



Proiect SGBD Oracle

Gestiunea unei baze de date in cadrul domeniului de comenzi, livrari si produse electronice- eMAG

Realizator:

Petrescu Rares-Mihnea,

Seria D, grupa 1054

Introducere- Prezentarea proiectului:

Baza de date are ca scop gestiunea companiei denumita eMAG. In cadrul bazei de date se vor gasi informatii legate despre clientii care achizitioneaza produse, despre fiecare bun oferit in magazinele online sau fizice, despre locatiile si vanzarile din fiecare magazin, despre serviciile oferite si angajati care aduc cea mai mare contributie financiara intreprinderii. Cu ajutorul acestor informatii, conducerea va putea sa decida care vor fi cele mai bune strategii pentru a atrage profitul, care sunt produsele care merita in continuare vandute, daca serviciile prestate sunt suficiente si care angajati vor merita o marire de salariu.

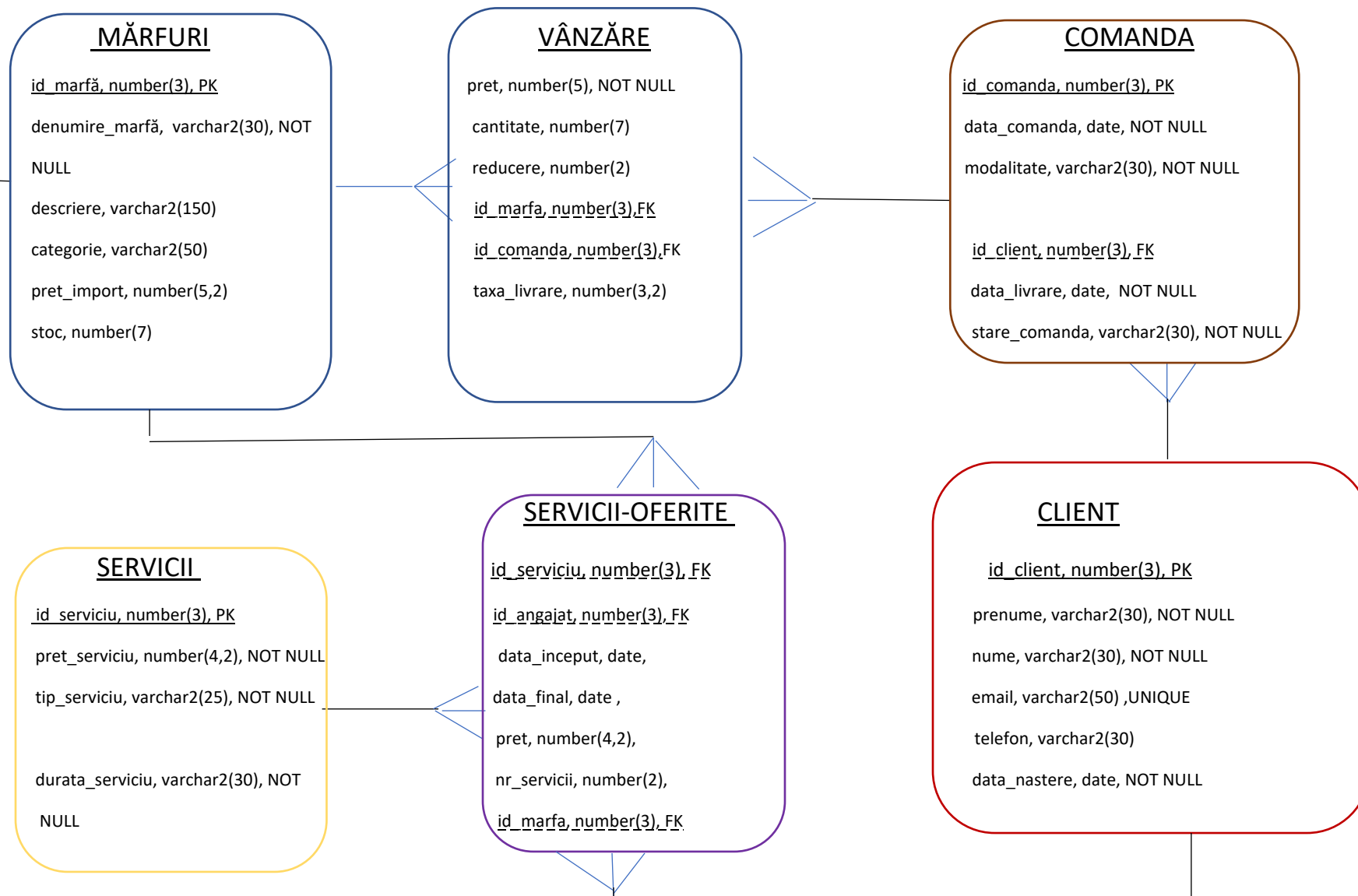
Baza de date va contine urmatoarele 10 tabele:

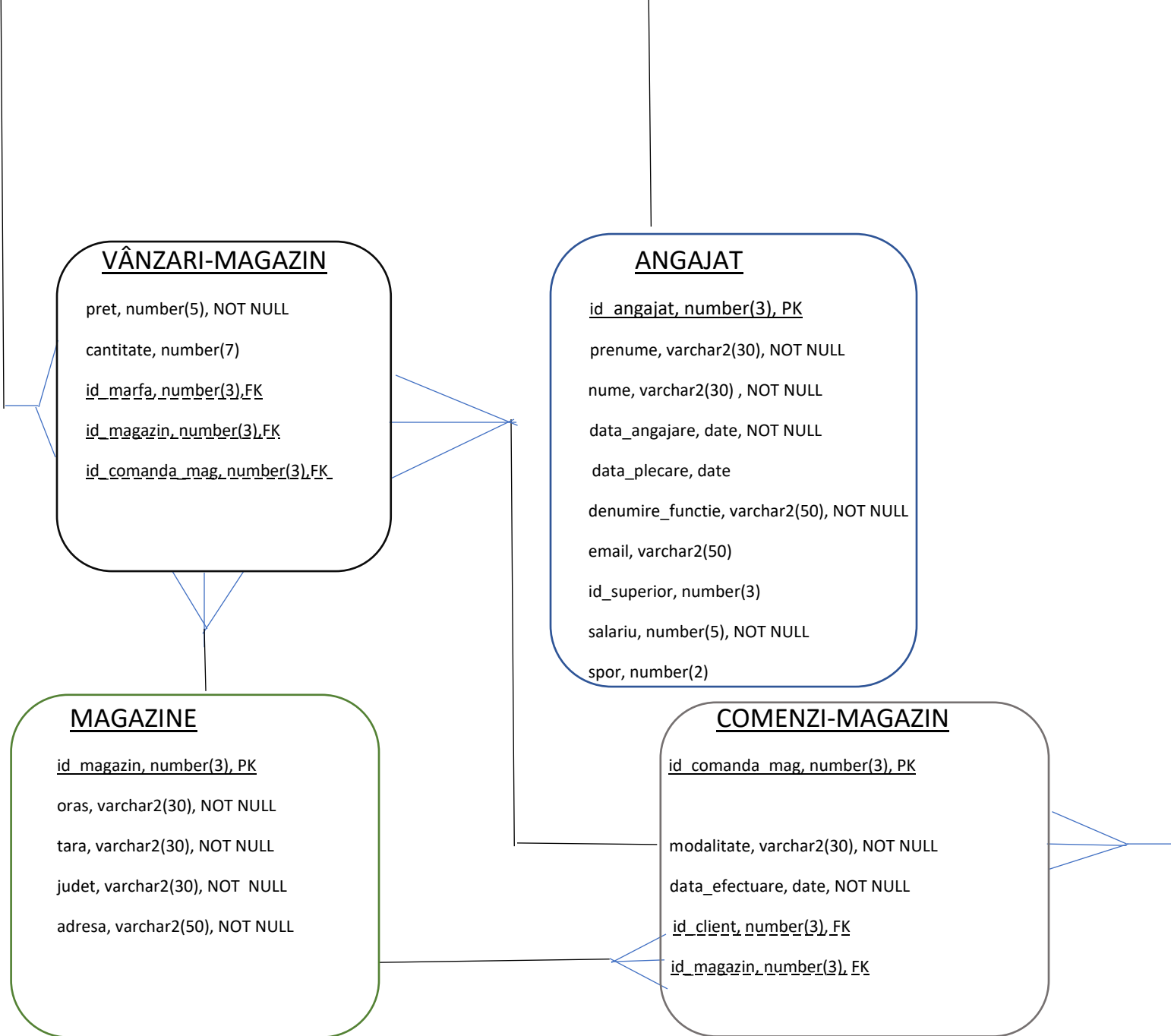
- Marfuri
- Vanzare
- Comanda
- Client
- Comenzi-Magazin
- Magazine
- Vanzari-Magazin
- Angajat
- Servicii
- Servicii-Oferite

Relatiile dintre tabele:

- Un client poate efectua mai multe comenzi atât în online, cât și în magazine. Astfel se va forma o relație de one to many între tabelele Client, Comanda si Comenzi-Magazin.
- Un angajat poate oferi mai multe servicii și un serviciu poate fi oferit de mai mulți angajați, ceea ce aduce o relație many to many. Pentru a putea sparge această relație se va crea un tabel de intersecție denumit servicii-oferite.
- O comandă poate conține mai multe produse/mărfuri, iar un bun poate apărea în mai multe comenzi. Aceasta este o relație many to many și pentru a fi pusă în funcțiune trebuie adăugat un tabel cu numele de Vânzări pentru orice comandă efectuată online. De asemenea, se va aplica aceeași regulă pentru comenzile din magazin, ceea ce va duce la crearea tabelului Vânzări-Magazin.
- Un produs poate fi însoțit de mai multe servicii, iar un serviciu poate fi efectuat pentru mai multe produse. Din nou, se observă formarea unei relații many to many, pentru care va fi necesar să fie utilizat tabelul servicii-oferite.
- Un magazin poate avea mai multe comenzi, iar o comandă aparține unui singur magazin. Se poate observa formarea unei relații de tip one to many între tabelele Magazine și Comenzi-Magazine.
- Mai multe magazine pot avea același produs, iar mai multe produse se pot găsi în diferite magazine. Astfel se va utiliza tabelul de intersecție Vânzări-Magazin pentru a efectua această relație de tip many to many.

Schema concpetuala a bazei de date:





Comenzi SQL pentru interactiunea cu serverul Oracle:

1. Sa se mareasca salariul fiecarui angajat care il are mai mic de 3000 de lei cu 100 de lei.

```
SET SERVEROUTPUT ON
```

```
BEGIN
```

```
UPDATE ANGAJAT
```

```
Set salariu=salariu+100
```

```
where salariu<3000;
```

```
END;
```

```
/
```

| | ID_ANGAJAT | NUME | SALARIU |
|----|------------|--------------|---------|
| 1 | 1 | Petrescu | 4000 |
| 2 | 2 | Alexandrescu | 3000 |
| 3 | 4 | Ciurea | 2500 |
| 4 | 5 | Marinescu | 4000 |
| 5 | 6 | Roman | 3100 |
| 6 | 7 | Ionescu | 2300 |
| 7 | 8 | Manulescu | 3500 |
| 8 | 9 | Superbus | 4000 |
| 9 | 10 | Lupascu | 3800 |
| 10 | 11 | Mulea | 2600 |
| 11 | 12 | Horthy | 3000 |
| 12 | 13 | Ionescu | 2750 |
| 13 | 14 | Marin | 3000 |
| 14 | 15 | Savantus | 3200 |
| 15 | 16 | Popa | 2600 |
| 16 | 3 | Dobroiescu | 4000 |

```
SET SERVEROUTPUT ON
```

```
BEGIN
```

```
UPDATE ANGAJAT
```

```
Set salariu=salariu+100
```

```
where salariu<3000;
```

```
END;
```

```
/
```

| | ID_ANGAJAT | NUME | SALARIU |
|----|------------|--------------|---------|
| 1 | 1 | Petrescu | 4000 |
| 2 | 2 | Alexandrescu | 3000 |
| 3 | 4 | Ciurea | 2600 |
| 4 | 5 | Marinescu | 4000 |
| 5 | 6 | Roman | 3100 |
| 6 | 7 | Ionescu | 2400 |
| 7 | 8 | Manulescu | 3500 |
| 8 | 9 | Superbus | 4000 |
| 9 | 10 | Lupascu | 3800 |
| 10 | 11 | Mulea | 2700 |
| 11 | 12 | Horthy | 3000 |
| 12 | 13 | Ionescu | 2850 |
| 13 | 14 | Marin | 3000 |
| 14 | 15 | Savantus | 3200 |
| 15 | 16 | Popa | 2700 |
| 16 | 3 | Dobroiescu | 4000 |

2.Sa se adauge in tabelul Magazine o noua coloana care spune daca magazinul este sau nu inchis duminica
SET SERVEROUTPUT ON

DECLARE

h_duminica varchar2(300);

BEGIN

h_duminica:='ALTER TABLE MAGAZINE ADD deschis_duminica varchar2(10)';

DBMS_OUTPUT.put_line(h_duminica);

execute immediate h_duminica;

END;

/

--Sa se adauge in tabelul Magazine o noua coloana care spune daca magazinul este sau nu inchis duminica

SET SERVEROUTPUT ON

DECLARE

h_duminica varchar2(300);

BEGIN

h_duminica:='ALTER TABLE MAGAZINE ADD deschis_duminica varchar2(10)';

DBMS_OUTPUT.put_line(h_duminica);

execute immediate h_duminica;

END;

/

ALTER TABLE MAGAZINE ADD deschis_duminica varchar2(10)

PL/SQL procedure successfully completed.

3. Sa se stearga din tabelul SERVICII optiunea denumita ,serviciu de livrare'.

SET SERVEROUTPUT ON

BEGIN

DELETE FROM SERVICII

where tip_serviciu='serviciu de livrare';

END;

/

The screenshot shows a SQL query execution window. The query script area contains the following text:

```
--Sa se stearga din tabelul servicii optiunea cu denumirea 'serviciu de livrare'.  
Select * FROM Servicii;
```

Below the script, the 'Query Result' tab is active, displaying the results of the query. The status bar indicates 'All Rows Fetched: 6 in 0.021 seconds'. The results are shown in a table with the following columns: ID_SERVICIU, PRET_SERVICIU, TIP_SERVICIU, and DURATA_SERVICIU.

| ID_SERVICIU | PRET_SERVICIU | TIP_SERVICIU | DURATA_SERVICIU |
|-------------|---------------|------------------------|-----------------|
| 1 | 3 | 45 extragarantie tip 1 | 1 an |
| 2 | 4 | 85 extragarantie tip 2 | 2 ani |
| 3 | 5 | 75 serviciu de livrare | o zi |
| 4 | 1 | 90 asigurare de baza | 1 an |
| 5 | 2 | 94 asisurare extinsa | 2 ani |
| 6 | 6 | 60 eMAG care | 1 an |

SET SERVEROUTPUT ON

BEGIN

DELETE FROM SERVICII

where tip_serviciu='serviciu de livrare';

END;

/

Script Output x Query Result x

SQL | All Rows Fetched: 5 in 0.02 seconds

| ID_SERVICIU | PRET_SERVICIU | TIP_SERVICIU | DURATA_SERVICIU |
|-------------|---------------|------------------------|-----------------|
| 1 | 3 | 45 extragarantie tip 1 | 1 an |
| 2 | 4 | 85 extragarantie tip 2 | 2 ani |
| 3 | 1 | 90 asigurare de baza | 1 an |
| 4 | 2 | 94 asisurare extinsa | 2 ani |
| 5 | 6 | 60 eMAG care | 1 an |

4. Sa se afiseze denumirea si stocul despre produsul cu id_marfa egal cu 7.

```
SET SERVEROUTPUT ON
```

```
DECLARE
```

```
v_denumire varchar2(50);
```

```
v_stoc number;
```

```
BEGIN
```

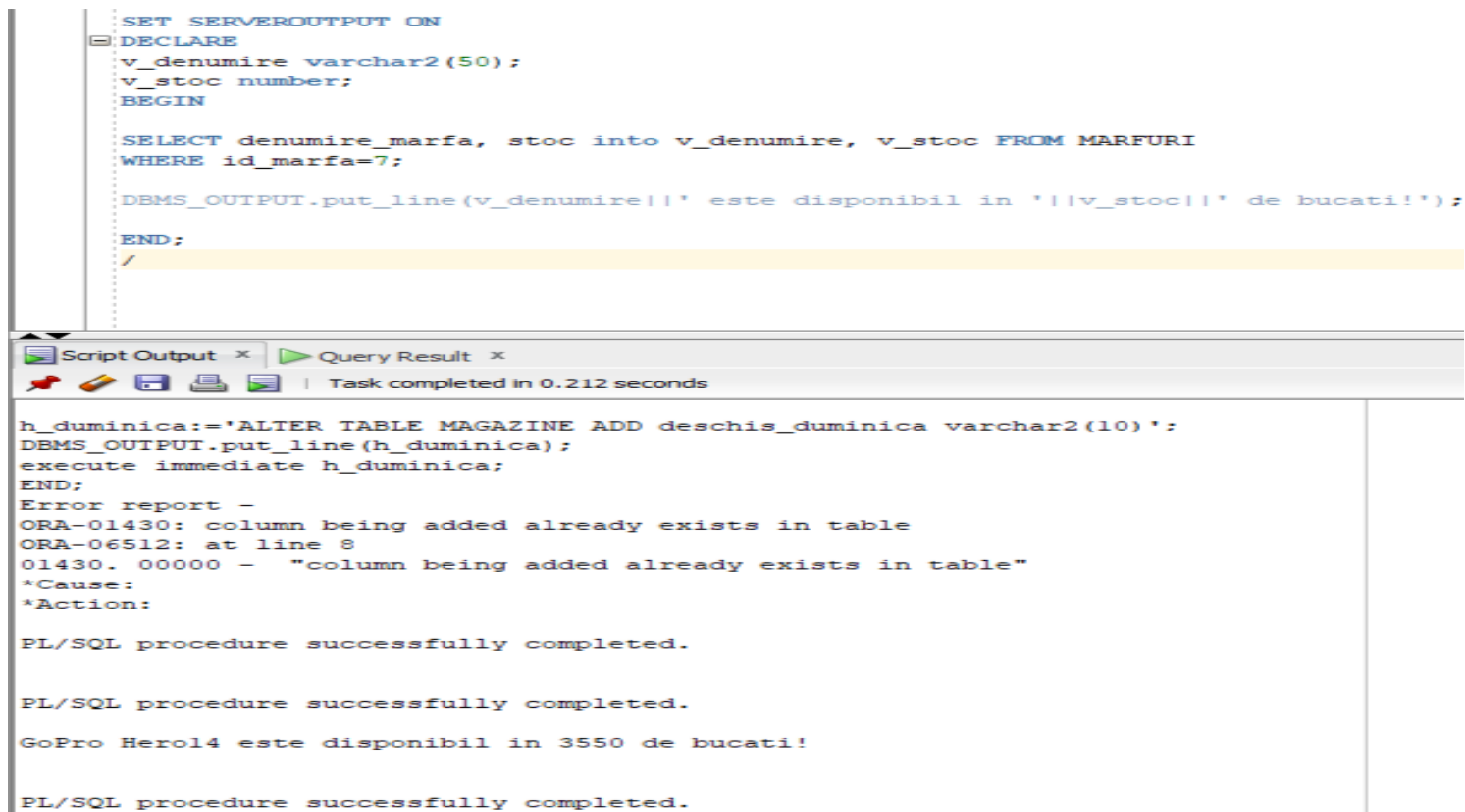
```
SELECT denumire_marfa, stoc into v_denumire, v_stoc FROM MARFURI
```

WHERE id_marfa=7;

DBMS_OUTPUT.put_line(v_denumire||' este disponibil in '||v_stoc||' de bucati!');

END;

/



The screenshot displays the Oracle SQL Developer environment. The top pane shows a PL/SQL script with the following code:

```
SET SERVEROUTPUT ON
DECLARE
v_denumire varchar2(50);
v_stoc number;
BEGIN

SELECT denumire_marfa, stoc into v_denumire, v_stoc FROM MARFURI
WHERE id_marfa=7;

DBMS_OUTPUT.put_line(v_denumire||' este disponibil in '||v_stoc||' de bucati!');

END;
/
```

The bottom pane shows the 'Script Output' window, which contains the following text:

```
h_duminica:='ALTER TABLE MAGAZINE ADD deschis_duminica varchar2(10)';
DBMS_OUTPUT.put_line(h_duminica);
execute immediate h_duminica;
END;
Error report -
ORA-01430: column being added already exists in table
ORA-06512: at line 8
01430. 00000 - "column being added already exists in table"
*Cause:
*Action:

PL/SQL procedure successfully completed.

PL/SQL procedure successfully completed.

GoPro Herol4 este disponibil in 3550 de bucati!

PL/SQL procedure successfully completed.
```

5. In tabela Vanzari_Online reduceti pretul comenzii atunci cand ai o reducere diferita de 0.

```
SET SERVEROUTPUT ON
```

```
BEGIN
```

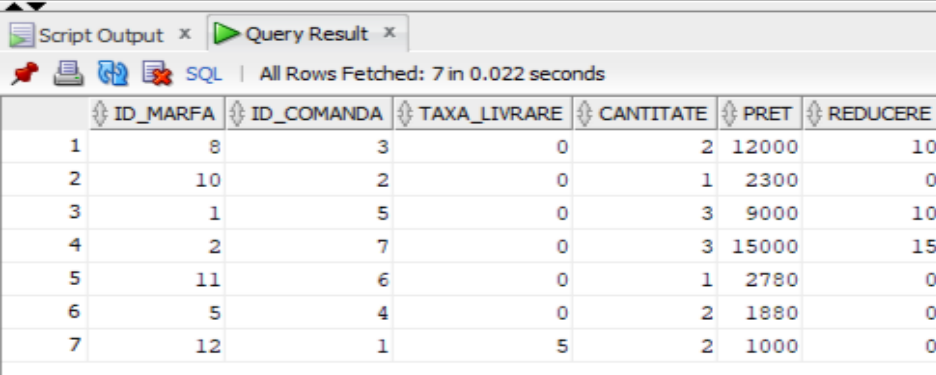
```
UPDATE VANZARI_ONLINE
```

```
SET pret=pret-(reducere/100)*pret
```

```
where reducere!=0;
```

```
END;
```

```
/
```



| | ID_MARFA | ID_COMANDA | TAXA_LIVRARE | CANTITATE | PRET | REDUCERE |
|---|----------|------------|--------------|-----------|-------|----------|
| 1 | 8 | 3 | 0 | 2 | 12000 | 10 |
| 2 | 10 | 2 | 0 | 1 | 2300 | 0 |
| 3 | 1 | 5 | 0 | 3 | 9000 | 10 |
| 4 | 2 | 7 | 0 | 3 | 15000 | 15 |
| 5 | 11 | 6 | 0 | 1 | 2780 | 0 |
| 6 | 5 | 4 | 0 | 2 | 1880 | 0 |
| 7 | 12 | 1 | 5 | 2 | 1000 | 0 |

```

SET SERVEROUTPUT ON

BEGIN

UPDATE VANZARI_ONLINE
SET pret=pret-(reducere/100)*pret
where reducere!=0;

END;
/

SELECT * from VANZARI_ONLINE;

```

Script Output x Query Result x

SQL | All Rows Fetched: 7 in 0.013 seconds

| | ID_MARFA | ID_COMANDA | TAXA_LIVRARE | CANTITATE | PRET | REDUCERE |
|---|----------|------------|--------------|-----------|-------|----------|
| 1 | 8 | 3 | 0 | 2 | 10800 | 10 |
| 2 | 10 | 2 | 0 | 1 | 2300 | 0 |
| 3 | 1 | 5 | 0 | 3 | 8100 | 10 |
| 4 | 2 | 7 | 0 | 3 | 12750 | 15 |
| 5 | 11 | 6 | 0 | 1 | 2780 | 0 |
| 6 | 5 | 4 | 0 | 2 | 1880 | 0 |
| 7 | 12 | 1 | 5 | 2 | 1000 | 0 |

6. Afiseaza angajatul care nu are superior

SET SERVEROUTPUT ON

DECLARE

TYPE tip is Record

(p_id_angajat angajat.id_angajat%type, p_nume angajat.nume%type, p_prenume varchar2(100));

re tip;

BEGIN

SELECT id_angajat, nume, prenume into re FROM ANGAJAT

WHERE id_superior is null;

DBMS_OUTPUT.put_line('Angajatul ' || re.p_nume || ' ' || re.p_prenume || ' este liderul.');

end;

/



```
SET SERVEROUTPUT ON
DECLARE
  TYPE tip is Record
  (p_id_angajat angajat.id_angajat%type, p_nume angajat.nume%type, p_prenume varchar2(100));

  re tip;

BEGIN

  SELECT id_angajat, nume, prenume into re FROM ANGAJAT
  WHERE id_superior is null;

  DBMS_OUTPUT.put_line('Angajatul ' || re.p_nume || ' ' || re.p_prenume || ' este liderul.');
```

Script Output x Query Result x

Task completed in 0.103 seconds

PL/SQL procedure successfully completed.

Angajatul Marinescu Iulia este liderul.

7. Sa se creeze o tabela identica cu Comanda

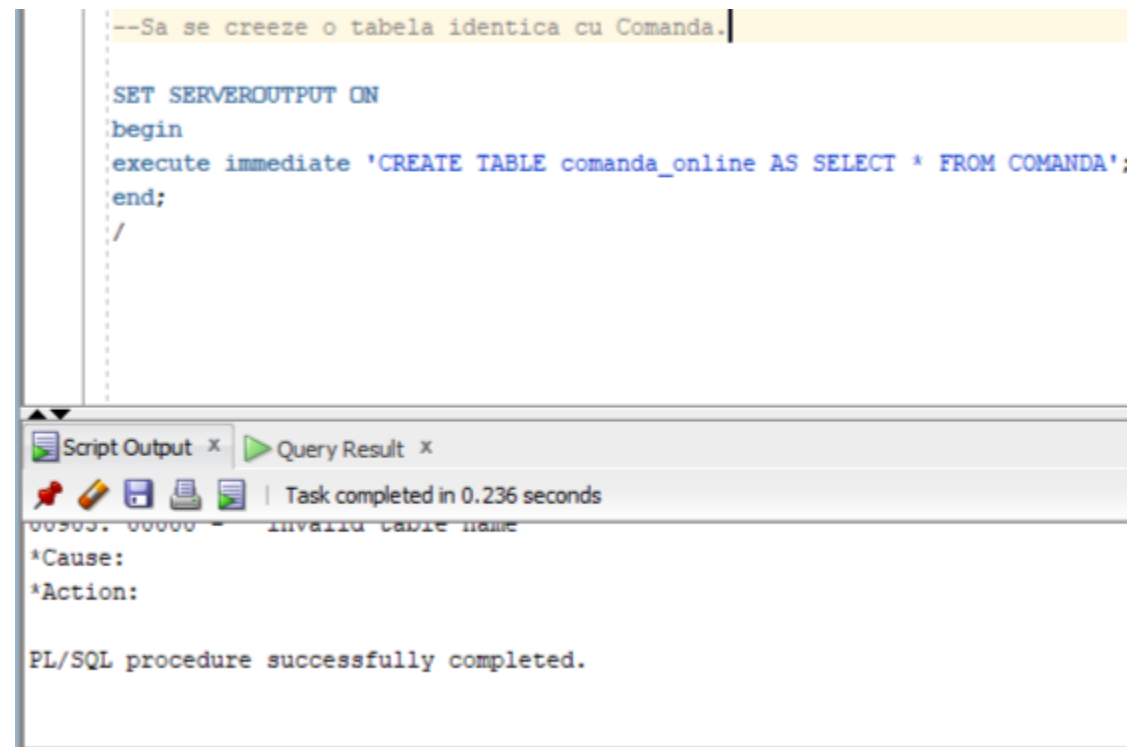
```
SET SERVEROUTPUT ON
```

```
begin
```

```
execute immediate 'CREATE TABLE comanda_online AS SELECT * FROM COMANDA';
```

```
end;
```

```
/
```



8. Sa se stearga tabela creata mai inainte.

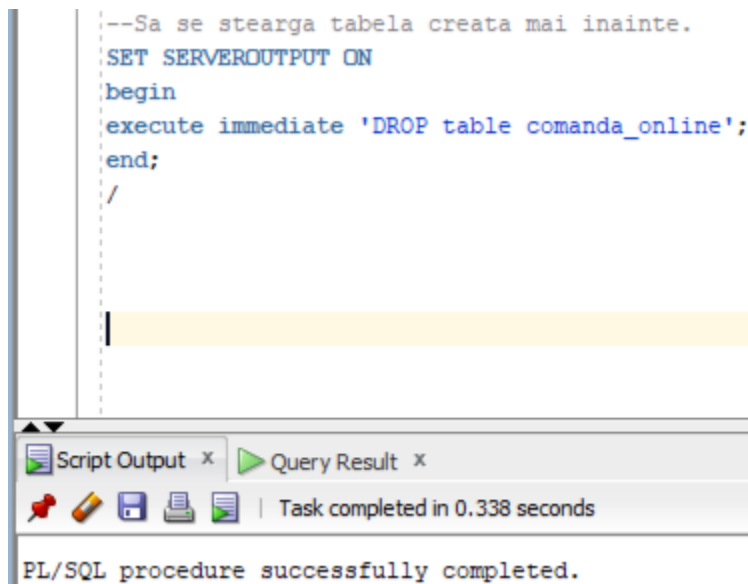
```
SET SERVEROUTPUT ON
```

```
begin
```

```
execute immediate 'DROP table comanda_online';
```

```
end;
```

```
/
```



```
--Sa se stearga tabela creata mai inainte.  
SET SERVEROUTPUT ON  
begin  
execute immediate 'DROP table comanda_online';  
end;  
/  
  
|
```

Script Output x Query Result x

Task completed in 0.338 seconds

PL/SQL procedure successfully completed.

Structuri de control

1. Sa se afiseze numele intreg al angajatilor care au numele egal cu prenumele, daca acestea au lungimi diferite se va afisa doar numele de familie.

```
SET SERVEROUTPUT on
```

```
DECLARE
```

```
CURSOR parcurs is(SELECT nume, prenume FROM ANGAJAT);
```

```
rec parcurs%rowtype;
```

```
BEGIN
```

```
for rec in parcurs loop
```

```
if (length(rec.nume)=length(rec.prenume)) then
```

```
DBMS_OUTPUT.put_line(rec.nume || ' ' || rec.prenume);
```

```
else
```

```
DBMS_OUTPUT.put_line(rec.nume);
```

```
END If;
```

```
end loop;
```

```
END;
```

```
/
```



```

SET SERVEROUTPUT on

DECLARE

CURSOR parcurs is (SELECT nume, prenume FROM ANGAJAT);

rec parcurs%rowtype;

BEGIN

for rec in parcurs loop

if (length(rec.nume)=length(rec.prenume)) then
DBMS_OUTPUT.put_line(rec.nume||' '||rec.prenume);
else
DBMS_OUTPUT.put_line(rec.nume);

END If;
end loop;
END;
/

```

Script Output x Query Result x

Task completed in 0.172 seconds

PL/SQL procedure successfully completed.

Petrescu
 Alexandrescu
 Ciurea Marius
 Marinescu
 Roman Mihai
 Ionescu
 Manulescu
 Superbus Septimus
 Lupascu Claudia
 Mulea Cezar
 Horthy Milkos
 Ionescu
 Marin Ioana
 Savantus
 Popa
 Dobroiescu

PL/SQL procedure successfully completed.

2. Se vor afisa vanzarile efectuate de intr-un magazin al carui id il citim de la tastatura. Sa se trateze exceptia atunci cand nu se gaseste nicio vanzare.

```
SET SERVEROUTPUT ON
```

```
DECLARE
```

```
v_id number:=&p_id;
```

```
cursor c is select * from VANZARI_MAGAZIN where id_magazin=v_id ;
```

```
v_id_magazin VANZARI_MAGAZIN.id_magazin%TYPE;
```

```
v_id_marfa VANZARI_MAGAZIN.id_marfa%TYPE;
```

```
v_id_comanda_mag VANZARI_MAGAZIN.id_comanda_mag%TYPE;
```

```
v_pret VANZARI_MAGAZIN.pret%TYPE;
```

```
v_cantitate VANZARI_MAGAZIN.cantitate%TYPE;
```

```
v_id_angajat VANZARI_MAGAZIN.id_angajat%TYPE;
```

```
BEGIN
```

```
open c;
```

```
loop
exit when c%NOTFOUND;
fetch c into v_id_magazin, v_id_marfa, v_id_comanda_mag, v_pret,v_cantitate,v_id_angajat ;

DBMS_OUTPUT.put_line('Comanda are un cost de ' || v_pret || '. S-a comandat in cantitate de ' || v_cantitate || '.');
end loop;
close c;
EXCEPTION
when NO_DATA_FOUND then
DBMS_OUTPUT.put_line('Nu exista aceset magazin');

end;
/
```

```
SET SERVEROUTPUT ON
```

DECLARE

```
v_id number:=sp_id;
cursor c is select * from VANZARI_MAGAZIN where id_magazin=v_id ;
v_id_magazin VANZARI_MAGAZIN.id_magazin%TYPE;
v_id_marfa VANZARI_MAGAZIN.id_marfa%TYPE;
v_id_comanda_mag VANZARI_MAGAZIN.id_comanda_mag%TYPE;
v_pret VANZARI_MAGAZIN.pret%TYPE;
v_cantitate VANZARI_MAGAZIN.cantitate%TYPE;
v_id_angajat VANZARI_MAGAZIN.id_angajat%TYPE;
BEGIN

open c;
loop
exit when c%NOTFOUND;
fetch c into v_id_magazin, v_id_marfa, v_id_comanda_mag, v_pret,v_cantitate,v_id_angajat ;

DBMS_OUTPUT.put_line('Comanda are un cost de '||v_pret||'. S-a comandat in cantitate de '||v_cantitate||'.');
end loop;
close c;
EXCEPTION
when NO_DATA_FOUND then
DBMS_OUTPUT.put_line('Nu exista aceset magazin');

end;
/
```

Script Output x Query Result x
Task completed in 3.545 seconds

```
Comanda are un cost de 5000. S-a comandat in cantitate de 1.
Comanda are un cost de 5000. S-a comandat in cantitate de 1.
Comanda are un cost de 6000. S-a comandat in cantitate de 2.
Comanda are un cost de 1020. S-a comandat in cantitate de 3.
Comanda are un cost de 2780. S-a comandat in cantitate de 1.
Comanda are un cost de 9650. S-a comandat in cantitate de 2.
Comanda are un cost de 4825. S-a comandat in cantitate de 1.
Comanda are un cost de 1355. S-a comandat in cantitate de 3.
Comanda are un cost de 5560. S-a comandat in cantitate de 2.
Comanda are un cost de 5560. S-a comandat in cantitate de 2.
```

PL/SQL procedure successfully completed.

3. Sa se afiseze pretul si modalitatea de plata pentru fiecare produs cumparat online. Daca s-a comandat mai mult de un produs atunci se va afisa si cantitatea.

```
SET SERVEROUTPUT ON
```

```
DECLARE
```

```
TYPE inregistrare is record(i_id_Comanda comanda.id_comanda%type, i_pret vanzari_online.pret%type,  
i_modalitate comanda.modalitate%type, i_cantitate vanzari_online.cantitate%type);
```

```
rand inregistrare;
```

```
i number;
```

```
BEGIN
```

```
i:=1;
```

```
while i<=7 loop
```

```
select c.id_comanda,v.pret,c.modalitate,v.cantitate into rand from Comanda c, VANZARI_ONLINE v
```

```
where c.id_comanda=v.id_comanda and c.id_comanda=i;
```

```
if(rand.i_cantitate>1) then
DBMS_OUTPUT.put_line(rand.i_id_comanda||' '||rand.i_pret||' '||rand.i_cantitate||' '||rand.i_modalitate);
else
DBMS_OUTPUT.put_line(rand.i_id_comanda||' '||rand.i_pret||' '||rand.i_modalitate);
end if;

i:=i+1;
end loop;

end;
/
```

```

SET SERVEROUTPUT ON

DECLARE

TYPE inregistrare IS RECORD(i_id_Comanda comanda.id_comanda%TYPE, i_pret vanzari_online.pret%TYPE,
i_modalitate comanda.modalitate%TYPE, i_cantitate vanzari_online.cantitate%TYPE);

rand inregistrare;
i NUMBER;
BEGIN
i:=1;
WHILE i<=7 LOOP

select c.id_comanda,v.pret,c.modalitate,v.cantitate into rand from Comanda c, VANZARI_ONLINE v
where c.id_comanda=v.id_comanda and c.id_comanda=i;

IF(rand.i_cantitate>1) THEN
DBMS_OUTPUT.put_line(rand.i_id_comanda||' '||rand.i_pret||' '||rand.i_cantitate||' '||rand.i_modalitate);
ELSE
DBMS_OUTPUT.put_line(rand.i_id_comanda||' '||rand.i_pret||' '||rand.i_modalitate);
END IF;

i:=i+1;
END LOOP;

END;
/

```

Script Output x Query Result x

Task completed in 0.16 seconds

PL/SQL procedure successfully completed.

```

1 1000 2 CASH
2 2300 CASH
3 10800 2 CARD
4 1880 2 CASH
5 8100 3 CARD
6 2780 CASH
7 12750 3 CARD

```

PL/SQL procedure successfully completed.

Tratarea Exceptiilor

1. Sa se calculeze rata care va fi platita pentru un produs din Vanzari-Magazin al carui cod este cict de la tastatura. Numarul de luni pe care se va plati se va prelua de la tastatura. Introduceti o exceptie atunci cand numarul de luni va fi egal cu 0 sau atunci cand sunt returnate mai multe randuri decat unul.

```
SET SERVEROUTPUT ON
```

```
DECLARE
```

```
v_luni number:=&p_luni;
```

```
v_pret number;
```

```
v_rata number;
```

```
BEGIN
```

```
SELECT pret into v_pret from VANZARI_MAGAZIN
```

```
WHERE id_marfa=&p_id_marfa;
```

```
v_rata:=v_pret/v_luni;
```

```
DBMS_OUTPUT.put_line('Se va plati o rata de ' || v_rata);
```


EXCEPTION

WHEN TOO_MANY_ROWS then

DBMS_OUTPUT.put_line('Exista mai multe produse cu acelasi cod!');

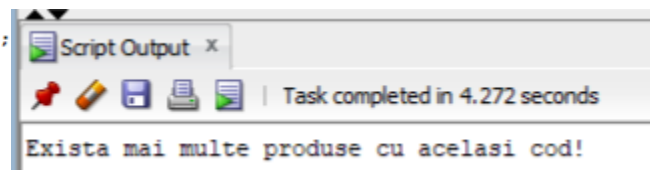
WHEN zero_divide then

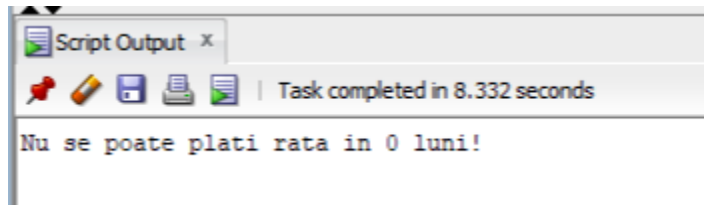
DBMS_OUTPUT.put_line('Nu se poate plati rata in 0 luni!');

END;

/

```
SET SERVEROUTPUT ON  
  
DECLARE  
  
    v_luni number:=ap_luni;  
    v_pret number;  
    v_rata number;  
  
BEGIN  
    SELECT pret into v_pret from VANZARI_MAGAZIN  
    WHERE id_marfa=ap_id_marfa;  
    v_rata:=v_pret/v_luni;  
    DBMS_OUTPUT.put_line('Se va plati o rata de '||v_rata);  
  
EXCEPTION  
WHEN TOO_MANY_ROWS then  
    DBMS_OUTPUT.put_line('Exista mai multe produse cu acelasi cod!');  
WHEN zero_divide then  
    DBMS_OUTPUT.put_line('Nu se poate plati rata in 0 luni!');  
  
END;  
/
```





2. Sa se afiseze comenzile efectuate de clientul cu numarul 3 din tabela Comenzi. Sa se trateze situatia in care cursorul este deschis de mai multe ori odata.

```
SET SERVEROUTPUT ON
```

```
DECLARE
```

```
cursor c is SELECT id_comanda, id_client, modalitate FROM COMANDA
```

```
where id_comanda=3;
```

```
rec c%ROWTYPE;
```

```
BEGIN
```

```
open c;
```

```
for rec in c loop
```

```
DBMS_OUTPUT.put_line('Clientul cu id 3 a efectuat comanda ' || rec.id_comanda || ' prin ' || rec.modalitate);
```

```
end loop;
```

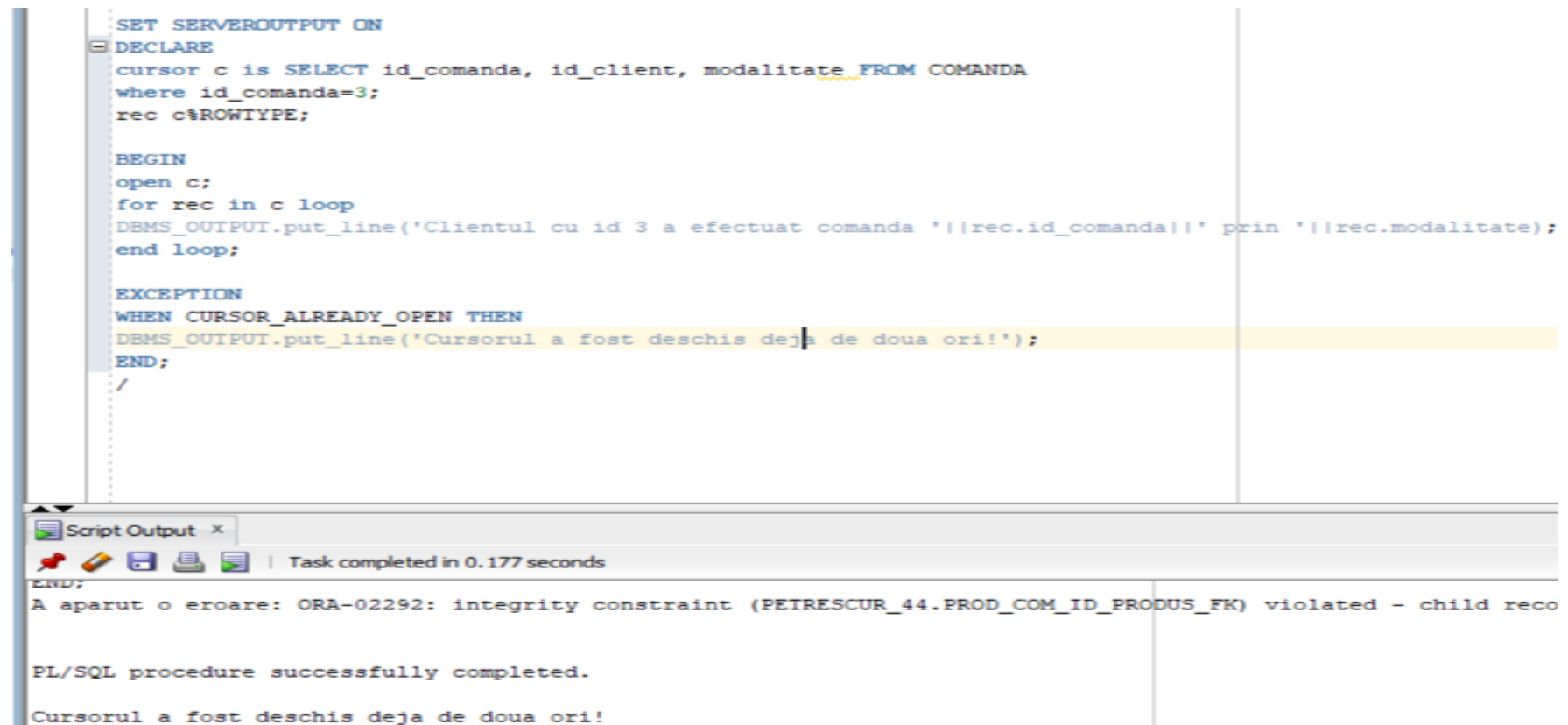
EXCEPTION

WHEN CURSOR_ALREADY_OPEN THEN

DBMS_OUTPUT.put_line('Cursorul a fost deschis deja de doua ori!');

END;

/



The screenshot displays the Oracle SQL Developer environment. The main window shows a PL/SQL script with the following code:

```
SET SERVEROUTPUT ON
DECLARE
cursor c is SELECT id_comanda, id_client, modalitate FROM COMANDA
where id_comanda=3;
rec c%ROWTYPE;

BEGIN
open c;
for rec in c loop
DBMS_OUTPUT.put_line('Clientul cu id 3 a efectuat comanda '||rec.id_comanda||' prin '||rec.modalitate);
end loop;

EXCEPTION
WHEN CURSOR_ALREADY_OPEN THEN
DBMS_OUTPUT.put_line('Cursorul a fost deschis deja de doua ori!');
END;
/
```

The line `DBMS_OUTPUT.put_line('Cursorul a fost deschis deja de doua ori!');` is highlighted in yellow. Below the script editor, the 'Script Output' window is visible, showing the execution results:

```
END;
A aparut o eroare: ORA-02292: integrity constraint (PETRESCUR_44.PROD_COM_ID_PRODUS_FK) violated - child reco

PL/SQL procedure successfully completed.
Cursorul a fost deschis deja de doua ori!
```

3. Sa se modifice denumirea bunului care are id-ul egal cu 20. Creati o exceptie in cazul in care nu se schimba nimic.

```
SET SERVEROUTPUT ON

DECLARE

find_exception EXCEPTION;

PRAGMA EXCEPTION_INIT(find_exception,-01323);

BEGIN

UPDATE MARFURI

set denumire_marfa='Cauciuc Debica'

where id_marfa=20;

if SQL%ROWCOUNT=0 then

raise find_exception;

end if;

EXCEPTION

when find_exception then

DBMS_OUTPUT.put_line('Nu exista un produs cu aces cod!');

when others then

DBMS_OUTPUT.put_line('A aparut o eroare! '||SQLERRM);
```

END;

/

```
SET SERVEROUTPUT ON

DECLARE

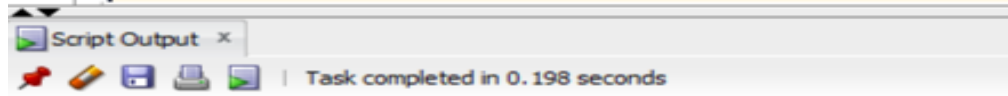
find_exception EXCEPTION;
PRAGMA EXCEPTION_INIT(find_exception,-01323);

BEGIN

UPDATE MARFURI
set denumire_marfa='Cauciuc Debica'
where id_marfa=20;

if SQL%ROWCOUNT=0 then
raise find_exception;
end if;
EXCEPTION
when find_exception then
DBMS_OUTPUT.put_line('Nu exista un produs cu aces cod!');
when others then
DBMS_OUTPUT.put_line('A aparut o eroare! '||SQLERRM);

END;
/
```



PL/SQL procedure successfully completed.

Nu exista un produs cu aces cod!

PL/SQL procedure successfully completed.

4. Sa se gaseasca produsele cu pretul de achizitie mai mare de 1000. Sa se trateze exceptia in cazul in care acestea nu exista.

```
SET SERVEROUTPUT ON
```

```
DECLARE
```

```
nu_avem_nimic exception;
```

```
PRAGMA EXCEPTION_INIT(nu_avem_nimic,-01357);
```

```
v_count NUMBER;
```

```
BEGIN
```

```
SELECT COUNT(*) INTO v_count FROM MARFURI WHERE pret_import > 1000;
```

```
if v_count>0 then
```

```
for rec in (SELECT id_marfa, denumire_marfa, pret_import from MARFURI where pret_import>1000) loop
```

```
DBMS_OUTPUT.put_line('Produsul ' || rec.denumire_marfa || ' are un pret de ' || rec.pret_import);
```

```
end loop;
```

```
else
```

```
raise nu_avem_nimic;
```

```
end if;
```

```
EXCEPTION
```

```
WHEN nu_avem_nimic then
```

```
DBMS_OUTPUT.put_line('Nu exista produs cu pret mai mare de 1000!');
```

```
WHEN OTHERS then
```

```
DBMS_OUTPUT.put_line('A aparut o eroare: '||SQLERRM);
```

```
END;
```

```
/
```

```
SET SERVEROUTPUT ON

DECLARE
nu_avem_nimic exception;
PRAGMA EXCEPTION_INIT(nu_avem_nimic,-01357);
v_count NUMBER;
BEGIN
SELECT COUNT(*) INTO v_count FROM MARFURI WHERE pret_import > 1000;
if v_count>0 then
for rec in (SELECT id_marfa, denumire_marfa, pret_import from MARFURI where pret_import>1000) loop
DBMS_OUTPUT.put_line('Produsul '||rec.denumire_marfa||' are un pret de '||rec.pret_import);
end loop;
else
raise nu_avem_nimic;
end if;
EXCEPTION
WHEN nu_avem_nimic then
DBMS_OUTPUT.put_line('Nu exista produs cu pret mai mare de 1000!');
WHEN OTHERS then
DBMS_OUTPUT.put_line('A aparut o eroare: '||SQLERRM);
END;
/
```

Script Output x

Task completed in 0.191 seconds

*Action:
Nu exista produs cu pret mai mare de 1000!

PL/SQL procedure successfully completed.

5. Sa se afiseze magazinele care se gasesc intr-un oras. Daca nu se inchide corespunzator cursorul, sa se trateze exceptia.

```
SET SERVEROUTPUT ON
```

```
DECLARE
```

```
v_oras varchar2(50):='Bucuresti';
```

```
--folosim bucuresti ca exemplu pt ca acolo avem deja magazine
```

```
cursor rand_magazine(p_oras v_oras%type) IS SELECT id_magazin, oras, tara FROM MAGAZINE
```

```
WHERE oras=p_oras;
```

```
rec rand_magazine%ROWTYPE;
```

```
BEGIN
```

```
for rec in rand_magazine(v_oras) loop
```

```
DBMS_OUTPUT.put_line('Magazinul ' || rec.id_magazin || ' se afla in ' || rec.oras || ', ' || rec.tara);
```

```
end loop;
```

```
close rand_magazine;
```

```
EXCEPTION
```

```
WHEN INVALID_CURSOR then
```

```
DBMS_OUTPUT.put_line('Ati inchis cursorul din grseala de doua ori!');
```

```
END;
```


/

```
SET SERVEROUTPUT ON
DECLARE
v_oras varchar2(50):='Bucuresti';
--folosim bucuresti ca exemplu pt ca acolo avem deja magazine
cursor rand_magazine(p_oras v_oras%type) IS SELECT id_magazin, oras, tara FROM MAGAZINE
WHERE oras=p_oras;
rec rand_magazine%ROWTYPE;
BEGIN
for rec in rand_magazine(v_oras) loop
DBMS_OUTPUT.put_line('Magazinul '||rec.id_magazin||' se afla in '||rec.oras||', '||rec.tara);
end loop;
close rand_magazine;
EXCEPTION
WHEN INVALID_CURSOR then
DBMS_OUTPUT.put_line('Ati inchis cursorul din grseala de doua ori!');
END;
/
```

Script Output x

Task completed in 0.132 seconds

The symbol "!=" was substituted for "(" to continue.
06550. 00000 - "line %s, column %s:\n%s"
*Cause: Usually a PL/SQL compilation error.
*Action:
Magazinul 1 se afla in Bucuresti, Romania
Ati inchis cursorul din grseala de doua ori!

Exercitii speciale pentru gestionarea cursorilor

1. Schimbati in tabela comanda toate modalitate pe plata CASH la CARD. Afisati cate randuri au fost modificate, dupa care anulati tranzactia.

```
SET SERVEROUTPUT on
```

```
BEGIN
```

```
UPDATE COMANDA
```

```
set modalitate='CARD'
```

```
where modalitate='CASH';
```

```
DBMS_OUTPUT.put_line('S-au modificat atatea ranudri: ' || SQL%ROWCOUNT);
```

```
ROLLBACK;
```

```
DBMS_OUTPUT.put_line('Au ramas ' || SQL%ROWCOUNT || ' linii modificate!');
```

```
END;
```

```
/
```

```
SET SERVEROUTPUT on
BEGIN
UPDATE COMANDA
set modalitate='CARD'
where modalitate='CASH';
DBMS_OUTPUT.put_line('S-au modificat atatea ranudri: '||SQL%ROWCOUNT);
ROLLBACK;
DBMS_OUTPUT.put_line('Au ramas '||SQL%ROWCOUNT||' linii modificate!');
END;
/
```

Script Output x

Task completed in 0.153 seconds

```
S-au modificat atatea ranudri: 4
Au ramas 0 linii modificate!

PL/SQL procedure successfully completed.
```

2. Sa se adauge un spor de 15% la fiecare angajat care lucreaza dinainte de 2020 si nu are un spor, folosind un cursor implicit.
Sa se confirme tranzactia.

SET SERVEROUTPUT ON

declare

```
v_an date:=to_date(2020,'yyyy');  
  
BEGIN  
  
UPDATE ANGAJAT  
  
set spor=15 where extract(year from data_angajare)<2020 and spor IS NULL;  
  
if sql%NOTFOUND then  
  
DBMS_OUTPUT.put_line('Nu avem angajati care lucreaza dinainte de 2020, care sa nu aibe spor!');  
  
else  
  
DBMS_OUTPUT.put_line('S-a modificat sporul pentru '||SQL%ROWCOUNT||' angajati!');  
  
end if;  
  
COMMIT;  
  
END;  
  
/
```

```

SET SERVEROUTPUT ON
declare
v_an date:=to_date(2020,'yyyy');
BEGIN
UPDATE ANGAJAT
set spor=15 where extract(year from data_angajare)<2020 and spor IS NULL;

if sql%NOTFOUND then
DBMS_OUTPUT.put_line('Nu avem angajati care lucreaza dinainte de 2020, care sa nu aibe spor!');
else
DBMS_OUTPUT.put_line('S-a modificat sporul pentru '||SQL%ROWCOUNT||' angajati!');
end if;
COMMIT;
END;
/

```

Script Output x



Task completed in 0.199 seconds

PL/SQL procedure successfully completed.

S-a modificat sporul pentru 2 angajati!

PL/SQL procedure successfully completed.

3. Alfati ce produse au un stoc mai mare disponibil decat cel pe care al carui id il vom primi ca parametru citit.

```
SET SERVEROUTPUT ON
```

```
declare
```

```
v_id varchar2(50):=&p_id;
```

```
stoc_referinta number;
```

```
cursor c(m_stoc number) is select id_marfa, denumire_marfa, stoc from Marfuri
```

```
where m_stoc<stoc;
```

```
begin
```

```
SELECT stoc into stoc_referinta from MARFURI
```

```
where id_marfa=v_id;
```

```
DBMS_OUTPUT.put_line('Produsele care au un stoc mai mare de ' || stoc_referinta || ' sunt:');
```

```
for rec in c(stoc_referinta) loop
```

```
DBMS_OUTPUT.put_line('Produsul ' || rec.denumire_marfa || ' are un stoc de produse disponibil egal cu: ' || rec.stoc);
```

```
end loop;
```

```
END;
```

```
/
```

```

SET SERVEROUTPUT ON
declare
v_id varchar2(50):=sp_id;
stoc_referinta number;
cursor c(m_stoc number) is select id_marfa, denumire_marfa, stoc from Marfuri
where m_stoc<stoc;

begin
SELECT stoc into stoc_referinta from MARFURI
where id_marfa=v_id;
DBMS_OUTPUT.put_line('Produsele care au un stoc mai mare de '||stoc_referinta||' sunt:');
for rec in c(stoc_referinta) loop
DBMS_OUTPUT.put_line('Produsul '||rec.denumire_marfa||' are un stoc de produse disponibil egal cu: '||rec.stoc);

end loop;

END;
/

```

Script Output x

Task completed in 3.054 seconds

```

END,
Produsele care au un stoc mai mare de 4100 sunt:
Produsul Karcher 250W are un stoc de produse disponibil egal cu: 17000
Produsul Toshiba Microwave50 are un stoc de produse disponibil egal cu: 5000
Produsul Gorenjie Boiler 12 are un stoc de produse disponibil egal cu: 10000
Produsul Apple Iphone 14 are un stoc de produse disponibil egal cu: 30000
Produsul Samsung Galaxy S22 are un stoc de produse disponibil egal cu: 7400
Produsul Huawei P50 are un stoc de produse disponibil egal cu: 5000

PL/SQL procedure successfully completed.

```

Functii, proceduri, pachete

1. Creati o functie care sa returneze suma de bani castigata in vanzarile online.

SET SERVEROUTPUT on

create or replace function profit

return number

is

v_suma number;

BEGIN

SELECT SUM(pret* cantitate) into v_suma FROM VANZARI_ONLINE;

return v_suma;

END;

/

SET SERVEROUTPUT on

declare

v_profit number;

begin

v_profit:=profit();

DBMS_OUTPUT.put_line('In vanzarile online se inregistreaza castiguri in valoare de : '||v_profit);

END;

/

```
SET SERVEROUTPUT on

create or replace function profit
return number
is
v_suma number;
BEGIN
SELECT SUM(pret* cantitate) into v_suma FROM VANZARI_ONLINE;

return v_suma;

END;
/

SET SERVEROUTPUT on
declare
v_profit number;
begin
v_profit:=profit();
DBMS_OUTPUT.put_line('In vanzarile online se inregistreaza castiguri in valoare de : '||v_profit);
END;
/
```

Script Output x Query Result x
Task completed in 0.461 seconds

PL/SQL procedure successfully completed.

Function PROFIT compiled

In vanzarile online se inregistreaza castiguri in valoare de : 94990

PL/SQL procedure successfully completed.

2. Sa se vada care este media salariala platita lunar cu ajutorul unei functii.

```
SET SERVEROUTPUT on
```

```
create or replace function salariu_mediu (val_medie out NUMBER)
```

```
return number IS
```

```
BEGIN
```

```
SELECT AVG(salariu) into val_medie from ANGAJAT;
```

```
return val_medie;
```

```
END;
```

```
/
```

```
SET SERVEROUTPUT ON
```

```
declare
```

```
sal_med number;
```

BEGIN

DBMS_OUTPUT.put_line(salariu_mediu(sal_med));

END;

/

```
SET SERVEROUTPUT on

create or replace function salariu_mediu (val_medie out NUMBER)
return number IS
BEGIN

SELECT AVG(salariu) into val_medie from ANGAJAT;

return val_medie;

END;
/
SET SERVEROUTPUT ON

declare
sal_med number;
BEGIN
DBMS_OUTPUT.put_line(salariu_mediu(sal_med));
END;
/
```

Script Output x Query Result x

Task completed in 2.486 seconds

PL/SQL procedure successfully completed.

Function SALARIU_MEDIU compiled

3240.625

PL/SQL procedure successfully completed.

3. Creati o procedura care sa afiseze cinci angajati folosind un tablou indexat.

SET SERVEROUTPUT on

create or replace procedure afis_angajati(inceput in number, sfarsit in number) IS

type ang_table is table of Angajat%ROWTYPE index by pls_integer;

tabel ang_table;

BEGIN

for i in inceput..sfarsit loop

SELECT * into tabel(i) from ANGAJAT where id_angajat=i;

end loop;

for i in tabel.first..tabel.last loop

DBMS_OUTPUT.put_line('Angajatul ' || tabel(i).nume || ' ' || tabel(i).prenume || ' lucreaza ca ' || tabel(i).denumire_functie || ' pe
un salariu de: ' || tabel(i).salariu);

end loop;

end;

/

call afis_angajati(1,5);

```
SET SERVEROUTPUT on

create or replace procedure afis_angajati(inceput in number, sfarsit in number) IS
type ang_table is table of Angajat%ROWTYPE index by pls_integer;
tabel ang_table;
BEGIN
for i in inceput..sfarsit loop
SELECT * into tabel(i) from ANGAJAT where id_angajat=i;
end loop;
for i in tabel.first..tabel.last loop
DBMS_OUTPUT.put_line('Angajatul '||tabel(i).nume||' '||tabel(i).prenume||' lucreaza ca '||tabel(i).denumire_functie||' pe un salariu de: '||tabel(i).salariu);
end loop;

end;
/

call afis_angajati(1,5);
```

Script Output x Query Result x

Task completed in 0.114 seconds

procedure afis_angajati compiled

| | |
|---|--|
| Angajatul Petrescu Rares-Mihnea lucreaza ca Supervizor-Departament pe un salariu de: 4000 | |
| Angajatul Alexandrescu Vlad lucreaza ca Vanzator pe un salariu de: 3000 | |
| Angajatul Dobroiescu Maria lucreaza ca Vanzator pe un salariu de: 4000 | |
| Angajatul Ciurea Marius lucreaza ca manipulant_marfa pe un salariu de: 2600 | |
| Angajatul Marinescu Iulia lucreaza ca Supervizor-Magazine pe un salariu de: 4000 | |

Call completed.

4. Creati o functie care primeste id_ul unui client si ii returneaza prenumele.

```
create or replace function adu_prenumele(c_id client.id_client%type)
```

```
return client.prenume%type is
```

```
v_prenume client.prenume%type;
```

```
BEGIN
```

```
SELECT prenume into v_prenume from Client
```

```
where id_client=c_id;
```

```
return v_prenume;
```

```
END;
```

```
/
```

```
SET SERVEROUTPUT on
```

```
DECLARE
```

```
prenume_gasit client.prenume%type;
```

```
ind number;
```

```
BEGIN
```

```
ind:=&p_ind;
```

```
prenume_gasit:=adu_prenumele(ind);
```

```
DBMS_OUTPUT.put_line('Avem prenumele pentru angajatul cu id-ul '||ind||' :'||prenume_gasit);  
END;  
/
```

```
-- create or replace function adu_prenumele(c_id client.id_client%type)  
-- return client.prenume%type is  
-- v_prenume client.prenume%type;  
-- BEGIN  
-- SELECT prenume into v_prenume from Client  
-- where id_client=c_id;  
-- return v_prenume;  
-- END;  
-- /  
  
SET SERVEROUTPUT on  
-- DECLARE  
-- prenume_gasit client.prenume%type;  
-- ind number;  
-- BEGIN  
-- ind:=sp_ind;  
-- prenume_gasit:=adu_prenumele(ind);  
-- DBMS_OUTPUT.put_line('Avem prenumele pentru angajatul cu id-ul '||ind||' :'||prenume_gasit);  
-- END;  
-- /
```

Script Output x Query Result x
Task completed in 1.617 seconds

```
END;  
Avem prenumele pentru angajatul cu id-ul 4 :Florinel
```

```
PL/SQL procedure successfully completed.
```

5. Creati o procedura in care se vor afisa denumirile serviciilor sub forma unui tabel imbricat.

```
SET SERVEROUTPUT ON
```

```
create type lista_servicii as table of varchar2(100);
```

```
create or replace procedure afis_servicii(tabel IN lista_servicii ) IS
```

```
BEGIN
```

```
for i in 1..tabel.count loop
```

```
DBMS_OUTPUT.put_line('Serviciul:' || tabel(i));
```

```
end loop;
```

```
END;
```

```
/
```

```
SET SERVEROUTPUT ON
```

```
declare
```

```
lista lista_servicii;
```



```
i number;  
tip servicii.tip_serviciu%type;  
BEGIN  
lista:=lista_servicii();  
Select count(*) into i FROM SERVICII;  
  
for j in 1..i loop  
Select tip_serviciu into tip from SERVICII  
where id_serviciu=j;  
lista.extend;  
lista(j):=tip;  
end loop;  
  
afis_servicii(lista);  
END;  
/
```

```

SET SERVEROUTPUT ON
create type lista_servicii as table of varchar2(100);
create or replace procedure afis_servicii(tabel IN lista_servicii ) IS

BEGIN
for i in 1..tabel.count loop
DBMS_OUTPUT.put_line('Serviciul:'||tabel(i));
end loop;

END;
/

SET SERVEROUTPUT ON
declare
lista lista_servicii;
i number;
tip serviciu.tip_serviciu%type;
BEGIN
lista:=lista_servicii();
Select count(*) into i FROM SERVICII;

for j in 1..i loop
Select tip_serviciu into tip from SERVICII
where id_serviciu=j;
lista.extend;
lista(j):=tip;
end loop;

afis_servicii(lista);
END;
/

```

Script Output x Query Result x

Task completed in 0.146 seconds

Procedure AFIS_SERVICII compiled

```

Serviciul:asigurare de baza
Serviciul:asigurare extinsa
Serviciul:extragarantie tip 1
Serviciul:extragarantie tip 2
Serviciul:serviciu de livrare
Serviciul:eMAG care

```

PL/SQL procedure successfully completed.

6. Creati o procedura care sa afiseze fiecare denumire de produs folosind un varray.

SET SERVEROUTPUT on

create type vector_variabil as Varray(20) of varchar2(100);

create or replace procedure afis_produce(multime IN vector_variabil) IS

BEGIN

for i in 1..multime.count loop

DBMS_OUTPUT.put_line('Acest produs are denumirea:' || multime(i));

END LOOP;

END;

/

SET SERVEROUTPUT ON

declare

lista_produce vector_variabil;

j number;

denumire marfuri.denumire_marfa%type;

BEGIN

lista_produce:=vector_variabil();

select count(denumire_marfa) into j from Marfuri;

```
for i in 1..j loop
lista_produce.extend;
select denumire_marfa into denumire from MARFURI
where id_marfa=i;
lista_produce(i):=denumire;
end loop;
afis_produce(lista_produce);
end;
/
```

```

SET SERVEROUTPUT on
create type vector_variabil as Varray(20) of varchar2(100);
create or replace procedure afis_produce( multime IN vector_variabil) IS
BEGIN
for i in 1..multime.count loop
DBMS_OUTPUT.put_line('Acest produs are denumirea:'||multime(i));
END LOOP;
END;
/
SET SERVEROUTPUT ON
declare
lista_produce vector_variabil;
j number;
denumire marfuri.denumire_marfa%type;
BEGIN
lista_produce:=vector_variabil();
select count(denumire_marfa) into j from Marfuri;

for i in 1..j loop
lista_produce.extend;
select denumire_marfa into denumire from MARFURI
where id_marfa=i;
lista_produce(i):=denumire;
end loop;

afis_produce(lista_produce);
end;
/

```

Script Output x

Query Result x

Task completed in 0.232 seconds

```

Acest produs are denumirea:Apple Iphone 14
Acest produs are denumirea:LG OLED 75
Acest produs are denumirea:Karcher 250W
Acest produs are denumirea:Samsung Galaxy S22
Acest produs are denumirea:Canon AD380
Acest produs are denumirea:Toshiba Microwave50
Acest produs are denumirea:GoPro Hero14
Acest produs are denumirea:Sony Bravia X57
Acest produs are denumirea:Acer Predator 17
Acest produs are denumirea:Samsung Curve 25
Acest produs are denumirea:Huawei P50
Acest produs are denumirea:Gorenjie Boiler 12

PL/SQL procedure successfully completed.

```

7. Creati un pachet care sa permita afisarea de inregistrari din Comenzi_Magazin pe baza modalitatii de plata.
Pachetul va contine si functii/proceduri care sa adauge TVA la vanzarile facute online sau in magazin.

```
SET SERVEROUTPUT ON
```

```
create or replace package prelucrare is
```

```
procedure afisare_comenzi_modalitate(p_modalitate IN comenzi_magazin.modalitate%type);
```

```
procedure adauga_TVA_online;
```

```
procedure adauga_TVA_magazin;
```

```
END;
```

```
/
```

```
create or replace package body prelucrare is
```

```
procedure afisare_comenzi_modalitate(p_modalitate IN comenzi_magazin.modalitate%type) is
```

```
BEGIN
```

```
for rec in (SELECT id_comanda_mag,data_efectuare, modalitate FROM COMENZI_MAGAZIN Where modalitate=p_modalitate)  
loop
```

```
DBMS_OUTPUT.put_line('Comanda ' || rec.id_comanda_mag || ' s-a efectuat la data de ' || rec.data_efectuare || ' sub forma de  
' || rec.modalitate);  
  
end loop;  
  
end;  
  
procedure adauga_TVA_online is  
  
BEGIN  
  
UPDATE VANZARI_ONLINE  
  
set pret=pret+0.19*pret;  
  
DBMS_OUTPUT.put_line('S-a introdus TVA-ul in pretul din vanzarile online!');  
  
end;  
  
procedure adauga_TVA_magazin is  
  
BEGIN  
  
UPDATE VANZARI_MAGAZIN  
  
set pret=pret+0.19*pret;  
  
DBMS_OUTPUT.put_line('S-a introdus TVA-ul in pretul din vanzarile fizice!');  
  
end;  
  
end prelucrare;
```

```

SET SERVEROUTPUT ON
create or replace package prelucrare is
procedure afisare_comenzi_modalitate(p_modalitate IN comenzi_magazin.modalitate%type);
procedure adauga_TVA_online;
procedure adauga_TVA_magazin;
END;
/
create or replace package body prelucrare is

procedure afisare_comenzi_modalitate(p_modalitate IN comenzi_magazin.modalitate%type) is
BEGIN
for rec in (SELECT id_comanda_mag,data_efectuare, modalitate FROM COMENZI_MAGAZIN Where modalitate=p_modalitate) loop
DBMS_OUTPUT.put_line('Comanda '||rec.id_comanda_mag||' s-a efectuat la data de '||rec.data_efectuare||' sub forma de '||rec.modalitate);
end loop;
end;

procedure adauga_TVA_online is
BEGIN
UPDATE VANZARI_ONLINE
set pret=pret+0.19*pret;
DBMS_OUTPUT.put_line('S-a introdus TVA-ul in pretul din vanzarile online!');
end;

procedure adauga_TVA_magazin is
BEGIN
UPDATE VANZARI_MAGAZIN
set pret=pret+0.19*pret;
DBMS_OUTPUT.put_line('S-a introdus TVA-ul in pretul din vanzarile fizice!');
end;
end prelucrare;

execute prelucrare.afisare_comenzi_modalitate('CARD');
execute prelucrare.adauga_TVA_online;
execute prelucrare.adauga_TVA_magazin;

```

Script Output x Query Result x

Task completed in 0.165 seconds

dependency.

*Action: delete dependencies first then parent or disable constraint.

Comanda 2 s-a efectuat la data de 19-JUL-22 sub forma de CARD

Comanda 5 s-a efectuat la data de 30-AUG-22 sub forma de CARD

Comanda 8 s-a efectuat la data de 01-JUN-22 sub forma de CARD

Comanda 12 s-a efectuat la data de 05-JUL-22 sub forma de CARD

Comanda 13 s-a efectuat la data de 29-JUL-22 sub forma de CARD

PL/SQL procedure successfully completed.

Exercitii cu trigger

1. Sa se creeze un trigger prin care sa nu se poata depasi valoare de 25 atunci cand se adauga un spor nou unui angajat.

SET SERVEROUTPUT ON

create or replace trigger spor_depasit

BEFORE UPDATE ON ANGAJAT

FOR EACH ROW

DECLARE

c_spor_limita constant number:=25;

BEGIN

if :new.spor>c_spor_limita then

RAISE_APPLICATION_ERROR(-20202,'Nu se poate aplica un spor mai mare de 25!');

end if;

END;

/

UPDATE ANGAJAT

set spor=35 where id_angajat=2;

```
SET SERVEROUTPUT ON

create or replace trigger spor_depasit
BEFORE UPDATE ON ANGAJAT
FOR EACH ROW
DECLARE
c_spor_limita constant number:=25;
BEGIN
if :new.spor>c_spor_limita then
RAISE_APPLICATION_ERROR(-20202,'Nu se poate aplica un spor mai mare de 25!');
end if;
END;
/

UPDATE ANGAJAT
set spor=35 where id_angajat=2;
```

Script Output x Query Result x

Task completed in 0.248 seconds

1 row inserted.

1 row deleted.

Rollback complete.

Trigger SPOR_DEPASIT compiled

Error starting at line : 610 in command -
UPDATE ANGAJAT
set spor=35 where id_angajat=2
Error report -
ORA-20202: Nu se poate aplica un spor mai mare de 25!
ORA-06512: at "PETRESCUR_44.SPOR_DEPASIT", line 5
ORA-04088: error during execution of trigger 'PETRESCUR_44.SPOR_DEPASIT'

2. Sa se creeze un trigger prin care nu se va putea face nicio editare in tabela Comenzi_Magazin dupa ora 17:00

```
SET SERVEROUTPUT ON
```

```
create or replace trigger ora_depasita
```

```
before insert or update or delete on COMENZI_MAGAZIN
```

```
DECLARE
```

```
BEGIN
```

```
if to_char(sysdate,'hh24mi')>'17:00' and INSERTING then
```

```
RAISE_APPLICATION_ERROR(-20001,'Nu se poate insera la ora asta nimic!');
```

```
elsif to_char(sysdate,'hh24mi')>'17:00' and UPDATING then
```

```
RAISE_APPLICATION_ERROR(-20001,'Nu se poate schimba la ora asta nimic!');
```

```
elsif to_char(sysdate,'hh24mi')>'17:00' and Deleting then
```

```
RAISE_APPLICATION_ERROR(-20001,'Nu se poate sterge la ora asta nimic!');
```

```
end if;
```

```
end;
```

```
/
```

```
Update Comenzi_Magazin
```

```
set modalitate='CASH' where id_comanda_mag=3;
```

delete from COMENZI_MAGAZIN

where modalitate='CARD';

The screenshot displays the Oracle SQL Developer environment. The top pane shows a SQL script with a trigger definition and two SQL statements. The bottom pane shows the execution results, including a successful trigger compilation and two error messages.

```
SET SERVEROUTPUT ON
create or replace trigger ora_depasita
before insert or update or delete on COMENZI_MAGAZIN

DECLARE
BEGIN
if to_char(sysdate,'hh24mi')>'17:00' and INSERTING then
RAISE_APPLICATION_ERROR(-20001,'Nu se poate insera la ora asta nimic!');
elsif to_char(sysdate,'hh24mi')>'17:00' and UPDATING then
RAISE_APPLICATION_ERROR(-20001,'Nu se poate schimba la ora asta nimic!');
elsif to_char(sysdate,'hh24mi')>'17:00' and Deleting then
RAISE_APPLICATION_ERROR(-20001,'Nu se poate sterge la ora asta nimic!');
end if;
end;
/

Update Comenzi_Magazin
set modalitate='CASH' where id_comanda_mag=3;

delete from COMENZI_MAGAZIN
where modalitate='CARD';
```

Script Output x Query Result x

Task completed in 0.12 seconds

Trigger ORA_DEPASITA compiled

Error starting at line : 632 in command -
Update Comenzi_Magazin
set modalitate='CASH' where id_comanda_mag=3
Error report -
ORA-20001: Nu se poate schimba la ora asta nimic!
ORA-06512: at "PETRESCUR_44.ORA_DEPASITA", line 6
ORA-04088: error during execution of trigger 'PETRESCUR_44.ORA_DEPASITA'

Error starting at line : 635 in command -
delete from COMENZI_MAGAZIN
where modalitate='CARD'
Error report -
ORA-20001: Nu se poate sterge la ora asta nimic!
ORA-06512: at "PETRESCUR_44.ORA_DEPASITA", line 8
ORA-04088: error during execution of trigger 'PETRESCUR_44.ORA_DEPASITA'

Noua schema concpetuala a bazei de date:

