

Proiect la disciplina “Sisteme de gestiune a bazelor de date”

# Hardware Haven

Nume și grupă:

- Sasu Alexandru-Cristian

- Grupa 242

## 1. Prezentarea bazei de date:

Modelul de date va gestiona informații despre un magazin care se ocupă cu vânzarea componentelor, accesoriilor, și serviciilor destinate calculatoarelor și laptop-urilor. Serviciile oferite constau în: diagnosticarea, montarea, asamblarea, și testarea produselor. În afara celor precizate anterior, magazinul mai are inclus în oferta sa atât sisteme deja asamblate, precum: calculatoare și laptop-uri.

Există mai multe categorii de produse, fiecare produs aparținându-i unei singure categorii.

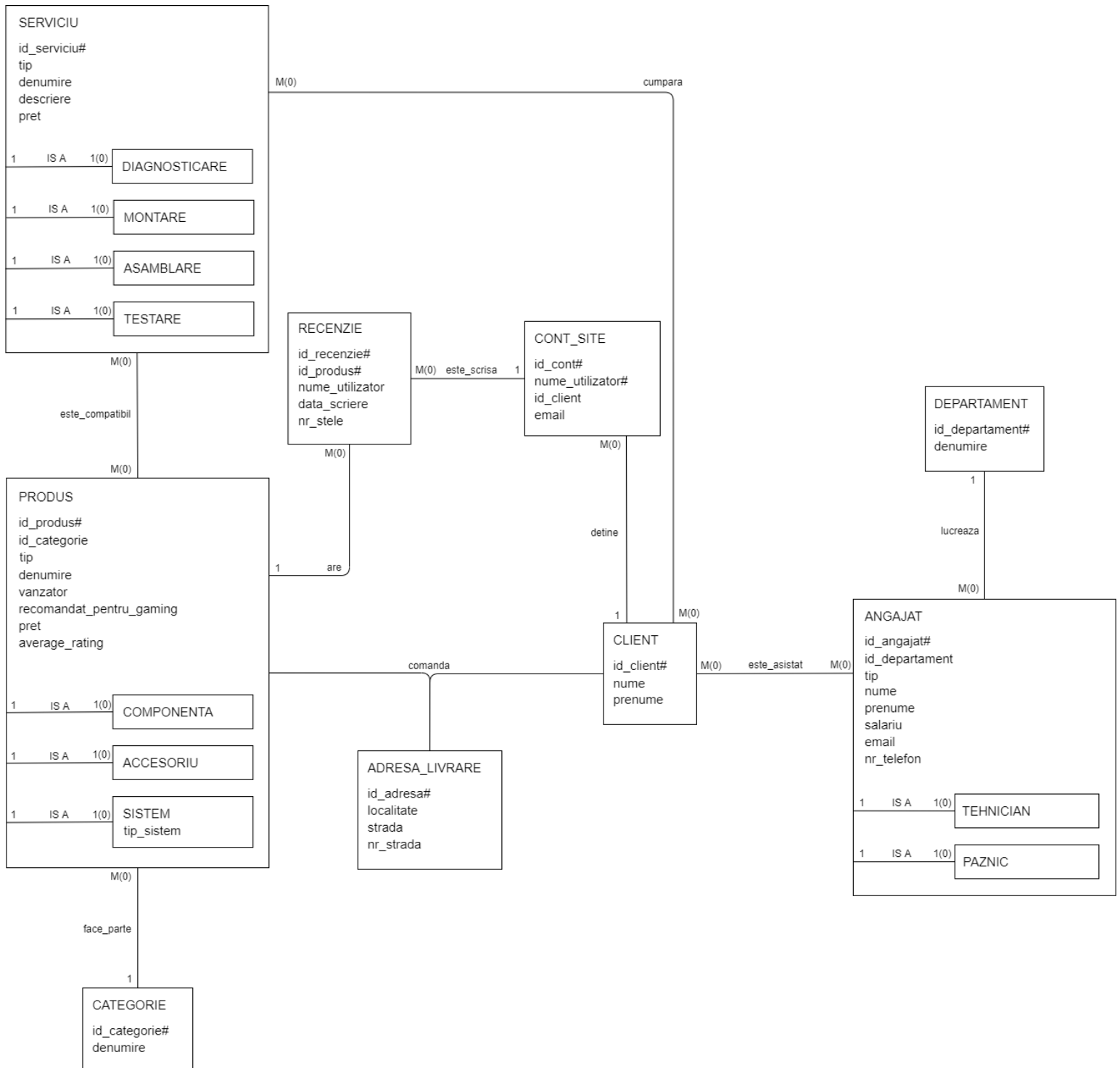
Magazinul deține și un website, prin intermediul căruia clienții își pot crea unul sau mai multe conturi, cu scopul de a explora ofertele magazinului, de a plasa comenzi, de a comunica cu angajații în legătură cu anumite nelămuriri sau eventuale sfaturi, cât și de a adăuga recenzii produselor cumpărate de aceștia.

Magazinul are două tipuri de angajați: paznici și tehnicieni. De asemenea, angajații pot asista clienții în legătură cu diferite întrebări ale acestora, atât în incinta magazinului, cât și prin intermediul website-ului.

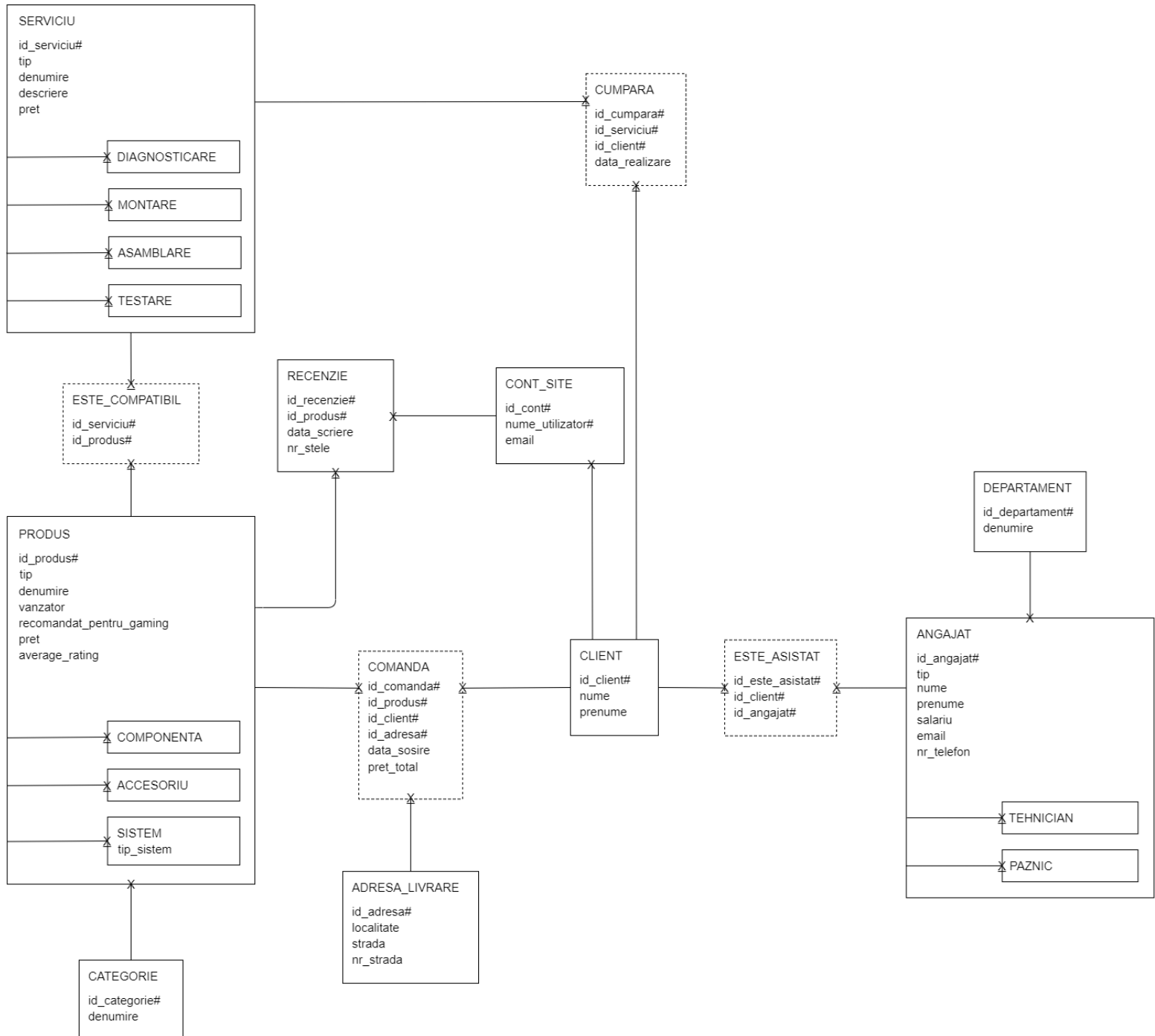
Prezentarea constrângerilor (restricții, reguli) impuse asupra modelului:

- Un client poate șterge doar recenziile scrise de acesta;
- Un client nu poate să adauge mai mult de o recenzie pentru același produs;
- Atât tehnicienii cât și paznicii lucrează într-un singur department, iar paznicii lucrează într-un singur tip de departament, anume cel de securitate;
- Clienții nu își pot șterge conturile;
- Nu există conturi cu același nume de utilizator;
- Un produs trebuie să facă parte dintr-o singură categorie;
- Orice cont trebuie să aibă specificată o adresă de e-mail;
- Orice angajat trebuie să aibă specificată o adresă de e-mail și un număr de telefon;
- Trebuie să se cunoască data sosirii și prețul total al tuturor comenzilor;
- Trebuie să se cunoască data realizării tuturor serviciilor.

## 2. Diagrama entitate-relație:



### 3. Diagrama conceptuală:



#### 4. Crearea tabelelor:

CREATE TABLE CATEGORIE

```
(id_categorie INTEGER,  
denumire VARCHAR2(40) CONSTRAINT denumire_categorie NOT NULL,  
CONSTRAINT pk_categorie PRIMARY KEY(id_categorie),  
CONSTRAINT u_denumire_categorie UNIQUE(denumire)  
);
```

CREATE TABLE PRODUS

```
(id_produs INTEGER,  
id_categorie INTEGER CONSTRAINT id_categorie_produs NOT NULL,  
tip VARCHAR(10) CONSTRAINT tip_produs NOT NULL,  
denumire VARCHAR2(200) CONSTRAINT denumire_produs NOT NULL,  
tip_sistem VARCHAR2(30),  
vanzator VARCHAR2(40) CONSTRAINT vanzator_produs NOT NULL,  
recomandat_pentru_gaming VARCHAR2(2),  
pret NUMBER CONSTRAINT pret_produs NOT NULL,  
average_rating NUMBER,  
CONSTRAINT pk_produs PRIMARY KEY(id_produs),  
CONSTRAINT fk_categorie_in_produs FOREIGN KEY(id_categorie) REFERENCES  
CATEGORIE(id_categorie) ON DELETE CASCADE  
);
```

CREATE TABLE SERVICIU

```
(id_serviciu INTEGER,  
tip VARCHAR(13) CONSTRAINT tip_serviciu NOT NULL,
```

```
denumire VARCHAR2(70) CONSTRAINT denumire_serviciu NOT NULL,  
descriere VARCHAR2(1000),  
pret NUMBER CONSTRAINT pret_serviciu NOT NULL,  
CONSTRAINT pk_serviciu PRIMARY KEY(id_serviciu),  
CONSTRAINT u_denumire_serviciu UNIQUE(denumire)  
);
```

```
CREATE TABLE ADRESA_LIVRARE
```

```
(id_adresa INTEGER,  
localitate VARCHAR2(30) CONSTRAINT localitate_adresa_livrare NOT NULL,  
strada VARCHAR2(80) CONSTRAINT strada_adresa_livrare NOT NULL,  
nr_strada INTEGER CONSTRAINT nr_strada_adresa_livrare NOT NULL,  
CONSTRAINT pk_adresa_livrare PRIMARY KEY(id_adresa)  
);
```

```
CREATE TABLE CLIENT
```

```
(id_client INTEGER,  
nume VARCHAR2(40) CONSTRAINT nume_client NOT NULL,  
prenume VARCHAR2(40) CONSTRAINT prenume_client NOT NULL,  
CONSTRAINT pk_client PRIMARY KEY(id_client)  
);
```

```
CREATE TABLE CONT_SITE
```

```
(id_cont INTEGER,  
nume_utilizator VARCHAR2(40),  
id_client INTEGER,
```

```
email VARCHAR2(40) CONSTRAINT email NOT NULL,  
CONSTRAINT pk_compus_cont_site PRIMARY KEY(id_cont, nume_utilizator),  
CONSTRAINT u_nume_utilizator UNIQUE(nume_utilizator),  
CONSTRAINT fk_client_in_cont_site FOREIGN KEY(id_client) REFERENCES CLIENT(id_client)  
ON DELETE CASCADE  
);
```

CREATE TABLE RECENZIE

```
(id_recenzie INTEGER,  
id_produs INTEGER CONSTRAINT id_produs_recenzie NOT NULL,  
nume_utilizator VARCHAR2(40) CONSTRAINT nume_utilizator_recenzie NOT NULL,  
data_scriere DATE CONSTRAINT data_scriere_recenzie NOT NULL,  
nr_stele NUMBER,  
CONSTRAINT pk_compus_recenzie PRIMARY KEY(id_recenzie, id_produs),  
CONSTRAINT fk_produs_in_recenzie FOREIGN KEY(id_produs) REFERENCES  
PRODUS(id_produs) ON DELETE CASCADE,  
CONSTRAINT fk_cont_site_in_recenzie FOREIGN KEY(nume_utilizator) REFERENCES  
CONT_SITE(nume_utilizator) ON DELETE CASCADE  
);
```

CREATE TABLE DEPARTAMENT

```
(id_departament INTEGER,  
denumire VARCHAR2(30) CONSTRAINT denumire_departament NOT NULL,  
CONSTRAINT pk_departament PRIMARY KEY(id_departament),  
CONSTRAINT u_denumire_departament UNIQUE(denumire)  
);
```

CREATE TABLE ANGAJAT

```
(id_angajat INTEGER,  
id_departament INTEGER,  
tip VARCHAR2(9) CONSTRAINT tip_angajat NOT NULL,  
nume VARCHAR2(30) CONSTRAINT nume_angajat NOT NULL,  
prenume VARCHAR2(30) CONSTRAINT prenume_angajat NOT NULL,  
salariu NUMBER CONSTRAINT salariu_angajat NOT NULL,  
email VARCHAR2(40) CONSTRAINT email_angajat NOT NULL,  
nr_telefon VARCHAR2(10) CONSTRAINT nr_telefon_angajat NOT NULL,  
CONSTRAINT pk_angajat PRIMARY KEY(id_angajat),  
CONSTRAINT fk_departament_in_angajat FOREIGN KEY(id_departament) REFERENCES  
DEPARTAMENT(id_departament) ON DELETE CASCADE,  
CONSTRAINT u_email_angajat UNIQUE(email)  
);
```

CREATE TABLE ESTE\_COMPATIBIL

```
(id_serviciu INTEGER,  
id_produs INTEGER,  
CONSTRAINT pk_compus_este_compatibil PRIMARY KEY(id_serviciu, id_produs),  
CONSTRAINT fk_serviciu_in_este_compatibil FOREIGN KEY(id_serviciu) REFERENCES  
SERVICIU(id_serviciu) ON DELETE CASCADE,  
CONSTRAINT fk_produs_in_este_compatibil FOREIGN KEY(id_produs) REFERENCES  
PRODUS(id_produs) ON DELETE CASCADE  
);
```

CREATE TABLE COMANDA

```
(id_comanda INTEGER,
```



```

    id_produs INTEGER,

    id_client INTEGER,

    id_adresa INTEGER,

    data_sosire DATE CONSTRAINT data_sosire_comanda NOT NULL,

    pret_total NUMBER CONSTRAINT pret_total_comanda NOT NULL,

    CONSTRAINT pk_compus_comanda PRIMARY KEY(id_comanda, id_produs, id_client,
id_adresa),

    CONSTRAINT fk_produs_in_comanda FOREIGN KEY(id_produs) REFERENCES
PRODUS(id_produs) ON DELETE CASCADE,

    CONSTRAINT fk_client_in_comanda FOREIGN KEY(id_client) REFERENCES CLIENT(id_client)
ON DELETE CASCADE,

    CONSTRAINT fk_adresa_livrare_in_comanda FOREIGN KEY(id_adresa) REFERENCES
ADRESA_LIVRARE(id_adresa) ON DELETE CASCADE

);

```

CREATE TABLE CUMPARA

```

    (id_cumpara INTEGER,

    id_serviciu INTEGER,

    id_client INTEGER,

    data_realizare DATE CONSTRAINT data_realizare_cumpara NOT NULL,

    CONSTRAINT pk_compus_cumpara PRIMARY KEY(id_cumpara, id_serviciu, id_client),

    CONSTRAINT fk_serviciu_in_cumpara FOREIGN KEY(id_serviciu) REFERENCES
SERVICIU(id_serviciu) ON DELETE CASCADE,

    CONSTRAINT fk_client_in_cumpara FOREIGN KEY(id_client) REFERENCES CLIENT(id_client)
ON DELETE CASCADE

);

```

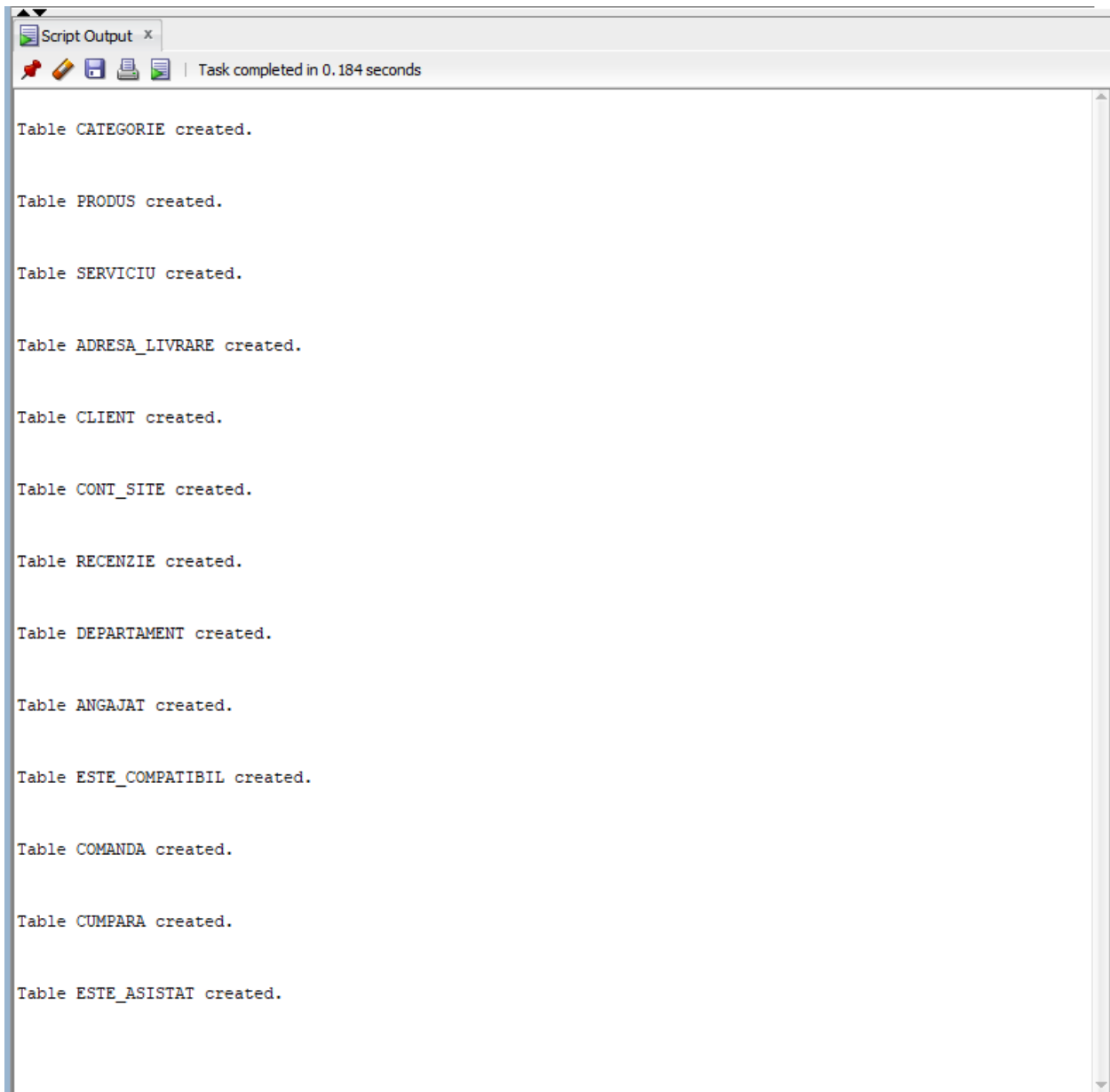
CREATE TABLE ESTE\_ASISTAT

```

    (id_este_asistat INTEGER,

```

```
id_client INTEGER,  
  
id_angajat INTEGER,  
  
CONSTRAINT pk_compus_este_asistat PRIMARY KEY(id_este_asistat, id_client, id_angajat),  
  
CONSTRAINT fk_client_in_este_asistat FOREIGN KEY(id_client) REFERENCES  
CLIENT(id_client) ON DELETE CASCADE,  
  
CONSTRAINT fk_angajat_in_este_asistat FOREIGN KEY(id_angajat) REFERENCES  
ANGAJAT(id_angajat) ON DELETE CASCADE  
  
);
```



## 5. Inserarea datelor în tabele:

-- Secventa pentru inserarea inregistrarilor in tabelul "CATEGORIE":

```
CREATE SEQUENCE SEQ_CATEGORIE
```

```
INCREMENT by 1
```

```
START WITH 0
```

```
MINVALUE -1
```

```
MAXVALUE 399
```

```
NOCYCLE;
```

```
-- DROP SEQUENCE SEQ_CATEGORIE;
```

```
INSERT INTO CATEGORIE
```

```
VALUES(SEQ_CATEGORIE.NEXTVAL, 'Placi video');
```

```
INSERT INTO CATEGORIE
```

```
VALUES(SEQ_CATEGORIE.NEXTVAL, 'Procesoare');
```

```
INSERT INTO CATEGORIE
```

```
VALUES(SEQ_CATEGORIE.NEXTVAL, 'Memorii');
```

```
INSERT INTO CATEGORIE
```

```
VALUES(SEQ_CATEGORIE.NEXTVAL, 'HDD');
```

```
INSERT INTO CATEGORIE
```

```
VALUES(SEQ_CATEGORIE.NEXTVAL, 'Adaptoare');
```

```
INSERT INTO CATEGORIE
```

```
VALUES(SEQ_CATEGORIE.NEXTVAL, 'PC-uri');
```

```
INSERT INTO CATEGORIE
```

```
VALUES(SEQ_CATEGORIE.NEXTVAL, 'Laptop-uri');
```

```
-- ROLLBACK;
```

```
-- COMMIT;
```

```
-- Secventa pentru inserarea inregistrarilor in tabelul "PRODUS":
```

```
CREATE SEQUENCE SEQ_PRODUS
```

```
INCREMENT by 5
```

```
START WITH 0
```

```
MINVALUE -1
```

```
NOCYCLE;
```

```
-- DROP SEQUENCE SEQ_PRODUS;
```

```
INSERT INTO PRODUS
```

```
VALUES(SEQ_PRODUS.NEXTVAL, 1, 'Componenta', 'Placa video Palit GeForce RTX 3090  
GamingPro 24GB GDDR6X 384-bit', NULL, 'GIGABYTE', 'Da', 14999.99, 5);
```

```
INSERT INTO PRODUS
```

```
VALUES(SEQ_PRODUS.NEXTVAL, 2, 'Componenta', 'Procesor AMD Ryzen 5 3600 3.6GHz box',  
NULL, 'AMD', 'Da', 999.99, 5);
```

```
INSERT INTO PRODUS
```

```
VALUES(SEQ_PRODUS.NEXTVAL, 5, 'Accesoriu', 'Adaptor Gembird 1x HDMI 1.4 Male - 1x VGA  
Female', NULL, 'Gembird', 'Nu', 35.99, NULL);
```

```
INSERT INTO PRODUS
```

```
VALUES(SEQ_PRODUS.NEXTVAL, 6, 'Sistem', 'PC Gaming Raptor5, Intel i5-9400F 2.9GHz Coffee  
Lake, 16GB DDR4, 960GB SSD, RX 5600 XT 6GB GDDR6, Iluminare RGB', 'PC', 'Raptor5', 'Da',  
4799.99, 4);
```

```
INSERT INTO PRODUS
```

```
VALUES(SEQ_PRODUS.NEXTVAL, 7, 'Sistem', 'Laptop Lenovo 15.6" ThinkPad E15 Gen 2, FHD,  
Procesor Intel® Core™ i5-1135G7 (8M Cache, up to 4.20 GHz), 8GB DDR4, 256GB SSD, Intel Iris  
Xe, No OS, Black', 'Lenovo', 'Laptop', 'Nu', 3398.99, 5);
```

```
INSERT INTO PRODUS
```

```
VALUES(SEQ_PRODUS.NEXTVAL, 1, 'Componenta', 'Placa video PowerColor Radeon RX 6900 XT Red Devil 16GB GDDR6 256-bit', NULL, 'Red Devil', 'Da', 11999.99, NULL);
```

```
INSERT INTO PRODUS
```

```
VALUES(SEQ_PRODUS.NEXTVAL, 3, 'Componenta', 'Memorie Corsair Vengeance LPX Black 16GB DDR4 3200MHz CL16 Dual Channel Kit', NULL, 'Corsair', 'Da', 514.99, NULL);
```

```
INSERT INTO PRODUS
```

```
VALUES(SEQ_PRODUS.NEXTVAL, 4, 'Componenta', 'Hard disk Seagate BarraCuda 2TB SATA-III 7200RPM 256MB', NULL, 'Seagate', 'Da', 262.99, NULL);
```

```
INSERT INTO PRODUS
```

```
VALUES(SEQ_PRODUS.NEXTVAL, 5, 'Accesoriu', 'Adaptor Gembird 1x HDMI 1.4 Male - 1x VGA Female', NULL, 'Vanzator 1', 'Nu', 47.99, 2);
```

```
-- ROLLBACK;
```

```
-- COMMIT;
```

```
-- Secventa pentru inserarea inregistrarilor in tabelul "SERVICIU":
```

```
CREATE SEQUENCE SEQ_SERVICIU
```

```
INCREMENT by 1
```

```
START WITH 0
```

```
MINVALUE -1
```

```
NOCYCLE;
```

```
-- DROP SEQUENCE SEQ_SERVICIU;
```

```
INSERT INTO SERVICIU
```

```
VALUES(SEQ_SERVICIU.NEXTVAL, 'Asamblare', 'Serviciu Asamblare Standard',
```

```
'Asamblare sistem de calcul desktop de catre un specialist calificat.
```

```
Instalare sistem de operare test. Verificare compatibilitate componente.
```

```
Preinstalare sisteme de operare si aplicatii (daca s-au achizitionat)',
```

159);

INSERT INTO SERVICIU

VALUES(SEQ\_SERVICIU.NEXTVAL, 'Testare', 'Serviciu Testare Produs',

'Verificare functionalitate produs.

Aplicabil oricarui produs comercializat.',

135.99);

INSERT INTO SERVICIU

VALUES(SEQ\_SERVICIU.NEXTVAL, 'Montare', 'Serviciu Montare Componenta',

'Montarea diverselor componente in sistemul de calcul: placa video,

procesor, RAM, HDD, cooler pentru procesor, etc.',

59.99);

INSERT INTO SERVICIU

VALUES(SEQ\_SERVICIU.NEXTVAL, 'Diagnosticare', 'Serviciu Diagnosticare Produs',

'Identificarea problemelor de functionare si furnizarea de recomandari.',

119.99);

INSERT INTO SERVICIU

VALUES(SEQ\_SERVICIU.NEXTVAL, 'Asamblare', 'Serviciu Asamblare Premium',

'Asamblare si testare sistem de calcul desktop de catre un specialist calificat.

Cooling management. Wire management. Instalare sistem de operare test.

Verificare compatibilitate componente. Testare sistem (rulare in conditii de stres 24 ore).

Generarea unui raport operatiune asamblare premium. Updateuri la ultimele versiuni stabile (BIOS placa de baza).

Preinstalare sisteme de operare si aplicatii (daca s-au achizitionat).',

249.99);

-- ROLLBACK;

-- COMMIT;

```
INSERT INTO ADRESA_LIVRARE
VALUES(1, 'Bucuresti', 'Mihai Eminescu', 1);

INSERT INTO ADRESA_LIVRARE
VALUES(2, 'Iasi', 'Morii', 12);

INSERT INTO ADRESA_LIVRARE
VALUES(3, 'Bucuresti', 'Unirii', 34);

INSERT INTO ADRESA_LIVRARE
VALUES(4, 'Constanta', 'Stejarul mic', 2);

INSERT INTO ADRESA_LIVRARE
VALUES(5, 'Bucuresti', 'Eroilor', 10);

INSERT INTO ADRESA_LIVRARE
VALUES(6, 'Bucuresti', 'Mihail Kogalniceanu', 8);

-- ROLLBACK;

-- COMMIT;
```

```
INSERT INTO CLIENT
VALUES(1, 'Aurel', 'Popescu-Mihai');

INSERT INTO CLIENT
VALUES(2, 'Carstea', 'Darian-Stefan');

INSERT INTO CLIENT
VALUES(3, 'Castan', 'Mirela');

INSERT INTO CLIENT
VALUES(4, 'Popa', 'Marcel-Radu');

INSERT INTO CLIENT
```

```
VALUES(5, 'Manole', 'Iuliana-Elena');
```

```
INSERT INTO CLIENT
```

```
VALUES(6, 'Florian', 'Andrei-Cosmin');
```

```
INSERT INTO CLIENT
```

```
VALUES(7, 'Aurel', 'Marcel-Stoian');
```

```
-- ROLLBACK;
```

```
-- COMMIT;
```

```
INSERT INTO CONT_SITE
```

```
VALUES(1, 'Utilizator Mihai', NULL, 'uzmihai2005@gmail.com');
```

```
INSERT INTO CONT_SITE
```

```
VALUES(2, 'Darian Stefan', 2, 'carsteastef@yahoo.com');
```

```
INSERT INTO CONT_SITE
```

```
VALUES(3, 'Mirela Castan', 3, 'mirela_c@gmail.com');
```

```
INSERT INTO CONT_SITE
```

```
VALUES(4, 'ANTON PAVEL MIHAI', NULL, 'apav_mihai@gmail.com');
```

```
INSERT INTO CONT_SITE
```

```
VALUES(5, 'Elena 2008', 5, 'elenamanole@yahoo.com');
```

```
-- ROLLBACK;
```

```
-- COMMIT;
```

```
INSERT INTO RECENZIE
```

```
VALUES(1, 5, 'Utilizator Mihai', TO_DATE('15-01-2021','dd-mm-yyyy'), 3);
```

```
INSERT INTO RECENZIE
```



```
VALUES(2, 10, 'Utilizator Mihai', TO_DATE('15-01-2021','dd-mm-yyyy'), 5);
INSERT INTO RECENZIE
VALUES(3, 20, 'Darian Stefan', TO_DATE('23-05-2020','dd-mm-yyyy'), 4);
INSERT INTO RECENZIE
VALUES(4, 25, 'Mirela Castan', TO_DATE('02-08-2019','dd-mm-yyyy'), 5);
INSERT INTO RECENZIE
VALUES(5, 25, 'Elena 2008', TO_DATE('11-09-2020','dd-mm-yyyy'), 5);
INSERT INTO RECENZIE
VALUES(6, 15, 'Darian Stefan', TO_DATE('15-02-2020','dd-mm-yyyy'), 3);
INSERT INTO RECENZIE
VALUES(7, 15, 'Utilizator Mihai', TO_DATE('14-07-2020','dd-mm-yyyy'), 3);
INSERT INTO RECENZIE
VALUES(8, 35, 'Utilizator Mihai', TO_DATE('06-09-2021','dd-mm-yyyy'), 4);

-- ROLLBACK;

-- COMMIT;
```

```
INSERT INTO DEPARTAMENT
VALUES(1, 'Relationare clienti');
INSERT INTO DEPARTAMENT
VALUES(2, 'Prestare servicii');
INSERT INTO DEPARTAMENT
VALUES(3, 'Intretinere website');
INSERT INTO DEPARTAMENT
VALUES(4, 'Inventar produse');
INSERT INTO DEPARTAMENT
```

```
VALUES(5, 'Securitate');
```

```
-- ROLLBACK;
```

```
-- COMMIT;
```

```
INSERT INTO ANGAJAT
```

```
VALUES(1, 2, 'Tehnician', 'Aurel', 'Tudor-Mihai', 7000, 'aurel_mihai@gmail.com', '0793333333');
```

```
INSERT INTO ANGAJAT
```

```
VALUES(2, 1, 'Tehnician', 'Mihail', 'Maria-Andreea', 4500, 'mariaandreea09@yahoo.com',  
'0791111111');
```

```
INSERT INTO ANGAJAT
```

```
VALUES(3, 1, 'Tehnician', 'Vladoi', 'Rares', 4000, 'raresVld@gmail.com', '0749999999');
```

```
INSERT INTO ANGAJAT
```

```
VALUES(4, 4, 'Tehnician', 'Grigorescu', 'Stefan', 4500, 'grgrsc_stf@yahoo.com', '0755555555');
```

```
INSERT INTO ANGAJAT
```

```
VALUES(5, 5, 'Paznic', 'Marian', 'Alexandru', 3300, 'marianalex3@yahoo.com', '0788888885');
```

```
-- ROLLBACK;
```

```
-- COMMIT;
```

```
INSERT INTO ESTE_COMPATIBIL
```

```
VALUES(2, 5);
```

```
INSERT INTO ESTE_COMPATIBIL
```

```
VALUES(2, 10);
```

```
INSERT INTO ESTE_COMPATIBIL
```

```
VALUES(2, 20);
```

```
INSERT INTO ESTE_COMPATIBIL
```

```
VALUES(2, 25);  
INSERT INTO ESTE_COMPATIBIL  
VALUES(2, 30);  
INSERT INTO ESTE_COMPATIBIL  
VALUES(4, 5);  
INSERT INTO ESTE_COMPATIBIL  
VALUES(4, 20);  
INSERT INTO ESTE_COMPATIBIL  
VALUES(4, 25);  
INSERT INTO ESTE_COMPATIBIL  
VALUES(3, 5);  
INSERT INTO ESTE_COMPATIBIL  
VALUES(3, 10);  
INSERT INTO ESTE_COMPATIBIL  
VALUES(3, 30);  
INSERT INTO ESTE_COMPATIBIL  
VALUES(1, 10);
```

```
-- ROLLBACK;
```

```
-- COMMIT;
```

```
INSERT INTO COMANDA  
VALUES(1, 5, 1, 1, TO_DATE('20-09-2020','dd-mm-yyyy'), 14999.99);  
INSERT INTO COMANDA  
VALUES(11, 5, 1, 1, TO_DATE('20-09-2020','dd-mm-yyyy'), 14999.99);  
INSERT INTO COMANDA
```

```
VALUES(2, 10, 1, 1, TO_DATE('20-09-2020','dd-mm-yyyy'), 999.99);
INSERT INTO COMANDA
VALUES(3, 20, 2, 3, TO_DATE('07-12-2019','dd-mm-yyyy'), 4799.99);
INSERT INTO COMANDA
VALUES(4, 25, 3, 4, TO_DATE('16-02-2019','dd-mm-yyyy'), 3398.99);
INSERT INTO COMANDA
VALUES(5, 25, 5, 2, TO_DATE('11-09-2020','dd-mm-yyyy'), 3398.99);
INSERT INTO COMANDA
VALUES(6, 30, 4, 5, TO_DATE('07-02-2021','dd-mm-yyyy'), 11999.99);
INSERT INTO COMANDA
VALUES(7, 15, 1, 5, TO_DATE('07-02-2021','dd-mm-yyyy'), 71.98);
INSERT INTO COMANDA
VALUES(8, 10, 4, 3, TO_DATE('07-02-2021','dd-mm-yyyy'), 999.99);
INSERT INTO COMANDA
VALUES(9, 15, 2, 3, TO_DATE('10-12-2019','dd-mm-yyyy'), 35.99);
INSERT INTO COMANDA
VALUES(10, 20, 6, 6, TO_DATE('20-05-2021','dd-mm-yyyy'), 4799.99);

-- ROLLBACK;

-- COMMIT;

INSERT INTO CUMPARA
VALUES(1, 2, 1, TO_DATE('20-09-2020','dd-mm-yyyy'));
INSERT INTO CUMPARA
VALUES(2, 5, 1, TO_DATE('20-09-2020','dd-mm-yyyy'));
INSERT INTO CUMPARA
```

```
VALUES(3, 2, 6, TO_DATE('14-04-2020','dd-mm-yyyy'));  
INSERT INTO CUMPARA  
VALUES(4, 3, 6, TO_DATE('14-04-2020','dd-mm-yyyy'));  
INSERT INTO CUMPARA  
VALUES(5, 2, 3, TO_DATE('16-02-2019','dd-mm-yyyy'));  
INSERT INTO CUMPARA  
VALUES(6, 4, 3, TO_DATE('16-02-2019','dd-mm-yyyy'));  
INSERT INTO CUMPARA  
VALUES(7, 2, 5, TO_DATE('11-09-2020','dd-mm-yyyy'));  
INSERT INTO CUMPARA  
VALUES(8, 4, 5, TO_DATE('11-09-2020','dd-mm-yyyy'));  
INSERT INTO CUMPARA  
VALUES(9, 2, 2, TO_DATE('07-12-2019','dd-mm-yyyy'));  
INSERT INTO CUMPARA  
VALUES(10, 2, 4, TO_DATE('07-02-2021','dd-mm-yyyy'));  
INSERT INTO CUMPARA  
VALUES(11, 3, 4, TO_DATE('08-02-2021','dd-mm-yyyy'));  
INSERT INTO CUMPARA  
VALUES(12, 2, 1, TO_DATE('20-10-2021','dd-mm-yyyy'));
```

```
-- ROLLBACK;
```

```
-- COMMIT;
```

```
INSERT INTO ESTE_ASISTAT
```

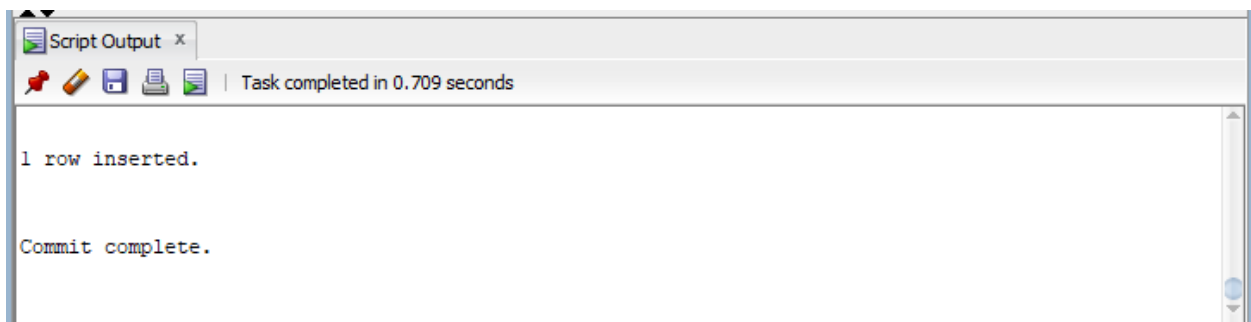
```
VALUES(1, 1, 1);
```

```
INSERT INTO ESTE_ASISTAT
```

```
VALUES(2, 1, 2);  
INSERT INTO ESTE_ASISTAT  
VALUES(3, 2, 1);  
INSERT INTO ESTE_ASISTAT  
VALUES(4, 2, 2);  
INSERT INTO ESTE_ASISTAT  
VALUES(5, 3, 1);  
INSERT INTO ESTE_ASISTAT  
VALUES(6, 3, 2);  
INSERT INTO ESTE_ASISTAT  
VALUES(7, 5, 1);  
INSERT INTO ESTE_ASISTAT  
VALUES(8, 5, 2);  
INSERT INTO ESTE_ASISTAT  
VALUES(9, 4, 3);  
INSERT INTO ESTE_ASISTAT  
VALUES(10, 4, 2);
```

```
-- ROLLBACK;
```

```
COMMIT;
```



Alex_Proiect_SGBD : SELECT * FROM CATEGORIE	
Script Output x	Query Re... x
SQL   All Rows Fetched: 7 in 0.004 seconds	
ID_CATEGORIE	DENUMIRE
1	1 Placi video
2	2 Procesoare
3	3 Memorii
4	4 HDD
5	5 Adaptoare
6	6 PC-uri
7	7 Laptop-uri

Alex_Proiect_SGBD : SELECT * FROM PRODUS			
Script Output x	Query Re... x		
SQL   All Rows Fetched: 8 in 0.003 seconds			
ID_PRODUS	ID_CATEGORIE	TIP	DENUMIRE
1	5	1 Componenta	Placa video Palit GeForce RTX 3090 GamingPro 24GB GDDR6X 3
2	10	2 Componenta	Procesor AMD Ryzen 5 3600 3.6GHz box
3	15	5 Accesoriu	Adaptor Gembird 1x HDMI 1.4 Male - 1x VGA Female
4	20	6 Sistem	PC Gaming Raptor5, Intel i5-9400F 2.9GHz Coffee Lake, 16GB
5	25	7 Sistem	Laptop Lenovo 15.6' ThinkPad E15 Gen 2, FHD, Procesor Inte
6	30	1 Componenta	Placa video PowerColor Radeon RX 6900 XT Red Devil 16GB GD
7	35	3 Componenta	Memorie Corsair Vengeance LPX Black 16GB DDR4 3200MHz CL16
8	40	4 Componenta	Hard disk Seagate BarraCuda 2TB SATA-III 7200RPM 256MB

Alex_Proiect_SGBD : SELECT * FROM SERVICIU			
Script Output x	Query Re... x		
SQL   All Rows Fetched: 5 in 0.003 seconds			
ID_SERVICIU	TIP	DENUMIRE	DESCRIERE
1	1 Asamblare	Serviciu Asamblare Standard	Asamblare sistem de calcul desktop de
2	2 Testare	Serviciu Testare Produs	Verificare functionalitate produs. Apl
3	3 Montare	Serviciu Montare Componenta	Montarea diverselor componente in sist
4	4 Diagnosticare	Serviciu Diagnosticare Produs	Identificarea problemelor de functiona
5	5 Asamblare	Serviciu Asamblare Premium	Asamblare si testare sistem de calcul









## 6. Subprogram stocat care utilizează două tipuri de colecții:

-- Cerinta 6.:

-- Enunt:

-- Pentru toti clientii care au acelasi nume ca un nume introdus de la tastatura,  
-- sa se afiseze id-ul, numele complet, si pretul tuturor produselor care au fost comandate  
-- de catre acestia, in ordine alfabetica a numelor produselor. Pentru fiecare produs  
-- al fiecarui client se va afisa un numar de ordine. In cazul in care un anumit client  
-- nu a comandat niciun produs, se va afisa un mesaj corespunzator.  
-- Sa se trateze exceptii care pot aparea.

CREATE OR REPLACE PROCEDURE EX6\_PROC

(v\_nume IN CLIENT.nume%TYPE)

IS

v\_prenume CLIENT.prenume%TYPE;

v\_id CLIENT.id\_client%TYPE;

TYPE tab\_imb\_id\_cli IS TABLE OF CLIENT.id\_client%TYPE;

t\_id\_cli tab\_imb\_id\_cli;

v\_nr\_clienti NUMBER;

TYPE tab\_imb\_id\_prod IS TABLE OF PRODUS.id\_produs%TYPE;

t\_id\_prod tab\_imb\_id\_prod;

TYPE tab\_ind\_nume\_prod IS TABLE OF PRODUS.denumire%TYPE

```

INDEX BY PLS_INTEGER;

t_nume_prod tab_ind_nume_prod;

TYPE vector_pret_prod IS VARRAY(2000) OF PRODUS.pret%TYPE;
vect_pret_prod vector_pret_prod := vector_pret_prod();

v_contor NUMBER;

BEGIN

SELECT COUNT(ume)
INTO v_nr_clienti
FROM CLIENT
WHERE UPPER(ume) = UPPER(v_ume);

IF v_nr_clienti = 0 THEN
    RAISE NO_DATA_FOUND;
ELSE
    SELECT id_client
    BULK COLLECT INTO t_id_cli
    FROM CLIENT
    WHERE UPPER(ume) = UPPER(v_ume);

    FOR i IN t_id_cli.FIRST..t_id_cli.LAST LOOP
        SELECT DISTINCT p.id_produs, denumire, pret
        BULK COLLECT INTO t_id_prod, t_nume_prod, vect_pret_prod
        FROM PRODUS p JOIN COMANDA co ON p.id_produs = co.id_produs
        JOIN CLIENT cl ON co.id_client = cl.id_client
    
```

```
WHERE cl.id_client = t_id_cli(i)
```

```
ORDER BY 2;
```

```
SELECT prenume
```

```
INTO v_prenume
```

```
FROM CLIENT
```

```
WHERE id_client = t_id_cli(i);
```

```
IF t_id_prod.COUNT = 0 THEN
```

```
    DBMS_OUTPUT.PUT_LINE('-----');
```

```
    DBMS_OUTPUT.PUT_LINE('Clientul cu id-ul ' || t_id_cli(i) || ', pe nume ' || v_nume  
|| ' ' || v_prenume || ', nu a comandat niciun produs!');
```

```
    DBMS_OUTPUT.PUT_LINE('-----');
```

```
ELSE
```

```
    v_contor := 1;
```

```
    DBMS_OUTPUT.PUT_LINE('-----');
```

```
    DBMS_OUTPUT.PUT_LINE('Clientul cu id-ul ' || t_id_cli(i) || ', pe nume ' || v_nume  
|| ' ' || v_prenume || ', a achizitionat produsele: ');
```

```
    DBMS_OUTPUT.NEW_LINE;
```

```
FOR i IN t_id_prod.FIRST..t_id_prod.LAST LOOP
```

```
    DBMS_OUTPUT.PUT_LINE(v_contor || '.');
```

```
    DBMS_OUTPUT.PUT_LINE('Id produs: ' || t_id_prod(i));
```

```
    DBMS_OUTPUT.PUT_LINE('Denumire produs: ' || t_nume_prod(i));
```

```
    DBMS_OUTPUT.PUT_LINE('Pret produs: ' || vect_pret_prod(i));
```

```

        IF i <> t_id_prod.LAST THEN
            DBMS_OUTPUT.NEW_LINE;
        END IF;

        v_contor := v_contor + 1;
    END LOOP;

    DBMS_OUTPUT.PUT_LINE('-----');
END IF;

    DBMS_OUTPUT.NEW_LINE;
END LOOP;
END IF;
EXCEPTION
    WHEN NO_DATA_FOUND THEN
        DBMS_OUTPUT.PUT_LINE('Nu exista niciun client cu numele ' || v_numa || '!');
    WHEN OTHERS THEN
        DBMS_OUTPUT.PUT_LINE('A aparut alta eroare!');
END EX6_PROC;
/

DECLARE
    v_numa CLIENT.numa%TYPE := '&p_numa';
BEGIN
    EX6_PROC(v_numa);
END;
```

Worksheet Query Builder

```

451 -- Cerinta 6.:
452
453 -- Enunt:
454 -- Pentru toti clientii care au acelasi nume ca un nume introdus de la tastatura,
455 -- sa se afiseze id-ul, numele complet, si pretul tuturor produselor care au fost comandate
456 -- de catre acestia, in ordine alfabetica a numelor produselor. Pentru fiecare produs
457 -- al fiecarui client se va afisa un numar de ordine. In cazul in care un anumit client
458 -- nu a comandat niciun produs, se va afisa un mesaj corespunzator.
459 -- Sa se trateze exceptiile care pot aparea.
460
461 CREATE OR REPLACE PROCEDURE EX6_PROC
462 (v_nume IN CLIENT.nume%TYPE)
463 IS
464     v_prenume CLIENT.prenume%TYPE;
465     v_id CLIENT.id_client%TYPE;
466
467     TYPE tab_imb_id_cli IS TABLE OF CLIENT.id_client%TYPE;
468     t_id_cli tab_imb_id_cli;
469
470     v_nr_clienti NUMBER;
471
472     TYPE tab_imb_id_prod IS TABLE OF PRODUS.id_produs%TYPE;
473     t_id_prod tab_imb_id_prod;
474
475     TYPE tab_ind_nume_prod IS TABLE OF PRODUS.denumire%TYPE
476     INDEX BY PLS_INTEGER;
477     t_nume_prod tab_ind_nume_prod;
478
479     TYPE vector_pret_prod IS VARRAY(2000) OF PRODUS.pret%TYPE;
480     vect_pret_prod vector_pret_prod := vector_pret_prod();
481
482     v_contor NUMBER;
483 BEGIN
484     SELECT COUNT(nume)
485     INTO v_nr_clienti
486     FROM CLIENT
487     WHERE UPPER(nume) = UPPER(v_nume);
488
489     IF v_nr_clienti = 0 THEN

```

Script Output x

Task completed in 0.023 seconds

Procedure EX6\_PROC compiled

Worksheet Query Builder

```

517 DBMS_OUTPUT.PUT_LINE('-----');
518 DBMS_OUTPUT.PUT_LINE('Clien');
519 DBMS_OUTPUT.NEW_LINE;
520
521 FOR i IN t_id_prod.FIRST..
522     DBMS_OUTPUT.PUT_LINE(v
523     DBMS_OUTPUT.PUT_LINE('
524     DBMS_OUTPUT.PUT_LINE('
525     DBMS_OUTPUT.PUT_LINE('
526
527     IF i <> t_id_prod.LAST
528     DBMS_OUTPUT.NEW_LI
529     END IF;
530
531     v_contor := v_contor +
532     END LOOP;
533
534     DBMS_OUTPUT.PUT_LINE('-----');
535     END IF;
536
537     DBMS_OUTPUT.NEW_LINE;
538     END LOOP;
539     END IF;
540 EXCEPTION
541     WHEN NO_DATA_FOUND THEN
542         DBMS_OUTPUT.PUT_LINE('Nu exista ni
543     WHEN OTHERS THEN
544         DBMS_OUTPUT.PUT_LINE('A aparut alt
545     END EX6_PROC;
546 /
547
548 DECLARE
549     v_nume CLIENT.nume%TYPE := 'sp_nume';
550 BEGIN
551     EX6_PROC(v_nume);
552 END;
553 /

```

Script Output x

Task completed in 1.391 seconds

EX6\_PROC(v\_nume);

END;

PL/SQL procedure successfully completed.

Alex\_Proiect\_SGBD x

Clientul cu id-ul 1, pe nume Aurel Popescu-Mihai, a achizitionat produsele:

- Id produs: 15  
Denumire produs: Adaptor Gembird lx HDMI 1.4 Male - lx VGA Female  
Pret produs: 35.99
- Id produs: 5  
Denumire produs: Placa video Palit GeForce RTX 3090 GamingPro 24GB GDDR6X 384-bit  
Pret produs: 14999.99
- Id produs: 10  
Denumire produs: Procesor AMD Ryzen 5 3600 3.6GHz box  
Pret produs: 999.99

Clientul cu id-ul 7, pe nume Aurel Marcel-Stoian, nu a comandat niciun produs!

Alex\_Proiect\_SGBD

Worksheet Query Builder

```
518 DBMS_OUTPUT.PUT_LINE('Clientul cu id-ul ' || t_id_cli(i) || ');
519 DBMS_OUTPUT.NEW_LINE;
520
521 FOR i IN t_id_prod.FIRST..t_id_prod.LAST LOOP
522     DBMS_OUTPUT.PUT_LINE(v_contor || '.');
523     DBMS_OUTPUT.PUT_LINE('Id produs: ' || t_id_prod(i));
524     DBMS_OUTPUT.PUT_LINE('Denumire produs: ' || t_nume_prod(i));
525     DBMS_OUTPUT.PUT_LINE('Pret produs: ' || vect_pret_prod(i));
526
527     IF i <> t_id_prod.LAST THEN
528         DBMS_OUTPUT.NEW_LINE;
529     END IF;
530
531     v_contor := v_contor + 1;
532 END LOOP;
533
534 DBMS_OUTPUT.PUT_LINE('-----');
535 END IF;
536
537 DBMS_OUTPUT.NEW_LINE;
538 END LOOP;
539 END IF;
540 EXCEPTION
541     WHEN NO_DATA_FOUND THEN
542         DBMS_OUTPUT.PUT_LINE('Nu exista niciun client cu numele ' || v_nume);
543     WHEN OTHERS THEN
544         DBMS_OUTPUT.PUT_LINE('A aparut alta eroare!');
545 END EX6_PROC;
546 /
547
548 DECLARE
549     v_nume CLIENT.nume%TYPE := 'sp_nume';
550 BEGIN
551     EX6_PROC(v_nume);
552 END;
553 /
554
```

Alex\_Proiect\_SGBD x

Nu exista niciun client cu numele dhgfvbdfbh!

Script Output x

Task completed in 2.882 seconds

EX6\_PROC(v\_nume);  
END;  
  
PL/SQL procedure successfully completed.



## 7. Subprogram stocat care utilizează un tip de cursor:

-- Cerinta 7.:

-- Enunt:

-- Prin intermediul unei functii stocate, sa se afiseze, pentru fiecare angajat care a  
-- asistat clienti, numele complet al acestuia (nume + prenume), alaturi de lista numelor  
-- complete a clientilor asistati, ordonand rezultatele alfabetic dupa numele angajatilor.  
-- Tot prin intermediul functiei stocate, se va returna numarul de angajati care au  
-- asistat clienti, urmand sa se afiseze mesajul "Personal inactiv!" in cazul in care  
-- numarul este mai mic decat jumatate din numarul total de angajati, respectiv mesajul  
-- "Personal activ!" in caz contrar, si se va obtine (printr-un parametru de tip OUT) si  
-- afisa numarul de clienti asistati de catre un angajat al carui cod este introdus de  
-- la tastatura.  
-- Sa se trateze exceptii care pot aparea.  
-- Rezolvati problema folosind ciclu cursoare cu subcereri, stiind ca nu este permisa  
-- folosirea colectiilor de date.

CREATE OR REPLACE FUNCTION EX7\_FUNC

(p\_id\_ang IN ANGAJAT.id\_angajat%TYPE,

p\_nr\_cli\_asist\_ang OUT NUMBER)

RETURN NUMBER

IS

v\_nr\_ang\_activi NUMBER;

v\_nr\_cli\_asist\_ang NUMBER;

BEGIN

p\_nr\_cli\_asist\_ang := 0;

FOR i IN (SELECT id\_angajat, nume || ' ' || prenume nume\_ang

FROM ANGAJAT a

ORDER BY 2) LOOP

SELECT COUNT(DISTINCT nume || ' ' || prenume)

INTO v\_nr\_cli\_asist\_ang

FROM CLIENT c JOIN ESTE\_ASISTAT ea ON c.id\_client = ea.id\_client

WHERE ea.id\_angajat = i.id\_angajat;

IF v\_nr\_cli\_asist\_ang > 0 THEN

v\_nr\_ang\_activi := v\_nr\_ang\_activi + 1;

DBMS\_OUTPUT.PUT\_LINE('-----');

DBMS\_OUTPUT.PUT\_LINE('Id angajat: ' || i.id\_angajat);

DBMS\_OUTPUT.PUT\_LINE('Nume angajat: ' || i.nume\_ang);

DBMS\_OUTPUT.NEW\_LINE;

DBMS\_OUTPUT.PUT\_LINE('Clienti asistati de catre angajat: ');

FOR j IN (SELECT DISTINCT nume || ' ' || prenume nume\_cli

FROM CLIENT c JOIN ESTE\_ASISTAT ea ON c.id\_client = ea.id\_client

```
WHERE ea.id_angajat = i.id_angajat) LOOP
```

```
DBMS_OUTPUT.PUT_LINE(j.num_ecli);
```

```
END LOOP;
```

```
DBMS_OUTPUT.PUT_LINE('-----');
```

```
DBMS_OUTPUT.NEW_LINE;
```

```
END IF;
```

```
IF i.id_angajat = p_id_ang THEN
```

```
    p_nr_ecli_asist_ang := v_nr_ecli_asist_ang;
```

```
END IF;
```

```
END LOOP;
```

```
RETURN v_nr_ang_activi;
```

```
EXCEPTION
```

```
    WHEN OTHERS THEN
```

```
        DBMS_OUTPUT.PUT_LINE('A aparut o eroare necunoscuta!');
```

```
END EX7_FUNC;
```

```
/
```

```
DECLARE
```

```
    v_id_ang ANGAJAT.id_angajat%TYPE := &p_id;
```

```
    v_nr_ecli_asist_ang NUMBER;
```

```

v_nr_ang NUMBER;

BEGIN

SELECT COUNT(id_angajat)

INTO v_nr_ang

FROM ANGAJAT;


SELECT id_angajat

INTO v_id_ang

FROM ANGAJAT

WHERE id_angajat = v_id_ang;


IF EX7_FUNC(v_id_ang, v_nr_cli_asist_ang) < v_nr_ang/2 THEN

    DBMS_OUTPUT.PUT_LINE('Status personal:');

    DBMS_OUTPUT.PUT_LINE('Personal inactiv!');

ELSE

    DBMS_OUTPUT.PUT_LINE('Status personal:');

    DBMS_OUTPUT.PUT_LINE('Personal activ!');

END IF;


DBMS_OUTPUT.NEW_LINE;


DBMS_OUTPUT.PUT_LINE('Numarul de clienti asistati de catre angajatul cu id-ul ' || v_id_ang
|| ': ' || v_nr_cli_asist_ang || '.');

```

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

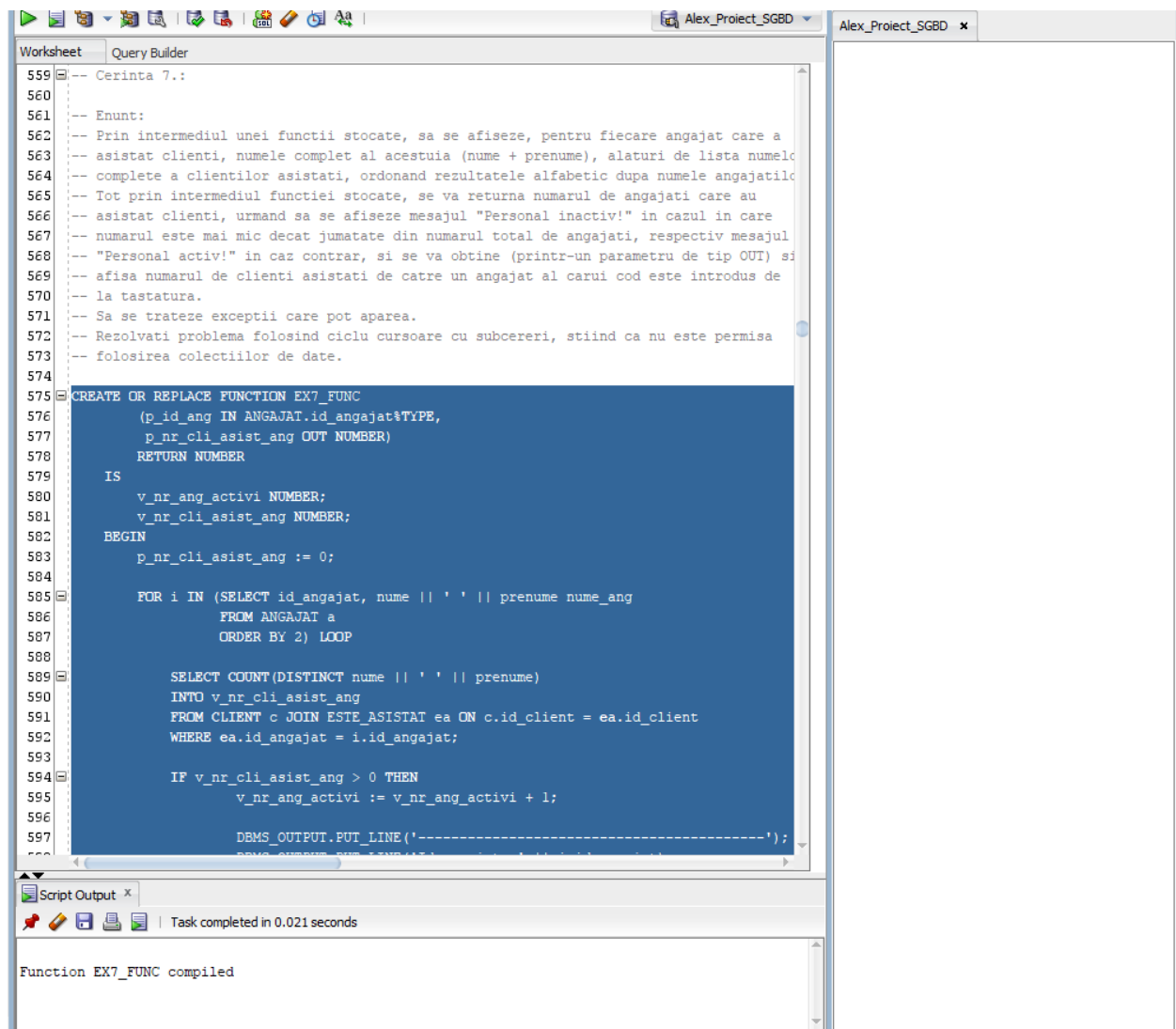
DBMS\_OUTPUT.PUT\_LINE('Nu exista niciun angajat cu id-ul ' || v\_id\_ang || '!');

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE('A aparut alta eroare!');

END;

/



```
559 -- Cerinta 7.:
560
561 -- Enunt:
562 -- Prin intermediul unei functii stocate, sa se afiseze, pentru fiecare angajat care a
563 -- asistat clienti, numele complet al acestuia (nume + prenume), alaturi de lista numelor
564 -- complete a clientilor asistati, ordonand rezultatele alfabetic dupa numele angajatilor.
565 -- Tot prin intermediul functiei stocate, se va returna numarul de angajati care au
566 -- asistat clienti, urmand sa se afiseze mesajul "Personal inactiv!" in cazul in care
567 -- numarul este mai mic decat jumatate din numarul total de angajati, respectiv mesajul
568 -- "Personal activ!" in caz contrar, si se va obtine (printr-un parametru de tip OUT) si
569 -- afisa numarul de clienti asistati de catre un angajat al carui cod este introdus de
570 -- la tastatura.
571 -- Sa se trateze exceptiile care pot aparea.
572 -- Rezolvati problema folosind ciclul cursorilor cu subcereri, stiind ca nu este permisa
573 -- folosirea colectiilor de date.
574
575 CREATE OR REPLACE FUNCTION EX7_FUNC
576 (p_id_ang IN ANGAJAT.id_angajat%TYPE,
577  p_nr_cli_asist_ang OUT NUMBER)
578 RETURN NUMBER
579 IS
580  v_nr_ang_activi NUMBER;
581  v_nr_cli_asist_ang NUMBER;
582 BEGIN
583  p_nr_cli_asist_ang := 0;
584
585  FOR i IN (SELECT id_angajat, nume || ' ' || prenume nume_ang
586            FROM ANGAJAT a
587            ORDER BY 2) LOOP
588
589      SELECT COUNT(DISTINCT nume || ' ' || prenume)
590      INTO v_nr_cli_asist_ang
591      FROM CLIENT c JOIN ESTE_ASISTAT ea ON c.id_client = ea.id_client
592      WHERE ea.id_angajat = i.id_angajat;
593
594      IF v_nr_cli_asist_ang > 0 THEN
595        v_nr_ang_activi := v_nr_ang_activi + 1;
596
597      DBMS_OUTPUT.PUT_LINE('-----');
598  END LOOP;
599
600  IF v_nr_ang_activi < (SELECT COUNT(*) FROM ANGAJAT) / 2 THEN
601    DBMS_OUTPUT.PUT_LINE('Personal inactiv!');
602  ELSE
603    DBMS_OUTPUT.PUT_LINE('Personal activ!');
604  END IF;
605
606  RETURN p_nr_cli_asist_ang;
607 END;
```

Script Output x

Task completed in 0.021 seconds

Function EX7\_FUNC compiled

Worksheet Query Builder

```
624 END EX7_FUNC;
625 /
626
627 DECLARE
628     v_id_ang ANGAJAT.id_angajat%TYPE := &p_id;
629     v_nr_cli_asist_ang NUMBER;
630
631     v_nr_ang NUMBER;
632 BEGIN
633     SELECT COUNT(id_angajat)
634     INTO v_nr_ang
635     FROM ANGAJAT;
636
637     SELECT id_angajat
638     INTO v_id_ang
639     FROM ANGAJAT
640     WHERE id_angajat = v_id_ang;
641
642     IF EX7_FUNC(v_id_ang, v_nr_cli_asist_ang) < v_nr_ang/2 THEN
643         DBMS_OUTPUT.PUT_LINE('Status personal:');
644         DBMS_OUTPUT.PUT_LINE('Personal inactiv!');
645     ELSE
646         DBMS_OUTPUT.PUT_LINE('Status personal:');
647         DBMS_OUTPUT.PUT_LINE('Personal activ!');
648     END IF;
649
650     DBMS_OUTPUT.NEW_LINE;
651
652     DBMS_OUTPUT.PUT_LINE('Numarul de clienti asistati de catre ang
653 EXCEPTION
654     WHEN NO_DATA_FOUND THEN
655         DBMS_OUTPUT.PUT_LINE('Nu exista niciun angajat cu id-ul '
656     WHEN OTHERS THEN
657         DBMS_OUTPUT.PUT_LINE('A aparut alta eroare!');
658 END;
659 /
660
```

Script Output x

Task completed in 1.729 seconds

DBMS\_OUTPUT.PUT\_LINE('A aparut alta eroare!');  
END;  
PL/SQL procedure successfully completed.

Alex\_Proiect\_SGBD x

Id angajat: 1  
Nume angajat: Aurel Tudor-Mihai

Clienti asistati de catre angajat:  
Carstea Darian-Stefan  
Manole Iuliana-Elena  
Aurel Popescu-Mihai  
Castan Mirela

Id angajat: 2  
Nume angajat: Mihail Maria-Andreea

Clienti asistati de catre angajat:  
Carstea Darian-Stefan  
Manole Iuliana-Elena  
Aurel Popescu-Mihai  
Castan Mirela  
Popa Marcel-Radu

Id angajat: 3  
Nume angajat: Vladoi Rares

Clienti asistati de catre angajat:  
Popa Marcel-Radu

Status personal:  
Personal activ!

Numarul de clienti asistati de catre angajatul cu id-ul 1: 4.

Worksheet Query Builder

```
624 END EX7_FUNC;
625 /
626
627 DECLARE
628     v_id_ang ANGAJAT.id_angajat%TYPE := &p_id;
629     v_nr_cli_asist_ang NUMBER;
630
631     v_nr_ang NUMBER;
632 BEGIN
633     SELECT COUNT(id_angajat)
634     INTO v_nr_ang
635     FROM ANGAJAT;
636
637     SELECT id_angajat
638     INTO v_id_ang
639     FROM ANGAJAT
640     WHERE id_angajat = v_id_ang;
641
642     IF EX7_FUNC(v_id_ang, v_nr_cli_asist_ang) < v_nr_ang/2 THEN
643         DBMS_OUTPUT.PUT_LINE('Status personal:');
644         DBMS_OUTPUT.PUT_LINE('Personal inactiv!');
645     ELSE
646         DBMS_OUTPUT.PUT_LINE('Status personal:');
647         DBMS_OUTPUT.PUT_LINE('Personal activ!');
648     END IF;
649
650     DBMS_OUTPUT.NEW_LINE;
651
652     DBMS_OUTPUT.PUT_LINE('Numarul de clienti asistati de catre angajatul cu id-ul ' || v
653 EXCEPTION
654     WHEN NO_DATA_FOUND THEN
655         DBMS_OUTPUT.PUT_LINE('Nu exista niciun angajat cu id-ul ' || v_id_ang || '!');
656     WHEN OTHERS THEN
657         DBMS_OUTPUT.PUT_LINE('A aparut alta eroare!');
658 END;
659 /
660
```

Script Output x

Task completed in 2.326 seconds

DBMS\_OUTPUT.PUT\_LINE('A aparut alta eroare!');  
END;  
PL/SQL procedure successfully completed.

Alex\_Proiect\_SGBD x

Nu exista niciun angajat cu id-ul 1111!

8. Subprogram stocat de tip funcție care utilizează într-o singură comandă SQL 3 dintre tabelele definite:

-- Cerinta 8.:

-- Enunt:

-- Pentru un cont al carui nume de utilizator este dat de la tastatura, sa se

-- afiseze id-ul clientului care il detine. In cazul in care contul nu este detinut

-- de catre un client cunoscut, se va afisa un mesaj corespunzator. Totodata, se va

-- afisa, pentru top 3 recenzii (in functie de numarul de stele) scrise de catre cont,

-- id-ul recenziei, numarul de stele, si numele produsului pentru care a fost scrisa

-- recenzia. In caz ca exista mai multe recenzii care au acelasi numar de stele, si

-- sunt eligibile pentru a se afla in top, se vor afisa informatii despre toate aceste

-- recenzii. De asemenea, daca utilizatorul nu a scris nicio recenzie, se va afisa un

-- mesaj corespunzator.

-- Mai mult, pentru numele de utilizator specificat, sa se returneze si afiseze numarul

-- de recenzii scrise de catre contul asociat.

-- Sa se trateze toate exceptiile care pot aparea.

CREATE OR REPLACE FUNCTION EX8\_FUNC

(v\_nume\_utilizator IN CONT\_SITE.nume\_utilizator%TYPE)

RETURN NUMBER

IS

v\_id\_cont CONT\_SITE.id\_cont%TYPE;

v\_id\_client CLIENT.id\_client%TYPE;

v\_id\_recenzie RECENZIE.id\_recenzie%TYPE;

v\_nr\_stele RECENZIE.nr\_stele%TYPE;

v\_nr\_stele\_precedent RECENZIE.nr\_stele%TYPE := 0;

v\_nume\_produs PRODUS.denumire%TYPE;

v\_nr\_stele\_min RECENZIE.nr\_stele%TYPE;

TYPE ref\_info\_recenzii IS REF CURSOR;

c\_info\_recenzii ref\_info\_recenzii;

CURSOR c\_recenzii IS

    SELECT id\_client,

        CURSOR (SELECT id\_recenzie, nr\_stele, denumire

                FROM RECENZIE r JOIN PRODUS p ON r.id\_produs = p.id\_produs

                WHERE nume\_utilizator = v\_nume\_utilizator

                ORDER BY 2 DESC)

    FROM CONT\_SITE

    WHERE nume\_utilizator = v\_nume\_utilizator;

v\_contor NUMBER := 0;

v\_top NUMBER := 0;

BEGIN



```
SELECT id_cont  
INTO v_id_cont  
FROM CONT_SITE  
WHERE nume_utilizator = v_nume_utilizator;
```

```
DBMS_OUTPUT.PUT_LINE('Id cont: ' || v_id_cont);  
DBMS_OUTPUT.PUT_LINE('Nume utilizator cont: ' || v_nume_utilizator);
```

```
OPEN c_recenzii;
```

```
FETCH c_recenzii INTO v_id_client, c_info_recenzii;
```

```
IF v_id_client IS NULL THEN
```

```
    DBMS_OUTPUT.PUT_LINE('Contul nu este detinut de catre un client cunoscut.');
```

```
EISE
```

```
    DBMS_OUTPUT.PUT_LINE('Id client: ' || v_id_client);
```

```
END IF;
```

```
DBMS_OUTPUT.NEW_LINE;
```

```
SELECT MIN(distincte)  
INTO v_nr_stele_min  
FROM (SELECT DISTINCT nr_stele distincte  
      FROM RECENZIE
```

WHERE nume\_utilizator = v\_nume\_utilizator

ORDER BY distincte DESC)

WHERE ROWNUM <= 3;

FETCH c\_info\_recenzii INTO v\_id\_recenzie, v\_nr\_stele, v\_nume\_produș;

IF c\_info\_recenzii%ROWCOUNT = 1 THEN

DBMS\_OUTPUT.PUT\_LINE('Top 3 recenzii ale contului: ');

LOOP

v\_contor := v\_contor + 1;

IF v\_nr\_stele <> v\_nr\_stele\_precedent THEN

v\_top := v\_top + 1;

v\_nr\_stele\_precedent := v\_nr\_stele;

END IF;

IF v\_nr\_stele >= v\_nr\_stele\_min THEN

DBMS\_OUTPUT.PUT\_LINE(v\_top || ' . Id recenzie: ' || v\_id\_recenzie || ' | Nr. stele: ' || v\_nr\_stele || ' | Nume produs: ' || v\_nume\_produș);

END IF;

FETCH c\_info\_recenzii INTO v\_id\_recenzie, v\_nr\_stele, v\_nume\_produș;

```

        EXIT WHEN c_info_recenzii%NOTFOUND;

    END LOOP;

ELSE

    DBMS_OUTPUT.PUT_LINE('Utilizatorul nu a scris nicio recenzie.');
```

END IF;

DBMS\_OUTPUT.NEW\_LINE;

CLOSE c\_recenzii;

RETURN v\_contor;

EXCEPTION

    WHEN NO\_DATA\_FOUND THEN

        RAISE\_APPLICATION\_ERROR(-20000, 'Nu exista niciun cont cu numele de utilizator ' ||  
v\_nume\_utilizator || '');

    WHEN OTHERS THEN

        RAISE\_APPLICATION\_ERROR(-20002, 'A aparut alta eroare!');

END EX8\_FUNC;

/

DECLARE

    v\_nume\_utilizator CONT\_SITE.nume\_utilizator%TYPE := '&p\_nume\_utilizator';

    v\_nr\_recenzii NUMBER;

BEGIN

```

v_nr_recenzii := EX8_FUNC(v_num_utilizator);

IF v_nr_recenzii > 0 THEN

    IF v_nr_recenzii > 1 THEN

        DBMS_OUTPUT.PUT_LINE('Contul cu numele de utilizator ' || v_num_utilizator || ' a
scris ' || v_nr_recenzii || ' recenzii.');
```

ELSE

```

        DBMS_OUTPUT.PUT_LINE('Contul cu numele de utilizator ' || v_num_utilizator || ' a
scris o recenzie.');
```

END IF;

END IF;

END;

/

```

666 -- Cerinta 8.:
667
668 -- Enunt:
669 -- Pentru un cont al carui nume de utilizator este dat de la tastatura, sa se
670 -- afiseze id-ul clientului care il detine. In cazul in care contul nu este detinut
671 -- de catre un client cunoscut, se va afisa un mesaj corespunzator. Totodata, se va
672 -- afisa, pentru top 3 recenzii (in functie de numarul de stele) scrise de catre cont,
673 -- id-ul recenziei, numarul de stele, si numele produsului pentru care a fost scrisa
674 -- recenzia. In caz ca exista mai multe recenzii care au acelasi numar de stele, si
675 -- sunt eligibile pentru a se afla in top, se vor afisa informatii despre toate aceste
676 -- recenzii. De asemenea, daca utilizatorul nu a scris nicio recenzie, se va afisa un
677 -- mesaj corespunzator.
678 -- Mai mult, pentru numele de utilizator specificat, sa se returneze si afiseze numarul
679 -- de recenzii scrise de catre contul asociat.
680 -- Sa se trateze toate exceptiile care pot aparea.
681
682 CREATE OR REPLACE FUNCTION EX8_FUNC
683 (v_num_utilizator IN CONT_SITE.num_utilizator%TYPE)
684 RETURN NUMBER
685 IS
686     v_id_cont CONT_SITE.id_cont%TYPE;
687
688     v_id_client CLIENT.id_client%TYPE;
689
690     v_id_recenzie RECENZIE.id_recenzie%TYPE;
691     v_nr_stele RECENZIE.nr_stele%TYPE;
692     v_nr_stele_precedent RECENZIE.nr_stele%TYPE := 0;
693     v_num_produs PRODUS.denumire%TYPE;
694
695     v_nr_stele_min RECENZIE.nr_stele%TYPE;
696
697     TYPE ref_info_recenzii IS REF CURSOR;
698     c_info_recenzii ref_info_recenzii;
699
700     CURSOR c_recenzii IS
701         SELECT id_client,
702                CURSOR (SELECT id_recenzie, nr_stele, denumire
703                        FROM RECENZIE r JOIN PRODUS p ON r.id_produs = p.id_produs
704                        WHERE num_utilizator = v_num_utilizator
705                        ORDER BY nr_stele DESC)
706                FROM CONT_SITE
707                WHERE num_utilizator = v_num_utilizator;
708
709 BEGIN
710     -- Afisare id-ul clientului care il detine.
711     SELECT id_client INTO v_id_client FROM CLIENT WHERE num_utilizator = v_num_utilizator;
712
713     IF v_id_client IS NULL THEN
714         DBMS_OUTPUT.PUT_LINE('Nu exista client cu numele de utilizator ' || v_num_utilizator);
715         RETURN 0;
716     END IF;
717
718     -- Afisare id-ul recenziei, numarul de stele, si numele produsului pentru care a fost scrisa
719     -- recenzia.
720     OPEN c_recenzii;
721     LOOP
722         FETCH c_recenzii INTO v_id_recenzie, v_nr_stele, v_num_produs;
723         EXIT WHEN c_recenzii%NOTFOUND;
724
725         IF v_nr_stele > v_nr_stele_precedent THEN
726             v_nr_stele_precedent := v_nr_stele;
727         END IF;
728
729         DBMS_OUTPUT.PUT_LINE('Recenzie: ' || v_id_recenzie || ' Stele: ' || v_nr_stele || ' Produs: ' || v_num_produs);
730     END LOOP;
731     CLOSE c_recenzii;
732
733     -- Afisare numarul de recenzii scrise de catre contul asociat.
734     SELECT COUNT(*) INTO v_nr_recenzii FROM RECENZIE WHERE id_client = v_id_client;
735
736     DBMS_OUTPUT.PUT_LINE('Numarul de recenzii scrise de catre contul asociat: ' || v_nr_recenzii);
737
738     RETURN v_nr_recenzii;
739 END;
```

Script Output x

Task completed in 0.025 seconds

Function EX8\_FUNC compiled

Worksheet Query Builder

```
755      END IF;
756
757      FETCH c_info_recenzii INTO v_id_recenzie, v_
758
759      EXIT WHEN c_info_recenzii%NOTFOUND;
760      END LOOP;
761      ELSE
762          DBMS_OUTPUT.PUT_LINE('Utilizatorul nu a scris ni
763      END IF;
764
765      DBMS_OUTPUT.NEW_LINE;
766
767      CLOSE c_recenzii;
768
769      RETURN v_contor;
770  EXCEPTION
771      WHEN NO_DATA_FOUND THEN
772          RAISE_APPLICATION_ERROR(-20000, 'Nu exista niciu
773      WHEN OTHERS THEN
774          RAISE_APPLICATION_ERROR(-20002, 'A aparut alta e
775  END EX@_FUNC;
776  /
777
778  DECLARE
779      v_nume_utilizator CONT_SITE.nume_utilizator%TYPE := 'sp_
780      v_nr_recenzii NUMBER;
781  BEGIN
782      v_nr_recenzii := EX@_FUNC(v_nume_utilizator);
783
784      IF v_nr_recenzii > 0 THEN
785          IF v_nr_recenzii > 1 THEN
786              DBMS_OUTPUT.PUT_LINE('Contul cu numele de utiliz
787          ELSE
788              DBMS_OUTPUT.PUT_LINE('Contul cu numele de utiliz
789          END IF;
790      END IF;
791  END;
792  /
793
```

Id cont: 1  
Nume utilizator cont: Utilizator Mihai  
Contul nu este detinut de catre un client cunoscut.

Top 3 recenzii ale contului:

- 1. Id recenzie: 1 | Nr. stele: 5 | Nume produs: Placa video Palit Gef
- 1. Id recenzie: 2 | Nr. stele: 5 | Nume produs: Procesor AMD Ryzen 5
- 2. Id recenzie: 8 | Nr. stele: 4 | Nume produs: Memorie Corsair Venge
- 3. Id recenzie: 7 | Nr. stele: 3 | Nume produs: Adaptor Gembird lx H

Contul cu numele de utilizator Utilizator Mihai a scris 4 recenzii.

Script Output x

Task completed in 7.58 seconds

PL/SQL procedure successfully completed.

Worksheet Query Builder

```
755      END IF;
756
757      FETCH c_info_recenzii INTO v_id_recenzie, v_nr_stele, v_nume
758
759      EXIT WHEN c_info_recenzii%NOTFOUND;
760      END LOOP;
761      ELSE
762          DBMS_OUTPUT.PUT_LINE('Utilizatorul nu a scris nicio recenzie.');
```

Id cont: 4  
Nume utilizator cont: ANTON PAVEL MIHAI  
Contul nu este detinut de catre un client cunoscut.

Utilizatorul nu a scris nicio recenzie.

Script Output x

Task completed in 0.554 seconds

PL/SQL procedure successfully completed.

Worksheet Query Builder Alex\_Proiect\_SGBD Alex\_Proiect\_SGBD x

```
762 DBMS_OUTPUT.PUT_LINE('Utilizatorul nu a scris nicio recenzie.');
```

```
763 END IF;
```

```
764
```

```
765 DBMS_OUTPUT.NEW_LINE;
```

```
766
```

```
767 CLOSE c_recenzii;
```

```
768
```

```
769 RETURN v_contor;
```

```
770 EXCEPTION
```

```
771 WHEN NO_DATA_FOUND THEN
```

```
772 RAISE_APPLICATION_ERROR(-20000, 'Nu exista niciun cont cu numele de utilizator ' ||
```

```
773 WHEN OTHERS THEN
```

```
774 RAISE_APPLICATION_ERROR(-20002, 'A aparut alta eroare!');
```

```
775 END EX8_FUNC;
```

```
776 /
```

```
777
```

```
778 DECLARE
```

```
779 v_num_utilizator CONT_SITE.num_utilizator%TYPE := 'ap_num_utilizator';
```

```
780 v_nr_recenzii NUMBER;
```

```
781 BEGIN
```

```
782 v_nr_recenzii := EX8_FUNC(v_num_utilizator);
```

```
783
```

```
784 IF v_nr_recenzii > 0 THEN
```

```
785 IF v_nr_recenzii > 1 THEN
```

```
786 DBMS_OUTPUT.PUT_LINE('Contul cu numele de utilizator ' || v_num_utilizator || ' a
```

```
787 ELSE
```

```
788 DBMS_OUTPUT.PUT_LINE('Contul cu numele de utilizator ' || v_num_utilizator || ' a
```

```
789 END IF;
```

```
790 END IF;
```

```
791 END;
```

```
792 /
```

```
793
```

Script Output x

Task completed in 2.287 seconds

```
END;
```

```
Error report -
```

```
ORA-20000: Nu exista niciun cont cu numele de utilizator dseubvs!
```

```
ORA-06512: at "ALEXSGBD.EX8_FUNC", line 91
```

```
ORA-06512: at line 5
```

```
20000. 00000 - "%s"
```

```
*Cause: The stored procedure 'raise_application_error'
```

```
was called which causes this error to be generated.
```

```
*Action: Correct the problem as described in the error message or contact
```

```
the application administrator or DBA for more information.
```

9. Subprogram stocat de tip procedură care utilizează într-o singură comandă SQL 5 dintre tabelele definite:

-- Cerinta 9.:

-- Enunt:

-- Pentru un produs al carui nume este introdus de la tastatura, sa se afiseze id-ul  
-- si numele tuturor serviciilor compatibile cu acesta. In plus, sa se afiseze si  
-- id-ul, alaturi de numele complet (nume + prenume) al tuturor clientilor care au  
-- cumparat un anumit serviciu compatibil cu produsul. In caz ca produsul nu este  
-- compatibil cu niciun serviciu, sau in caz ca nu exista clienti care au cumparat  
-- un anumit serviciu compatibil cu produsul, se vor afisa mesaje corespunzatoare.  
-- Rezultatele se vor afisa in ordine alfabetica a numelor serviciilor, cat si a  
-- numelor clientilor.  
-- Sa se trateze toate exceptiile care pot aparea.

CREATE OR REPLACE PROCEDURE EX9\_PROC

(v\_nume\_produs IN PRODUS.denumire%TYPE)

IS

v\_id\_produs PRODUS.id\_produs%TYPE;

v\_id\_serviciu SERVICIU.id\_serviciu%TYPE;

v\_nume\_serviciu SERVICIU.denumire%TYPE;

TYPE tab\_imb\_id\_cli IS TABLE OF CLIENT.id\_client%TYPE;

```
t_id_cli tab_imb_id_cli;
```

```
TYPE tab_imb_nume_cli IS TABLE OF CLIENT.nume%TYPE;
```

```
t_nume_cli tab_imb_nume_cli;
```

```
TYPE ref_info_cli IS REF CURSOR;
```

```
c_info_cli ref_info_cli;
```

```
CURSOR c_date_servicii IS
```

```
    SELECT s.id_serviciu, s.denumire,
```

```
           CURSOR (SELECT DISTINCT cl.id_client, nume || ' ' || prenume
```

```
                     FROM CLIENT cl JOIN CUMPARA cu ON cl.id_client = cu.id_client
```

```
                     JOIN SERVICIU si ON cu.id_serviciu = si.id_serviciu
```

```
                     WHERE si.id_serviciu = s.id_serviciu
```

```
                     ORDER BY 2)
```

```
    FROM SERVICIU s JOIN ESTE_COMPATIBIL ec ON s.id_serviciu = ec.id_serviciu
```

```
           JOIN PRODUS p ON ec.id_produs = p.id_produs
```

```
    WHERE UPPER(p.denumire) = UPPER(v_nume_produs)
```

```
    ORDER BY 2;
```

```
BEGIN
```

```
    SELECT id_produs
```

```
    INTO v_id_produs
```

```
    FROM PRODUS
```

```
    WHERE UPPER(denumire) = UPPER(v_nume_produs);
```



```
OPEN c_date_servicii;
```

```
DBMS_OUTPUT.PUT_LINE('Id produs: ' || v_id_produs);
```

```
DBMS_OUTPUT.PUT_LINE('Nume produs: ' || v_nume_produs);
```

```
DBMS_OUTPUT.NEW_LINE;
```

```
DBMS_OUTPUT.NEW_LINE;
```

```
DBMS_OUTPUT.PUT_LINE('Lista serviciilor compatibile cu produsul:');  
  
LOOP
```

```
    FETCH c_date_servicii INTO v_id_serviciu, v_nume_serviciu, c_info_cli;
```

```
    IF c_date_servicii%NOTFOUND THEN
```

```
        IF c_date_servicii%ROWCOUNT = 0 THEN
```

```
            DBMS_OUTPUT.PUT_LINE('Nu exista servicii compatibile!');
```

```
        END IF;
```

```
        EXIT;
```

```
    END IF;
```

```
    DBMS_OUTPUT.PUT_LINE('-----');
```

```
    DBMS_OUTPUT.PUT_LINE('Id serviciu: ' || v_id_serviciu);
```

```
    DBMS_OUTPUT.PUT_LINE('Nume serviciu: ' || v_nume_serviciu);
```

```
    DBMS_OUTPUT.NEW_LINE;
```

```

    FETCH c_info_cli BULK COLLECT INTO t_id_cli, t_nume_cli;

    DBMS_OUTPUT.PUT_LINE('Lista clientilor care au cumparat serviciul:');

    IF t_id_cli.COUNT = 0 THEN

        DBMS_OUTPUT.PUT_LINE('Nu exista clienti care au cumparat serviciul.');
```

ELSE

```

        FOR i IN t_id_cli.FIRST..t_id_cli.LAST LOOP

            DBMS_OUTPUT.PUT_LINE('Id client: ' || t_id_cli(i) || ' | Nume client: ' ||
t_nume_cli(i));

            END LOOP;

        END IF;

        DBMS_OUTPUT.PUT_LINE('-----');

        DBMS_OUTPUT.NEW_LINE;

    END LOOP;

    CLOSE c_date_servicii;

    EXCEPTION

    WHEN NO_DATA_FOUND THEN

        RAISE_APPLICATION_ERROR(-20000, 'Nu exista niciun produs cu numele ' ||
v_nume_produș || '!');

    WHEN TOO_MANY_ROWS THEN

```

```
        RAISE_APPLICATION_ERROR(-20001, 'Exista mai multe produse cu numele ' ||  
v_nume_produș || '');
```

```
    WHEN OTHERS THEN
```

```
        RAISE_APPLICATION_ERROR(-20002, 'A aparut alta eroare!');
```

```
    END EX9_PROC;
```

```
    /
```

```
DECLARE
```

```
    v_nume_produș PRODUS.denumire%TYPE := '&p_nume_produș';
```

```
BEGIN
```

```
    EX9_PROC(v_nume_produș);
```

```
END;
```

```
    /
```

The screenshot displays a SQL development environment with a 'Worksheet' tab and a 'Query Builder' tab. The 'Worksheet' tab contains a PL/SQL procedure named EX9\_PROC. The procedure is designed to find compatible services for a given product. It includes several local variables and cursors. The main logic involves a cursor that joins the CLIENT, SERVICIU, and PRODUS tables to find services compatible with a specific product. The procedure is compiled successfully, as indicated by the 'Script Output' tab at the bottom, which shows the message 'Procedure EX9\_PROC compiled'.

```
005  -- Cerinta 9.:  
006  
007  -- Enunt:  
008  -- Pentru un produs al carui nume este introdus de la tastatura, sa se afiseze id-ul  
009  -- si numele tuturor serviciilor compatibile cu acesta. In plus, sa se afiseze si  
010  -- id-ul, alaturi de numele complet (nume + prenume) al tuturor clientilor care au  
011  -- cumparat un anumit serviciu compatibil cu produsul. In caz ca produsul nu este  
012  -- compatibil cu niciun serviciu, sau in caz ca nu exista clienti care au cumparat  
013  -- un anumit serviciu compatibil cu produsul, se vor afisa mesaje corespunzatoare.  
014  -- Rezultatele se vor afisa in ordine alfabetica a numelor serviciilor, cat si a  
015  -- numelor clientilor.  
016  -- Sa se trateze toate exceptiile care pot aparea.  
017  
018  CREATE OR REPLACE PROCEDURE EX9_PROC  
019  (v_nume_produș IN PRODUS.denumire%TYPE)  
020  IS  
021      v_id_produș PRODUS.id_produș%TYPE;  
022  
023      v_id_serviciu SERVICIU.id_serviciu%TYPE;  
024      v_nume_serviciu SERVICIU.denumire%TYPE;  
025  
026      TYPE tab_imb_id_cli IS TABLE OF CLIENT.id_client%TYPE;  
027      t_id_cli tab_imb_id_cli;  
028  
029      TYPE tab_imb_nume_cli IS TABLE OF CLIENT.nume%TYPE;  
030      t_nume_cli tab_imb_nume_cli;  
031  
032      TYPE ref_info_cli IS REF CURSOR;  
033      c_info_cli ref_info_cli;  
034  
035      CURSOR c_date_servicii IS  
036          SELECT s.id_serviciu, s.denumire,  
037              CURSOR (SELECT DISTINCT cl.id_client, nume || ' ' || prenume  
038                      FROM CLIENT cl JOIN CUMPARA cu ON cl.id_client = cu.id_client  
039                      JOIN SERVICIU si ON cu.id_serviciu = si.id_serviciu  
040                      WHERE si.id_serviciu = s.id_serviciu  
041                      ORDER BY 2)  
042          FROM SERVICIU s JOIN ESTE_COMPATIBIL ec ON s.id_serviciu = ec.id_serviciu  
043          JOIN PRODUS p ON ec.id_produș = p.id_produș  
044  
045  BEGIN  
046      v_id_produș := v_nume_produș;  
047  
048      OPEN c_info_cli FOR  
049          SELECT id_client, nume || ' ' || prenume  
050          FROM CLIENT cl JOIN CUMPARA cu ON cl.id_client = cu.id_client  
051          JOIN SERVICIU si ON cu.id_serviciu = si.id_serviciu  
052          WHERE si.id_serviciu = v_id_serviciu  
053          ORDER BY 2;  
054  
055      LOOP  
056          FETCH c_info_cli INTO t_id_cli, t_nume_cli;  
057          EXIT WHEN t_id_cli IS NULL;  
058  
059          RAISE_APPLICATION_ERROR(-20001, 'Exista mai multe produse cu numele ' ||  
060              v_nume_produș || '');
```

Alex\_Proiect\_SGBD

```

872 DBMS_OUTPUT.PUT_LINE('Id serviciu: ' || v_id_serviciu);
873 DBMS_OUTPUT.PUT_LINE('Nume serviciu: ' || v_nume_serviciu);
874 DBMS_OUTPUT.NEW_LINE;
875
876 FETCH c_info_cli BULK COLLECT INTO t_id_cli, t_nume_cli;
877
878 DBMS_OUTPUT.PUT_LINE('Lista clientilor care au cumparat serviciu:');
879
880 IF t_id_cli.COUNT = 0 THEN
881     DBMS_OUTPUT.PUT_LINE('Nu exista clienti care au cumparat serviciu');
882 ELSE
883     FOR i IN t_id_cli.FIRST..t_id_cli.LAST LOOP
884         DBMS_OUTPUT.PUT_LINE('Id client: ' || t_id_cli(i) || ' | Nume client: ' || t_nume_cli(i));
885     END LOOP;
886 END IF;
887
888 DBMS_OUTPUT.PUT_LINE('-----');
889 DBMS_OUTPUT.NEW_LINE;
890
891 END LOOP;
892
893 CLOSE c_date_servicii;
894 EXCEPTION
895 WHEN NO_DATA_FOUND THEN
896     RAISE_APPLICATION_ERROR(-20000, 'Nu exista niciun produs cu numele ' || v_nume_produs);
897 WHEN TOO_MANY_ROWS THEN
898     RAISE_APPLICATION_ERROR(-20001, 'Exista mai multe produse cu numele ' || v_nume_produs);
899 WHEN OTHERS THEN
900     RAISE_APPLICATION_ERROR(-20002, 'A aparut alta eroare!');
901 END EX9_PROC;
902 /
903
904 DECLARE
905     v_nume_produs PRODUS.denumire%TYPE := 'sp_nume_produs';
906 BEGIN
907     EX9_PROC(v_nume_produs);
908 END;
909 /
910

```

Id produs: 5  
Nume produs: Placa video Palit GeForce RTX 3090 Gaming

Lista serviciilor compatibile cu produsul:

Id serviciu: 4  
Nume serviciu: Serviciu Diagnosticare Produs

Lista clientilor care au cumparat serviciul:

Id client: 3 | Nume client: Castan Mirela  
Id client: 5 | Nume client: Manole Iuliana-Elena

Id serviciu: 3  
Nume serviciu: Serviciu Montare Componenta

Lista clientilor care au cumparat serviciul:

Id client: 6 | Nume client: Florian Andrei-Cosmin  
Id client: 4 | Nume client: Popa Marcel-Radu

Id serviciu: 2  
Nume serviciu: Serviciu Testare Produs

Lista clientilor care au cumparat serviciul:

Id client: 1 | Nume client: Aurel Popescu-Mihai  
Id client: 2 | Nume client: Carstea Darian-Stefan  
Id client: 3 | Nume client: Castan Mirela  
Id client: 6 | Nume client: Florian Andrei-Cosmin  
Id client: 5 | Nume client: Manole Iuliana-Elena  
Id client: 4 | Nume client: Popa Marcel-Radu

Script Output x Query Result x

Task completed in 0.589 seconds

PL/SQL procedure successfully completed.

Alex\_Proiect\_SGBD

```

874 DBMS_OUTPUT.PUT_LINE('');
875
876 FETCH c_info_cli BULK COLLECT INTO t_id_cli, t_nume_cli;
877
878 DBMS_OUTPUT.PUT_LINE('Lista clientilor care au cumparat serviciul:');
879
880 IF t_id_cli.COUNT = 0 THEN
881     DBMS_OUTPUT.PUT_LINE('Nu exista clienti care au cumparat serviciul');
882 ELSE
883     FOR i IN t_id_cli.FIRST..t_id_cli.LAST LOOP
884         DBMS_OUTPUT.PUT_LINE('Id client: ' || t_id_cli(i) || ' | Nume client: ' || t_nume_cli(i));
885     END LOOP;
886 END IF;
887
888 DBMS_OUTPUT.PUT_LINE('-----');
889 DBMS_OUTPUT.NEW_LINE;
890
891 END LOOP;
892
893 CLOSE c_date_servicii;
894 EXCEPTION
895 WHEN NO_DATA_FOUND THEN
896     RAISE_APPLICATION_ERROR(-20000, 'Nu exista niciun produs cu numele ' || v_nume_produs);
897 WHEN TOO_MANY_ROWS THEN
898     RAISE_APPLICATION_ERROR(-20001, 'Exista mai multe produse cu numele ' || v_nume_produs);
899 WHEN OTHERS THEN
900     RAISE_APPLICATION_ERROR(-20002, 'A aparut alta eroare!');
901 END EX9_PROC;
902 /
903
904 DECLARE
905     v_nume_produs PRODUS.denumire%TYPE := 'sp_nume_produs';
906 BEGIN
907     EX9_PROC(v_nume_produs);
908 END;
909 /
910

```

Id produs: 40  
Nume produs: Hard disk Seagate BarraCuda 2TB

Lista serviciilor compatibile cu produsul:  
Nu exista servicii compatibile!

Script Output x Query Result x

Task completed in 0.658 seconds

EX9\_PROC(v\_nume\_produs);  
END;

PL/SQL procedure successfully completed.

Worksheet Query Builder

```

876 DBMS_OUTPUT.PUT_LINE('Nume serviciu: ' || v_nume_serviciu);
877 DBMS_OUTPUT.NEW_LINE;
878
879 FETCH c_info_cli BULK COLLECT INTO t_id_cli, t_nume_cli;
880
881 DBMS_OUTPUT.PUT_LINE('Lista clientilor care au cumparat serviciul');
882
883 IF t_id_cli.COUNT = 0 THEN
884   DBMS_OUTPUT.PUT_LINE('Nu exista clienti care au cumparat serv
885 ELSE
886   FOR i IN t_id_cli.FIRST..t_id_cli.LAST LOOP
887     DBMS_OUTPUT.PUT_LINE('Id client: ' || t_id_cli(i) || ' |
888   END LOOP;
889 END IF;
890
891 DBMS_OUTPUT.PUT_LINE('-----');
892 DBMS_OUTPUT.NEW_LINE;
893
894 END LOOP;
895
896 CLOSE c_date_servicii;
897 EXCEPTION
898 WHEN NO_DATA_FOUND THEN
899   RAISE_APPLICATION_ERROR(-20000, 'Nu exista niciun produs cu numel
900 WHEN TOO_MANY_ROWS THEN
901   RAISE_APPLICATION_ERROR(-20001, 'Exista mai multe produse cu nume
902 WHEN OTHERS THEN
903   RAISE_APPLICATION_ERROR(-20002, 'A aparut alta eroare!');
904 END EX9_PROC;
905 /
906
907 DECLARE
908   v_nume_produs PRODUS.denumire%TYPE := 'sp_nume_produs';
909 BEGIN
910   EX9_PROC(v_nume_produs);
911 END;
912 /
913

```

Script Output x Query Result x

Task completed in 1.1 seconds

EX9\_PROC(v\_nume\_produs);

END;

PL/SQL procedure successfully completed.

Alex\_Proiect\_SGBD x

Id produs: 10  
Nume produs: Procesor AMD Ryzen 5 3600 3.6GHz box

Lista serviciilor compatibile cu produsul:

Id serviciu: 1  
Nume serviciu: Serviciu Asamblare Standard

Lista clientilor care au cumparat serviciul:  
Nu exista clienti care au cumparat serviciul.

Id serviciu: 3  
Nume serviciu: Serviciu Montare Componenta

Lista clientilor care au cumparat serviciul:  
Id client: 6 | Nume client: Florian Andrei-Cosmin  
Id client: 4 | Nume client: Popa Marcel-Radu

Id serviciu: 2  
Nume serviciu: Serviciu Testare Produs

Lista clientilor care au cumparat serviciul:  
Id client: 1 | Nume client: Aurel Popescu-Mihai  
Id client: 2 | Nume client: Carstea Darian-Stefan  
Id client: 3 | Nume client: Castan Mirela  
Id client: 6 | Nume client: Florian Andrei-Cosmin  
Id client: 5 | Nume client: Manole Iuliana-Elena  
Id client: 4 | Nume client: Popa Marcel-Radu

Worksheet Query Builder

```

880
881 DBMS_OUTPUT.PUT_LINE('Lista clientilor care au cumparat serviciul');
882
883 IF t_id_cli.COUNT = 0 THEN
884   DBMS_OUTPUT.PUT_LINE('Nu exista clienti care au cumparat serviciul.');
```

```

885 ELSE
886   FOR i IN t_id_cli.FIRST..t_id_cli.LAST LOOP
887     DBMS_OUTPUT.PUT_LINE('Id client: ' || t_id_cli(i) || ' | Nume client: ' || t
888   END LOOP;
889 END IF;
890
891 DBMS_OUTPUT.PUT_LINE('-----');
892 DBMS_OUTPUT.NEW_LINE;
893
894 END LOOP;
895
896 CLOSE c_date_servicii;
897 EXCEPTION
898 WHEN NO_DATA_FOUND THEN
899   RAISE_APPLICATION_ERROR(-20000, 'Nu exista niciun produs cu numele ' || v_nume_produ
900 WHEN TOO_MANY_ROWS THEN
901   RAISE_APPLICATION_ERROR(-20001, 'Exista mai multe produse cu numele ' || v_nume_produ
902 WHEN OTHERS THEN
903   RAISE_APPLICATION_ERROR(-20002, 'A aparut alta eroare!');
```

```

904 END EX9_PROC;
905 /
906
907 DECLARE
908   v_nume_produs PRODUS.denumire%TYPE := 'sp_nume_produs';
909 BEGIN
910   EX9_PROC(v_nume_produs);
911 END;
912 /
913

```

Script Output x Query Result x

Task completed in 4.718 seconds

Error report -  
ORA-20000: Nu exista niciun produs cu numele mnj,hbnfhj!  
ORA-06512: at "ALEXSGBD.EX9\_PROC", line 79  
ORA-06512: at line 4  
20000. 00000 - "%s"  
\*Cause: The stored procedure 'raise\_application\_error' was called which causes this error to be generated.  
\*Action: Correct the problem as described in the error message or contact the application administrator or DBA for more information.

Alex\_Proiect\_SGBD x

Alex\_Proiect\_SGBD

Worksheet Query Builder

```
880 DBMS_OUTPUT.PUT_LINE('Lista clientilor care au cumparat serviciul:');
881
882 IF t_id_cli.COUNT = 0 THEN
883     DBMS_OUTPUT.PUT_LINE('Nu exista clienti care au cumparat serviciul.');
```

885 ELSE

```
886     FOR i IN t_id_cli.FIRST..t_id_cli.LAST LOOP
887         DBMS_OUTPUT.PUT_LINE('Id client: ' || t_id_cli(i) || ' | Nume client: ' || t
888     END LOOP;
889 END IF;
890
891 DBMS_OUTPUT.PUT_LINE('-----');
892 DBMS_OUTPUT.NEW_LINE;
893
894 END LOOP;
895
896 CLOSE c_date_servicii;
897 EXCEPTION
898     WHEN NO_DATA_FOUND THEN
899         RAISE_APPLICATION_ERROR(-20000, 'Nu exista niciun produs cu numele ' || v_nume_produ
900     WHEN TOO_MANY_ROWS THEN
901         RAISE_APPLICATION_ERROR(-20001, 'Exista mai multe produse cu numele ' || v_nume_produ
902     WHEN OTHERS THEN
903         RAISE_APPLICATION_ERROR(-20002, 'A aparut alta eroare!');
```

904 END EX9\_PROC;

905 /

906

907 DECLARE

```
908     v_nume_produ PRODUS.denumire%TYPE := '&p_nume_produ';
909 BEGIN
910     EX9_PROC(v_nume_produ);
911 END;
```

912 /

913

Script Output x Query Result x

Task completed in 0.673 seconds

```
v_nume_produ PRODUS.denumire%TYPE := '&p_nume_produ';
BEGIN
    EX9_PROC(v_nume_produ);
END;
Error report -
ORA-20001: Exista mai multe produse cu numele Adaptor Gembird lx HDMI 1.4 Male - lx VGA Female!
ORA-06512: at "ALEXSGBD.EX9_PROC", line 81
ORA-06512: at line 4
```

## 10. Trigger de tip LMD la nivel de comandă:

-- Cerinta 10.:

```
CREATE OR REPLACE PROCEDURE EX10_PROC_AUX
IS
    TYPE tab_imb_tipuri_servicii IS TABLE OF SERVICIU.tip%TYPE;
    t_tipuri_servicii tab_imb_tipuri_servicii;

    v_nr_servicii NUMBER;
BEGIN
    SELECT DISTINCT tip
    BULK COLLECT INTO t_tipuri_servicii
    FROM SERVICIU
    ORDER BY 1;

    IF t_tipuri_servicii.COUNT > 0 THEN
        FOR i IN t_tipuri_servicii.FIRST..t_tipuri_servicii.LAST LOOP
            SELECT COUNT(*)
            INTO v_nr_servicii
            FROM SERVICIU
            WHERE tip = t_tipuri_servicii(i);

            IF v_nr_servicii = 1 THEN
                DBMS_OUTPUT.PUT_LINE(' un serviciu de tipul ' || t_tipuri_servicii(i));
```

```

        ELSE

            DBMS_OUTPUT.PUT_LINE(' ' || v_nr_servicii || ' servicii de tipul ' ||
t_tipuri_servicii(i));

        END IF;

    END LOOP;

END IF;

END EX10_PROC_AUX;

/

```

```

CREATE OR REPLACE TRIGGER EX10_TRIG_LMD_COM_BEFORE

```

```

    BEFORE INSERT OR DELETE ON SERVICIU

```

```

DECLARE

```

```

    v_nr_inregistrari NUMBER;

```

```

BEGIN

```

```

    SELECT COUNT (*) INTO v_nr_inregistrari FROM SERVICIU;

```

```

    IF INSERTING THEN

```

```

        IF v_nr_inregistrari >= 1000 THEN

```

```

            RAISE_APPLICATION_ERROR(-20003, 'Nu se mai pot insera inregistrari in tabel,
deoarece acesta este prea plin!');

```

```

        END IF;

```

```

        DBMS_OUTPUT.PUT_LINE('-----');

```

```

        IF v_nr_inregistrari <> 1 THEN

```



```

        DBMS_OUTPUT.PUT('Inainte de inserare, in tabelul SERVICIU se afla ' ||
v_nr_inregistrari || ' servicii');

        IF v_nr_inregistrari = 0 THEN

            DBMS_OUTPUT.PUT_LINE('.');

        ELSE

            DBMS_OUTPUT.PUT_LINE(':');

        END IF;

    ELSE

        DBMS_OUTPUT.PUT_LINE('Inainte de inserare, in tabelul SERVICIU se afla un
serviciu:');

        END IF;

    ELSE

        IF v_nr_inregistrari <= 0 THEN

            RAISE_APPLICATION_ERROR(-20003, 'Nu se mai pot sterge inregistrari din tabel,
deoarece acesta este gol!');

        END IF;

        DBMS_OUTPUT.PUT_LINE('-----');

        IF v_nr_inregistrari <> 1 THEN

            DBMS_OUTPUT.PUT('Inainte de stergere, in tabelul SERVICIU se afla ' ||
v_nr_inregistrari || ' servicii');

            IF v_nr_inregistrari = 0 THEN

                DBMS_OUTPUT.PUT_LINE('.');

            ELSE

                DBMS_OUTPUT.PUT_LINE(':');

            END IF;

        END IF;

    END IF;

```

```

        END IF;

    ELSE

        DBMS_OUTPUT.PUT_LINE('Inainte de stergere, in tabelul SERVICIU se afla un
serviciu:');

        END IF;

    END IF;

EX10_PROC_AUX;

END EX10_TRIG_LMD_COM_BEFORE;

/

CREATE OR REPLACE TRIGGER EX10_TRIG_LMD_COM_AFTER

    AFTER INSERT OR DELETE ON SERVICIU

DECLARE

    v_nr_inregistrari NUMBER;

BEGIN

    SELECT COUNT (*) INTO v_nr_inregistrari FROM SERVICIU;

    DBMS_OUTPUT.NEW_LINE;

    IF INSERTING THEN

        IF v_nr_inregistrari <> 1 THEN

            DBMS_OUTPUT.PUT('Dupa inserare, in tabelul SERVICIU se afla ' || v_nr_inregistrari ||
' servicii');

            IF v_nr_inregistrari = 0 THEN

```

```

        DBMS_OUTPUT.PUT_LINE('.');

    ELSE

        DBMS_OUTPUT.PUT_LINE(':');

    END IF;

ELSE

    DBMS_OUTPUT.PUT_LINE('Dupa inserare, in tabelul SERVICIU se afla un serviciu:');

END IF;

ELSE

    IF v_nr_inregistrari <> 1 THEN

        DBMS_OUTPUT.PUT('Dupa stergere, in tabelul SERVICIU au mai ramas ' ||
v_nr_inregistrari || ' servicii');

        IF v_nr_inregistrari = 0 THEN

            DBMS_OUTPUT.PUT_LINE('.');

        ELSE

            DBMS_OUTPUT.PUT_LINE(':');

        END IF;

    ELSE

        DBMS_OUTPUT.PUT_LINE('Dupa stergere, in tabelul SERVICIU a mai ramas un
serviciu:');

    END IF;

END IF;

EX10_PROC_AUX;

DBMS_OUTPUT.PUT_LINE('-----');

```

```
DBMS_OUTPUT.NEW_LINE;
```

```
END EX10_TRIG_LMD_COM_AFTER;
```

```
/
```

```
-- ALTER TRIGGER EX10_TRIG_LMD_COM_BEFORE DISABLE;
```

```
-- ALTER TRIGGER EX10_TRIG_LMD_COM_BEFORE ENABLE;
```

```
-- DROP TRIGGER EX10_TRIG_LMD_COM_BEFORE;
```

```
-- ALTER TRIGGER EX10_TRIG_LMD_COM_AFTER DISABLE;
```

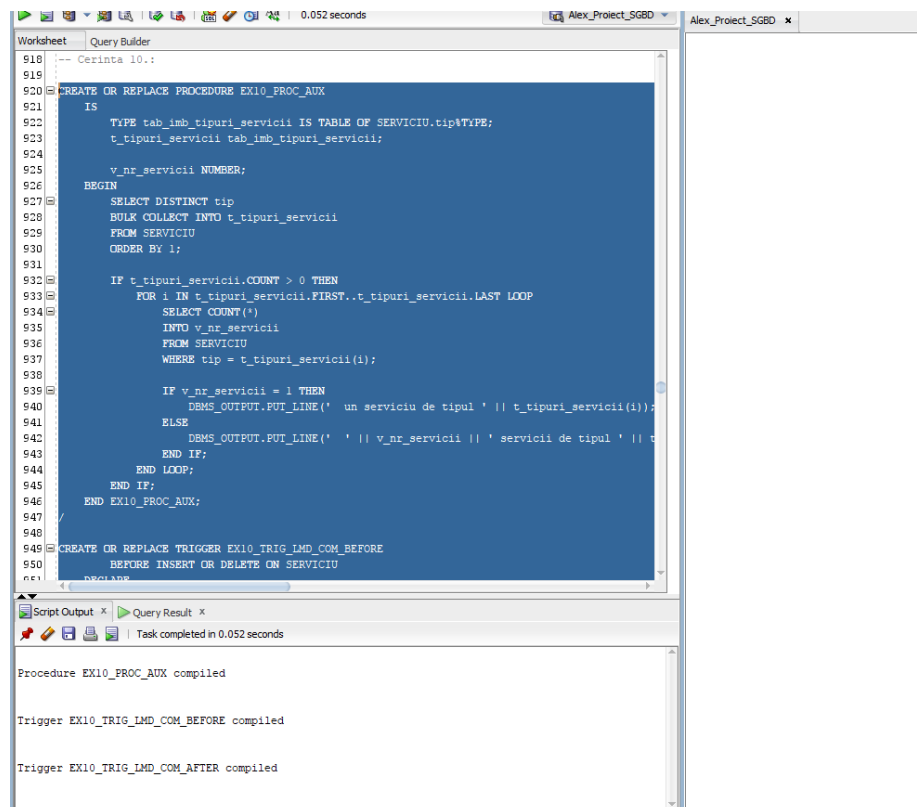
```
-- ALTER TRIGGER EX10_TRIG_LMD_COM_AFTER ENABLE;
```

```
-- DROP TRIGGER EX10_TRIG_LMD_COM_AFTER;
```

```
DELETE
```

```
FROM SERVICIU
```

```
WHERE id_serviciu >= 1;
```



The screenshot displays a SQL Developer window with a script titled 'Cerinta 10.' in the 'Query Builder' tab. The script defines a procedure 'EX10\_PROC\_AUX' and a trigger 'EX10\_TRIG\_LMD\_COM\_BEFORE'. The procedure logic involves selecting distinct tips from a table, looping through them, and using 'DBMS\_OUTPUT.PUT\_LINE' to display information. The trigger is set to fire before insert or delete operations on the 'SERVICIU' table. The 'Script Output' tab at the bottom shows the compilation status: 'Procedure EX10\_PROC\_AUX compiled', 'Trigger EX10\_TRIG\_LMD\_COM\_BEFORE compiled', and 'Trigger EX10\_TRIG\_LMD\_COM\_AFTER compiled'. The 'Query Result' tab is empty. The top status bar indicates the task was completed in 0.052 seconds.

```
918 -- Cerinta 10.:
919
920 CREATE OR REPLACE PROCEDURE EX10_PROC_AUX
921 IS
922     TYPE tab_tipuri_servicii IS TABLE OF SERVICIU.tip%TYPE;
923     t_tipuri_servicii tab_tipuri_servicii;
924
925     v_nr_servicii NUMBER;
926
927 BEGIN
928     SELECT DISTINCT tip
929     BULK COLLECT INTO t_tipuri_servicii
930     FROM SERVICIU
931     ORDER BY 1;
932
933     IF t_tipuri_servicii.COUNT > 0 THEN
934         FOR i IN t_tipuri_servicii.FIRST..t_tipuri_servicii.LAST LOOP
935             SELECT COUNT(*)
936             INTO v_nr_servicii
937             FROM SERVICIU
938             WHERE tip = t_tipuri_servicii(i);
939
940             IF v_nr_servicii = 1 THEN
941                 DBMS_OUTPUT.PUT_LINE(' un serviciu de tipul ' || t_tipuri_servicii(i));
942             ELSE
943                 DBMS_OUTPUT.PUT_LINE(' ' || v_nr_servicii || ' servicii de tipul ' || t_tipuri_servicii(i));
944             END IF;
945         END LOOP;
946     END IF;
947 END EX10_PROC_AUX;
948
949 CREATE OR REPLACE TRIGGER EX10_TRIG_LMD_COM_BEFORE
950 BEFORE INSERT OR DELETE ON SERVICIU
951
```

Script Output x Query Result x

Task completed in 0.052 seconds

Procedure EX10\_PROC\_AUX compiled

Trigger EX10\_TRIG\_LMD\_COM\_BEFORE compiled

Trigger EX10\_TRIG\_LMD\_COM\_AFTER compiled

0.039 seconds

Worksheet Query Builder

```
1057 START WITH 0
1058 MINVALUE -1
1059 NOCYCLE;
1060 -- DROP SEQUENCE SEQ_SERVICIU;
1061
1062 INSERT INTO SERVICIU
1063 VALUES (SEQ_SERVICIU.NEXTVAL, 'Asamblare', 'Serviciu Asamblare St
1064 'Asamblare sistem de calcul desktop de catre un specialist calif
1065 Instalare sistem de operare test. Verificare compatibilitate co
1066 Preinstalare sisteme de operare si aplicatii (daca s-au achizit
1067 159);
1068 INSERT INTO SERVICIU
1069 VALUES (SEQ_SERVICIU.NEXTVAL, 'Testare', 'Serviciu Testare Produs
1070 'Verificare functionalitate produs.
1071 Aplicabil oricarui produs comercializat.',
1072 135.99);
1073 INSERT INTO SERVICIU
1074 VALUES (SEQ_SERVICIU.NEXTVAL, 'Montare', 'Serviciu Montare Compon
1075 'Montarea diverselor componente in sistemul de calcul: placa vid
1076 procesor, RAM, HDD, cooler pentru procesor, etc.',
1077 59.99);
1078 INSERT INTO SERVICIU
1079 VALUES (SEQ_SERVICIU.NEXTVAL, 'Diagnosticare', 'Serviciu Diagnost
1080 'Identificarea problemelor de functionare si furnizarea de recom
1081 119.99);
1082 INSERT INTO SERVICIU
1083 VALUES (SEQ_SERVICIU.NEXTVAL, 'Asamblare', 'Serviciu Asamblare Pr
1084 'Asamblare si testare sistem de calcul desktop de catre un speci
1085 Cooling management. Wire management. Instalare sistem de operar
1086 Verificare compatibilitate componente. Testare sistem (rulare i
1087 Generarea unui raport operatiune asamblare premium. Updateuri l
1088 Preinstalare sisteme de operare si aplicatii (daca s-au achizit
1089 249.99);
1090
1091 DELETE
1092 FROM SERVICIU
1093 WHERE id_serviciu >= 2;
1094
```

Script Output x

Task completed in 0.039 seconds

1 row inserted.

1 row inserted.

Alex\_Proiect\_SGBD x

Inainte de inserare, in tabelul SERVICIU se afla 0 servicii.

Dupa inserare, in tabelul SERVICIU se afla un serviciu:  
un serviciu de tipul Asamblare

Inainte de inserare, in tabelul SERVICIU se afla un serviciu:  
un serviciu de tipul Asamblare

Dupa inserare, in tabelul SERVICIU se afla 2 servicii:  
un serviciu de tipul Asamblare  
un serviciu de tipul Testare

Inainte de inserare, in tabelul SERVICIU se afla 2 servicii:  
un serviciu de tipul Asamblare  
un serviciu de tipul Testare

Dupa inserare, in tabelul SERVICIU se afla 3 servicii:  
un serviciu de tipul Asamblare  
un serviciu de tipul Montare  
un serviciu de tipul Testare

Inainte de inserare, in tabelul SERVICIU se afla 3 servicii:  
un serviciu de tipul Asamblare  
un serviciu de tipul Montare  
un serviciu de tipul Testare

Dupa inserare, in tabelul SERVICIU se afla 4 servicii:  
un serviciu de tipul Asamblare  
un serviciu de tipul Diagnosticare  
un serviciu de tipul Montare  
un serviciu de tipul Testare

Inainte de inserare, in tabelul SERVICIU se afla 4 servicii:  
un serviciu de tipul Asamblare  
un serviciu de tipul Diagnosticare  
un serviciu de tipul Montare

Alex\_Proiect\_SGBD

Worksheet Query Builder

```
1057 START WITH 0
1058 MINVALUE -1
1059 NOCYCLE;
1060 -- DROP SEQUENCE SEQ_SERVICIU;
1061
1062 INSERT INTO SERVICIU
1063 VALUES (SEQ_SERVICIU.NEXTVAL, 'Asamblare', 'Serviciu Asamblare St
1064 'Asamblare sistem de calcul desktop de catre un specialist calif
1065 Instalare sistem de operare test. Verificare compatibilitate co
1066 Preinstalare sisteme de operare si aplicatii (daca s-au achizit
1067 159);
1068 INSERT INTO SERVICIU
1069 VALUES (SEQ_SERVICIU.NEXTVAL, 'Testare', 'Serviciu Testare Produs
1070 'Verificare functionalitate produs.
1071 Aplicabil oricarui produs comercializat.',
1072 135.99);
1073 INSERT INTO SERVICIU
1074 VALUES (SEQ_SERVICIU.NEXTVAL, 'Montare', 'Serviciu Montare Compon
1075 'Montarea diverselor componente in sistemul de calcul: placa vid
1076 procesor, RAM, HDD, cooler pentru procesor, etc.',
1077 59.99);
1078 INSERT INTO SERVICIU
1079 VALUES (SEQ_SERVICIU.NEXTVAL, 'Diagnosticare', 'Serviciu Diagnost
1080 'Identificarea problemelor de functionare si furnizarea de recom
1081 119.99);
1082 INSERT INTO SERVICIU
1083 VALUES (SEQ_SERVICIU.NEXTVAL, 'Asamblare', 'Serviciu Asamblare Pr
1084 'Asamblare si testare sistem de calcul desktop de catre un speci
1085 Cooling management. Wire management. Instalare sistem de operar
1086 Verificare compatibilitate componente. Testare sistem (rulare i
1087 Generarea unui raport operatiune asamblare premium. Updateuri l
1088 Preinstalare sisteme de operare si aplicatii (daca s-au achizit
1089 249.99);
1090
1091 DELETE
1092 FROM SERVICIU
1093 WHERE id_serviciu >= 2;
1094
```

Script Output x

Task completed in 0.016 seconds

4 rows deleted.

Alex\_Proiect\_SGBD x

Inainte de stergere, in tabelul SERVICIU se afla 5 servicii:  
2 servicii de tipul Asamblare  
un serviciu de tipul Diagnosticare  
un serviciu de tipul Montare  
un serviciu de tipul Testare

Dupa stergere, in tabelul SERVICIU a mai ramas un serviciu:  
un serviciu de tipul Asamblare

Worksheet Query Builder

```
1057 START WITH 0
1058 MINVALUE -1
1059 NOCYCLE;
1060 -- DROP SEQUENCE SEQ_SERVICIU;
1061
1062 INSERT INTO SERVICIU
1063 VALUES(SEQ_SERVICIU.NEXTVAL, 'Asamblare', 'Serviciu Asamblare St
1064 'Asamblare sistem de calcul desktop de catre un specialist calif
1065 Instalare sistem de operare test. Verificare compatibilitate co
1066 Preinstalare sisteme de operare si aplicatii (daca s-au achizit
1067 159);
1068 INSERT INTO SERVICIU
1069 VALUES(SEQ_SERVICIU.NEXTVAL, 'Testare', 'Serviciu Testare Produs
1070 'Verificare functionalitate produs.
1071 Aplicabil oricarui produs comercializat.',
1072 135.99);
1073 INSERT INTO SERVICIU
1074 VALUES(SEQ_SERVICIU.NEXTVAL, 'Montare', 'Serviciu Montare Compon
1075 'Montarea diverselor componente in sistemul de calcul: placa vid
1076 procesor, RAM, HDD, cooler pentru procesor, etc.',
1077 59.99);
1078 INSERT INTO SERVICIU
1079 VALUES(SEQ_SERVICIU.NEXTVAL, 'Diagnosticare', 'Serviciu Diagnost
1080 'Identificarea problemelor de functionare si furnizarea de recom
1081 119.99);
1082 INSERT INTO SERVICIU
1083 VALUES(SEQ_SERVICIU.NEXTVAL, 'Asamblare', 'Serviciu Asamblare Pr
1084 'Asamblare si testare sistem de calcul desktop de catre un speci
1085 Cooling management. Wire management. Instalare sistem de operar
1086 Verificare compatibilitate componente. Testare sistem (rulare i
1087 Generarea unui raport operatiune asamblare premium. Updateuri l
1088 Preinstalare sisteme de operare si aplicatii (daca s-au achizit
1089 249.99);
1090
1091 DELETE
1092 FROM SERVICIU
1093 WHERE id_serviciu >= 1;
1094
```

Script Output x

Task completed in 0.016 seconds

1 row deleted.

Alex\_Proiect\_SGBD x

Inainte de stergere, in tabelul SERVICIU se afla un serviciu:  
un serviciu de tipul Asamblare

Dupa stergere, in tabelul SERVICIU au mai ramas 0 servicii.

Worksheet Query Builder

```
1060 -- DROP SEQUENCE SEQ_SERVICIU;
1061
1062 INSERT INTO SERVICIU
1063 VALUES(SEQ_SERVICIU.NEXTVAL, 'Asamblare', 'Serviciu Asamblare Standard',
1064 'Asamblare sistem de calcul desktop de catre un specialist calificat.
1065 Instalare sistem de operare test. Verificare compatibilitate componente.
1066 Preinstalare sisteme de operare si aplicatii (daca s-au achizitionat)',
1067 159);
1068 INSERT INTO SERVICIU
1069 VALUES(SEQ_SERVICIU.NEXTVAL, 'Testare', 'Serviciu Testare Produs',
1070 'Verificare functionalitate produs.
1071 Aplicabil oricarui produs comercializat.',
1072 135.99);
1073 INSERT INTO SERVICIU
1074 VALUES(SEQ_SERVICIU.NEXTVAL, 'Montare', 'Serviciu Montare Componenta',
1075 'Montarea diverselor componente in sistemul de calcul: placa video,
1076 procesor, RAM, HDD, cooler pentru procesor, etc.',
1077 59.99);
1078 INSERT INTO SERVICIU
1079 VALUES(SEQ_SERVICIU.NEXTVAL, 'Diagnosticare', 'Serviciu Diagnosticare Produ
1080 'Identificarea problemelor de functionare si furnizarea de recomandari.',
1081 119.99);
1082 INSERT INTO SERVICIU
1083 VALUES(SEQ_SERVICIU.NEXTVAL, 'Asamblare', 'Serviciu Asamblare Premium',
1084 'Asamblare si testare sistem de calcul desktop de catre un specialist califi
1085 Cooling management. Wire management. Instalare sistem de operare test.
1086 Verificare compatibilitate componente. Testare sistem (rulare in conditii
1087 Generarea unui raport operatiune asamblare premium. Updateuri la ultimele
1088 Preinstalare sisteme de operare si aplicatii (daca s-au achizitionat).',
1089 249.99);
1090
1091 DELETE
1092 FROM SERVICIU
1093 WHERE id_serviciu >= 1;
1094
```

Script Output x

Task completed in 0.024 seconds

DELETE  
FROM SERVICIU  
WHERE id\_serviciu >= 1  
Error report -  
ORA-20003: Nu se mai pot sterge inregistrari din tabel, deoarece acesta este gol!  
ORA-06512: at "ALEXSGBD.EX10\_TRIG\_LMD\_COM\_BEFORE", line 25  
ORA-04088: error during execution of trigger 'ALEXSGBD.EX10\_TRIG\_LMD\_COM\_BEFORE'

Alex\_Proiect\_SGBD x

## 11. Trigger de tip LMD la nivel de linie:

-- Cerinta 11.:

```
CREATE OR REPLACE PACKAGE EX11_PACHET_AUX
```

```
AS
```

```
    TYPE tab_imb_id_min IS TABLE OF SERVICIU.id_serviciu%TYPE;
```

```
    TYPE tab_imb_id_max IS TABLE OF SERVICIU.id_serviciu%TYPE;
```

```
    TYPE PRET_SERVICIU_REC IS RECORD
```

```
    (min SERVICIU.pret%TYPE, t_id_min tab_imb_id_min,
```

```
    max SERVICIU.pret%TYPE, t_id_max tab_imb_id_max,
```

```
    avg SERVICIU.pret%TYPE);
```

```
    v_pret PRET_SERVICIU_REC;
```

```
END EX11_PACHET_AUX;
```

```
/
```

```
CREATE OR REPLACE TRIGGER EX11_TRIG_LMD_COM_BEFORE
```

```
    BEFORE UPDATE OF pret ON SERVICIU
```

```
BEGIN
```

```
    SELECT MIN(pret), MAX(pret), ROUND(AVG(pret), 2)
```

```
    INTO EX11_PACHET_AUX.v_pret.min, EX11_PACHET_AUX.v_pret.max,  
    EX11_PACHET_AUX.v_pret.avg
```

```
    FROM SERVICIU;
```

```
SELECT id_serviciu  
  
BULK COLLECT INTO EX11_PACHET_AUX.v_pret.t_id_min  
  
FROM SERVICIU  
  
WHERE pret = EX11_PACHET_AUX.v_pret.min;
```

```
SELECT id_serviciu  
  
BULK COLLECT INTO EX11_PACHET_AUX.v_pret.t_id_max  
  
FROM SERVICIU  
  
WHERE pret = EX11_PACHET_AUX.v_pret.max;
```

```
END EX11_TRIG_LMD_COM_BEFORE;
```

```
/
```

```
CREATE OR REPLACE TRIGGER EX11_TRIG_LMD_LIN_BEFORE
```

```
BEFORE UPDATE OF pret ON SERVICIU
```

```
FOR EACH ROW
```

```
BEGIN
```

```
DBMS_OUTPUT.PUT_LINE('-----');
```

```
DBMS_OUTPUT.PUT_LINE('Serviciul cu id-ul ' || :NEW.id_serviciu || ', si pretul egal cu ' ||  
:OLD.pret || ', va avea pretul egal cu ' || :NEW.pret || '.');
```

```
DBMS_OUTPUT.NEW_LINE;
```

```
DBMS_OUTPUT.PUT_LINE('Date actuale despre preturile din tabelul SERVICIU:');
```



```

        DBMS_OUTPUT.PUT(' Id-ul serviciilor care au pretul minim, adica pretul de ' ||
EX11_PACHET_AUX.v_pret.min || ': ');

        FOR i IN
EX11_PACHET_AUX.v_pret.t_id_min.FIRST..EX11_PACHET_AUX.v_pret.t_id_min.LAST LOOP

            DBMS_OUTPUT.PUT(EX11_PACHET_AUX.v_pret.t_id_min(i) || ' ');

        END LOOP;

        DBMS_OUTPUT.NEW_LINE;


        DBMS_OUTPUT.PUT(' Id-ul serviciilor care au pretul maxim, adica pretul de ' ||
EX11_PACHET_AUX.v_pret.max || ': ');

        FOR i IN
EX11_PACHET_AUX.v_pret.t_id_max.FIRST..EX11_PACHET_AUX.v_pret.t_id_max.LAST LOOP

            DBMS_OUTPUT.PUT(EX11_PACHET_AUX.v_pret.t_id_max(i) || ' ');

        END LOOP;

        DBMS_OUTPUT.NEW_LINE;


        DBMS_OUTPUT.PUT(' Pretul mediu al serviciilor: ' || EX11_PACHET_AUX.v_pret.avg);

        DBMS_OUTPUT.NEW_LINE;

        DBMS_OUTPUT.PUT_LINE('-----');


        DBMS_OUTPUT.NEW_LINE;

        DBMS_OUTPUT.NEW_LINE;

    END EX11_TRIG_LMD_LIN_BEFORE;

/

```

```
-- ALTER TRIGGER EX11_TRIG_LMD_COM_BEFORE DISABLE;

-- ALTER TRIGGER EX11_TRIG_LMD_COM_BEFORE ENABLE;

-- DROP TRIGGER EX11_TRIG_LMD_COM_BEFORE;
```

```
-- ALTER TRIGGER EX11_TRIG_LMD_LIN_BEFORE DISABLE;

-- ALTER TRIGGER EX11_TRIG_LMD_LIN_BEFORE ENABLE;

-- DROP TRIGGER EX11_TRIG_LMD_LIN_BEFORE;
```

UPDATE SERVICIU

SET pret = 1.1 \* pret

WHERE id\_serviciu >= 3;

DELETE

FROM SERVICIU

WHERE id\_serviciu >= 1;

The screenshot shows a SQL development environment with a script editor and a script output window. The script editor contains the following SQL code:

```
1052 -- Cerinta 11.:
1053
1054 CREATE OR REPLACE PACKAGE EX11_PACHET_AUX
1055 AS
1056     TYPE tab_imb_id_min IS TABLE OF SERVICIU.id_serviciu%TYPE;
1057     TYPE tab_imb_id_max IS TABLE OF SERVICIU.id_serviciu%TYPE;
1058
1059     TYPE PRET_SERVICIU_REC IS RECORD
1060     (min SERVICIU.pret%TYPE, t_id_min tab_imb_id_min,
1061      max SERVICIU.pret%TYPE, t_id_max tab_imb_id_max,
1062      avg SERVICIU.pret%TYPE);
1063
1064     v_pret PRET_SERVICIU_REC;
1065 END EX11_PACHET_AUX;
1066 /
1067
1068 CREATE OR REPLACE TRIGGER EX11_TRIG_LMD_COM_BEFORE
1069 BEFORE UPDATE OF pret ON SERVICIU
1070 BEGIN
1071     SELECT MIN(pret), MAX(pret), ROUND(AVG(pret), 2)
1072     INTO EX11_PACHET_AUX.v_pret.min, EX11_PACHET_AUX.v_pret.max, EX11_PACHET_AUX.v_pret.avg
1073     FROM SERVICIU;
1074
1075     SELECT id_serviciu
1076     BULK COLLECT INTO EX11_PACHET_AUX.v_pret.t_id_min
1077     FROM SERVICIU
1078     WHERE pret = EX11_PACHET_AUX.v_pret.min;
1079
1080     SELECT id_serviciu
1081     BULK COLLECT INTO EX11_PACHET_AUX.v_pret.t_id_max
1082     FROM SERVICIU
1083     WHERE pret = EX11_PACHET_AUX.v_pret.max;
1084 END EX11_TRIG_LMD_COM_BEFORE;
1085
```

The script output window shows the following messages:

```
Task completed in 0.045 seconds

Package EX11_PACHET_AUX compiled

Trigger EX11_TRIG_LMD_COM_BEFORE compiled

Trigger EX11_TRIG_LMD_LIN_BEFORE compiled
```



Alex\_Proiect\_SGBD x

Worksheet Query Builder

```
1098 DBMS_OUTPUT.PUT(' Id-ul serviciilor ca:');
1099 FOR i IN EX11_PACHET_AUX.v_pret.t_id_mi
1100     DBMS_OUTPUT.PUT(EX11_PACHET_AUX.v_p
1101 END LOOP;
1102 DBMS_OUTPUT.NEW_LINE;
1103
1104 DBMS_OUTPUT.PUT(' Id-ul serviciilor ca:');
1105 FOR i IN EX11_PACHET_AUX.v_pret.t_id_ma:
1106     DBMS_OUTPUT.PUT(EX11_PACHET_AUX.v_p
1107 END LOOP;
1108 DBMS_OUTPUT.NEW_LINE;
1109
1110 DBMS_OUTPUT.PUT(' Pretul mediu al serv
1111 DBMS_OUTPUT.NEW_LINE;
1112 DBMS_OUTPUT.PUT_LINE('-----');
1113
1114 DBMS_OUTPUT.NEW_LINE;
1115 DBMS_OUTPUT.NEW_LINE;
1116 END EX11_TRIG_LMD_LIN_BEFORE;
1117 /
1118
1119 -- ALTER TRIGGER EX11_TRIG_LMD_COM_BEFORE DISAB
1120 -- ALTER TRIGGER EX11_TRIG_LMD_COM_BEFORE ENABL
1121 -- DROP TRIGGER EX11_TRIG_LMD_COM_BEFORE;
1122
1123 -- ALTER TRIGGER EX11_TRIG_LMD_LIN_BEFORE DISAB
1124 -- ALTER TRIGGER EX11_TRIG_LMD_LIN_BEFORE ENABL
1125 -- DROP TRIGGER EX11_TRIG_LMD_LIN_BEFORE;
1126
1127 UPDATE SERVICIU
1128 SET pret = 1.1 * pret
1129 WHERE id_serviciu >= 3;
1130
1131 DELETE
1132 FROM SERVICIU
1133 WHERE id_serviciu >= 1;
1134
```

Script Output x Query Result x

Task completed in 0.021 seconds

5 rows deleted.

0 rows updated.

## 12. Trigger de tip LDD:

```
CREATE TABLE INFORMATII_COMANDA_LDD
```

```
(nume_bd VARCHAR2(50),  
user_logat VARCHAR2(30),  
eveniment VARCHAR2(20),  
data_comanda TIMESTAMP(3),  
tip_obiect_referit VARCHAR2(30),  
nume_obiect_referit VARCHAR2(30)  
);
```

```
CREATE OR REPLACE TRIGGER EX12_TRIG_LDD
```

```
AFTER CREATE OR DROP ON SCHEMA
```

```
BEGIN
```

```
IF SYS.SYSEVENT = 'DROP' THEN
```

```
    DBMS_OUTPUT.PUT_LINE('A avut loc un eveniment de tipul DROP asupra schemei  
personale, pe un obiect de tipul ' || SYS.DICTIONARY_OBJ_TYPE || ', numit ' ||  
SYS.DICTIONARY_OBJ_NAME || '.');
```

```
ELSE
```

```
    DBMS_OUTPUT.PUT_LINE('A avut loc un eveniment de tipul CREATE asupra schemei  
personale, pe un obiect de tipul ' || SYS.DICTIONARY_OBJ_TYPE || ', numit ' ||  
SYS.DICTIONARY_OBJ_NAME || '.');
```

```
END IF;
```

```
INSERT INTO INFORMATII_COMANDA_LDD
```

```
VALUES (SYS.DATABASE_NAME,
```

```

SYS.LOGIN_USER,

SYS.SYSEVENT,

SYSTIMESTAMP(3),

SYS.DICTIONARY_OBJ_TYPE,

SYS.DICTIONARY_OBJ_NAME

);

END EX12_TRIG_LDD;

/

-- ALTER TRIGGER EX12_TRIG_LDD DISABLE;

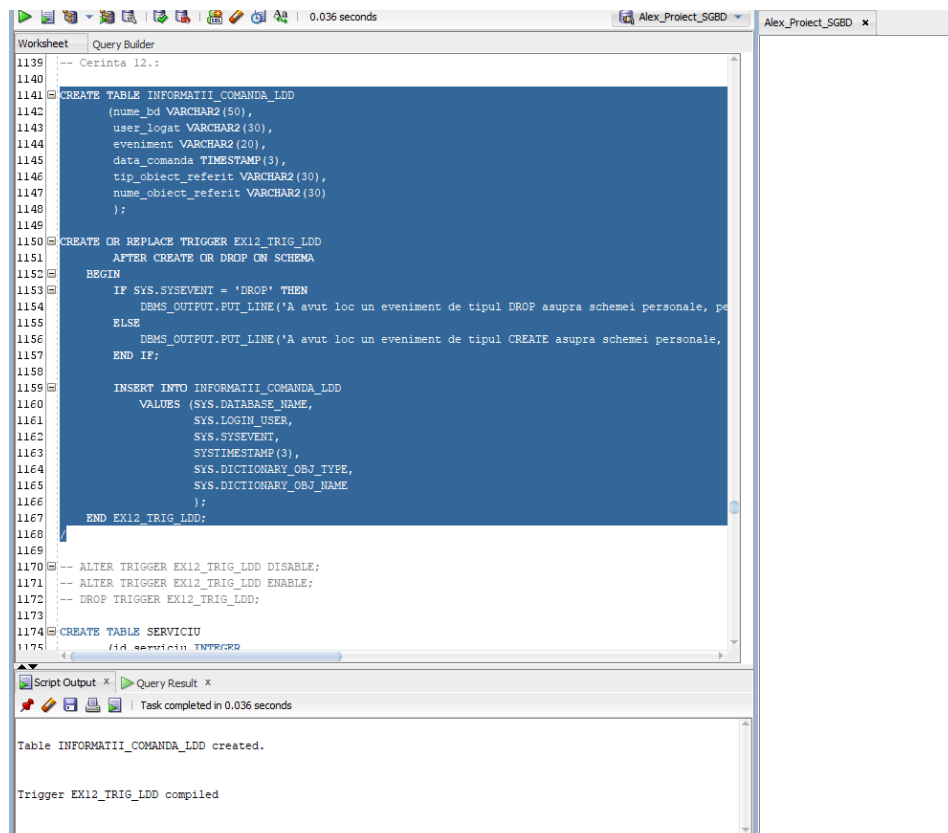
-- ALTER TRIGGER EX12_TRIG_LDD ENABLE;

-- DROP TRIGGER EX12_TRIG_LDD;

DROP TABLE SERVICIU;

SELECT * FROM INFORMATII_COMANDA_LDD;

```



Worksheet Query Builder

```
1173 CREATE TABLE SERVICIU
1174 (
1175     id_serviciu INTEGER,
1176     tip VARCHAR(13) CONSTRAINT
1177     denumire VARCHAR2(70) CO
1178     descriere VARCHAR2(1000)
1179     pret NUMBER CONSTRAINT p
1180     CONSTRAINT pk_serviciu P
1181     CONSTRAINT u_denumire_se
1182 );
1183
1184 -- Secventa pentru inserarea in
1185 CREATE SEQUENCE SEQ_SERVICIU
1186 INCREMENT by 1
1187 START WITH 0
1188 MINVALUE -1
1189 NOCYCLE;
1190 -- DROP SEQUENCE SEQ_SERVICIU;
1191
1192 INSERT INTO SERVICIU
1193 VALUES(SEQ_SERVICIU.NEXTVAL, 'As
1194 'Asamblare sistem de calcul desk
1195 Instalare sistem de operare tes
1196 Preinstalare sisteme de operare
1197 159);
1198 INSERT INTO SERVICIU
1199 VALUES(SEQ_SERVICIU.NEXTVAL, 'Te
1200 'Verificare functionalitate prod
1201 Aplicabil oricarui produs comer
1202 135.99);
1203 INSERT INTO SERVICIU
1204 VALUES(SEQ_SERVICIU.NEXTVAL, 'Mo
1205 'Montarea diverselor componente
1206 procesor, RAM, HDD, cooler pent
1207 59.99);
1208 INSERT INTO SERVICIU
1209 VALUES(SEQ_SERVICIU.NEXTVAL, 'Id
```

A avut loc un eveniment de tipul CREATE asupra schemei personale, pe un obiect de tipul INDEX, nu

A avut loc un eveniment de tipul CREATE asupra schemei personale, pe un obiect de tipul INDEX, nu

A avut loc un eveniment de tipul CREATE asupra schemei personale, pe un obiect de tipul TABLE, nu

A avut loc un eveniment de tipul CREATE asupra schemei personale, pe un obiect de tipul SEQUENCE,

Script Output x Query Result x

Task completed in 0.057 seconds

1 row inserted.

1 row inserted.

Worksheet Query Builder

```
1191
1192 INSERT INTO SERVICIU
1193 VALUES(SEQ_SERVICIU.NEXTVAL,
1194 'Asamblare sistem de calcul c
1195 Instalare sistem de operare
1196 Preinstalare sisteme de ope
1197 159);
1198 INSERT INTO SERVICIU
1199 VALUES(SEQ_SERVICIU.NEXTVAL,
1200 'Verificare functionalitate p
1201 Aplicabil oricarui produs co
1202 135.99);
1203 INSERT INTO SERVICIU
1204 VALUES(SEQ_SERVICIU.NEXTVAL,
1205 'Montarea diverselor componer
1206 procesor, RAM, HDD, cooler p
1207 59.99);
1208 INSERT INTO SERVICIU
1209 VALUES(SEQ_SERVICIU.NEXTVAL,
1210 'Identificarea problemelor de
1211 119.99);
1212 INSERT INTO SERVICIU
1213 VALUES(SEQ_SERVICIU.NEXTVAL,
1214 'Asamblare si testare sistem
1215 Cooling management. Wire mar
1216 Verificare compatibilitate c
1217 Generarea unui raport operat
1218 Preinstalare sisteme de ope
1219 249.99);
1220
1221 SELECT * FROM INFORMATII_COM
1222 DROP SEQUENCE SEQ_SERVICIU;
1223 DROP TABLE SERVICIU;
1224
1225
1226
1227
```

A avut loc un eveniment de tipul DROP asupra schemei personale, pe un obiect de tipul SEQUENCE, numi

A avut loc un eveniment de tipul DROP asupra schemei personale, pe un obiect de tipul TABLE, numit S

Script Output x Query Result x

Task completed in 0.043 s...

Sequence SEQ\_SERVICIU dropped.

Table SERVICIU dropped.

WorksheetQuery Builder

1191

1192

1193

1194

1195

1196

1197

1198

1199

1200

1201

1202

1203

1204

1205

1206

1207

1208

1209

1210

1211

1212

1213

1214

1215

1216

1217

1218

1219

1220

1221

1222

1223

1224

```
1192 INSERT INTO SERVICIU
1193 VALUES (SEQ_SERVICIU.NEXTVAL, 'Asamblare', 'Serviciu Asamblare Standard',
1194 'Asamblare sistem de calcul desktop de catre un specialist calificat.
1195 Instalare sistem de operare test. Verificare compatibilitate componente.
1196 Preinstalare sisteme de operare si aplicatii (daca s-au achizitionat)',
1197 159);
1198 INSERT INTO SERVICIU
1199 VALUES (SEQ_SERVICIU.NEXTVAL, 'Testare', 'Serviciu Testare Produs',
1200 'Verificare functionalitate produs.
1201 Aplicabil oricarui produs comercializat.',
1202 135.99);
1203 INSERT INTO SERVICIU
1204 VALUES (SEQ_SERVICIU.NEXTVAL, 'Montare', 'Serviciu Montare Componenta',
1205 'Montarea diverselor componente in sistemul de calcul: placa video,
1206 procesor, RAM, HDD, cooler pentru procesor, etc.',
1207 59.99);
1208 INSERT INTO SERVICIU
1209 VALUES (SEQ_SERVICIU.NEXTVAL, 'Diagnosticare', 'Serviciu Diagnosticare Produs',
1210 'Identificarea problemelor de functionare si furnizarea de recomandari.',
1211 119.99);
1212 INSERT INTO SERVICIU
1213 VALUES (SEQ_SERVICIU.NEXTVAL, 'Asamblare', 'Serviciu Asamblare Premium',
1214 'Asamblare si testare sistem de calcul desktop de catre un specialist calificat.
1215 Cooling management. Wire management. Instalare sistem de operare test.
1216 Verificare compatibilitate componente. Testare sistem (rulare in conditii de stres 24 ore).
1217 Generarea unui raport operatiune asamblare premium. Updateuri la ultimele versiuni stabile (BIOS placa
1218 Preinstalare sisteme de operare si aplicatii (daca s-au achizitionat).',
1219 249.99);
1220
1221 SELECT * FROM INFORMATII_COMANDA_LDD;
1222 DROP SEQUENCE SEQ_SERVICIU;
1223 DROP TABLE SERVICIU;
1224
```

Script OutputQuery Result

SQLAll Rows Fetched: 6 in 0.002 seconds

	NUME_BD	USER_LOGAT	EVENIMENT	DATA_COMANDA	TIP_OBIECT_REFERIT	NUME_OBIECT_REFERIT
1	O11G	ALEXSGBD	CREATE	01-JAN-22 09.21.43.575000000	PM INDEX	PK_SERVICIU
2	O11G	ALEXSGBD	CREATE	01-JAN-22 09.21.43.578000000	PM INDEX	U_DENUMIRE_SERVICIU
3	O11G	ALEXSGBD	CREATE	01-JAN-22 09.21.43.581000000	PM TABLE	SERVICIU
4	O11G	ALEXSGBD	CREATE	01-JAN-22 09.21.43.588000000	PM SEQUENCE	SEQ_SERVICIU
5	O11G	ALEXSGBD	DROP	01-JAN-22 09.24.02.396000000	PM SEQUENCE	SEQ_SERVICIU
6	O11G	ALEXSGBD	DROP	01-JAN-22 09.24.02.421000000	PM TABLE	SERVICIU



### 13. Pachet care conține toate obiectele definite în cadrul proiectului:

-- Cerinta 13.:

```
CREATE OR REPLACE PACKAGE EX13_PACK
```

```
IS
```

```
    PROCEDURE EX6_PROC(v_nume IN CLIENT.nume%TYPE);
```

```
    FUNCTION EX7_FUNC(p_id_ang IN ANGAJAT.id_angajat%TYPE,  
                      p_nr_cli_asist_ang OUT NUMBER)
```

```
        RETURN NUMBER;
```

```
    FUNCTION EX8_FUNC(v_nume_utilizator IN CONT_SITE.nume_utilizator%TYPE)  
        RETURN NUMBER;
```

```
    PROCEDURE EX9_PROC(v_nume_produs IN PRODUS.denumire%TYPE);
```

```
    PROCEDURE EX10_PROC_AUX;
```

```
END EX13_PACK;
```

```
/
```

```
CREATE OR REPLACE PACKAGE BODY EX13_PACK
```

```
IS
```

```
    PROCEDURE EX6_PROC
```

```
        (v_nume IN CLIENT.nume%TYPE)
```

```
    IS
```

```
        v_prenume CLIENT.prenume%TYPE;
```

```
        v_id CLIENT.id_client%TYPE;
```

```
TYPE tab_imb_id_cli IS TABLE OF CLIENT.id_client%TYPE;
```

```
t_id_cli tab_imb_id_cli;
```

```
v_nr_clienti NUMBER;
```

```
TYPE tab_imb_id_prod IS TABLE OF PRODUS.id_produs%TYPE;
```

```
t_id_prod tab_imb_id_prod;
```

```
TYPE tab_ind_num_prod IS TABLE OF PRODUS.denumire%TYPE
```

```
INDEX BY PLS_INTEGER;
```

```
t_num_prod tab_ind_num_prod;
```

```
TYPE vector_pret_prod IS VARRAY(2000) OF PRODUS.pret%TYPE;
```

```
vect_pret_prod vector_pret_prod := vector_pret_prod();
```

```
v_contor NUMBER;
```

```
BEGIN
```

```
SELECT COUNT(num)
```

```
INTO v_nr_clienti
```

```
FROM CLIENT
```

```
WHERE UPPER(num) = UPPER(v_num);
```

```
IF v_nr_clienti = 0 THEN
```

```
RAISE NO_DATA_FOUND;
```

ELSE

SELECT id\_client

BULK COLLECT INTO t\_id\_cli

FROM CLIENT

WHERE UPPER(ume) = UPPER(v\_ume);

FOR i IN t\_id\_cli.FIRST..t\_id\_cli.LAST LOOP

SELECT DISTINCT p.id\_produs, denumire, pret

BULK COLLECT INTO t\_id\_prod, t\_ume\_prod, vect\_pret\_prod

FROM PRODUS p JOIN COMANDA co ON p.id\_produs = co.id\_produs

JOIN CLIENT cl ON co.id\_client = cl.id\_client

WHERE cl.id\_client = t\_id\_cli(i)

ORDER BY 2;

SELECT preume

INTO v\_preume

FROM CLIENT

WHERE id\_client = t\_id\_cli(i);

IF t\_id\_prod.COUNT = 0 THEN

DBMS\_OUTPUT.PUT\_LINE('-----');

DBMS\_OUTPUT.PUT\_LINE('Clientul cu id-ul ' || t\_id\_cli(i) || ', pe ume ' ||  
v\_ume || ' ' || v\_preume || ', nu a comandat niciun produs!');

DBMS\_OUTPUT.PUT\_LINE('-----');

ELSE

v\_contor := 1;

DBMS\_OUTPUT.PUT\_LINE('-----');

DBMS\_OUTPUT.PUT\_LINE('Clientul cu id-ul ' || t\_id\_cli(i) || ', pe nume ' ||  
v\_nume || ' ' || v\_prenume || ', a achizitionat produsele: ');

DBMS\_OUTPUT.NEW\_LINE;

FOR i IN t\_id\_prod.FIRST..t\_id\_prod.LAST LOOP

DBMS\_OUTPUT.PUT\_LINE(v\_contor || ':');

DBMS\_OUTPUT.PUT\_LINE('Id produs: ' || t\_id\_prod(i));

DBMS\_OUTPUT.PUT\_LINE('Denumire produs: ' || t\_nume\_prod(i));

DBMS\_OUTPUT.PUT\_LINE('Pret produs: ' || vect\_pret\_prod(i));

IF i <> t\_id\_prod.LAST THEN

DBMS\_OUTPUT.NEW\_LINE;

END IF;

v\_contor := v\_contor + 1;

END LOOP;

DBMS\_OUTPUT.PUT\_LINE('-----');

END IF;

```
        DBMS_OUTPUT.NEW_LINE;

    END LOOP;

END IF;

EXCEPTION

    WHEN NO_DATA_FOUND THEN

        DBMS_OUTPUT.PUT_LINE('Nu exista niciun client cu numele ' || v_nume || '!');

    WHEN OTHERS THEN

        DBMS_OUTPUT.PUT_LINE('A aparut alta eroare!');

END EX6_PROC;
```

```
FUNCTION EX7_FUNC

    (p_id_ang IN ANGAJAT.id_angajat%TYPE,

    p_nr_cli_asist_ang OUT NUMBER)

    RETURN NUMBER

IS

    v_nr_ang_activi NUMBER;

    v_nr_cli_asist_ang NUMBER;

BEGIN

    p_nr_cli_asist_ang := 0;
```

```
FOR i IN (SELECT id_angajat, nume || ' ' || prenume nume_ang
          FROM ANGAJAT a
          ORDER BY 2) LOOP
```

```
SELECT COUNT(DISTINCT nume || ' ' || prenume)
INTO v_nr_cli_asist_ang
FROM CLIENT c JOIN ESTE_ASISTAT ea ON c.id_client = ea.id_client
WHERE ea.id_angajat = i.id_angajat;
```

```
IF v_nr_cli_asist_ang > 0 THEN
```

```
    v_nr_ang_activi := v_nr_ang_activi + 1;
```

```
    DBMS_OUTPUT.PUT_LINE('-----');
```

```
    DBMS_OUTPUT.PUT_LINE('Id angajat: ' || i.id_angajat);
```

```
    DBMS_OUTPUT.PUT_LINE('Nume angajat: ' || i.nume_ang);
```

```
    DBMS_OUTPUT.NEW_LINE;
```

```
    DBMS_OUTPUT.PUT_LINE('Clienti asistati de catre angajat: ');
```

```
    FOR j IN (SELECT DISTINCT nume || ' ' || prenume nume_cli
              FROM CLIENT c JOIN ESTE_ASISTAT ea ON c.id_client = ea.id_client
              WHERE ea.id_angajat = i.id_angajat) LOOP
```

```
        DBMS_OUTPUT.PUT_LINE(j.nume_cli);
```

```
END LOOP;
```

```
DBMS_OUTPUT.PUT_LINE('-----');
```

```
DBMS_OUTPUT.NEW_LINE;
```

```
END IF;
```

```
IF i.id_angajat = p_id_ang THEN
```

```
    p_nr_cli_asist_ang := v_nr_cli_asist_ang;
```

```
END IF;
```

```
END LOOP;
```

```
RETURN v_nr_ang_activi;
```

```
EXCEPTION
```

```
WHEN OTHERS THEN
```

```
    DBMS_OUTPUT.PUT_LINE('A aparut o eroare necunoscuta!');
```

```
END EX7_FUNC;
```

```
FUNCTION EX8_FUNC
```

```
(v_nume_utilizator IN CONT_SITE.nume_utilizator%TYPE)
```

```
RETURN NUMBER
```

IS

v\_id\_cont CONT\_SITE.id\_cont%TYPE;

v\_id\_client CLIENT.id\_client%TYPE;

v\_id\_recenzie RECENZIE.id\_recenzie%TYPE;

v\_nr\_stele RECENZIE.nr\_stele%TYPE;

v\_nr\_stele\_precedent RECENZIE.nr\_stele%TYPE := 0;

v\_nume\_produs PRODUS.denumire%TYPE;

v\_nr\_stele\_min RECENZIE.nr\_stele%TYPE;

TYPE ref\_info\_recenzii IS REF CURSOR;

c\_info\_recenzii ref\_info\_recenzii;

CURSOR c\_recenzii IS

SELECT id\_client,

CURSOR (SELECT id\_recenzie, nr\_stele, denumire

FROM RECENZIE r JOIN PRODUS p ON r.id\_produs = p.id\_produs

WHERE nume\_utilizator = v\_nume\_utilizator

ORDER BY 2 DESC)

FROM CONT\_SITE

WHERE nume\_utilizator = v\_nume\_utilizator;



```
v_contor NUMBER := 0;

v_top NUMBER := 0;

BEGIN

    SELECT id_cont
    INTO v_id_cont
    FROM CONT_SITE
    WHERE nume_utilizator = v_nume_utilizator;

    DBMS_OUTPUT.PUT_LINE('Id cont: ' || v_id_cont);
    DBMS_OUTPUT.PUT_LINE('Nume utilizator cont: ' || v_nume_utilizator);

    OPEN c_recenzii;

    FETCH c_recenzii INTO v_id_client, c_info_recenzii;

    IF v_id_client IS NULL THEN
        DBMS_OUTPUT.PUT_LINE('Contul nu este detinut de catre un client cunoscut.');
```

EISE

```
        DBMS_OUTPUT.PUT_LINE('Id client: ' || v_id_client);
    END IF;

    DBMS_OUTPUT.NEW_LINE;

    SELECT MIN(distincte)
```

```

    INTO v_nr_stele_min

    FROM (SELECT DISTINCT nr_stele distincte

          FROM RECENZIE

          WHERE nume_utilizator = v_nume_utilizator

          ORDER BY distincte DESC)

    WHERE ROWNUM <= 3;

    FETCH c_info_recenzii INTO v_id_recenzie, v_nr_stele, v_nume_produ;

    IF c_info_recenzii%ROWCOUNT = 1 THEN

        DBMS_OUTPUT.PUT_LINE('Top 3 recenzii ale contului: ');

    LOOP

        v_contor := v_contor + 1;

        IF v_nr_stele <> v_nr_stele_precedent THEN

            v_top := v_top + 1;

            v_nr_stele_precedent := v_nr_stele;

        END IF;

        IF v_nr_stele >= v_nr_stele_min THEN

            DBMS_OUTPUT.PUT_LINE(v_top || '. Id recenzie: ' || v_id_recenzie || ' | Nr.
stele: ' || v_nr_stele || ' | Nume produs: ' || v_nume_produ);

        END IF;

```

```

        FETCH c_info_recenzii INTO v_id_recenzie, v_nr_stele, v_nume_produ;

        EXIT WHEN c_info_recenzii%NOTFOUND;

    END LOOP;

    ELSE

        DBMS_OUTPUT.PUT_LINE('Utilizatorul nu a scris nicio recenzie.');
```

END IF;

```

    DBMS_OUTPUT.NEW_LINE;

    CLOSE c_recenzii;

    RETURN v_contor;

EXCEPTION

    WHEN NO_DATA_FOUND THEN

        RAISE_APPLICATION_ERROR(-20000, 'Nu exista niciun cont cu numele de utilizator '
|| v_nume_utilizator || '!');

    WHEN OTHERS THEN

        RAISE_APPLICATION_ERROR(-20002, 'A aparut alta eroare!');

END EX8_FUNC;
```

PROCEDURE EX9\_PROC

(v\_nume\_produs IN PRODUS.denumire%TYPE)

IS

v\_id\_produs PRODUS.id\_produs%TYPE;

v\_id\_serviciu SERVICIU.id\_serviciu%TYPE;

v\_nume\_serviciu SERVICIU.denumire%TYPE;

TYPE tab\_imb\_id\_cli IS TABLE OF CLIENT.id\_client%TYPE;

t\_id\_cli tab\_imb\_id\_cli;

TYPE tab\_imb\_nume\_cli IS TABLE OF CLIENT.nume%TYPE;

t\_nume\_cli tab\_imb\_nume\_cli;

TYPE ref\_info\_cli IS REF CURSOR;

c\_info\_cli ref\_info\_cli;

CURSOR c\_date\_servicii IS

SELECT s.id\_serviciu, s.denumire,

CURSOR (SELECT DISTINCT cl.id\_client, nume || ' ' || prenume

FROM CLIENT cl JOIN CUMPARA cu ON cl.id\_client = cu.id\_client

JOIN SERVICIU si ON cu.id\_serviciu = si.id\_serviciu

```

        WHERE si.id_serviciu = s.id_serviciu

        ORDER BY 2)

FROM SERVICIU s JOIN ESTE_COMPATIBIL ec ON s.id_serviciu = ec.id_serviciu

        JOIN PRODUS p ON ec.id_produs = p.id_produs

WHERE UPPER(p.denumire) = UPPER(v_nume_produs)

ORDER BY 2;

BEGIN

SELECT id_produs

INTO v_id_produs

FROM PRODUS

WHERE UPPER(denumire) = UPPER(v_nume_produs);

OPEN c_date_servicii;

DBMS_OUTPUT.PUT_LINE('Id produs: ' || v_id_produs);

DBMS_OUTPUT.PUT_LINE('Nume produs: ' || v_nume_produs);

DBMS_OUTPUT.NEW_LINE;

DBMS_OUTPUT.NEW_LINE;

DBMS_OUTPUT.PUT_LINE('Lista serviciilor compatibile cu produsul:');

LOOP

    FETCH c_date_servicii INTO v_id_serviciu, v_nume_serviciu, c_info_cli;

    IF c_date_servicii%NOTFOUND THEN

```

```

IF c_date_servicii%ROWCOUNT = 0 THEN

    DBMS_OUTPUT.PUT_LINE('Nu exista servicii compatibile!');

END IF;


EXIT;

END IF;


DBMS_OUTPUT.PUT_LINE('-----');

DBMS_OUTPUT.PUT_LINE('Id serviciu: ' || v_id_serviciu);

DBMS_OUTPUT.PUT_LINE('Nume serviciu: ' || v_nume_serviciu);

DBMS_OUTPUT.NEW_LINE;


FETCH c_info_cli BULK COLLECT INTO t_id_cli, t_nume_cli;


DBMS_OUTPUT.PUT_LINE('Lista clientilor care au cumparat serviciul:');


IF t_id_cli.COUNT = 0 THEN

    DBMS_OUTPUT.PUT_LINE('Nu exista clienti care au cumparat serviciul.');
```

ELSE

```

    FOR i IN t_id_cli.FIRST..t_id_cli.LAST LOOP

        DBMS_OUTPUT.PUT_LINE('Id client: ' || t_id_cli(i) || ' | Nume client: ' ||
t_nume_cli(i));

    END LOOP;

END IF;

```

```

        DBMS_OUTPUT.PUT_LINE('-----');

        DBMS_OUTPUT.NEW_LINE;

    END LOOP;

    CLOSE c_date_servicii;

    EXCEPTION

    WHEN NO_DATA_FOUND THEN

        RAISE_APPLICATION_ERROR(-20000, 'Nu exista niciun produs cu numele ' ||
v_nume_produș || '!');

    WHEN TOO_MANY_ROWS THEN

        RAISE_APPLICATION_ERROR(-20001, 'Exista mai multe produse cu numele ' ||
v_nume_produș || '!');

    WHEN OTHERS THEN

        RAISE_APPLICATION_ERROR(-20002, 'A aparut alta eroare!');

    END EX9_PROC;

```

```

PROCEDURE EX10_PROC_AUX

```

```

IS

```

```

    TYPE tab_imb_tipuri_servicii IS TABLE OF SERVICIU.tip%TYPE;

```

```

t_tipuri_servicii tab_imb_tipuri_servicii;

v_nr_servicii NUMBER;

BEGIN

SELECT DISTINCT tip

BULK COLLECT INTO t_tipuri_servicii

FROM SERVICIU

ORDER BY 1;

IF t_tipuri_servicii.COUNT > 0 THEN

FOR i IN t_tipuri_servicii.FIRST..t_tipuri_servicii.LAST LOOP

SELECT COUNT(*)

INTO v_nr_servicii

FROM SERVICIU

WHERE tip = t_tipuri_servicii(i);

IF v_nr_servicii = 1 THEN

DBMS_OUTPUT.PUT_LINE(' un serviciu de tipul ' || t_tipuri_servicii(i));

ELSE

DBMS_OUTPUT.PUT_LINE(' ' || v_nr_servicii || ' servicii de tipul ' ||
t_tipuri_servicii(i));

END IF;

END LOOP;

END IF;

```

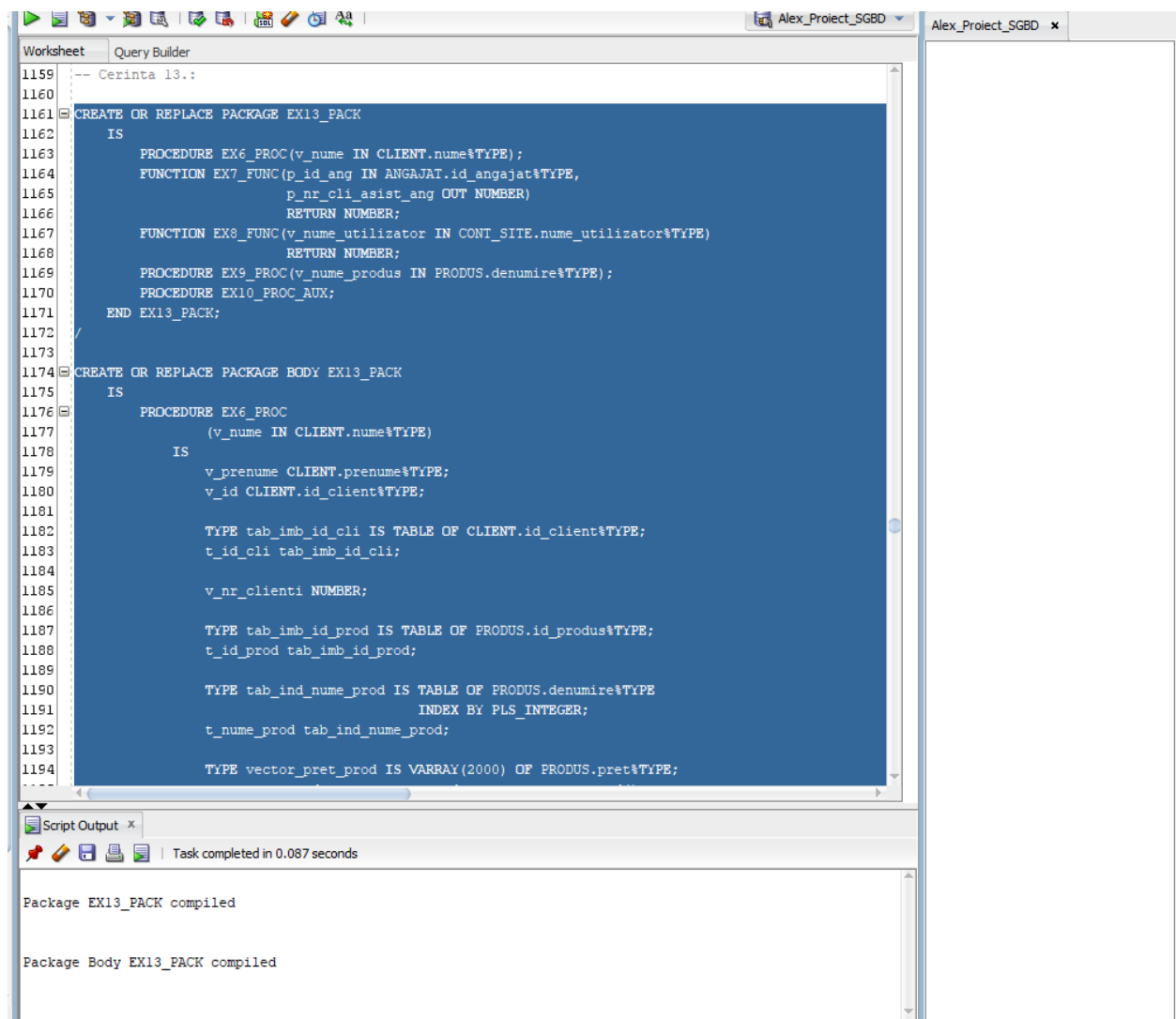


END EX10\_PROC\_AUX;

END EX13\_PACK;

/

-- DROP PACKAGE EX13\_PACK;



```
-- PENTRU EX13_PACK.EX6_PROC
```

```
DECLARE
```

```
    v_nume CLIENT.nume%TYPE := '&p_nume';
```

```
BEGIN
```

```
    EX13_PACK.EX6_PROC(v_nume);
```

```
END;
```

```
/
```

```
-- PENTRU EX13_PACK.EX7_FUNC
```

```
DECLARE
```

```
    v_id_ang ANGAJAT.id_angajat%TYPE := &p_id;
```

```
    v_nr_cli_asist_ang NUMBER;
```

```
    v_nr_ang NUMBER;
```

```
BEGIN
```

```
    SELECT COUNT(id_angajat)
```

```
    INTO v_nr_ang
```

```
    FROM ANGAJAT;
```

```
    SELECT id_angajat
```

```
    INTO v_id_ang
```

```
    FROM ANGAJAT
```

```
    WHERE id_angajat = v_id_ang;
```

```

IF EX13_PACK.EX7_FUNC(v_id_ang, v_nr_cli_asist_ang) < v_nr_ang/2 THEN

    DBMS_OUTPUT.PUT_LINE('Status personal:');

    DBMS_OUTPUT.PUT_LINE('Personal inactiv!');

ELSE

    DBMS_OUTPUT.PUT_LINE('Status personal:');

    DBMS_OUTPUT.PUT_LINE('Personal activ!');

END IF;


DBMS_OUTPUT.NEW_LINE;


DBMS_OUTPUT.PUT_LINE('Numarul de clienti asistati de catre angajatul cu id-ul ' || v_id_ang
|| ':' || v_nr_cli_asist_ang || '.');

EXCEPTION

    WHEN NO_DATA_FOUND THEN

        DBMS_OUTPUT.PUT_LINE('Nu exista niciun angajat cu id-ul ' || v_id_ang || '!');

    WHEN OTHERS THEN

        DBMS_OUTPUT.PUT_LINE('A aparut alta eroare!');

END;

/


-- PENTRU EX13_PACK.EX8_FUNC

DECLARE

    v_nume_utilizator CONT_SITE.nume_utilizator%TYPE := '&p_nume_utilizator';

```

```

    v_nr_recenzii NUMBER;

BEGIN

    v_nr_recenzii := EX13_PACK.EX8_FUNC(v_num_utilizator);

    IF v_nr_recenzii > 0 THEN

        IF v_nr_recenzii > 1 THEN

            DBMS_OUTPUT.PUT_LINE('Contul cu numele de utilizator ' || v_num_utilizator || ' a
scris ' || v_nr_recenzii || ' recenzii.');
```

```

        ELSE

            DBMS_OUTPUT.PUT_LINE('Contul cu numele de utilizator ' || v_num_utilizator || ' a
scris o recenzie.');
```

```

        END IF;

    END IF;

END;

/

-- PENTRU EX13_PACK.EX9_PROC

DECLARE

    v_num_produ PRODU.denumire%TYPE := '&p_num_produ';

BEGIN

    EX13_PACK.EX9_PROC(v_num_produ);

END;

/

-- PENTRU EX13_PACK.EX10_PROC_AUX
```

/

The screenshot shows the SQL Developer interface with two panes. The top pane, titled "Worksheet", contains the following PL/SQL code:

```
-- Cerinta 13.:
1181
1182
1183 CREATE OR REPLACE PACKAGE EX13_PACK...
1193 /
1194
1195 CREATE OR REPLACE PACKAGE BODY EX13_PACK...
1526 /
1527
1528
1529
1530 -- PENTRU EX13_PACK.EX6_PROC
1531 DECLARE
1532     v_name CLIENT.name%TYPE := 'sp_name';
1533 BEGIN
1534     EX13_PACK.EX6_PROC(v_name);
1535 END;
1536 /
1537
1538
1539 -- PENTRU EX13_PACK.EX7_FUNC
1540 DECLARE
1541     v_id_ang ANGAJAT.id_angajat%TYPE := sp_id;
1542     v_nr_cli_asist_ang NUMBER;
1543
1544     v_nr_ang NUMBER;
1545 BEGIN
1546     SELECT COUNT(id_angajat)
1547     INTO v_nr_ang
1548     FROM ANGAJAT;
1549
1550     SELECT id_angajat
1551     INTO v_id_ang
1552     FROM ANGAJAT
1553     WHERE id_angajat = v_id_ang;
1554
1555     IF EX13_PACK.EX7_FUNC(v_id_ang, v_nr_cli_a
1556         DBMS_OUTPUT.PUT_LINE('Status personal:');

```

The bottom pane, titled "Script Output", shows the result of the execution:

```
Script Output x
Task completed in 2.154 seconds
BEGIN
    EX13_PACK.EX6_PROC(v_name);
END;
PL/SQL procedure successfully completed.
```

Worksheet Query Builder

```
1181 -- Cerinta 13.:
1182
1183 CREATE OR REPLACE PACKAGE EX13_PACK...
1184 /
1185 CREATE OR REPLACE PACKAGE BODY EX13_PACK...
1186 /
1187
1188 -- PENTRU EX13_PACK.EX6_PROC
1189 DECLARE
1190     v_nume CLIENT.nume%TYPE := 'sp_nume';
1191 BEGIN
1192     EX13_PACK.EX6_PROC(v_nume);
1193 END;
1194 /
1195
1196 -- PENTRU EX13_PACK.EX7_FUNC
1197 DECLARE
1198     v_id_ang ANGAJAT.id_angajat%TYPE := sp_id;
1199     v_nr_cli_asist_ang NUMBER;
1200
1201     v_nr_ang NUMBER;
1202 BEGIN
1203     SELECT COUNT(id_angajat)
1204     INTO v_nr_ang
1205     FROM ANGAJAT;
1206
1207     SELECT id_angajat
1208     INTO v_id_ang
1209     FROM ANGAJAT
1210     WHERE id_angajat = v_id_ang;
1211
1212     IF EX13_PACK.EX7_FUNC(v_id_ang, v_nr_cli_asist_ang) < v_nr_ang
1213     DBMS_OUTPUT.PUT_LINE('Status personal:');
```

Id angajat: 1  
Nume angajat: Aurel Tudor-Mihai

Clienti asistati de catre angajat:  
Carstea Darian-Stefan  
Manole Iuliana-Elena  
Aurel Popescu-Mihai  
Castan Mirela

Id angajat: 2  
Nume angajat: Mihail Maria-Andreea

Clienti asistati de catre angajat:  
Carstea Darian-Stefan  
Manole Iuliana-Elena  
Aurel Popescu-Mihai  
Castan Mirela  
Popa Marcel-Radu

Id angajat: 3  
Nume angajat: Vladoi Rares

Clienti asistati de catre angajat:  
Popa Marcel-Radu

Status personal:  
Personal activ!

Numarul de clienti asistati de catre angajatul cu id-ul 1: 4.

Script Output x

Task completed in 2.304 seconds

DBMS\_OUTPUT.PUT\_LINE('A aparut alta eroare!');

END;

PL/SQL procedure successfully completed.

Worksheet Query Builder

```
1181 -- Cerinta 13.:
1182
1183 CREATE OR REPLACE PACKAGE EX13_PACK...
1184 /
1185 CREATE OR REPLACE PACKAGE BODY EX13_PACK...
1186 /
1187
1188 -- PENTRU EX13_PACK.EX6_PROC
1189 DECLARE
1190     v_nume CLIENT.nume%TYPE := 'sp_nume';
1191 BEGIN
1192     EX13_PACK.EX6_PROC(v_nume);
1193 END;
1194 /
1195
1196 -- PENTRU EX13_PACK.EX7_FUNC
1197 DECLARE
1198     v_id_ang ANGAJAT.id_angajat%TYPE := sp_id;
1199     v_nr_cli_asist_ang NUMBER;
1200
1201     v_nr_ang NUMBER;
1202 BEGIN
1203     SELECT COUNT(id_angajat)
1204     INTO v_nr_ang
1205     FROM ANGAJAT;
1206
1207     SELECT id_angajat
1208     INTO v_id_ang
1209     FROM ANGAJAT
1210     WHERE id_angajat = v_id_ang;
1211
1212     IF EX13_PACK.EX7_FUNC(v_id_ang, v_nr_cli_asist_ang) < v_nr_ang/2 THEN
1213     DBMS_OUTPUT.PUT_LINE('Status personal:');
```

Nu exista niciun angajat cu id-ul 1111!

Script Output x

Task completed in 1.684 seconds

DBMS\_OUTPUT.PUT\_LINE('A aparut alta eroare!');

END;

PL/SQL procedure successfully completed.

Worksheet Query Builder

```
1564 DBMS_OUTPUT.PUT_LINE('Numarul de clienti asistati de catre angajatul cu id-ul ' || v_id_angajat);
1565
1566 EXCEPTION
1567 WHEN NO_DATA_FOUND THEN
1568     DBMS_OUTPUT.PUT_LINE('Nu exista niciun angajat cu id-ul ' || v_id_angajat);
1569 WHEN OTHERS THEN
1570     DBMS_OUTPUT.PUT_LINE('A aparut alta eroare!');
1571 END;
1572 /
1573
1574 -- PENTRU EX13_PACK.EX8_FUNC
1575
1576 DECLARE
1577     v_nume_utilizator CONT_SITE.nume_utilizator%TYPE := 'ap_nume_utilizator';
1578     v_nr_recenzii NUMBER;
1579 BEGIN
1580     v_nr_recenzii := EX13_PACK.EX8_FUNC(v_nume_utilizator);
1581
1582     IF v_nr_recenzii > 0 THEN
1583         IF v_nr_recenzii > 1 THEN
1584             DBMS_OUTPUT.PUT_LINE('Contul cu numele de utilizator ' || v_nume_utilizator
1585                                 || ' a scris ' || v_nr_recenzii || ' recenzii.');
1586         ELSE
1587             DBMS_OUTPUT.PUT_LINE('Contul cu numele de utilizator ' || v_nume_utilizator
1588                                 || ' a scris ' || v_nr_recenzii || ' recenzie.');
1589         END IF;
1590     END IF;
1591 END;
1592 /
1593 -- PENTRU EX13_PACK.EX9_PROC
1594 DECLARE
1595     v_nume_produc PRODUS.denumire%TYPE := 'ap_nume_produc';
1596 BEGIN
1597     EX13_PACK.EX9_PROC(v_nume_produc);
1598 END;
1599 /
1600
```

Script Output x

Task completed in 5.386 seconds

END IF;  
END;  
  
PL/SQL procedure successfully completed.

Alex\_Proiect\_SGBD x

Id cont: 1  
Nume utilizator cont: Utilizator Mihai  
Contul nu este detinut de catre un client cunoscut.

Top 3 recenzii ale contului:  
1. Id recenzie: 1 | Nr. stele: 5 | Nume produs: Placa video Palit Geforce RTX 3080  
2. Id recenzie: 2 | Nr. stele: 5 | Nume produs: Procesor AMD Ryzen 5 3600  
3. Id recenzie: 3 | Nr. stele: 4 | Nume produs: Memorie Corsair Vengeance

Contul cu numele de utilizator Utilizator Mihai a scris 4 recenzii.

Worksheet Query Builder

```
1564 DBMS_OUTPUT.PUT_LINE('Numarul de clienti asistati de catre angajatul cu id-ul ' || v_id_angajat);
1565
1566 EXCEPTION
1567 WHEN NO_DATA_FOUND THEN
1568     DBMS_OUTPUT.PUT_LINE('Nu exista niciun angajat cu id-ul ' || v_id_angajat);
1569 WHEN OTHERS THEN
1570     DBMS_OUTPUT.PUT_LINE('A aparut alta eroare!');
1571 END;
1572 /
1573
1574 -- PENTRU EX13_PACK.EX8_FUNC
1575
1576 DECLARE
1577     v_nume_utilizator CONT_SITE.nume_utilizator%TYPE := 'ap_nume_utilizator';
1578     v_nr_recenzii NUMBER;
1579 BEGIN
1580     v_nr_recenzii := EX13_PACK.EX8_FUNC(v_nume_utilizator);
1581
1582     IF v_nr_recenzii > 0 THEN
1583         IF v_nr_recenzii > 1 THEN
1584             DBMS_OUTPUT.PUT_LINE('Contul cu numele de utilizator ' || v_nume_utilizator
1585                                 || ' a scris ' || v_nr_recenzii || ' recenzii.');
1586         ELSE
1587             DBMS_OUTPUT.PUT_LINE('Contul cu numele de utilizator ' || v_nume_utilizator
1588                                 || ' a scris ' || v_nr_recenzii || ' recenzie.');
1589         END IF;
1590     END IF;
1591 END;
1592 /
1593 -- PENTRU EX13_PACK.EX9_PROC
1594 DECLARE
1595     v_nume_produc PRODUS.denumire%TYPE := 'ap_nume_produc';
1596 BEGIN
1597     EX13_PACK.EX9_PROC(v_nume_produc);
1598 END;
1599 /
1600
```

Script Output x

Task completed in 7.022 seconds

END IF;  
END;  
  
PL/SQL procedure successfully completed.

Alex\_Proiect\_SGBD x

Id cont: 4  
Nume utilizator cont: ANTON PAVEL MIHAI  
Contul nu este detinut de catre un client cunoscut.

Utilizatorul nu a scris nicio recenzie.





Worksheet Query Builder

```

1563 DBMS_OUTPUT.NEW_LINE;
1564
1565 DBMS_OUTPUT.PUT_LINE('Numarul de clienti asistati de catre angajatul cu id-ul ' || v_id_ang);
1566 EXCEPTION
1567 WHEN NO_DATA_FOUND THEN
1568 DBMS_OUTPUT.PUT_LINE('Nu exista niciun angajat cu id-ul ' || v_id_ang || '!');
1569 WHEN OTHERS THEN
1570 DBMS_OUTPUT.PUT_LINE('A aparut alta eroare!');
1571 END;
1572 /
1573
1574
1575 -- PENTRU EX13_PACK.EX8_FUNC
1576 DECLARE
1577 v_num_utilizator CONT_SITE.num_utilizator%TYPE := 'ap_num_utilizator';
1578 v_nr_recenzii NUMBER;
1579 BEGIN
1580 v_nr_recenzii := EX13_PACK.EX8_FUNC(v_num_utilizator);
1581
1582 IF v_nr_recenzii > 0 THEN
1583 IF v_nr_recenzii > 1 THEN
1584 DBMS_OUTPUT.PUT_LINE('Contul cu numele de utilizator ' || v_num_utilizator || ' are ' || v_nr_recenzii || ' recenzii');
1585 ELSE
1586 DBMS_OUTPUT.PUT_LINE('Contul cu numele de utilizator ' || v_num_utilizator || ' are ' || v_nr_recenzii || ' recenzii');
1587 END IF;
1588 END IF;
1589 END;
1590 /
1591
1592
1593 -- PENTRU EX13_PACK.EX9_PROC
1594 DECLARE
1595 v_num_produs PRODUS.denumire%TYPE := 'ap_num_produs';
1596 BEGIN
1597 EX13_PACK.EX9_PROC(v_num_produs);
1598 END;
1599 /
1600

```

Script Output x Query Result x

Task completed in 0.876 seconds

```

EX13_PACK.EX9_PROC(v_num_produs);
END;

PL/SQL procedure successfully completed.

```

Alex\_Proiect\_SGBD x

Id produs: 40  
Nume produs: Hard disk Seagate BarraCuda 2TB

Lista serviciilor compatibile cu produsul:  
Nu exista servicii compatibile!

Worksheet Query Builder

```

1563 DBMS_OUTPUT.NEW_LINE;
1564
1565 DBMS_OUTPUT.PUT_LINE('Numarul de clienti asistati de catre angajatul cu id-ul ' || v_id_ang);
1566 EXCEPTION
1567 WHEN NO_DATA_FOUND THEN
1568 DBMS_OUTPUT.PUT_LINE('Nu exista niciun angajat cu id-ul ' || v_id_ang || '!');
1569 WHEN OTHERS THEN
1570 DBMS_OUTPUT.PUT_LINE('A aparut alta eroare!');
1571 END;
1572 /
1573
1574
1575 -- PENTRU EX13_PACK.EX8_FUNC
1576 DECLARE
1577 v_num_utilizator CONT_SITE.num_utilizator%TYPE := 'ap_num_utilizator';
1578 v_nr_recenzii NUMBER;
1579 BEGIN
1580 v_nr_recenzii := EX13_PACK.EX8_FUNC(v_num_utilizator);
1581
1582 IF v_nr_recenzii > 0 THEN
1583 IF v_nr_recenzii > 1 THEN
1584 DBMS_OUTPUT.PUT_LINE('Contul cu numele de utilizator ' || v_num_utilizator || ' are ' || v_nr_recenzii || ' recenzii');
1585 ELSE
1586 DBMS_OUTPUT.PUT_LINE('Contul cu numele de utilizator ' || v_num_utilizator || ' are ' || v_nr_recenzii || ' recenzii');
1587 END IF;
1588 END IF;
1589 END;
1590 /
1591
1592
1593 -- PENTRU EX13_PACK.EX9_PROC
1594 DECLARE
1595 v_num_produs PRODUS.denumire%TYPE := 'ap_num_produs';
1596 BEGIN
1597 EX13_PACK.EX9_PROC(v_num_produs);
1598 END;
1599 /
1600

```

Script Output x Query Result x

Task completed in 0.551 seconds

```

EX13_PACK.EX9_PROC(v_num_produs);
END;

PL/SQL procedure successfully completed.

```

Alex\_Proiect\_SGBD x

Id produs: 10  
Nume produs: Procesor AMD Ryzen 5 3600 3.6GHz box

Lista serviciilor compatibile cu produsul:

Id serviciu: 1  
Nume serviciu: Serviciu Asamblare Standard

Lista clientilor care au cumparat serviciul:  
Nu exista clienti care au cumparat serviciul.

Id serviciu: 3  
Nume serviciu: Serviciu Montare Componenta

Lista clientilor care au cumparat serviciul:  
Id client: 6 | Nume client: Florian Andrei-Cosmin  
Id client: 4 | Nume client: Popa Marcel-Radu

Id serviciu: 2  
Nume serviciu: Serviciu Testare Produs

Lista clientilor care au cumparat serviciul:  
Id client: 1 | Nume client: Aurel Popescu-Mihai  
Id client: 2 | Nume client: Carstea Darian-Stefan  
Id client: 3 | Nume client: Castan Mirela  
Id client: 6 | Nume client: Florian Andrei-Cosmin  
Id client: 5 | Nume client: Manole Iuliana-Elena  
Id client: 4 | Nume client: Popa Marcel-Radu

Worksheet Query Builder

```
1568 DBMS_OUTPUT.PUT_LINE('Nu exista niciun angajat cu id-ul ' || v_id_angajat);
1569 WHEN OTHERS THEN
1570     DBMS_OUTPUT.PUT_LINE('A aparut alta eroare!');
1571 END;
1572 /
1573
1574
1575 -- PENTRU EX13_PACK.EX8_FUNC
1576 DECLARE
1577     v_nume_utilizator CONT_SITE.nume_utilizator%TYPE := 'sp_nume_utilizator';
1578     v_nr_recenzii NUMBER;
1579 BEGIN
1580     v_nr_recenzii := EX13_PACK.EX8_FUNC(v_nume_utilizator);
1581
1582     IF v_nr_recenzii > 0 THEN
1583         IF v_nr_recenzii > 1 THEN
1584             DBMS_OUTPUT.PUT_LINE('Contul cu numele de utilizator ' || v_nume_utilizator);
1585         ELSE
1586             DBMS_OUTPUT.PUT_LINE('Contul cu numele de utilizator ' || v_nume_utilizator);
1587         END IF;
1588     END IF;
1589 END;
1590 /
1591
1592
1593 -- PENTRU EX13_PACK.EX9_PROC
1594 DECLARE
1595     v_nume_produus PRODUS.denumire%TYPE := 'sp_nume_produus';
1596 BEGIN
1597     EX13_PACK.EX9_PROC(v_nume_produus);
1598 END;
1599 /
1600
```

Script Output x Query Result x

Task completed in 5.328 seconds

END;  
Error report -  
ORA-20000: Nu exista niciun produs cu numele luisseabrf!  
ORA-06512: at "ALEXSGBD.EX13\_PACK", line 325  
ORA-06512: at line 4  
20000. 00000 - "%s"  
Cause: The stored procedure 'raise\_application\_error' was called which causes this error to be generated.  
Action: Correct the problem as described in the error message or contact the application administrator or DBA for more information.

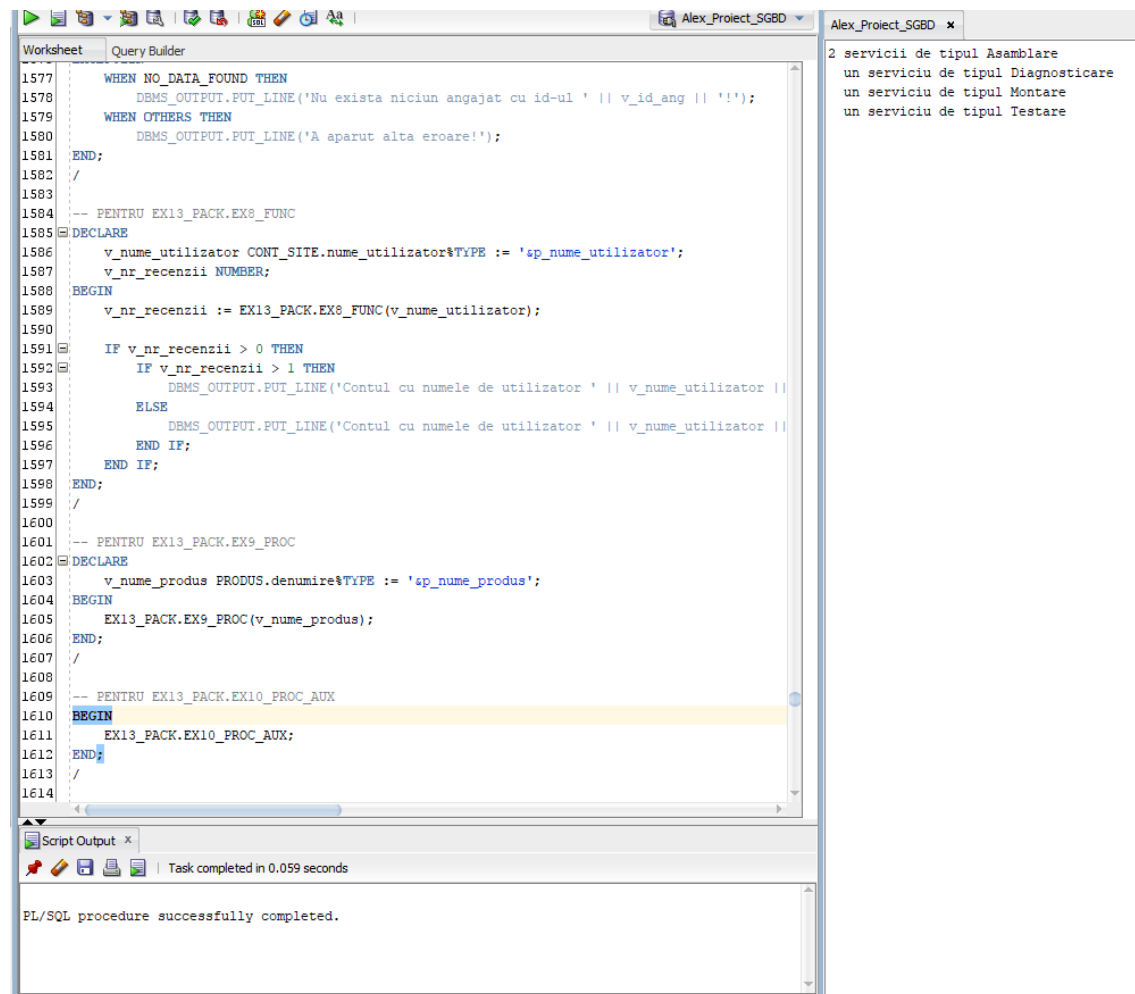
Worksheet Query Builder

```
1568 DBMS_OUTPUT.PUT_LINE('Nu exista niciun angajat cu id-ul ' || v_id_angajat);
1569 WHEN OTHERS THEN
1570     DBMS_OUTPUT.PUT_LINE('A aparut alta eroare!');
1571 END;
1572 /
1573
1574
1575 -- PENTRU EX13_PACK.EX8_FUNC
1576 DECLARE
1577     v_nume_utilizator CONT_SITE.nume_utilizator%TYPE := 'sp_nume_utilizator';
1578     v_nr_recenzii NUMBER;
1579 BEGIN
1580     v_nr_recenzii := EX13_PACK.EX8_FUNC(v_nume_utilizator);
1581
1582     IF v_nr_recenzii > 0 THEN
1583         IF v_nr_recenzii > 1 THEN
1584             DBMS_OUTPUT.PUT_LINE('Contul cu numele de utilizator ' || v_nume_utilizator);
1585         ELSE
1586             DBMS_OUTPUT.PUT_LINE('Contul cu numele de utilizator ' || v_nume_utilizator);
1587         END IF;
1588     END IF;
1589 END;
1590 /
1591
1592
1593 -- PENTRU EX13_PACK.EX9_PROC
1594 DECLARE
1595     v_nume_produus PRODUS.denumire%TYPE := 'sp_nume_produus';
1596 BEGIN
1597     EX13_PACK.EX9_PROC(v_nume_produus);
1598 END;
1599 /
1600
```

Script Output x Query Result x

Task completed in 0.551 seconds

v\_nume\_produus PRODUS.denumire%TYPE := 'sp\_nume\_produus';  
BEGIN  
EX13\_PACK.EX9\_PROC(v\_nume\_produus);  
END;  
Error report -  
ORA-20001: Exista mai multe produse cu numele Adaptor Gembird lx HDMI 1.4 Male - lx V  
ORA-06512: at "ALEXSGBD.EX13\_PACK", line 327  
ORA-06512: at line 4



14. Pachet care include tipuri de date complexe și obiecte necesare unui flux de acțiuni integrate, specifice bazei de date definite:

-- Cerinta 14.:

-- Enunt:

-- Definiti un pachet, care contine minimum 2 tipuri de date, minimum 2 functii, si minimum 2 proceduri.

-- Prin intermediul pachetului, determinati, pentru un produs al carui id este dat de la tastatura, daca

-- i se poate aplica vreuna din reducerile:

-- - In caz ca produsul are cel mult 10 recenzii, iar numarul mediu de stele al sau este cel mult 3,

-- i se va aplica o reducere de 10%. In caz contrar, produsul nu este eligibil pentru reducere;

-- - In caz ca produsul a fost cumparat de cel mult 10 clienti, i se va aplica o reducere de 10%.

-- In caz contrar, produsul nu este eligibil pentru reducere.

-- Sa se afiseze date despre produs, cat si despre recenziiile si comenzile sale.

CREATE OR REPLACE PACKAGE EX14\_PACK

IS

TYPE DATE\_RECENZIE\_PROD IS RECORD

(id\_recenzie RECENZIE.id\_recenzie%TYPE,  
stele\_recenzie RECENZIE.nr\_stele%TYPE);

TYPE tab\_ind\_date\_recenzii\_prod IS TABLE OF DATE\_RECENZIE\_PROD

INDEX BY PLS\_INTEGER;

t\_date\_recenzii\_prod tab\_ind\_date\_recenzii\_prod;

v\_total\_nr\_stele\_prod NUMBER;

TYPE DATE\_COMANDA\_PROD IS RECORD

(id\_comanda COMANDA.id\_comanda%TYPE,  
pret\_comanda COMANDA.pret\_total%TYPE);

TYPE tab\_ind\_date\_comenzi\_prod IS TABLE OF DATE\_COMANDA\_PROD

INDEX BY PLS\_INTEGER;

t\_date\_comenzi\_prod tab\_ind\_date\_comenzi\_prod;

TYPE DATE\_PRODUS IS RECORD

(id\_produs PRODUS.id\_produs%TYPE,

nume\_produs PRODUS.denumire%TYPE,

nume\_categorie\_produs CATEGORIE.denumire%TYPE,

pret\_produs PRODUS.pret%TYPE);

v\_date\_produs DATE\_PRODUS;

FUNCTION EX14\_FUNC1

(id\_prod PRODUS.id\_produs%TYPE)

RETURN NUMBER;

FUNCTION EX14\_FUNC2

(id\_prod PRODUS.id\_produs%TYPE)

RETURN NUMBER;

PROCEDURE EX14\_PROC1

(id\_prod IN OUT PRODUS.id\_produs%TYPE);

PROCEDURE EX14\_PROC2

(id\_prod PRODUS.id\_produs%TYPE,

id\_apel NUMBER);

END EX14\_PACK;

/

```
CREATE OR REPLACE PACKAGE BODY EX14_PACK
```

```
IS
```

```
    PROCEDURE EX14_PROC2
```

```
        (id_prod PRODUS.id_produs%TYPE,
```

```
        id_apel NUMBER)
```

```
    IS
```

```
        exceptie EXCEPTION;
```

```
    BEGIN
```

```
        IF id_apel = 1 THEN
```

```
            IF (t_date_recenzii_prod.COUNT <= 10) AND  
(v_total_nr_stele_prod/t_date_recenzii_prod.COUNT <= 3) THEN
```

```
                UPDATE PRODUS
```

```
                SET pret = pret * 0.9
```

```
                WHERE id_produs = id_prod;
```

```
                DBMS_OUTPUT.PUT_LINE('Produsul cu id-ul ' || id_prod || ' a primit o reducere a  
pretului de 10%, deoarece are un numar de recenzii mai mic sau egal cu 10, iar media  
numarului de stele este mai mica sau egala cu 3.');
```

```
                DBMS_OUTPUT.PUT_LINE('Medie nr. stele: ' ||  
ROUND(v_total_nr_stele_prod/t_date_recenzii_prod.COUNT, 2));
```

```
            ELSE
```

```
                DBMS_OUTPUT.PUT_LINE('Produsul nu este eligibil pentru reducere, deoarece fie  
are un numar de recenzii mai mare decat 10, fie media numarului de stele este mai mare decat  
3, fie ambele.');
```

```
                DBMS_OUTPUT.PUT_LINE('Medie nr. stele: ' ||  
ROUND(v_total_nr_stele_prod/t_date_recenzii_prod.COUNT, 2));
```

```
            END IF;
```

```

ELSIF id_apel = 2 THEN

    IF t_date_comenzi_prod.COUNT <= 10 THEN

        UPDATE PRODUS

        SET pret = pret * 0.9

        WHERE id_produs = id_prod;

        DBMS_OUTPUT.PUT_LINE('Produsul cu id-ul ' || id_prod || ' a primit o reducere a
pretului de 10%, deoarece a fost cumparat de catre un numar mai mic sau egal cu 10 clienti.');
```

ELSE

```

        DBMS_OUTPUT.PUT_LINE('Produsul nu este eligibil pentru reducere, deoarece a
fost cumparat de catre un numar mai mare de 10 clienti.');
```

END IF;

```

ELSIF id_apel <> 1 OR id_apel <> 2 THEN

    RAISE exceptie;

END IF;

EXCEPTION

    WHEN exceptie THEN

        RAISE_APPLICATION_ERROR(-20003, 'Id-ul apelului este incorect!');
```

END EX14\_PROC2;

```

FUNCTION EX14_FUNC1

(id_prod PRODUS.id_produs%TYPE)

RETURN NUMBER

IS
```

```

v_recenzie_rec DATE_RECENZIE_PROD;

v_counter NUMBER := 0;

BEGIN

v_total_nr_stele_prod := 0;


FOR i IN (SELECT id_recenzie id
          FROM RECENZIE
          WHERE id_produs = id_prod) LOOP

v_counter := v_counter + 1;


SELECT id_recenzie, nr_stele
INTO v_recenzie_rec
FROM RECENZIE
WHERE id_recenzie = i.id;


t_date_recenzii_prod(v_counter) := v_recenzie_rec;

v_total_nr_stele_prod := v_total_nr_stele_prod + v_recenzie_rec.stele_recenzie;

END LOOP;


EX14_PROC2(id_prod, 1);


RETURN t_date_recenzii_prod.COUNT;

END EX14_FUNC1;

```



FUNCTION EX14\_FUNC2

(id\_prod PRODUS.id\_produs%TYPE)

RETURN NUMBER

IS

v\_comanda\_rec DATE\_COMANDA\_PROD;

v\_counter NUMBER := 0;

BEGIN

FOR i IN (SELECT id\_comanda id

FROM PRODUS p JOIN COMANDA c ON p.id\_produs = c.id\_produs

WHERE p.id\_produs = id\_prod) LOOP

v\_counter := v\_counter + 1;

SELECT id\_comanda, pret\_total

INTO v\_comanda\_rec

FROM COMANDA

WHERE id\_comanda = i.id;

t\_date\_comenzi\_prod(v\_counter) := v\_comanda\_rec;

END LOOP;

EX14\_PROC2(id\_prod, 2);

RETURN t\_date\_comenzi\_prod.COUNT;

```
END EX14_FUNC2;
```

```
PROCEDURE EX14_PROC1
```

```
(id_prod IN OUT PRODUS.id_produs%TYPE)
```

```
IS
```

```
BEGIN
```

```
SELECT id_produs, p.denumire, c.denumire, pret
```

```
INTO v_date_produs.id_produs, v_date_produs.ume_produs,  
v_date_produs.ume_categorie_produs, v_date_produs.pret_produs
```

```
FROM PRODUS p JOIN CATEGORIE c USING (id_categorie)
```

```
WHERE id_produs = id_prod;
```

```
DBMS_OUTPUT.PUT_LINE('Id produs: ' || v_date_produs.id_produs);
```

```
DBMS_OUTPUT.PUT_LINE('Nume produs: ' || v_date_produs.ume_produs);
```

```
DBMS_OUTPUT.PUT_LINE('Nume categorie: ' ||  
v_date_produs.ume_categorie_produs);
```

```
DBMS_OUTPUT.PUT_LINE('Pret produs: ' || ROUND(v_date_produs.pret_produs, 2));
```

```
DBMS_OUTPUT.NEW_LINE;
```

```
DBMS_OUTPUT.NEW_LINE;
```

```
DBMS_OUTPUT.PUT_LINE('Nr. recenzii produs = ' || EX14_FUNC1(id_prod) || ':');
```

```
FOR i IN t_date_recenzii_prod.FIRST..t_date_recenzii_prod.LAST LOOP
```

```
DBMS_OUTPUT.PUT_LINE(' Id recenzie: ' || t_date_recenzii_prod(i).id_recenzie || '  
| Nr. stele recenzie: ' || t_date_recenzii_prod(i).stele_recenzie);
```

```
END LOOP;
```

```
DBMS_OUTPUT.NEW_LINE;
```

```
DBMS_OUTPUT.PUT_LINE('Nr. comenzi produs = ' || EX14_FUNC2(id_prod) || ':');
```

```
FOR i IN t_date_comenzi_prod.FIRST..t_date_comenzi_prod.LAST LOOP
```

```
DBMS_OUTPUT.PUT_LINE(' Id comanda: ' || t_date_comenzi_prod(i).id_comanda  
|| ' | Pret comanda: ' || t_date_comenzi_prod(i).pret_comanda);
```

```
END LOOP;
```

```
DBMS_OUTPUT.NEW_LINE;
```

```
SELECT pret
```

```
INTO v_date_produs.pret_produs
```

```
FROM PRODUS
```

```
WHERE id_produs = id_prod;
```

```
DBMS_OUTPUT.PUT_LINE('Pretul final al produsului: ' ||  
ROUND(v_date_produs.pret_produs, 2));
```

```
t_date_recenzii_prod.DELETE;
```

```
t_date_comenzi_prod.DELETE;
```

```
EXCEPTION
```

```
WHEN NO_DATA_FOUND THEN
```

```

        RAISE_APPLICATION_ERROR(-20000, 'Nu exista niciun produs cu id-ul ' || id_prod ||
        '!');

    END EX14_PROC1;

    END EX14_PACK;

/

-- DROP PACKAGE EX14_PACK;

DECLARE

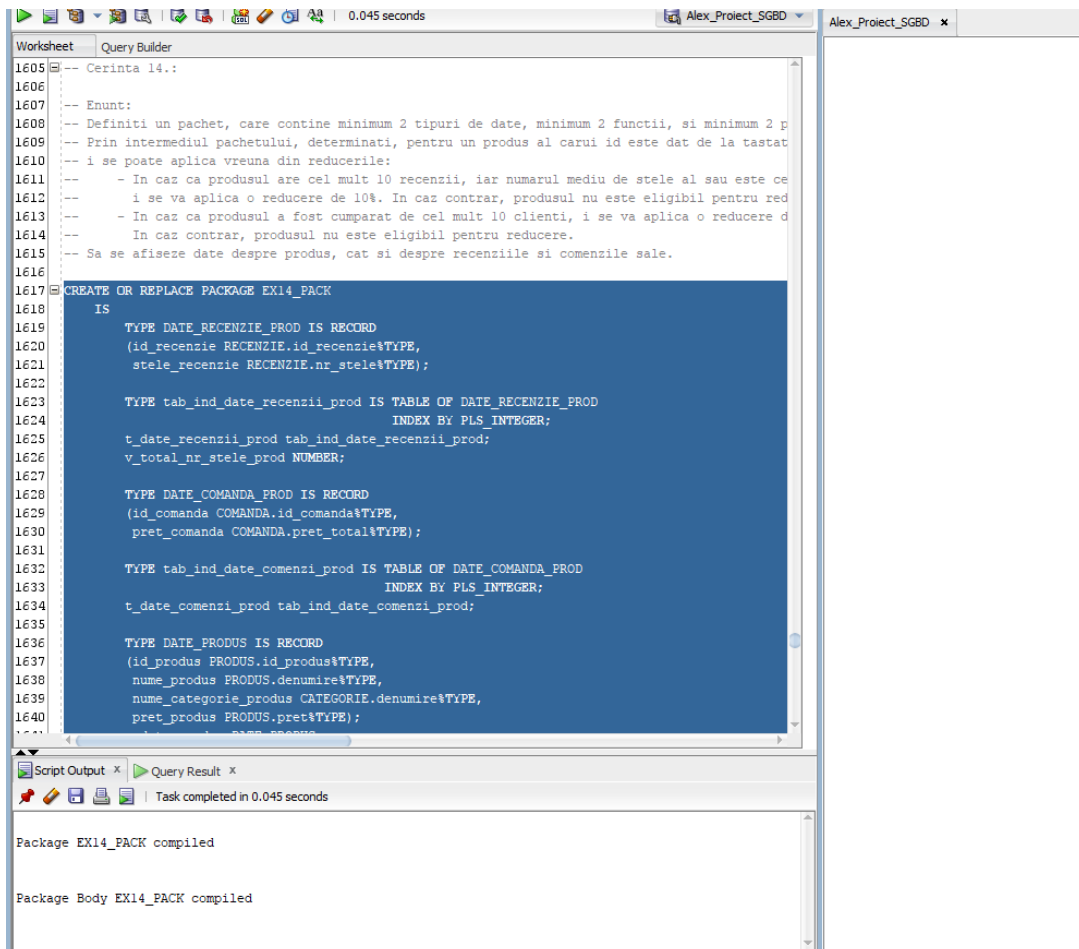
    v_id_produs PRODUS.id_produs%TYPE := 223;

BEGIN

    EX14_PACK.EX14_PROC1(v_id_produs);

END;

```



Worksheet Query Builder

```

1765 DBMS_OUTPUT.NEW_LINE;
1766 DBMS_OUTPUT.PUT_LINE('Nr. recenzii produs = ' || EX14
1767
1768 DBMS_OUTPUT.PUT_LINE('Nr. recenzii produs = ' || EX14
1769
1770 FOR i IN t_date_recenzii_prod.FIRST..t_date_recenzii
1771 DBMS_OUTPUT.PUT_LINE(' Id recenzie: ' || t_date
1772 END LOOP;
1773
1774 DBMS_OUTPUT.NEW_LINE;
1775
1776 DBMS_OUTPUT.PUT_LINE('Nr. comenzi produs = ' || EX14
1777
1778 FOR i IN t_date_comenzi_prod.FIRST..t_date_comenzi_px
1779 DBMS_OUTPUT.PUT_LINE(' Id comanda: ' || t_date_c
1780 END LOOP;
1781
1782 DBMS_OUTPUT.NEW_LINE;
1783 SELECT pret
1784 INTO v_date_produs.pret_produs
1785 FROM PRODUS
1786 WHERE id_produs = id_prod;
1787 DBMS_OUTPUT.PUT_LINE('Pretul final al produsului: '
1788
1789 t_date_recenzii_prod.DELETE;
1790 t_date_comenzi_prod.DELETE;
1791 EXCEPTION
1792 WHEN NO_DATA_FOUND THEN
1793 RAISE_APPLICATION_ERROR(-20000, 'Nu exista niciur
1794 END EX14_PROCI;
1795 END EX14_PACK;
1796 /
1797
1798 -- DROP PACKAGE EX14_PACK;
1799
1800 DECLARE
1801 v_id_produs PRODUS.id_produs%TYPE := 15;
1802 BEGIN
1803 EX14_PACK.EX14_PROCI(v_id_produs);
1804 END;

```

Script Output x Query Result x

Task completed in 0.033 seconds

PL/SQL procedure successfully completed.

Alex\_Proiect\_SGBD x

Id produs: 15  
 Nume produs: Adaptor Gembird 1x HDMI 1.4 Male - 1x VGA Femal  
 Nume categorie: Adaptoare  
 Pret produs: 35.99

Produsul cu id-ul 15 a primit o reducere a pretului de 10%,  
 Medie nr. stele: 3  
 Nr. recenzii produs = 2:  
 Id recenzie: 6 | Nr. stele recenzie: 3  
 Id recenzie: 7 | Nr. stele recenzie: 3

Produsul cu id-ul 15 a primit o reducere a pretului de 10%,  
 Nr. comenzi produs = 2:  
 Id comanda: 7 | Pret comanda: 71.98  
 Id comanda: 9 | Pret comanda: 35.99

Pretul final al produsului: 29.15

Worksheet Query Builder

```

1765 DBMS_OUTPUT.NEW_LINE;
1766 DBMS_OUTPUT.PUT_LINE('Nr. recenzii produs = ' || EX1
1767
1768 DBMS_OUTPUT.PUT_LINE('Nr. recenzii produs = ' || EX1
1769
1770 FOR i IN t_date_recenzii_prod.FIRST..t_date_recenzii
1771 DBMS_OUTPUT.PUT_LINE(' Id recenzie: ' || t_date
1772 END LOOP;
1773
1774 DBMS_OUTPUT.NEW_LINE;
1775
1776 DBMS_OUTPUT.PUT_LINE('Nr. comenzi produs = ' || EX14
1777
1778 FOR i IN t_date_comenzi_prod.FIRST..t_date_comenzi_p
1779 DBMS_OUTPUT.PUT_LINE(' Id comanda: ' || t_date_
1780 END LOOP;
1781
1782 DBMS_OUTPUT.NEW_LINE;
1783 SELECT pret
1784 INTO v_date_produs.pret_produs
1785 FROM PRODUS
1786 WHERE id_produs = id_prod;
1787 DBMS_OUTPUT.PUT_LINE('Pretul final al produsului: '
1788
1789 t_date_recenzii_prod.DELETE;
1790 t_date_comenzi_prod.DELETE;
1791 EXCEPTION
1792 WHEN NO_DATA_FOUND THEN
1793 RAISE_APPLICATION_ERROR(-20000, 'Nu exista niciu
1794 END EX14_PROCI;
1795 END EX14_PACK;
1796 /
1797
1798 -- DROP PACKAGE EX14_PACK;
1799
1800 DECLARE
1801 v_id_produs PRODUS.id_produs%TYPE := 5;
1802 BEGIN
1803 EX14_PACK.EX14_PROCI(v_id_produs);
1804 END;

```

Script Output x Query Result x

Task completed in 0.02 seconds

PL/SQL procedure successfully completed.

Alex\_Proiect\_SGBD x

Id produs: 5  
 Nume produs: Placa video Palit GeForce RTX 3090 GamingPro 240  
 Nume categorie: Placi video  
 Pret produs: 14999.99

Produsul nu este eligibil pentru reducere, deoarece fie are u  
 Medie nr. stele: 5  
 Nr. recenzii produs = 1:  
 Id recenzie: 1 | Nr. stele recenzie: 5

Produsul cu id-ul 5 a primit o reducere a pretului de 10%, de  
 Nr. comenzi produs = 2:  
 Id comanda: 1 | Pret comanda: 14999.99  
 Id comanda: 11 | Pret comanda: 14999.99

Pretul final al produsului: 13499.99

Worksheet Query Builder

```
1772      END LOOP;
1773
1774      DBMS_OUTPUT.NEW_LINE;
1775
1776      DBMS_OUTPUT.PUT_LINE('Nr. comenzi produs = ' || EX14_FUNC2(id_prod) || ':');
1777
1778      FOR i IN t_date_comenzi_prod.FIRST..t_date_comenzi_prod.LAST LOOP
1779          DBMS_OUTPUT.PUT_LINE('  Id comanda: ' || t_date_comenzi_prod(i).id_comanda || '
1780      END LOOP;
1781
1782      DBMS_OUTPUT.NEW_LINE;
1783      SELECT pret
1784      INTO v_date_produs.pret_produs
1785      FROM PRODUS
1786      WHERE id_produs = id_prod;
1787      DBMS_OUTPUT.PUT_LINE('Pretul final al produsului: ' || ROUND(v_date_produs.pret_produs
1788
1789      t_date_recenzii_prod.DELETE;
1790      t_date_comenzi_prod.DELETE;
1791      EXCEPTION
1792      WHEN NO_DATA_FOUND THEN
1793          RAISE_APPLICATION_ERROR(-20000, 'Nu exista niciun produs cu id-ul ' || id_prod |
1794      END EX14_PROCL;
1795      END EX14_PACK;
1796  /
1797
1798  -- DROP PACKAGE EX14_PACK;
1799
1800  DECLARE
1801      v_id_produs PRODUS.id_produs%TYPE := 223;
1802  BEGIN
1803      EX14_PACK.EX14_PROCL(v_id_produs);
1804  END;
```

Script Output x Query Result x

Task completed in 0.026 seconds

END;  
Error report -  
ORA-20000: Nu exista niciun produs cu id-ul 223!  
ORA-06512: at "ALEXSGBD.EX14\_PACK", line 137  
ORA-06512: at line 4  
20000. 00000 - "%s"  
\*Cause: The stored procedure 'raise\_application\_error'  
was called which causes this error to be generated.  
\*Action: Correct the problem as described in the error message or contact  
the application administrator or DBA for more information.