

Gestiunea unui site destinat vanzarii de tablouri

Aldea Alexia, grupa 244

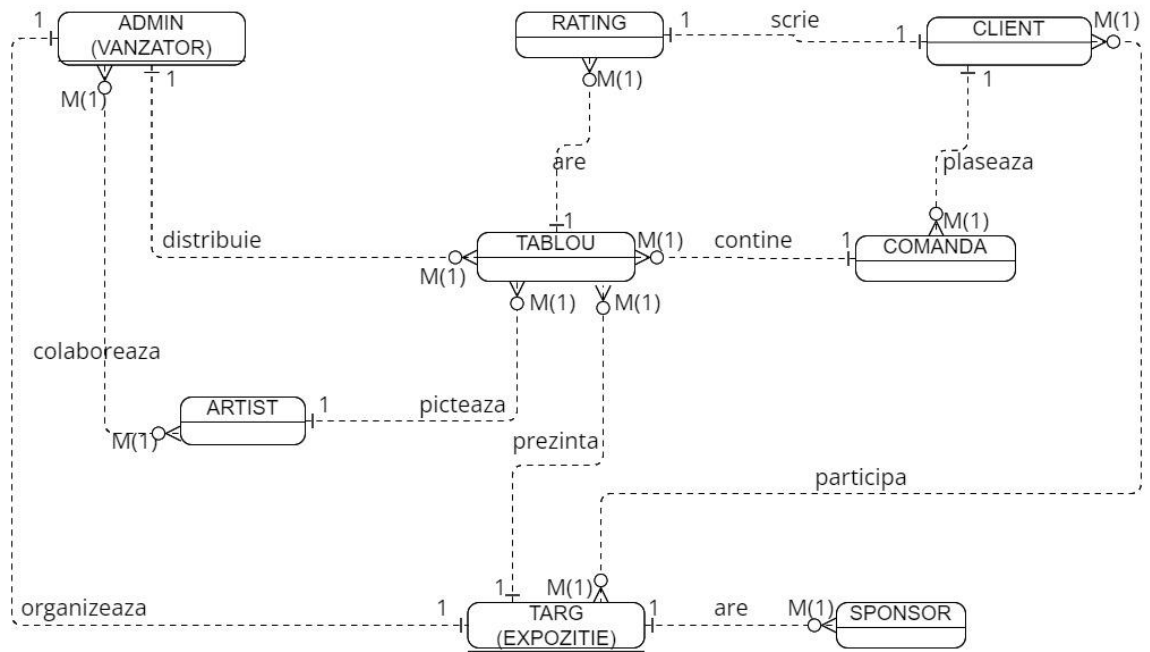
1. Prezentați pe scurt baza de date (utilitatea ei).

Baza de date contine informatii cu privire la tablourile realizate de diversi artisti contemporani, care sunt incredintate unui admin (al site-ului) si, mai apoi, sunt oferite spre vanzare celor pasionati de arta. Clientii pot lasa un rating (review) fiecarui tablou cumparat. De asemenea, acestia pot lua parte si la evenimentele (targurile) organizate de admin-ul site-ului, prin cumpararea unui bilet. Aceste evenimente sunt sustinute de diferiti sponsori. Scopul crearii acestei baze de date este de a putea tine evidenta comenzilor plasate pe site si evidenta tablourilor oferite de artisti spre vanzare. Astfel, micii artisti contemporani isi pot arata picturile lumii.

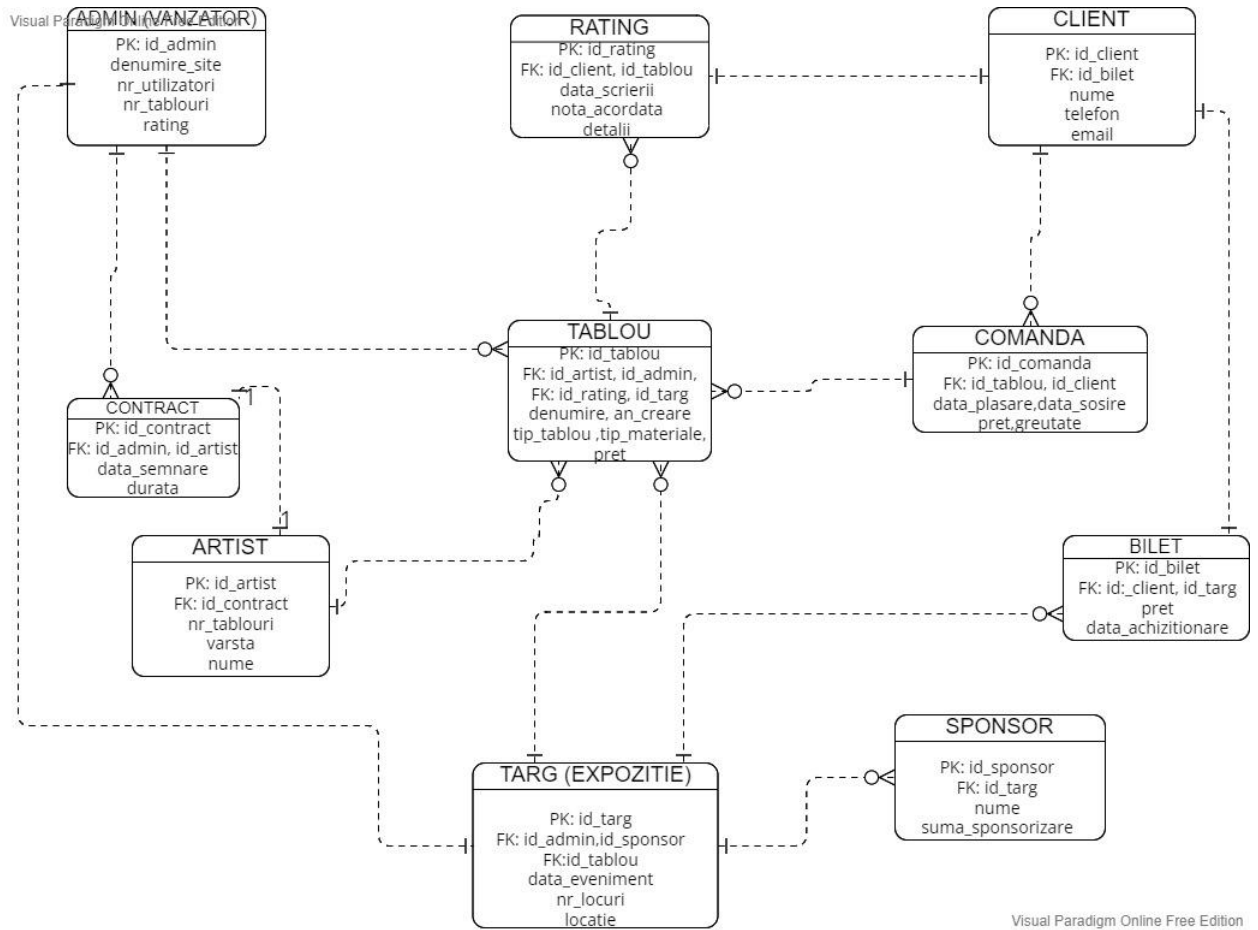
Un tablou este vandut admin-ului de catre artist, care, la randul sau, il pune la vanzare pe site, pentru a-l vedea si altii. Acesta poate fi comandat de catre clienti sau vandut direct la targuri/evenimente organizate de admin. De asemenea, un tablou poate avea unul sau mai multe rating-uri primite de la clienti.

Clientii pot plasa comenzi pe site, pot scrie review-uri tablourilor cumparate (un review per tablou) si isi pot cumpara bilet (un client => un bilet) pentru a lua parte la targ. Targurile sunt organizate periodic de catre admin.

2. Realizați diagrama entitate-relație (ERD).



3. Pornind de la diagrama entitate-relație realizați diagrama conceptuală a modelului propus, integrând toate atributele necesare.



4. Implementați în Oracle diagrama conceptuală realizată: definiți toate tabelele, implementând toate constrângerile de integritate necesare (chei primare, cheile externe etc).

-----ADMIN-----

```

CREATE TABLE ADMIN( id_admin NUMBER(5) CONSTRAINT pk_admin PRIMARY KEY,

denumire_site VARCHAR(100) CONSTRAINT denumire_site_admin NOT NULL,

nr_utilizatori NUMBER(5) CONSTRAINT nr_utilizatori_admin NOT NULL,

nr_tablouri NUMBER(5) CONSTRAINT nr_tablouri_admin NOT NULL,

rating NUMBER(5) CONSTRAINT rating_admin NOT NULL);
    
```

```
-----ADMIN-----
CREATE TABLE ADMIN( id_admin NUMBER(5) CONSTRAINT pk_admin PRIMARY KEY,
                     denumire_site VARCHAR(100) CONSTRAINT denumire_site_admin NOT NULL,
                     nr_utilizatori NUMBER(5) CONSTRAINT nr_utilizatori_admin NOT NULL,
                     nr_tablouri NUMBER(5) CONSTRAINT nr_tablouri_admin NOT NULL,
                     rating NUMBER(5) CONSTRAINT rating_admin NOT NULL);
```

Script Output x

Task completed in 0.165 seconds

Table ADMIN created.

-----ARTIST-----

```
CREATE TABLE ARTIST(id_artist NUMBER(5) CONSTRAINT pk_artist PRIMARY KEY,
                     nr_tablouri NUMBER(5) CONSTRAINT nr_tablouri_artist NOT NULL,
                     varsta NUMBER(5) CONSTRAINT varsta_artist NOT NULL,
                     nume VARCHAR(100) CONSTRAINT nume_artist NOT NULL);
```

```
-----ARTIST-----
CREATE TABLE ARTIST(id_artist NUMBER(5) CONSTRAINT pk_artist PRIMARY KEY,
                     nr_tablouri NUMBER(5) CONSTRAINT nr_tablouri_artist NOT NULL,
                     varsta NUMBER(5) CONSTRAINT varsta_artist NOT NULL,
                     nume VARCHAR(100) CONSTRAINT nume_artist NOT NULL);
```

Script Output x Query Result x

Task completed in 0.155 seconds

1 row inserted.

Commit complete.

Table ARTIST created.

-----CONTRACT-----

```
CREATE TABLE CONTRACT(id_contract NUMBER(5) CONSTRAINT pk_contract PRIMARY KEY,
                       data_semnare DATE CONSTRAINT data_semnare_contract NOT NULL,
                       durata VARCHAR(100) CONSTRAINT durata_contract NOT NULL,
```

```

        id_admin NUMBER(5), CONSTRAINT fk_contr FOREIGN KEY(id_admin) REFERENCES
ADMIN(id_admin),

        id_artist NUMBER(5), CONSTRAINT fk_cont FOREIGN KEY(id_artist) REFERENCES ARTIST(id_artist)

);

```

```

-----CONTRACT-----
CREATE TABLE CONTRACT(id_contract NUMBER(5) CONSTRAINT pk_contract PRIMARY KEY,
                        data_semnare DATE CONSTRAINT data_semnare_contract NOT NULL,
                        durata VARCHAR(100) CONSTRAINT durata_contract NOT NULL,
                        id_admin NUMBER(5),
                        CONSTRAINT fk_contr FOREIGN KEY(id_admin) REFERENCES ADMIN(id_admin),
                        id_artist NUMBER(5),
                        CONSTRAINT fk_cont FOREIGN KEY(id_artist) REFERENCES ARTIST(id_artist)
);

```

Script Output x Query Result x

Task completed in 0.071 seconds

*Cause:

*Action:

Table CONTRACT created.

```

-----CLIENT-----

CREATE TABLE CLIENT(id_client NUMBER(5) CONSTRAINT pk_client primary key,

                    nome VARCHAR(100) CONSTRAINT nome_client NOT NULL,

                    telefon VARCHAR(100) CONSTRAINT telefon_client NOT NULL,

                    email VARCHAR(100) CONSTRAINT email_client NOT NULL);

```

```

-----CLIENT-----
CREATE TABLE CLIENT(id_client NUMBER(5) CONSTRAINT pk_client primary key,
                    nome VARCHAR(100) CONSTRAINT nome_client NOT NULL,
                    telefon VARCHAR(100) CONSTRAINT telefon_client NOT NULL,
                    email VARCHAR(100) CONSTRAINT email_client NOT NULL);

```

Script Output x Query Result x

Task completed in 0.385 seconds

Table CLIENT created.

```

-----SPONSOR-----

CREATE TABLE SPONSOR(id_sponsor NUMBER(5) CONSTRAINT pk_sponsor primary key,

                    nome VARCHAR(100) CONSTRAINT nome_sponsor not null,

```

suma_sponsorizare NUMBER(5) CONSTRAINT suma_spons_sponsor not null);

```
-----SPONSOR-----
CREATE TABLE SPONSOR(id_sponsor NUMBER(5) CONSTRAINT pk_sponsor primary key,
                        nume VARCHAR(100) CONSTRAINT nume_sponsor not null,
                        suma_sponsorizare NUMBER(5) CONSTRAINT suma_spons_sponsor not null);

insert into sponsor
values(26, 'TotalSoft', 34000);

insert into sponsor
```

Script Output x Query Result x

Task completed in 0.038 seconds

Error report -
Unknown Command

Table SPONSOR created.

-----TARG/EXPOZITIE-----

```
CREATE TABLE TARG(id_targ NUMBER(5) CONSTRAINT pk_targ primary key,
                    data_eveniment DATE CONSTRAINT data_ev_targ not null,
                    nr_locuri NUMBER(5) CONSTRAINT nr_loc_targ not null,
                    locatie VARCHAR(100) CONSTRAINT locatie_targ not null,
                    id_admin NUMBER(5), CONSTRAINT fk_tr FOREIGN KEY (id_admin) REFERENCES ADMIN(id_admin),
                    id_sponsor NUMBER(5), CONSTRAINT fk_tg FOREIGN KEY (id_sponsor) REFERENCES
SPONSOR(id_sponsor) );
```

```
-----TARG/EXPOZITIE-----
CREATE TABLE TARG(id_targ NUMBER(5) CONSTRAINT pk_targ primary key,
                    data_eveniment DATE CONSTRAINT data_ev_targ not null,
                    nr_locuri NUMBER(5) CONSTRAINT nr_loc_targ not null,
                    locatie VARCHAR(100) CONSTRAINT locatie_targ not null,
                    id_admin NUMBER(5),
                    CONSTRAINT fk_tr FOREIGN KEY (id_admin) REFERENCES ADMIN(id_admin),
                    id_sponsor NUMBER(5),
                    CONSTRAINT fk_tg FOREIGN KEY (id_sponsor) REFERENCES SPONSOR(id_sponsor) );
```

Script Output x Query Result x

Task completed in 0.039 seconds

Table TARG created.

-----TABLOU-----

```

CREATE TABLE TABLOU(id_tablou NUMBER(5) CONSTRAINT pk_tablou primary key,

    denumire VARCHAR(100) CONSTRAINT denumire_tablou not null,

    an_creatoare NUMBER(5) CONSTRAINT an_cr_tablou not null,

    tip_tablou VARCHAR(100) CONSTRAINT tip_tabl_tablou not null,

    tip_materiale VARCHAR(100) CONSTRAINT tip_mat_tablou not null,

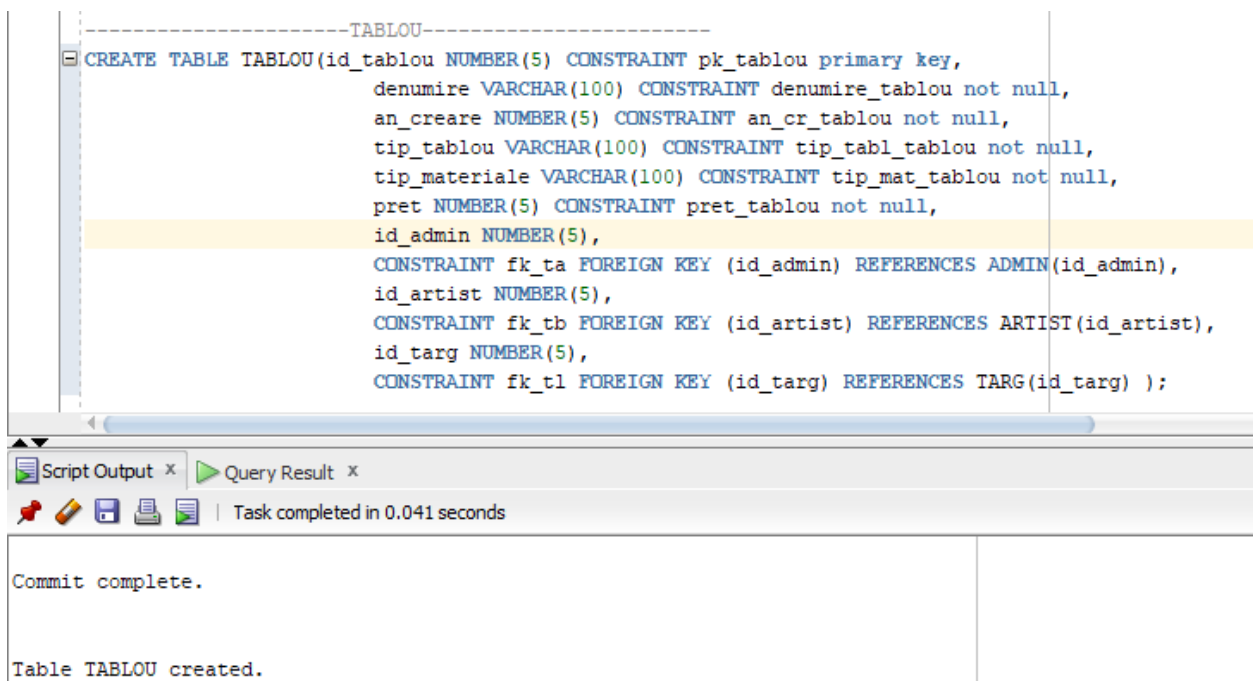
    pret NUMBER(5) CONSTRAINT pret_tablou not null,

    id_admin NUMBER(5), CONSTRAINT fk_ta FOREIGN KEY (id_admin) REFERENCES ADMIN(id_admin),

    id_artist NUMBER(5), CONSTRAINT fk_tb FOREIGN KEY (id_artist) REFERENCES ARTIST(id_artist),

    id_targ NUMBER(5), CONSTRAINT fk_tl FOREIGN KEY (id_targ) REFERENCES TARG(id_targ) );

```



```

-----TABLOU-----
CREATE TABLE TABLOU(id_tablou NUMBER(5) CONSTRAINT pk_tablou primary key,
    denumire VARCHAR(100) CONSTRAINT denumire_tablou not null,
    an_creatoare NUMBER(5) CONSTRAINT an_cr_tablou not null,
    tip_tablou VARCHAR(100) CONSTRAINT tip_tabl_tablou not null,
    tip_materiale VARCHAR(100) CONSTRAINT tip_mat_tablou not null,
    pret NUMBER(5) CONSTRAINT pret_tablou not null,
    id_admin NUMBER(5),
    CONSTRAINT fk_ta FOREIGN KEY (id_admin) REFERENCES ADMIN(id_admin),
    id_artist NUMBER(5),
    CONSTRAINT fk_tb FOREIGN KEY (id_artist) REFERENCES ARTIST(id_artist),
    id_targ NUMBER(5),
    CONSTRAINT fk_tl FOREIGN KEY (id_targ) REFERENCES TARG(id_targ) );

```

Script Output x Query Result x

Task completed in 0.041 seconds

Commit complete.

Table TABLOU created.

```

-----RATING-----

CREATE TABLE RATING(id_rating NUMBER(5) CONSTRAINT pk_rating primary key,

    data_scrierii DATE CONSTRAINT data_scr_rating not null,

    nota_acordata NUMBER(5) CONSTRAINT nota_ac_rating not null,

    detalii VARCHAR(100) CONSTRAINT detalii_rating not null,

    id_client NUMBER(5), CONSTRAINT fk_ra FOREIGN KEY (id_client) REFERENCES CLIENT(id_client),

    id_tablou NUMBER(5), CONSTRAINT fk_rt FOREIGN KEY (id_tablou) REFERENCES TABLOU(id_tablou) );

```



```
from tablou;
```

```
-----RATING-----
CREATE TABLE RATING(id_rating NUMBER(5) CONSTRAINT pk_rating primary key,
    data_scrierii DATE CONSTRAINT data_scr_rating not null,
    nota_acordata NUMBER(5) CONSTRAINT nota_ac_rating not null,
    detalii VARCHAR(100) CONSTRAINT detalii_rating not null,
    id_client NUMBER(5),
    CONSTRAINT fk_ra FOREIGN KEY (id_client) REFERENCES CLIENT(id_client),
    id_tablou NUMBER(5),
    CONSTRAINT fk_rt FOREIGN KEY (id_tablou) REFERENCES TABLOU(id_tablou) );

insert into rating aa
```

Script Output x Query Result x

Task completed in 0.062 seconds

Table RATING created.

-----COMANDA-----

```
CREATE TABLE COMANDA(id_comanda NUMBER(5) CONSTRAINT pk_comanda primary key,
    data_plasare DATE CONSTRAINT data_plas_comanda not null,
    data_sosire DATE CONSTRAINT data_sos_comanda not null,
    pret NUMBER(5) CONSTRAINT pret_comanda not null,
    greutate NUMBER(5) CONSTRAINT greutate_comanda not null,
    id_client NUMBER(5), CONSTRAINT fk_com FOREIGN KEY (id_client) REFERENCES CLIENT(id_client),
    id_tablou NUMBER(5), CONSTRAINT fk_co FOREIGN KEY (id_tablou) REFERENCES TABLOU(id_tablou) );
```

```
-----COMANDA-----
CREATE TABLE COMANDA(id_comanda NUMBER(5) CONSTRAINT pk_comanda primary key,
    data_plasare DATE CONSTRAINT data_plas_comanda not null,
    data_sosire DATE CONSTRAINT data_sos_comanda not null,
    pret NUMBER(5) CONSTRAINT pret_comanda not null,
    greutate NUMBER(5) CONSTRAINT greutate_comanda not null,
    id_client NUMBER(5),
    CONSTRAINT fk_com FOREIGN KEY (id_client) REFERENCES CLIENT(id_client),
    id_tablou NUMBER(5),
    CONSTRAINT fk_co FOREIGN KEY (id_tablou) REFERENCES TABLOU(id_tablou) );
```

Script Output x Query Result x

Task completed in 0.05 seconds

Commit complete.

Table COMANDA created.

-----BILET-----

```
CREATE TABLE BILET (id_bilet NUMBER(5) CONSTRAINT pk_bilet primary key,
    pret NUMBER(5) CONSTRAINT pret_bilet not null,
```

```

data_achizitionare DATE CONSTRAINT data_achiz_bilet not null,

id_client NUMBER(5), CONSTRAINT fk_bil FOREIGN KEY (id_client) REFERENCES CLIENT(id_client),

id_targ NUMBER(5), CONSTRAINT fk_blt FOREIGN KEY (id_targ) REFERENCES TARG(id_targ) );

```

The screenshot shows a database IDE with a SQL editor and a results pane. The SQL editor contains the following code:

```

-----BILET-----
CREATE TABLE BILET (id_bilet NUMBER(5) CONSTRAINT pk_bilet primary key,
    pret NUMBER(5) CONSTRAINT pret_bilet not null,
    data_achizitionare DATE CONSTRAINT data_achiz_bilet not null,
    id_client NUMBER(5),
    CONSTRAINT fk_bil FOREIGN KEY (id_client) REFERENCES CLIENT(id_client),
    id_targ NUMBER(5),
    CONSTRAINT fk_blt FOREIGN KEY (id_targ) REFERENCES TARG(id_targ) );

insert into bilet aa

```

The results pane shows the following output:

```

Script Output x Query Result x
Task completed in 0.04 seconds
1 row inserted.

Commit complete.

Table BILET created.

```

5. Adăugați informații coerente în tabelele create (minim 5 înregistrări pentru fiecare entitate independentă; minim 10 înregistrări pentru tabela asociativă).

```

insert into admin values(1, 'ProfiArt', 5000, 1243, 8);

insert into admin values(2, 'ArtaInPapuci', 234, 200, 6);

insert into admin values(3, 'SHL', 2323, 6463, 10);

insert into admin values(4, 'PictenzaCuNoi', 543, 234, 6);

insert into admin values(5, 'FloriVesele', 5426, 23556, 8);

```

```

insert into admin
values(1, 'ProfiArt', 5000, 1243, 8);

insert into admin
values(2, 'ArtaInPapuci', 234, 200, 6);

insert into admin
values(3, 'SHL', 2323, 6463, 10);

insert into admin
values(4, 'PictaazaCuNoi', 543, 234, 6);

insert into admin
values(5, 'FloriVesele', 5426, 23556, 8);

commit;
select *
from admin;

```

cript Output x Query Result x

SQL | All Rows Fetched: 5 in 0.045 seconds

ID_ADMIN	DENUMIRE_SITE	NR_UTILIZATORI	NR_TABL...	RATING
1	1 ProfiArt	5000	1243	8
2	2 ArtaInPapuci	234	200	6
3	3 SHL	2323	6463	10
4	4 PictaazaCuNoi	543	234	6
5	5 FloriVesele	5426	23556	8

insert into artist values(6, 300,45,'Popescu Valentin');

insert into artist values(7, 23, 19, 'Mirciulescu Cristina');

insert into artist values(8, 345, 65, 'Floarea Andrei');

insert into artist values(9, 543, 63, 'Dumitrescu Florin');

insert into artist values(10, 36, 25, 'Popa Alexandru');

```

insert into artist
values(6, 300,45, 'Popescu Valentin');

insert into artist
values(7, 23, 19, 'Mirciulescu Cristina');


insert into artist
values(8, 345, 65, 'Floarea Andrei');




insert into artist
values(9, 543, 63, 'Dumitrescu Florin');

insert into artist
values(10, 36, 25, 'Popa Alexandru');

commit;
select *
from artist;

```

cript Output x  Query Result x

   SQL | All Rows Fetched: 5 in 0.01 seconds

ID_ARTIST	NR_TABLOURI	VARSTA	NUME
1	6	300	45 Popescu Valentin
2	7	23	19 Mirciulescu Cristina
3	8	345	65 Floarea Andrei
4	9	543	63 Dumitrescu Florin
5	10	36	25 Popa Alexandru

```

insert into contract values(11, TO_DATE('26-01-2020','dd-mm-yyyy'), '3 ani',1,7);

insert into contract values(12, TO_DATE('14-01-2020','dd-mm-yyyy'), '5 ani', 2, 6);

insert into contract values(13, TO_DATE('27-09-2019','dd-mm-yyyy'), '6 ani', 3, 10);

insert into contract values(14, TO_DATE('12-06-2022', 'dd-mm-yyyy'), '6 luni', 4, 8);

insert into contract values(15, TO_DATE('15-08-2022', 'dd-mm-yyyy'), '1 an', 5, 9);

insert into contract values(16, TO_DATE('24-06-2022', 'dd-mm-yyyy'), '10 ani', 3, 7);

insert into contract values(17, TO_DATE('17-07-2021', 'dd-mm-yyyy'), '4 ani', 1, 6);

insert into contract values(18, TO_DATE('15-02-2021', 'dd-mm-yyyy'), '2 ani', 5, 8);

insert into contract values(19, TO_DATE('10-12-2020', 'dd-mm-yyyy'), '2 ani', 4,10);

```

insert into contract values(20, TO_DATE('11-10-2022', 'dd-mm-yyyy'), '3 luni', 3, 9);

```
insert into contract
values(11, TO_DATE('26-01-2020', 'dd-mm-yyyy'), '3 ani', 1, 7);

insert into contract
values(12, TO_DATE('14-01-2020', 'dd-mm-yyyy'), '5 ani', 2, 6);

insert into contract
values(13, TO_DATE('27-09-2019', 'dd-mm-yyyy'), '6 ani', 3, 10);

insert into contract
values(14, TO_DATE('12-06-2022', 'dd-mm-yyyy'), '6 luni', 4, 8);

insert into contract
values(15, TO_DATE('15-08-2022', 'dd-mm-yyyy'), '1 an', 5, 9);

insert into contract
values(16, TO_DATE('24-06-2022', 'dd-mm-yyyy'), '10 ani', 3, 7);

insert into contract
values(17, TO_DATE('17-07-2021', 'dd-mm-yyyy'), '4 ani', 1, 6);

insert into contract
values(18, TO_DATE('15-02-2021', 'dd-mm-yyyy'), '2 ani', 5, 8);

insert into contract
values(19, TO_DATE('10-12-2020', 'dd-mm-yyyy'), '2 ani', 4, 10);

insert into contract
```

Script Output x Query Result x

SQL | All Rows Fetched: 10 in 0.018 seconds

	ID_CONTRACT	DATA_SEMNARE	DURATA	ID_ADMIN	ID_ARTIST
1	11	26-JAN-20	3 ani	1	7
2	12	14-JAN-20	5 ani	2	6
3	13	27-SEP-19	6 ani	3	10
4	14	12-JUN-22	6 luni	4	8
5	15	15-AUG-22	1 an	5	9
6	16	24-JUN-22	10 ani	3	7
7	17	17-JUL-21	4 ani	1	6
8	18	15-FEB-21	2 ani	5	8
9	19	10-DEC-20	2 ani	4	10
10	20	11-OCT-22	3 luni	3	9

insert into client values(21, 'Dinculescu Cosmina', '0756234543', 'cosmina123@gmail.com');

insert into client values(22, 'Georgescu Diana', '0745387163', 'didi@gmail.com');

insert into client values(23, 'Anghelescu Radu', '0788576209', 'rodede@gmail.com');

insert into client values(24, 'Pentu Maria', '0746012463', 'mery34@gmai.com');

insert into client values(25, 'Florescu Marina', '0789076713', 'flory_marinush@gmail.com');

```
insert into client
values(21, 'Dinculescu Cosmina', '0756234543', 'cosminal23@gmail.com');

insert into client
values(22, 'Georgescu Diana', '0745387163', 'didi@gmail.com');

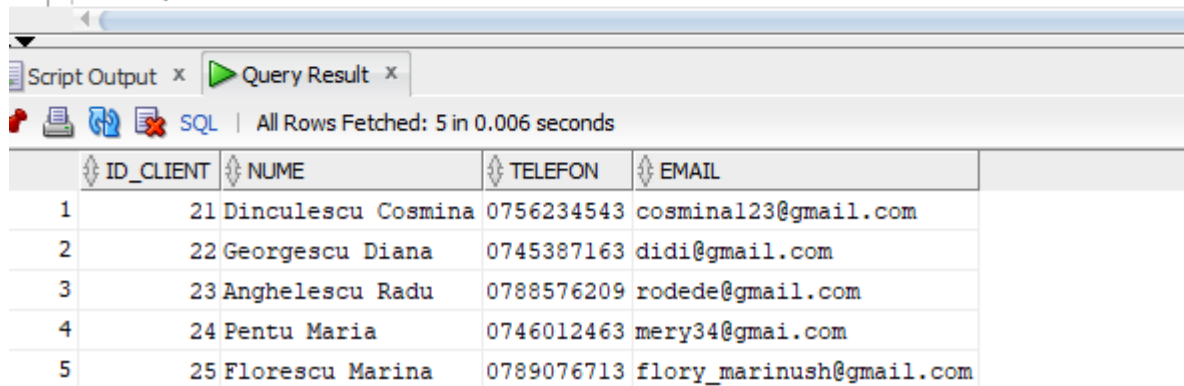
insert into client
values(23, 'Anghelescu Radu', '0788576209', 'rodede@gmail.com');

insert into client
values(24, 'Pentu Maria', '0746012463', 'mery34@gmai.com');

insert into client
values(25, 'Florescu Marina', '0789076713', 'flory_marinush@gmail.com');

select*
from client;

commit;
```



The screenshot shows a database client window with two tabs: 'Script Output' and 'Query Result'. The 'Query Result' tab is active, displaying the results of the SQL script. The results are shown in a table with 5 rows and 4 columns: ID_CLIENT, NUME, TELEFON, and EMAIL. The data is as follows:

ID_CLIENT	NUME	TELEFON	EMAIL
1	21 Dinculescu Cosmina	0756234543	cosminal23@gmail.com
2	22 Georgescu Diana	0745387163	didi@gmail.com
3	23 Anghelescu Radu	0788576209	rodede@gmail.com
4	24 Pentu Maria	0746012463	mery34@gmai.com
5	25 Florescu Marina	0789076713	flory_marinush@gmail.com

insert into sponsor values(26, 'TotalSoft', 34000);

insert into sponsor values(27, 'RedBull', 24000);

insert into sponsor values(28, 'HatzSRL', 4500);

insert into sponsor values(29, 'MaryandAlex', 7000);

insert into sponsor values(30, 'Ikea', 20000);

```

insert into sponsor
values(26, 'TotalSoft', 34000);

insert into sponsor
values(27, 'RedBull', 24000);

insert into sponsor
values(28, 'HatzSRL', 4500);

insert into sponsor
values(29, 'MaryandAlex', 7000);

insert into sponsor
values(30, 'Ikea', 20000);

```

Script Output x Query Result x			
SQL All Rows Fetched: 5 in 0.007 seconds			
	ID_SPONSOR	NUME	SUMA_SPONSORIZARE
1	26	TotalSoft	34000
2	27	RedBull	24000
3	28	HatzSRL	4500
4	29	MaryandAlex	7000
5	30	Ikea	20000

insert into targ values(31, TO_DATE('01-11-2022', 'dd-mm-yyyy'),400, 'Str Kogalniceanu 9', 1, 27);

insert into targ values(32, TO_DATE('1-12-2022', 'dd-mm-yyyy'), 2000,'Sala Palatului', 4, 30);

insert into targ values(33, TO_DATE('02-12-2022', 'dd-mm-yyyy'), 30000, 'Palatul Parlamentului', 2, 26);

insert into targ values(34, TO_DATE('15-01-2023', 'dd-mm-yyyy'), 2400, 'Sala Palatului', 3, 28);

insert into targ values(35, TO_DATE('01-02-2023', 'dd-mm-yyyy'), 450, 'Str Kogalniceanu 9', 5, 29);

```

insert into targ
values(31, TO_DATE('01-11-2022', 'dd-mm-yyyy'),400, 'Str Kogalniceanu 9', 1, 27);

insert into targ
values(32, TO_DATE('1-12-2022', 'dd-mm-yyyy'), 2000,'Sala Palatului', 4, 30);

insert into targ
values(33, TO_DATE('02-12-2022', 'dd-mm-yyyy'), 30000, 'Palatul Parlamentului', 2, 26);

insert into targ
values(34, TO_DATE('15-01-2023', 'dd-mm-yyyy'), 2400, 'Sala Palatului', 3, 28);

insert into targ
values(35, TO_DATE('01-02-2023', 'dd-mm-yyyy'), 450, 'Str Kogalniceanu 9', 5, 29);

```

Script Output x Query Result x

SQL | All Rows Fetched: 5 in 0.007 seconds

	ID_TARG	DATA_EVENIMENT	NR_LOCURI	LOCATIE	ID_ADMIN	ID_SPONSOR
1	31	01-NOV-22	400	Str Kogalniceanu 9	1	27
2	32	01-DEC-22	2000	Sala Palatului	4	30
3	33	02-DEC-22	30000	Palatul Parlamentului	2	26
4	34	15-JAN-23	2400	Sala Palatului	3	28
5	35	01-FEB-23	450	Str Kogalniceanu 9	5	29

insert into tablou values(36, 'Iarna', 2021, 'pictura', 'acrilice', 340, 1, 8,34);

insert into tablou values(37, 'Iarna la schi', 2022, 'desen', 'creion', 200, 4, 9, 31);

insert into tablou values(38, 'Flori de camp', 2020, 'pictura', 'ulei', 500, 2, 6, 35);

insert into tablou values(39, 'Nuante de gri', 2021, 'desen', 'creion', 250, 3,7, 32);

insert into tablou values(40, 'Foc de tabara', 2020, 'pictura', 'acrilice', 700, 5,10, 33);


```

insert into tablou
values(36, 'Iarna', 2021, 'pictura', 'acrilice', 340, 1, 8 ,34);

insert into tablou
values(37, 'Iarna la schi', 2022, 'desen', 'creion', 200, 4, 9, 31);

insert into tablou
values(38, 'Flori de camp', 2020, 'pictura', 'ulei', 500, 2, 6, 35);

insert into tablou
values(39, 'Nuante de gri', 2021, 'desen', 'creion', 250, 3,7, 32);

insert into tablou
values(40, 'Foc de tabara', 2020, 'pictura', 'acrilice', 700, 5,10, 33);

```

Script Output x Query Result x									
SQL All Rows Fetched: 5 in 0.374 seconds									
	ID_TABLOU	DENUMIRE	AN_CREARE	TIP_TABLOU	TIP_MATERIALE	PRET	ID_ADMIN	ID_ARTIST	ID_TARG
1	36	Iarna	2021	pictura	acrilice	340	1	8	34
2	37	Iarna la schi	2022	desen	creion	200	4	9	31
3	38	Flori de camp	2020	pictura	ulei	500	2	6	35
4	39	Nuante de gri	2021	desen	creion	250	3	7	32
5	40	Foc de tabara	2020	pictura	acrilice	700	5	10	33

insert into rating values(41, TO_DATE('01-02-2022','dd-mm-yyyy'), 7, 'foarte frumos', 25, 36);

insert into rating values(42, TO_DATE('01-10-2022','dd-mm-yyyy'), 10, 'impresionant', 22, 37);

insert into rating values(43, TO_DATE('13-04-2021','dd-mm-yyyy'), 8, 'o adevarata capodopera', 24, 38);

insert into rating values(44, TO_DATE('23-05-2022','dd-mm-yyyy'), 5, 'satisfacator', 21, 39);

insert into rating values(45, TO_DATE('01-12-2021','dd-mm-yyyy'), 10, 'sunt mare fan',23,40);

```

insert into rating
values(41, TO_DATE('01-02-2022','dd-mm-yyyy'), 7, 'foarte frumos', 25, 36);

insert into rating
values(42, TO_DATE('01-10-2022','dd-mm-yyyy'), 10, 'impresionant', 22, 37);

insert into rating
values(43, TO_DATE('13-04-2021','dd-mm-yyyy'), 8, 'o adevarata capodopera', 24, 38);

insert into rating
values(44, TO_DATE('23-05-2022','dd-mm-yyyy'), 5, 'satisfacator', 21, 39);

insert into rating
values(45, TO_DATE('01-12-2021','dd-mm-yyyy'), 10, 'sunt mare fan',23,40);

```

Script Output x Query Result x

SQL | All Rows Fetched: 5 in 0.016 seconds

ID_RATING	DATA_SCRIERII	NOTA_ACORDATA	DETALII	ID_CLIENT	ID_TABLOU
1	41	01-FEB-22	7 foarte frumos	25	36
2	42	01-OCT-22	10 impresionant	22	37
3	43	13-APR-21	8 o adevarata capodopera	24	38
4	44	23-MAY-22	5 satisfacator	21	39
5	45	01-DEC-21	10 sunt mare fan	23	40

insert into comanda values(46, TO_DATE('24-08-2022','dd-mm-yyyy'), TO_DATE('01-12-2022', 'dd-mm-yyyy'), 400, 5, 21, 37);

insert into comanda values(47, TO_DATE('25-10-2022','dd-mm-yyyy'), TO_DATE('05-12-2022','dd-mm-yyyy'), 625, 3, 24, 36);

insert into comanda values(48, TO_DATE('13-10-2022','dd-mm-yyyy'), TO_DATE('23-11-2022', 'dd-mm-yyyy'), 1000, 7, 22, 40);

insert into comanda values(49, TO_DATE('25-09-2022','dd-mm-yyyy'), TO_DATE('31-10-2022','dd-mm-yyyy'), 450, 2, 23,38);

insert into comanda values(50, TO_DATE('17-07-2022','dd-mm-yyyy'), TO_DATE('31-10-2022','dd-mm-yyyy'), 500, 3, 25, 39);

```

insert into comanda
values(46, TO_DATE('24-08-2022','dd-mm-yyyy'), TO_DATE('01-12-2022', 'dd-mm-yyyy'), 400, 5, 21, 37);

insert into comanda
values(47, TO_DATE('25-10-2022','dd-mm-yyyy'), TO_DATE('05-12-2022','dd-mm-yyyy'), 625, 3, 24, 36);

insert into comanda
values(48, TO_DATE('13-10-2022','dd-mm-yyyy'), TO_DATE('23-11-2022', 'dd-mm-yyyy'), 1000, 7, 22, 40);

insert into comanda
values(49, TO_DATE('25-09-2022','dd-mm-yyyy'), TO_DATE('31-10-2022','dd-mm-yyyy'), 450, 2, 23,38);

insert into comanda
values(50, TO_DATE('17-07-2022','dd-mm-yyyy'), TO_DATE('31-10-2022','dd-mm-yyyy'), 500, 3, 25, 39);

select*
from comanda:

```

Script Output x Query Result x

SQL | All Rows Fetched: 5 in 0.02 seconds

	ID_COMANDA	DATA_PLASARE	DATA_SOSIRE	PRET	GREUTATE	ID_CLIENT	ID_TABLOU
1	46	24-AUG-22	01-DEC-22	400	5	21	37
2	47	25-OCT-22	05-DEC-22	625	3	24	36
3	48	13-OCT-22	23-NOV-22	1000	7	22	40
4	49	25-SEP-22	31-OCT-22	450	2	23	38
5	50	17-JUL-22	31-OCT-22	500	3	25	39

```

insert into bilete values(51, 50, TO_DATE('31-07-2022','dd-mm-yyyy'), 22, 31);
insert into bilete values(52, 100, TO_DATE('01-10-2022','dd-mm-yyyy'), 21, 35);
insert into bilete values(53, 50, TO_DATE('23-06-2022', 'dd-mm-yyyy'), 24,33);
insert into bilete values(54, 150, TO_DATE('12-10-2022','dd-mm-yyyy'), 22, 34);
insert into bilete values(55, 65, TO_DATE('31-05-2022','dd-mm-yyyy'), 23, 32);
insert into bilete values(56, 50, TO_DATE('23-06-2022','dd-mm-yyyy'), 25, 34);
insert into bilete values(57, 100, TO_DATE('14-07-2022','dd-mm-yyyy'), 23, 31);
insert into bilete values(58, 55, TO_DATE('15-09-2022', 'dd-mm-yyyy'), 21, 35);
insert into bilete values(59, 150, TO_DATE('25-09-2022','dd-mm-yyyy'), 25, 33);
insert into bilete values(60, 50, TO_DATE('03-03-2022','dd-mm-yyyy'), 24, 31);

```

worksheet

Query Builder

```

insert into billet
values(51, 50, TO_DATE('31-07-2022','dd-mm-yyyy'), 22, 31);

insert into billet
values(52, 100, TO_DATE('01-10-2022','dd-mm-yyyy'), 21, 35);

insert into billet
values(53, 50, TO_DATE('23-06-2022', 'dd-mm-yyyy'), 24,33);

insert into billet
values(54, 150, TO_DATE('12-10-2022','dd-mm-yyyy'), 22, 34);

insert into billet
values(55, 65, TO_DATE('31-05-2022','dd-mm-yyyy'), 23, 32);

insert into billet
values(56, 50, TO_DATE('23-06-2022','dd-mm-yyyy'), 25, 34);

insert into billet
values(57, 100, TO_DATE('14-07-2022','dd-mm-yyyy'), 23, 31);

insert into billet
values(58, 55, TO_DATE('15-09-2022', 'dd-mm-yyyy'), 21, 35);

insert into billet
values(59, 150, TO_DATE('25-09-2022','dd-mm-yyyy'), 25, 33);

insert into billet
values(60, 50, TO_DATE('03-03-2022','dd-mm-yyyy'), 24, 31);

```

Script Output x

Query Result x

SQL | All Rows Fetched: 10 in 0.008 seconds

	ID_BILET	PRET	DATA_ACHIZITIONARE	ID_CLIENT	ID_TARG
1	51	50	31-JUL-22	22	31
2	52	100	01-OCT-22	21	35
3	53	50	23-JUN-22	24	33
4	54	150	12-OCT-22	22	34
5	55	65	31-MAY-22	23	32
6	56	50	23-JUN-22	25	34
7	57	100	14-JUL-22	23	31
8	58	55	15-SEP-22	21	35
9	59	150	25-SEP-22	25	33
10	60	50	03-MAR-22	24	31

6. Formulați în limbaj natural o problemă pe care să o rezolvați folosind un subprogram stocat independent care să utilizeze două tipuri diferite de colecții studiate. Apelați subprogramul.

-- Pentru fiecare tablou sa se afiseze data targului la care va fi expus si pretul mediu al tablourilor pt acel targ

CREATE OR REPLACE PROCEDURE proceduraex6 IS

-- vom crea un tip de date de inregistrare pt a pastra data targului si locatia

TYPE record1 IS RECORD

(data_ev TARG.data_eveniment%TYPE,

locatie TARG.locatie%TYPE,

pret_mediu TABLOU.pret%TYPE);

--toate datele despre targuri vor fi retinute intr-un tablou imbricat

TYPE tabel_imbricat IS TABLE OF record1;

--vom folosi un vector pentru a retine datele pentru fiecare targ

TYPE vector IS VARRAY(8) OF TARG.data_eveniment%TYPE;

--cu ajutorul unui cursor vom parcurge toate targurile

CURSOR c IS

SELECT id_targ, data_eveniment, locatie

FROM TARG;

t tabel_imbricat := tabel_imbricat();

r record1;

v vector := vector();

BEGIN

FOR i in c LOOP

--vom salva datele pentru pretul mediu al tablourilor pt targul respectiv

SELECT AVG(pret)

INTO r.pret_mediu

FROM TABLOU

WHERE id_targ = i.id_targ;

v.EXTEND;

SELECT data_eveniment

INTO v(v.LAST)

```

FROM TARG

WHERE id_targ=i.id_targ;

r.data_ev := i.data_eveniment;

r.locatie := i.locatie;

--adaugsm variabila in tabloul imbricat

t.EXTEND;

t(t.LAST) := r;

DBMS_OUTPUT.PUT_LINE('La targul din data ' || t(t.LAST).data_ev || ' din locatie ' || t(t.LAST).locatie || '
pretul mediu al tablourilor este ' || t(t.LAST).pret_mediu);

END LOOP;

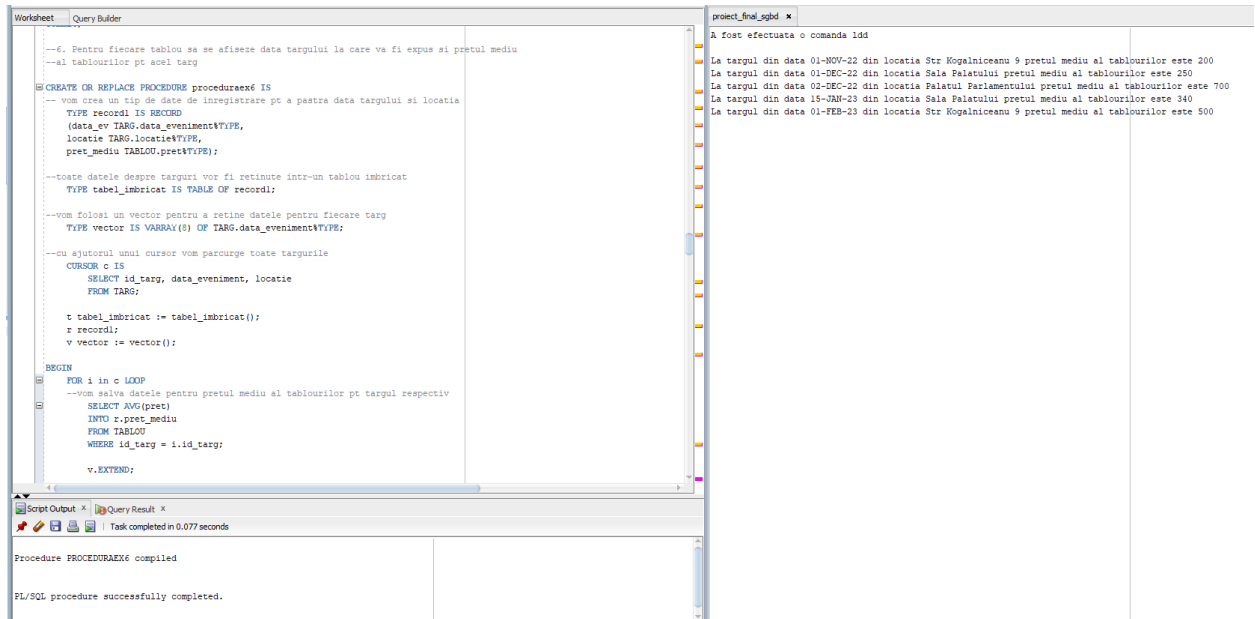
END proceduraex6;

/

EXECUTE proceduraex6;

/

```



7. Formulați în limbaj natural o problemă pe care să o rezolvați folosind un subprogram stocat independent care să utilizeze 2 tipuri diferite de cursoare studiate, unul dintre acestea fiind cursor parametrizat. Apelați subprogramul.

-- Sa se mareasca pretul tablourilor care au rating >=8 si artistii lor au varsta >=50.

```

CREATE OR REPLACE PROCEDURE proceduraex7 IS
    CURSOR tb IS
        SELECT id_tablou
        FROM TABLOU;
    CURSOR p (id_tablou TABLOU.id_tablou%TYPE) IS
        SELECT *
        FROM ARTIST a
        RIGHT JOIN TABLOU t ON t.id_artist= a.id_artist
        LEFT JOIN RATING ra ON t.id_tablou=ra.id_tablou
        WHERE t.id_tablou = 36 and ra.nota_acordata>=4 and a.varsta>=50
        FOR UPDATE OF t.pret NOWAIT;
BEGIN
    FOR r in tb LOOP
        FOR i in p(r.id_tablou) LOOP
            UPDATE TABLOU
            SET pret = pret + ((pret * 25)/100)
            WHERE CURRENT OF p;
        END LOOP;
    END LOOP;
END proceduraex7;
/
EXECUTE proceduraex7;
SELECT * FROM TABLOU;

```

```
--7. Sa se mareasca pretul tablourilor care au rating >=8 si artistii lor  
--au varsta >=50.
```

```
CREATE OR REPLACE PROCEDURE proceduraex7 IS  
  
    CURSOR tb IS  
        SELECT id_tablou  
        FROM TABLOU;  
  
    CURSOR p (id_tablou TABLOU.id_tablou%TYPE) IS  
        SELECT *  
        FROM ARTIST a  
        RIGHT JOIN TABLOU t ON t.id_artist= a.id_artist  
        LEFT JOIN RATING ra ON t.id_tablou=ra.id_tablou  
        WHERE t.id_tablou = 36 and ra.nota_acordata>=4 and a.varsta>=50  
        FOR UPDATE OF t.pret NOWAIT;  
  
BEGIN  
  
    FOR r in tb LOOP  
        FOR i in p(r.id_tablou) LOOP  
            UPDATE TABLOU  
            SET pret = pret + ((pret * 25)/100)  
            WHERE CURRENT OF p;  
        END LOOP;  
    END LOOP;  
  
END proceduraex7;  
/  
EXECUTE proceduraex7;  
SELECT * FROM TABLOU;
```

Script Output x Query Result x
Task completed in 0.069 seconds

Procedure PROCEDURAEX7 compiled

PL/SQL procedure successfully completed.

The screenshot shows a SQL query execution window. The query is: `SELECT * FROM TABLOU;`. The results are displayed in a table with 9 columns: ID_TABLOU, DENUMIRE, AN_CREARE, TIP_TABLOU, TIP_MATERIALE, PRET, ID_ADMIN, ID_ARTIST, and ID_TARG. There are 5 rows of data.

ID_TABLOU	DENUMIRE	AN_CREARE	TIP_TABLOU	TIP_MATERIALE	PRET	ID_ADMIN	ID_ARTIST	ID_TARG
1	36 Iarna	2021	pictura	acrilice	29530	1	8	34
2	37 Iarna la schi	2022	desen	creion	200	4	9	31
3	38 Flori de camp	2020	pictura	ulei	500	2	6	35
4	39 Nuante de gri	2021	desen	creion	250	3	7	32
5	40 Foc de tabara	2020	pictura	acrilice	700	5	10	33

8. Formulați în limbaj natural o problemă pe care să o rezolvați folosind un subprogram stocat independent de tip funcție care să utilizeze într-o singură comandă SQL3 dintre tabelele definite. Definiți minim 2 excepții. Apelați subprogramul astfel încât să evidențiați toate cazurile tratate.

-- Se citește numele unui client de la tastatură. Să se returneze pretul total al comenzii făcute de acesta (pretul comenzii + pretul tabloului comandat). În cazul în care un client
 -- are mai multe comenzi, se va returna suma tuturor prețurilor totale. Se vor arunca erori
 -- în cazul în care: există mai mulți clienți cu numele dat, nu există un client cu numele dat -- sau nu există comenzi făcute de acest client.

-- ne vom folosi de tabelele comanda, tablou și client în această funcție

```
CREATE OR REPLACE FUNCTION functieex8(v_nume CLIENT.nume%TYPE)
```

```
RETURN COMANDA.pret%TYPE IS
```

```
    pret_total COMANDA.pret%TYPE;
```

```
    contor NUMBER;
```

```
BEGIN
```

```
    -- tratăm întâi excepțiile
```

```
    -- numărăm câți clienți există cu numele dat în input
```

```
    SELECT COUNT (*) INTO contor
```

```
    FROM CLIENT WHERE INITCAP(v_nume) = INITCAP(nume);
```

```
    -- tratăm cazul în care nu există niciun client și cazul în care există mai mulți clienți cu același nume
```

```
    IF contor = 0
```

```
    THEN RAISE_APPLICATION_ERROR(-20000, 'Nu există un client cu acest nume');
```

```
    ELSIF contor > 1
```

```
    THEN RAISE_APPLICATION_ERROR(-20001, 'Există mai mulți clienți cu acest nume');
```

```
    END IF;
```

--folosim aceeași variabilă pentru a număra câte comenzi există pentru clientul dat, iar --dacă nu există niciuna, vom arunca o eroare

```
SELECT COUNT(*) INTO contor
FROM CLIENT RIGHT OUTER JOIN COMANDA USING (id_client)
WHERE INITCAP(v_nume) = INITCAP(ume);
IF contor = 0
THEN RAISE_APPLICATION_ERROR(-20002, 'Nu există comenzi pe acest nume');
END IF;
```

--funcția de SUM va trata atât cazul în care există o singură comandă, cât și cazul în care există mai multe

```
SELECT SUM(c.pret+ t.pret)
INTO pret_total
FROM COMANDA c JOIN TABLOU t USING(id_tablou) JOIN CLIENT USING (id_client)
WHERE INITCAP(v_nume) = INITCAP(CLIENT.ume);
RETURN pret_total;
END functieex8;
```

/

```
select* from client;
SELECT functieex8('Dinculescu Cosmina') FROM DUAL;
SELECT functieex8('Popescu Carmen') FROM DUAL;
```

```

THEN RAISE_APPLICATION_ERROR(-20001, 'Exista mai multi clienti cu acest nume');
END IF;

--folosim aceeaasi variabila pentru a numara cate comenzi exista pentru clientul dat, iar daca nu
--exista niciuna, vom arunca o eroare
SELECT COUNT(*) INTO contor
FROM CLIENT RIGHT OUTER JOIN COMANDA USING (id_client)
WHERE INITCAP(v_nume) = INITCAP(numa);

IF contor = 0
THEN RAISE_APPLICATION_ERROR(-20002, 'Nu exista comenzi pe acest nume');
END IF;

--functia de SUM va trata atat cazul in care exista o singura comanda, cat si cazul in care exista
--mai multe
SELECT SUM(c.pret+ t.pret)
INTO pret_total
FROM COMANDA c JOIN TABLOU t USING(id_tablou) JOIN CLIENT USING (id_client)
WHERE INITCAP(v_nume) = INITCAP(CLIENT.numa);
RETURN pret_total;

END functieex8;
/

```

Script Output x Query Result x

Task completed in 0.077 seconds

'L/SQL procedure successfully completed.

unction FUNCTIEEX8 compiled

```
SELECT functieex8('Dinculescu Cosmina') FROM DUAL;
```

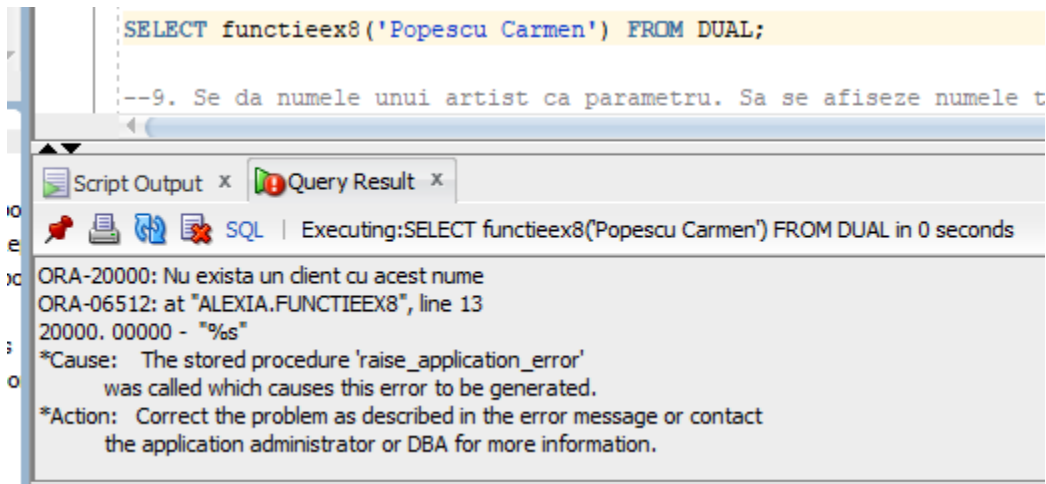
```
SELECT functieex8('Popescu Carmen') FROM DUAL;
```

--9. Se da numele unui artist ca parametru. Sa se afiseze nu

Script Output x Query Result x

SQL | All Rows Fetched: 1 in 0.041 seconds

	FUNCTIEEX8('DINCULESCUCOSMINA')
1	600



9. Formulați în limbaj natural o problemă pe care să o rezolvați folosind un subprogram stocat independent de tip procedură care să utilizeze într-o singură comandă SQL 5 dintre tabelele definite. Tratați toate excepțiile care pot apărea, incluzând excepțiile NO_DATA_FOUND și TOO_MANY_ROWS. Apelați subprogramul astfel încât să evidențiați toate cazurile tratate.

-- Se da numele unui artist ca parametru. Sa se afiseze numele tabloului oferit spre cumparare, ratingul acestuia, idul contractului si numele site-ului unde isi vinde tabloul.

CREATE OR REPLACE PROCEDURE proceduraex9 (nume_artist ARTIST.nume%TYPE) IS

--vom declara variabilele in care vom salva datele care trebuiesc selectate si afisate

nume_tablou TABLOU.denumire%TYPE;

rating_tablou RATING.nota_acordata%TYPE;

id_contr CONTRACT.id_contract%TYPE;

nume_site ADMIN.denumire_site%TYPE;

BEGIN

--vom folosi join pentru a accesa si selecta ce ne intereseaza

SELECT t.denumire, r.nota_acordata, c.id_contract, a.denumire_site

INTO nume_tablou, rating_tablou, id_contr, nume_site

FROM ARTIST ar RIGHT JOIN TABLOU t ON (ar.id_artist = t.id_artist)

JOIN RATING r ON (t.id_tablou = r.id_tablou) JOIN CONTRACT c ON (ar.id_artist = c.id_artist)

JOIN ADMIN a ON (c.id_admin= a.id_admin)

WHERE INITCAP(nume_artist) = INITCAP(ar.nume)

GROUP BY t.denumire, r.nota_acordata, c.id_contract, a.denumire_site;

--afisam in consola datele selectate mai sus

DBMS_OUTPUT.PUT_LINE('Numele artistului este: ' || INITCAP(nume_artist));

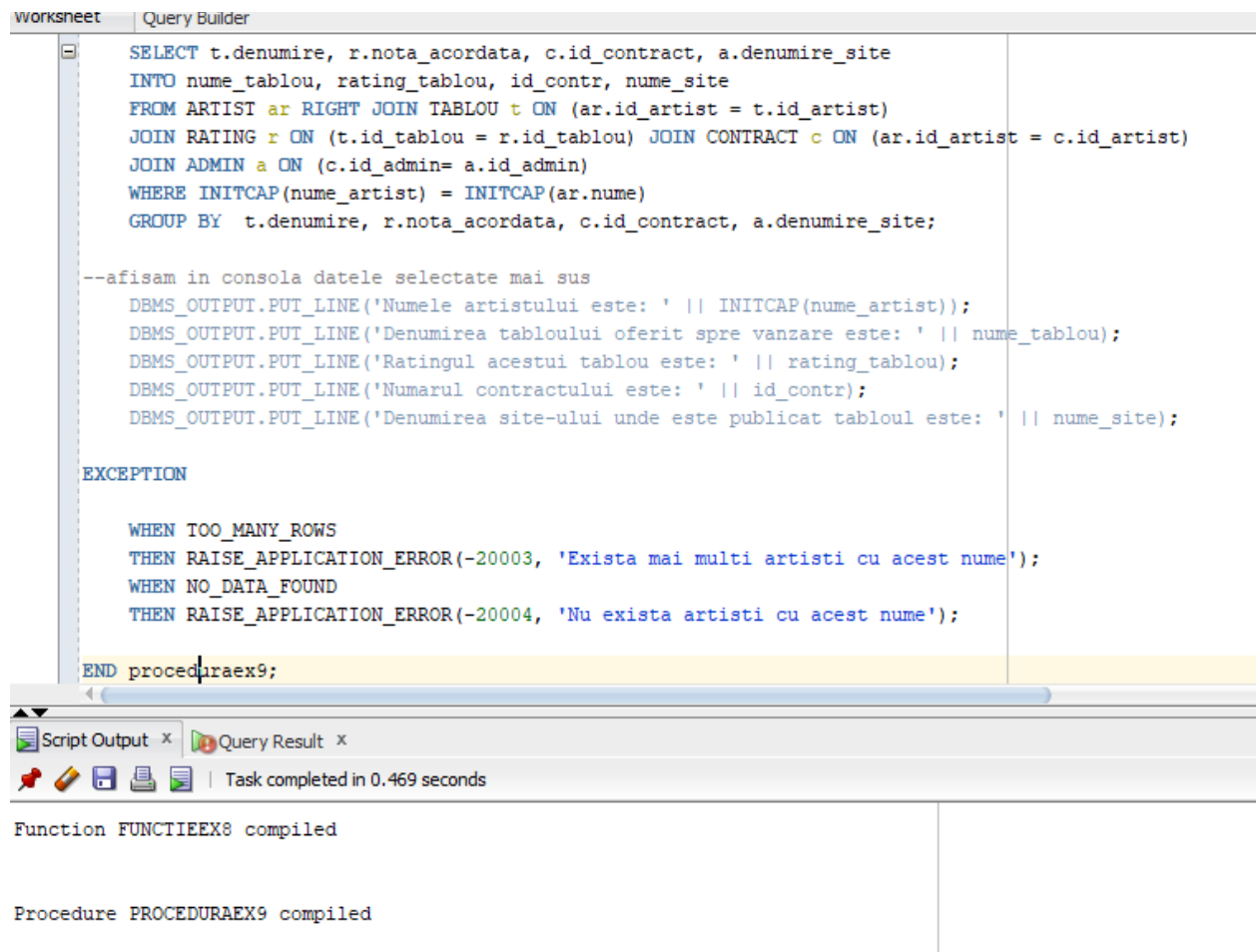
DBMS_OUTPUT.PUT_LINE('Denumirea tabloului oferit spre vanzare este: ' || nume_tablou);

```

DBMS_OUTPUT.PUT_LINE('Ratingul acestui tablou este: ' || rating_tablou);
DBMS_OUTPUT.PUT_LINE('Numarul contractului este: ' || id_contr);
DBMS_OUTPUT.PUT_LINE('Denumirea site-ului unde este publicat tabloul este: ' || nume_site);
EXCEPTION
    WHEN TOO_MANY_ROWS
    THEN RAISE_APPLICATION_ERROR(-20003, 'Exista mai multi artisti cu acest nume');
    WHEN NO_DATA_FOUND
    THEN RAISE_APPLICATION_ERROR(-20004, 'Nu exista artisti cu acest nume');
END proceduraex9;
/

select* from artist;
EXECUTE proceduraex9('Popescu Valentin');
EXECUTE proceduraex9('Aldea Alexia');

```



The screenshot shows a SQL IDE interface with a 'worksneet' tab and a 'Query Builder' tab. The main editor displays a SQL script for a procedure named 'proceduraex9'. The script includes a SELECT statement with joins on ARTIST, TABLOU, RATING, CONTRACT, and ADMIN tables, followed by an EXCEPTION block and a call to the procedure with two artist names: 'Popescu Valentin' and 'Aldea Alexia'.

Below the editor, the 'Script Output' and 'Query Result' tabs are visible. The 'Script Output' tab shows the execution results, indicating that the function 'FUNCTIEEX8' was compiled and the procedure 'PROCEDURAEX9' was compiled. The 'Query Result' tab shows the results of the SELECT statement, displaying columns: nume_tablou, rating_tablou, id_contr, nume_site. The results are grouped by the artist's name, showing data for 'Popescu Valentin' and 'Aldea Alexia'.

```

SELECT t.denumire, r.nota_acordata, c.id_contract, a.denumire_site
INTO nume_tablou, rating_tablou, id_contr, nume_site
FROM ARTIST ar RIGHT JOIN TABLOU t ON (ar.id_artist = t.id_artist)
JOIN RATING r ON (t.id_tablou = r.id_tablou) JOIN CONTRACT c ON (ar.id_artist = c.id_artist)
JOIN ADMIN a ON (c.id_admin= a.id_admin)
WHERE INITCAP(nume_artist) = INITCAP(ar.nume)
GROUP BY t.denumire, r.nota_acordata, c.id_contract, a.denumire_site;

--afisam in consola datele selectate mai sus
DBMS_OUTPUT.PUT_LINE('Numele artistului este: ' || INITCAP(nume_artist));
DBMS_OUTPUT.PUT_LINE('Denumirea tabloului oferit spre vanzare este: ' || nume_tablou);
DBMS_OUTPUT.PUT_LINE('Ratingul acestui tablou este: ' || rating_tablou);
DBMS_OUTPUT.PUT_LINE('Numarul contractului este: ' || id_contr);
DBMS_OUTPUT.PUT_LINE('Denumirea site-ului unde este publicat tabloul este: ' || nume_site);

EXCEPTION

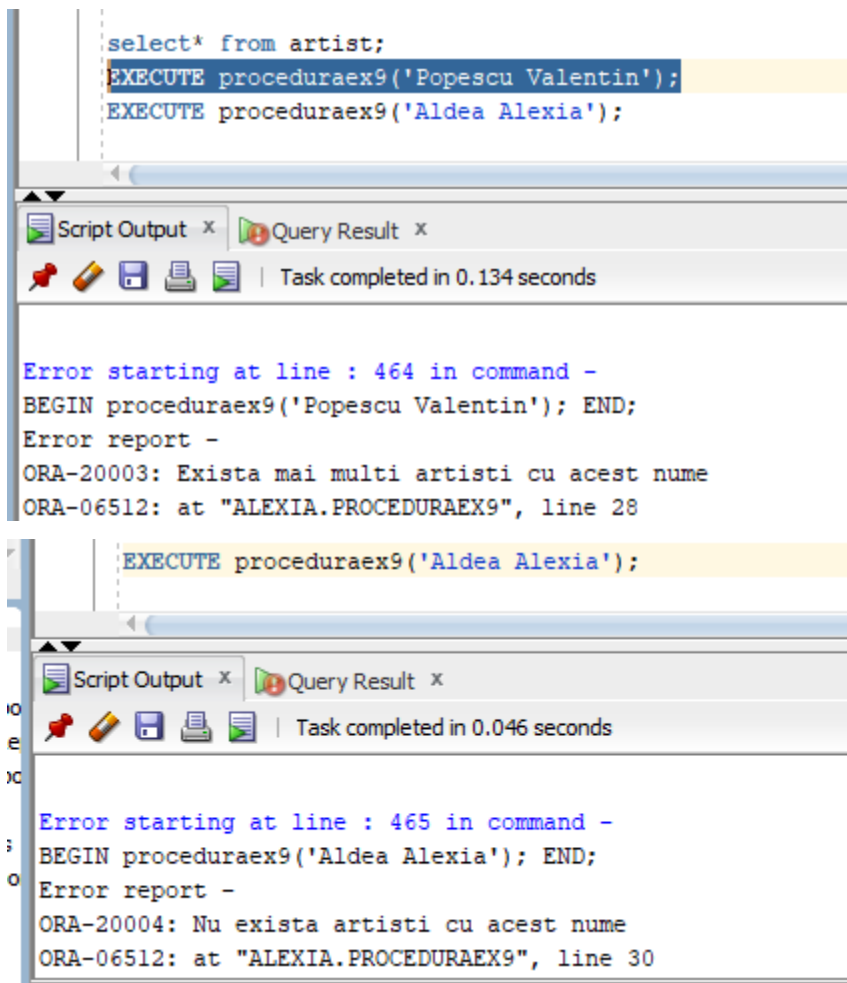
    WHEN TOO_MANY_ROWS
    THEN RAISE_APPLICATION_ERROR(-20003, 'Exista mai multi artisti cu acest nume');
    WHEN NO_DATA_FOUND
    THEN RAISE_APPLICATION_ERROR(-20004, 'Nu exista artisti cu acest nume');

END proceduraex9;

```

Function FUNCTIEEX8 compiled

Procedure PROCEDURAEX9 compiled



10. Definiți un trigger de tip LMD la nivel de comandă. Declanșați trigger-ul.

-- Trigger care interzice stergerea datelor din tabelul targ

/

CREATE OR REPLACE TRIGGER triggerex10

BEFORE DELETE ON TARG

BEGIN

RAISE_APPLICATION_ERROR(-20005, 'Nu se pot sterge date din tabelul TARG');

END;

/

DELETE FROM TARG

WHERE id_targ= 31;

```
--10. Trigger care interzice stergerea datelor din tabelul targ
/
CREATE OR REPLACE TRIGGER triggerex10
  BEFORE DELETE ON TARG
BEGIN
  RAISE_APPLICATION_ERROR(-20005, 'Nu se pot sterge date din tabelul TARG');
END;
/
```

Script Output x Query Result x

Task completed in 0.083 seconds

ORA-06512: at "ALEXIA.TRIGGEREX10", line 30

ORA-06512: at line 1

Trigger TRIGGEREX10 compiled

```
--10. Trigger care interzice stergerea datelor din tabelul targ
/
CREATE OR REPLACE TRIGGER triggerex10
  BEFORE DELETE ON TARG
BEGIN
  RAISE_APPLICATION_ERROR(-20005, 'Nu se pot sterge date din tabelul TARG');
END;
/

DELETE FROM TARG
WHERE id_targ= 31;

--11. Trigger care interzice cresterea pretului unui bilet cu mai mult de 100
/
CREATE OR REPLACE TRIGGER triggerex11
```

Script Output x Query Result x

Task completed in 0.068 seconds

Error report -

ORA-20005: Nu se pot sterge date din tabelul TARG

ORA-06512: at "ALEXIA.TRIGGEREX10", line 2

ORA-04088: error during execution of trigger 'ALEXIA.TRIGGEREX10'

11. Definiți un trigger de tip LMD la nivel de linie. Declanșați trigger-ul.

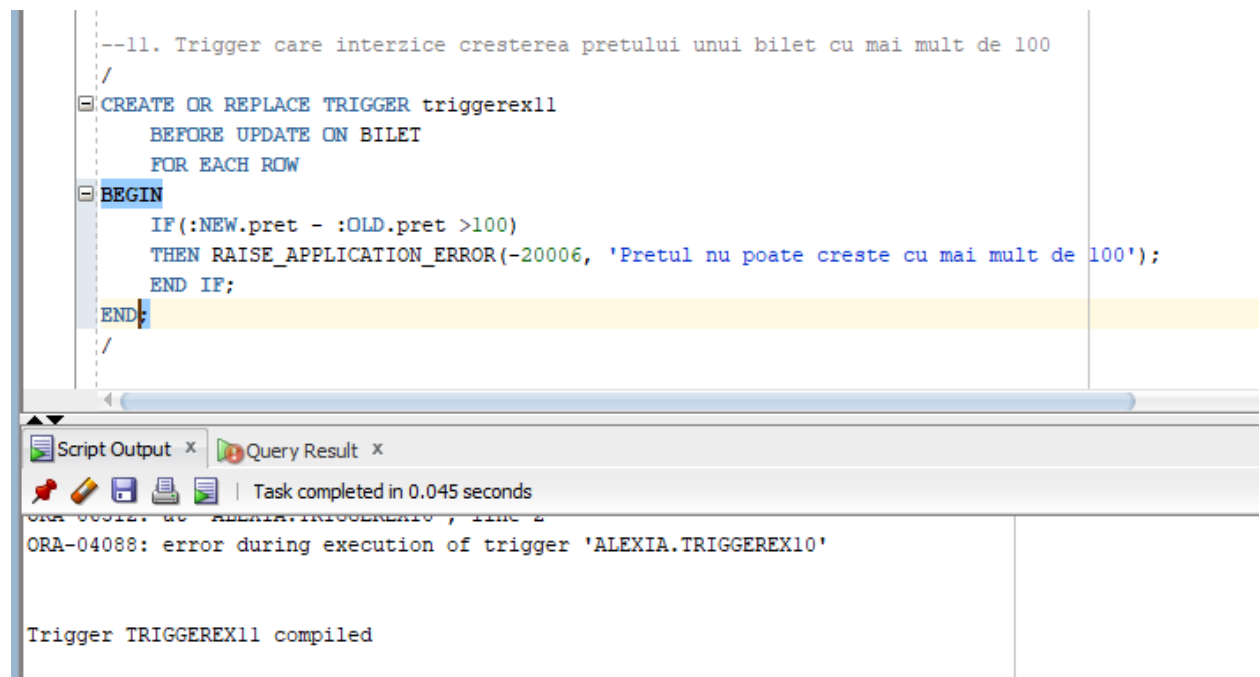
```
-- Trigger care interzice cresterea pretului unui bilet cu mai mult de 100
/
```

```

CREATE OR REPLACE TRIGGER triggerex11
  BEFORE UPDATE ON BILET
  FOR EACH ROW
BEGIN
  IF(:NEW.pret - :OLD.pret >100)
  THEN RAISE_APPLICATION_ERROR(-20006, 'Pretul nu poate creste cu mai mult de 100');
  END IF;
END;
/

UPDATE BILET
SET pret = pret+ 101
WHERE id_bilet = 57;

```



```

--11. Trigger care interzice cresterea pretului unui bilet cu mai mult de 100
/
CREATE OR REPLACE TRIGGER triggerex11
  BEFORE UPDATE ON BILET
  FOR EACH ROW
BEGIN
  IF(:NEW.pret - :OLD.pret >100)
  THEN RAISE_APPLICATION_ERROR(-20006, 'Pretul nu poate creste cu mai mult de 100');
  END IF;
END;
/

```

Script Output x Query Result x

Task completed in 0.045 seconds

ORA-04088: error during execution of trigger 'ALEXIA.TRIGGEREX10'

Trigger TRIGGEREX11 compiled


```
--11. Trigger care interzice cresterea pretului unui bilet cu mai mult de 100
/
CREATE OR REPLACE TRIGGER triggerex11
BEFORE UPDATE ON BILET
FOR EACH ROW
BEGIN
    IF (:NEW.pret - :OLD.pret > 100)
    THEN RAISE_APPLICATION_ERROR(-20006, 'Pretul nu poate creste cu mai mult de 100');
    END IF;
END;
/

UPDATE BILET
SET pret = pret+ 101
WHERE id_bilet = 57;

--12. Triger care afiseaza un mesaj de fiecare data cand este rulata o comanda ldd
```

Script Output x Query Result x

Task completed in 0.042 seconds

Error starting at line : 491 in command -

```
UPDATE BILET
SET pret = pret+ 101
WHERE id_bilet = 57
Error report -
ORA-20006: Pretul nu poate creste cu mai mult de 100
ORA-06512: at "ALEXIA.TRIGGEREX11", line 3
ORA-04088: error during execution of trigger 'ALEXIA.TRIGGEREX11'
```

12. Definiți un trigger de tip LDD. Declanșați trigger-ul.

--Triger care afiseaza un mesaj de fiecare data cand este rulata o comanda ldd

/

CREATE OR REPLACE TRIGGER triggerex12

AFTER CREATE OR ALTER OR DROP ON SCHEMA

BEGIN

DBMS_OUTPUT.PUT_LINE('A fost efectuata o comanda ldd');

END;

/

ALTER TABLE CLIENT

ADD varsta NUMBER(5);

ALTER TABLE CLIENT

DROP COLUMN varsta;

```
--12. Triger care afiseaza un mesaj de fiecare data cand este rulata o comanda ldd
/
CREATE OR REPLACE TRIGGER triggerex12
  AFTER CREATE OR ALTER OR DROP ON SCHEMA
BEGIN
  DBMS_OUTPUT.PUT_LINE('A fost efectuata o comanda ldd');
END;
```

Script Output x Query Result x

Task completed in 0.047 seconds

ORA-04088: error during execution of trigger 'ALEXIA.TRIGGEREX11'

Trigger TRIGGEREX12 compiled

```
--12. Triger care afiseaza un mesaj de fiecare data cand este rulata o comanda ldd
/
CREATE OR REPLACE TRIGGER triggerex12
  AFTER CREATE OR ALTER OR DROP ON SCHEMA
BEGIN
  DBMS_OUTPUT.PUT_LINE('A fost efectuata o comanda ldd');
END;
/

ALTER TABLE CLIENT
ADD varsta NUMBER(5);

ALTER TABLE CLIENT
DROP COLUMN varsta;
```

Script Output x Query Result x

Task completed in 0.426 seconds

Trigger TRIGGEREX12 compiled

Table CLIENT altered.

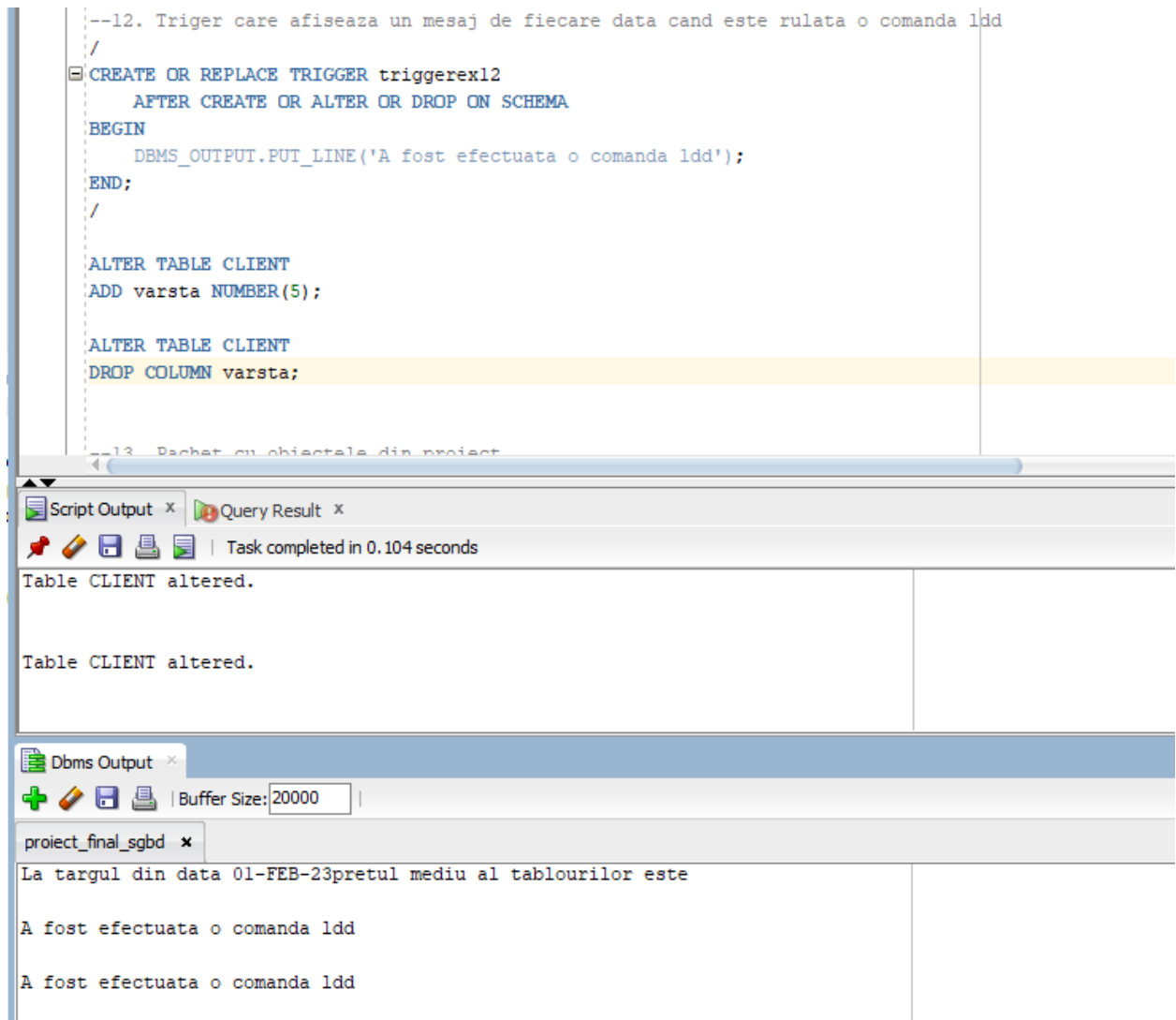
Dbms Output x

Buffer Size: 20000

proiect_final_sgbd x

La targul din data 02-DEC-22pretul mediu al tablourilor este
La targul din data 15-JAN-23pretul mediu al tablourilor este
La targul din data 01-FEB-23pretul mediu al tablourilor este

A fost efectuata o comanda ldd



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

```
/
CREATE OR REPLACE PACKAGE pachetex13 AS
  FUNCTION functieex8 (v_nume CLIENT.nume%TYPE) RETURN COMANDA.pret%TYPE;
  PROCEDURE proceduraex6;
  PROCEDURE proceduraex7;
  PROCEDURE proceduraex9(nume_artist ARTIST.nume%TYPE);
END pachetex13;
/

CREATE OR REPLACE PACKAGE BODY pachetex13 AS
--pretul total pentru comenzile facute de un client
  FUNCTION functieex8(v_nume CLIENT.nume%TYPE)
  RETURN COMANDA.pret%TYPE IS
    pret_total COMANDA.pret%TYPE;
    contor NUMBER;
  BEGIN
    SELECT COUNT(*) INTO contor
```

```

FROM CLIENT WHERE INITCAP(v_num) = INITCAP(num);
IF contor = 0
THEN RAISE_APPLICATION_ERROR(-20000, 'Nu exista un client cu acest nume');
ELSIF contor > 1
THEN RAISE_APPLICATION_ERROR(-20001, 'Exista mai multi clienti cu acest nume');
END IF;
SELECT COUNT(*) INTO contor
FROM CLIENT RIGHT OUTER JOIN COMANDA USING (id_client)
WHERE INITCAP(v_num) = INITCAP(num);
IF contor = 0
THEN RAISE_APPLICATION_ERROR(-20002, 'Nu exista comenzi pe acest nume');
END IF;
SELECT SUM(c.pret+ t.pret)
INTO pret_total
FROM COMANDA c JOIN TABLOU t USING(id_tablou) JOIN CLIENT USING (id_client)
WHERE INITCAP(v_num) = INITCAP(CLIENT.num);
RETURN pret_total;
END functieex8;
--pretul mediu al unui tablou
PROCEDURE proceduraex6 IS
TYPE record1 IS RECORD
(data_ev TARG.data_eveniment%TYPE,
locatie TARG.locatie%TYPE,
pret_mediu TABLOU.pret%TYPE);
TYPE tabel_imbricat IS TABLE OF record1;
TYPE vector IS VARRAY(8) OF TARG.data_eveniment%TYPE;
CURSOR c IS
SELECT id_targ, data_eveniment, locatie
FROM TARG;
t tabel_imbricat := tabel_imbricat();
r record1;
v vector := vector();
BEGIN
FOR i in c LOOP
SELECT AVG(pret)
INTO r.pret_mediu
FROM TABLOU
WHERE id_targ = i.id_targ;
v.EXTEND;
SELECT data_eveniment
INTO v(v.LAST)
FROM TARG
WHERE id_targ=i.id_targ;
r.data_ev := i.data_eveniment;
r.locatie := i.locatie;
t.EXTEND;
t(t.LAST) := r;

```

```

        DBMS_OUTPUT.PUT_LINE('La targul din data ' || t(t.LAST).data_ev || ' din locatia ' ||
t(t.LAST).locatie || ' pretul mediu al tablourilor este ' || t(t.LAST).pret_mediu);
    END LOOP;
END proceduraex6;
--proceduraex7
PROCEDURE proceduraex7 IS
    CURSOR tb IS
        SELECT id_tablou
        FROM TABLOU;
    CURSOR p (id_tablou TABLOU.id_tablou%TYPE) IS
        SELECT *
        FROM ARTIST a
        RIGHT JOIN TABLOU t ON t.id_artist= a.id_artist
        LEFT JOIN RATING ra ON t.id_tablou=ra.id_tablou
        WHERE t.id_tablou = 36 and ra.nota_acordata>=4 and a.varsta>=50
        FOR UPDATE OF t.pret NOWAIT;
BEGIN
    FOR r in tb LOOP
        FOR i in p(r.id_tablou) LOOP
            UPDATE TABLOU
            SET pret = pret + ((pret * 25)/100)
            WHERE CURRENT OF p;
        END LOOP;
    END LOOP;
END proceduraex7;
--date despre un artist dat ca parametru
PROCEDURE proceduraex9(ume_artist ARTIST.ume%TYPE) IS
    ume_tablou TABLOU.denumire%TYPE;
    rating_tablou RATING.nota_acordata%TYPE;
    id_contr CONTRACT.id_contract%TYPE;
    ume_site ADMIN.denumire_site%TYPE;
BEGIN
    SELECT t.denumire, r.nota_acordata, c.id_contract, a.denumire_site
    INTO ume_tablou, rating_tablou, id_contr, ume_site
    FROM ARTIST ar RIGHT JOIN TABLOU t ON (ar.id_artist = t.id_artist)
    JOIN RATING r ON (t.id_tablou = r.id_tablou) JOIN CONTRACT c ON (ar.id_artist = c.id_artist)
    JOIN ADMIN a ON (c.id_admin= a.id_admin)
    WHERE INITCAP(ume_artist) = INITCAP(ar.ume)
    GROUP BY t.denumire, r.nota_acordata, c.id_contract, a.denumire_site;
    DBMS_OUTPUT.PUT_LINE('Numele artistului este: ' || INITCAP(ume_artist));
    DBMS_OUTPUT.PUT_LINE('Denumirea tabloului oferit spre vanzare este: ' || ume_tablou);
    DBMS_OUTPUT.PUT_LINE('Ratingul acestui tablou este: ' || rating_tablou);
    DBMS_OUTPUT.PUT_LINE('Numarul contractului este: ' || id_contr);
    DBMS_OUTPUT.PUT_LINE('Denumirea site-ului unde este publicat tabloul este: ' || ume_site);
EXCEPTION
    WHEN TOO_MANY_ROWS
    THEN RAISE_APPLICATION_ERROR(-20003, 'Exista mai multi artisti cu acest nume');
    WHEN NO_DATA_FOUND
    THEN RAISE_APPLICATION_ERROR(-20004, 'Nu exista artisti cu acest nume');

```

```

END proceduraex9;
END pachetex13;
/
SELECT pachetex13.functieex8('Dinculescu Cosmina') FROM DUAL;
EXECUTE pachetex13.proceduraex6;
EXECUTE pachetex13.proceduraex7;
EXECUTE pachetex13.proceduraex9('Aldea Alexia');

```

Worksheet | Query Builder

```

540
541 --13. Pachet cu obiectele din proiect
542 /
543 CREATE OR REPLACE PACKAGE pachetex13 AS
544     FUNCTION functieex8 (v_nume CLIENT.nume%TYPE) RETURN COMANDA.pret%TYPE;
545     PROCEDURE proceduraex6;
546     PROCEDURE proceduraex7;
547     PROCEDURE proceduraex9(nume_artist ARTIST.nume%TYPE);
548 END pachetex13;
549 /
550 CREATE OR REPLACE PACKAGE BODY pachetex13 AS
551 --pretul total pentru comenzile facute de un client
552 FUNCTION functieex8(v_nume CLIENT.nume%TYPE)
553 RETURN COMANDA.pret%TYPE IS
554     pret_total COMANDA.pret%TYPE;
555     contor NUMBER;
556 BEGIN
557     SELECT COUNT(*) INTO contor
558     FROM CLIENT WHERE INITCAP(v_nume) = INITCAP(nume);
559
560     IF contor = 0
561     THEN RAISE_APPLICATION_ERROR(-20000, 'Nu exista un client cu acest nume');
562     ELSIF contor > 1
563     THEN RAISE_APPLICATION_ERROR(-20001, 'Exista mai multi clienti cu acest nume');
564     END IF;
565
566     SELECT COUNT(*) INTO contor
567     FROM CLIENT RIGHT OUTER JOIN COMANDA USING (id_client)
568     WHERE INITCAP(v_nume) = INITCAP(nume);
569
570     IF contor = 0

```

Script Output x Query Result x

Task completed in 0.146 seconds

Package PACHETEX13 compiled

Package Body PACHETEX13 compiled

```
687
688 SELECT pachetex13.functieex8('Dinculescu Cosmina') FROM DUAL;
689 EXECUTE pachetex13.proceduraex6;
690 EXECUTE pachetex13.proceduraex7;
691 EXECUTE pachetex13.proceduraex9('Aldea Alexia');
692
693
```

Script Output x Query Result x

SQL | All Rows Fetched: 1 in 0.018 seconds

PACHETEX13.FUNCTIEEX8('DINCULESCUCOSMINA')
1 600

Worksheet Query Builder

```
676 EXCEPTION
677
678 WHEN TOO_MANY_ROWS
679 THEN RAISE_APPLICATION_ERROR(-20003, 'Exista mai multi artisti cu acest nume');
680
681 WHEN NO_DATA_FOUND
682 THEN RAISE_APPLICATION_ERROR(-20004, 'Nu exista artisti cu acest nume');
683
684 END proceduraex9;
685
686 END pachetex13;
687
688 /
689
690 SELECT pachetex13.functieex8('Dinculescu Cosmina') FROM DUAL;
691 EXECUTE pachetex13.proceduraex6;
692 EXECUTE pachetex13.proceduraex7;
693 EXECUTE pachetex13.proceduraex9('Aldea Alexia');
694
```

Script Output x Query Result x

Task completed in 0.127 seconds

Package PACHETEX13 compiled

Package Body PACHETEX13 compiled

PL/SQL procedure successfully completed.

proiect_final_sghd x

La targul din data 01-NOV-22 din locatia Str Kogalniceanu 9 pretul mediu al tablourilor este 200
La targul din data 01-DEC-22 din locatia Sala Palatului pretul mediu al tablourilor este 250
La targul din data 02-DEC-22 din locatia Palatul Parlamentului pretul mediu al tablourilor este 700
La targul din data 15-JAN-23 din locatia Sala Palatului pretul mediu al tablourilor este 3170
La targul din data 01-FEB-23 din locatia Str Kogalniceanu 9 pretul mediu al tablourilor este 500

```
687
688 SELECT pachetex13.functieex8('Dinculescu Cosmina') FROM DUAL;
689 EXECUTE pachetex13.proceduraex6;
690 EXECUTE pachetex13.proceduraex7;
691 EXECUTE pachetex13.proceduraex9('Aldea Alexia');
692
693
694
```

Script Output x Query Result x

Task completed in 0.147 seconds

Package Body PACHETEX13 compiled

PL/SQL procedure successfully completed.

PL/SQL procedure successfully completed.

```
687  
688 SELECT pachetex13.functieex8('Dinculescu Cosmina') FROM DUAL;  
689 EXECUTE pachetex13.proceduraex6;  
690 EXECUTE pachetex13.proceduraex7;  
691 EXECUTE pachetex13.proceduraex9('Aldea Alexia');  
692  
693  
694
```

Script Output x

Query Result x

Task completed in 0.147 seconds

Error starting at line : 691 in command -
BEGIN pachetex13.proceduraex9('Aldea Alexia'); END;
Error report -
ORA-20004: Nu exista artisti cu acest nume
ORA-06512: at "ALEXIA.PACHETEX13", line 132
ORA-06512: at line 1