=1;
192
```

Script Output x Query Result x

Task completed in 0.08 sec

```
1 row deleted.
```

References

- [1] Oracle Data Types, [link](#).