

GESTIUNEA SALOANELOR COSMETICE DIN
ROMÂNIA

Voinea Stefania Alexandra

Grupa 242

Utilitatea bazei de date

Exemplele din acest capitol se referă la proiectarea unui model de date ce furnizează informații despre saloanele cosmetice din România, un loc frecventat mai mult sau mai puțin de orice cetățean.

Pentru acest model voi construi diagrama ERD corespunzătoare, dar și diagrama conceptuală.

Modelul de date va gestiona informații legate de organizarea și funcționarea saloanelor cosmetice din România. Există furnizori pentru aceste saloane – ei se află în colaborare cu saloanele pentru a le aproviziona cu produse cosmetice (vopsea, foarfece, ceară, etc.).

Fiecare salon se află într-o anumită locație, este curățat de îngrijitori și are mai mulți angajați. Îngrijitorii vin la finalul zilei pentru a pregăti salonul de o nouă zi, ceea ce le permite să fie arondați la mai multe saloane.

Angajații au fiecare câte un job (manichiurist, frizer, etc.) și trebuie să își satisfacă toți clienții. Clienții își fac o programare la un anumit serviciu. Având în vedere că fiecare angajat are specializarea sa, clienții trebuie să-și facă câte o programare pentru fiecare serviciu dorit (de exemplu, dacă un client dorește să se tundă și să-și facă manichiura, el trebuie să-și facă două programări separate).

Modelul de date respectă anumite restricții de funcționare:

- Angajații se ocupă de curățenia din timpul programului (nu este luat în considerare acest aspect), iar îngrijitorii se ocupă de curățenia de la final de zi, astfel încât angajații nu stau peste program.
- Furnizorii aprovizionează salonul cu produse cosmetice și diferite materiale, dar în baza de date apare doar suma totală de bani pe care o valorează aceste produse.
- Orice salon are o locație unică.

Diagrama entitate-relație

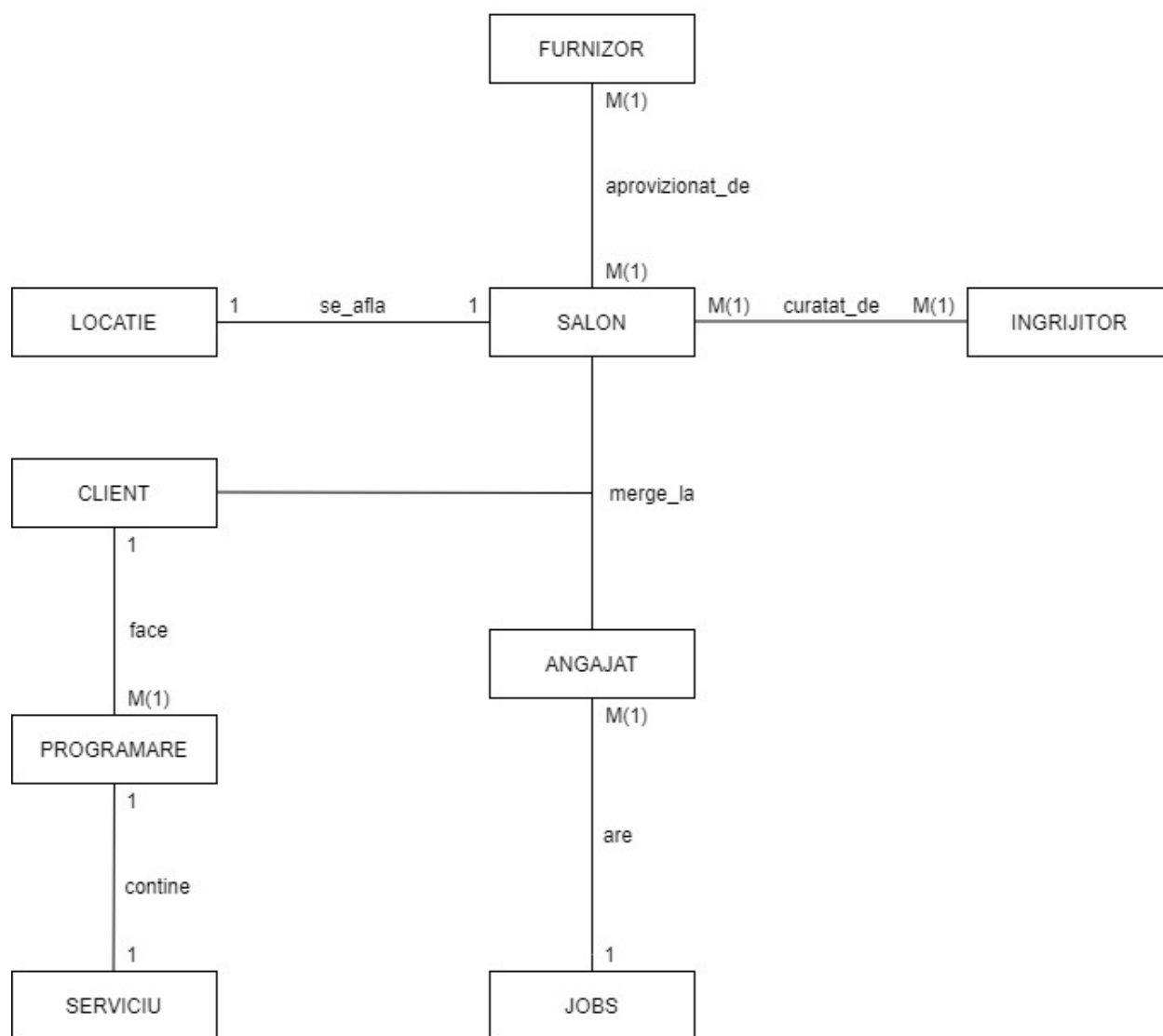
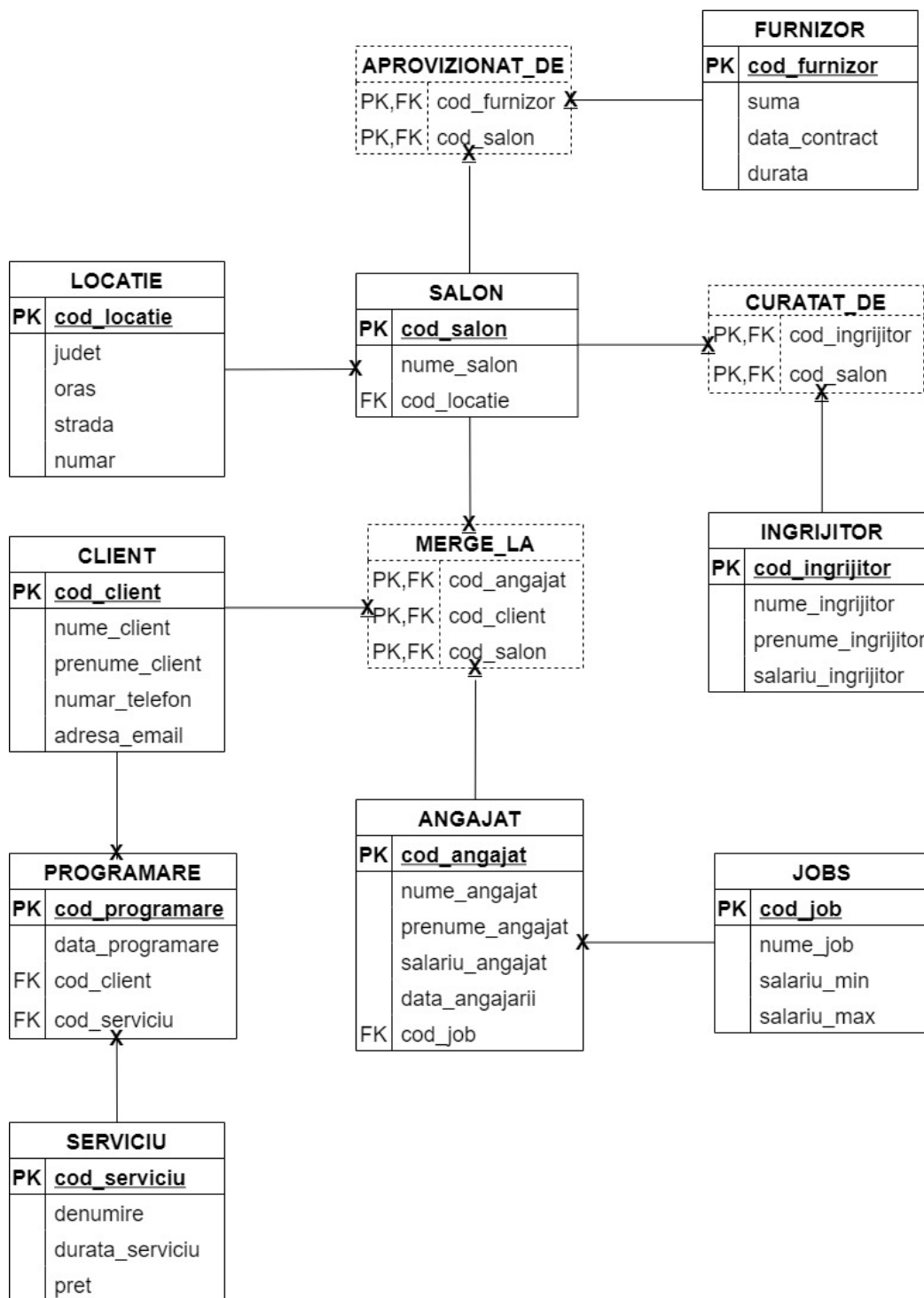


Diagrama conceptuala

Crearea - inserarea în Oracle

1. Tabela independentă LOCATIE

```
CREATE TABLE LOCATIE
```

```
(cod_locatie number(6),  
judet varchar2(20) default 'Bucuresti',  
oras varchar2(20) default 'Bucuresti',  
strada varchar2(30) constraint strada_loc not null,  
numar number(3) constraint numar_loc not null,  
constraint pk_locatie primary key(cod_locatie),  
check(numar > 0)  
);
```

```
CREATE SEQUENCE SEQ_LOCATIE
```

```
START WITH 1
```

```
INCREMENT BY 1
```

```
MINVALUE 0
```

```
MAXVALUE 10
```

```
NOCYCLE;
```

```
INSERT INTO LOCATIE VALUES (SEQ_LOCATIE.NEXTVAL, 'Cluj', 'Cluj-Napoca', 'Bogdan  
Petriceicu Hasdeu', 82);
```

```
INSERT INTO LOCATIE VALUES (SEQ_LOCATIE.NEXTVAL, 'Bucuresti', 'Bucuresti', '1  
Decembrie 1918', 27);
```

```
INSERT INTO LOCATIE VALUES (SEQ_LOCATIE.NEXTVAL, 'Maramures', 'Baia Mare',  
'Transilvaniei', 8);
```

```
INSERT INTO LOCATIE VALUES (SEQ_LOCATIE.NEXTVAL, 'Bihor', 'Oradea', 'Calea  
Republicii', 3);
```

```
INSERT INTO LOCATIE VALUES (SEQ_LOCATIE.NEXTVAL, 'Constanta', 'Constanta',
'Avram Iancu', 63);
```

```
INSERT INTO LOCATIE VALUES (SEQ_LOCATIE.NEXTVAL, 'Brasov', 'Brasov',
'Bulevardul Tineretului', 13);
```

```
COMMIT;
```

```
SELECT * FROM LOCATIE;
```

	COD_LOCATIE	JUDET	ORAS	STRADA	NUMAR
1	1	Cluj	Cluj-Napoca	Boqdan Petriceicu Hasdeu	82
2	2	Bucuresti	Bucuresti	1 Decembrie 1918	27
3	3	Maramures	Baia Mare	Transilvaniei	8
4	4	Bihor	Oradea	Calea Republicii	3
5	5	Constanta	Constanta	Avram Iancu	63
6	6	Brasov	Brasov	Bulevardul Tineretului	13

2. Tabela independentă SALON

```
CREATE TABLE SALON
```

```
(cod_salon number(6),
```

```
nume_salon varchar2(25) constraint nume_sal not null,
```

```
cod_locatie number(6),
```

```
constraint pk_salon primary key(cod_salon),
```

```
constraint fk_salon_locatie foreign key(cod_locatie) references locatie(cod_locatie)
```

```
);
```

```
INSERT INTO SALON VALUES (1, 'Rstyle', 3);
```

```
INSERT INTO SALON VALUES (2, 'Black Beauty', 5);
```

```
INSERT INTO SALON VALUES (3, 'Why Not', 2);
```

```
INSERT INTO SALON VALUES (4, 'Bella', 4);
```

```
INSERT INTO SALON VALUES (5, 'Alisa', 1);
```

```
INSERT INTO SALON VALUES (6, 'Why Not', 6);
```

```
COMMIT;
```

```
SELECT * FROM SALON;
```

	COD_SALON	NUME_SALON	COD_LOCATIE
1	1	Rstyle	3
2	2	Black Beauty	5
3	3	Why Not	2
4	4	Bella	4
5	5	Alisa	1
6	6	Why Not	6

3. Tabela independentă INGRIJITOR

```
CREATE TABLE INGRIJITOR
```

```
(cod_ingrijitor number(6),
nume_ingrijitor varchar2(20) constraint nume_in not null,
prenume_ingrijitor varchar2(20) constraint prenume_in not null,
salariu_ingrijitor number(6),
constraint pk_ingrijitor primary key(cod_ingrijitor)
);
```

```
INSERT INTO INGRIJITOR VALUES(1, 'Constantinescu', 'Adrian', 2200);
```

```
INSERT INTO INGRIJITOR VALUES(2, 'Popescu', 'Radu', 2340);
```

```
INSERT INTO INGRIJITOR VALUES(3, 'Ionescu', 'Paul', 2500);
```

```
INSERT INTO INGRIJITOR VALUES(4, 'Paunescu', 'George', 2105);
```

```
INSERT INTO INGRIJITOR VALUES(5, 'Antonescu', 'Ana', 2550);
```

```
COMMIT;
```

```
SELECT * FROM INGRIJITOR;
```

	COD_INGRIJITOR	NUME_INGRIJITOR	PRENUME_INGRIJITOR	SALARIU_INGRIJITOR
1	1	Constantinescu	Adrian	2200
2	2	Popescu	Radu	2340
3	3	Ionescu	Paul	2500
4	4	Paunescu	George	2105
5	5	Antonescu	Ana	2550

4. Tabela independentă CLIENT

```
CREATE TABLE CLIENT
```

```
(cod_client number(6),
nume_client varchar(20) constraint nume_cli not null,
prenume_client varchar(20) constraint prenume_cli not null,
numar_telefon varchar(13),
adresa_email varchar2(30),
constraint pk_client primary key(cod_client)
);
```

```
INSERT INTO CLIENT VALUES (100, 'Voicu', 'Stefania', '0768486544',
'stefania.voicu@hotmail.com');
```

```
INSERT INTO CLIENT VALUES (200, 'Sandu', 'Andreea', '0765465145',
'sandu_andreea@gmail.com');
```

```
INSERT INTO CLIENT VALUES (300, 'Aioanei', 'Lia', '0765321231', 'liavoicu@yahoo.com');
```

```
INSERT INTO CLIENT VALUES (400, 'Trandafir', 'Anton', '0764351235', 'trandafir-anton@gmail.com');
```

```
INSERT INTO CLIENT VALUES (500, 'Radulescu', 'Alexandru', '0745216644',
'alexandru.radulescu@yahoo.com');
```

```
INSERT INTO CLIENT VALUES (600, 'Mihai', 'Ioana', '0745657516',
'mihai.ioana@hotmail.com');
```

```
INSERT INTO CLIENT VALUES (700, 'Radu', 'Cristina', '0748165297',
'cristinaradu@gmail.com');
```

```
INSERT INTO CLIENT VALUES (800, 'Panait', 'Ovidiu', '0735482156',
'panait_ovidiu@yahoo.com');
```

```
INSERT INTO CLIENT VALUES (900, 'Preda', 'Nicu', '0724568154', 'nicu-preda@gmail.com');
```

```
INSERT INTO CLIENT VALUES (1000, 'Oprea', 'Petru', '0756842154',
'petru.oprea@yahoo.com');
```

```
COMMIT;
```



```
SELECT * FROM CLIENT;
```

	COD_CLIENT	NUME_CLIENT	PRENUME_CLIENT	NUMAR_TELEFON	ADRESA_EMAIL
1	100	Voicu	Stefania	0768486544	stefania.voicu@hotmail.com
2	200	Sandu	Andreea	0765465145	sandu andreea@gmail.com
3	300	Aioanei	Lia	0765321231	liavoicu@yahoo.com
4	400	Trandafir	Anton	0764351235	trandafir-anton@gmail.com
5	500	Radulescu	Alexandru	0745216644	alexandru.radulescu@yahoo.com
6	600	Mihai	Ioana	0745657516	mihai.ioana@hotmail.com
7	700	Radu	Cristina	0748165297	cristinaradu@gmail.com
8	800	Panait	Ovidiu	0735482156	panait ovidiu@yahoo.com
9	900	Preda	Nicu	0724568154	nicu-preda@gmail.com
10	1000	Oprea	Petru	0756842154	petru.oprea@yahoo.com

5. Tabela independent SERVICIU

```
CREATE TABLE SERVICIU
```

```
(cod_serviciu number(6),
```

```
denumire varchar2(20) constraint denumire_serv not null,
```

```
durata_serviciu number(3, 2) default 0,
```

```
pret number(3) constraint pret_serv not null,
```

```
constraint pk_serviciu primary key(cod_serviciu)
```

```
);
```

```
INSERT INTO SERVICIU VALUES (1, 'Tuns', 0.5, 30);
```

```
INSERT INTO SERVICIU VALUES (2, 'Vopsit', 2, 100);
```

```
INSERT INTO SERVICIU VALUES (3, 'Manichiura', 0.5, 40);
```

```
INSERT INTO SERVICIU VALUES (4, 'Pedichiura', 1, 50);
```

```
INSERT INTO SERVICIU VALUES (5, 'Masaj', 1, 50);
```

```
COMMIT;
```

```
SELECT * FROM SERVICIU;
```

	COD_SERVICIU	DENUMIRE	DURATA_SERVICIU	PRET
1	1	Tuns	0.5	30
2	2	Vopsit	2	100
3	3	Manichiura	0.5	40
4	4	Pedichiura	1	50
5	5	Masaaj	1	50

6. Tabela independentă PROGRAMARE

```
CREATE TABLE PROGRAMARE
```

```
(cod_programare number(6),
data_programare date default sysdate,
cod_client number(6),
cod_serviciu number(6),
constraint pk_programare primary key(cod_programare),
constraint fk_programare_client foreign key(cod_client) references client(cod_client),
constraint fk_programare_serviciu foreign key(cod_serviciu) references serviciu(cod_serviciu)
);
```

```
INSERT INTO PROGRAMARE VALUES (1, TO_DATE('01-07-2021 13:30', 'dd-mm-yyyy hh24:mi'), 300, 5);
```

```
INSERT INTO PROGRAMARE VALUES (2, TO_DATE('10-07-2021 08:30', 'dd-mm-yyyy hh24:mi'), 100, 3);
```

```
INSERT INTO PROGRAMARE VALUES (3, TO_DATE('05-07-2021 10:30', 'dd-mm-yyyy hh24:mi'), 200, 2);
```

```
INSERT INTO PROGRAMARE VALUES (4, TO_DATE('02-07-2021 18:30', 'dd-mm-yyyy hh24:mi'), 500, 5);
```

```
INSERT INTO PROGRAMARE VALUES (5, TO_DATE('07-07-2021 12:30', 'dd-mm-yyyy hh24:mi'), 400, 1);
```

```
INSERT INTO PROGRAMARE VALUES (6, TO_DATE('01-07-2021 13:30', 'dd-mm-yyyy hh24:mi'), 800, 2);
```

```
INSERT INTO PROGRAMARE VALUES (7, TO_DATE('10-07-2021 08:30', 'dd-mm-yyyy
hh24:mi'), 1000, 3);
```

```
INSERT INTO PROGRAMARE VALUES (8, TO_DATE('05-07-2021 10:30', 'dd-mm-yyyy
hh24:mi'), 900, 4);
```

```
INSERT INTO PROGRAMARE VALUES (9, TO_DATE('02-07-2021 18:30', 'dd-mm-yyyy
hh24:mi'), 900, 1);
```

```
INSERT INTO PROGRAMARE VALUES (10, TO_DATE('07-07-2021 12:30', 'dd-mm-yyyy
hh24:mi'), 200, 1);
```

```
COMMIT;
```

```
SELECT * FROM PROGRAMARE;
```

	⚡ COD_PROGRAMARE	⚡ DATA_PROGRAMARE	⚡ COD_CLIENT	⚡ COD_SERVICIU
1	1	01-JUL-21	300	5
2	2	10-JUL-21	100	3
3	3	05-JUL-21	200	2
4	4	02-JUL-21	500	5
5	5	07-JUL-21	400	1
6	6	01-JUL-21	800	2
7	7	10-JUL-21	1000	3
8	8	05-JUL-21	900	4
9	9	02-JUL-21	900	1
10	10	07-JUL-21	200	1

7. Tabela independentă JOBS

```
CREATE TABLE JOBS
```

```
(cod_job number(6),
```

```
nume_job varchar2(20) constraint nume_job not null,
```

```
salariu_min number(5),
```

```
salariu_max number(5),
```

```
constraint pk_jobs primary key(cod_job)
```

```
);
```

```

INSERT INTO JOBS VALUES(1, 'Frizer', 2000, 3000);
INSERT INTO JOBS VALUES(2, 'Coafeza', 2200, 3300);
INSERT INTO JOBS VALUES(3, 'Manichiurist', 2000, 2500);
INSERT INTO JOBS VALUES(4, 'Pedichiurist', 1800, 2500);
INSERT INTO JOBS VALUES(5, 'Maseur', 3000, 3500);
COMMIT;

```

```
SELECT * FROM JOBS;
```

	COD_JOB	NUME_JOB	SALARIU_MIN	SALARIU_MAX
1	1	Frizer	2000	3000
2	2	Coafeza	2200	3300
3	3	Manichiurist	2000	2500
4	4	Pedichiurist	1800	2500
5	5	Maseur	3000	3500

8. Tabela independentă FURNIZOR

```
CREATE TABLE FURNIZOR
```

```

(cod_furnizor number(6),
suma number(10) constraint suma_furn not null,
data_contract date default sysdate,
durata number(3,2) default null,
constraint pk_furnizor primary key(cod_furnizor)
);

```

```

INSERT INTO FURNIZOR VALUES(10, 5000, TO_DATE('01-05-2021', 'dd-mm-yyyy'), 0.25);
INSERT INTO FURNIZOR VALUES(20, 7500, TO_DATE('10-04-2021', 'dd-mm-yyyy'), 0.5);
INSERT INTO FURNIZOR VALUES(30, 10000, TO_DATE('12-03-2021', 'dd-mm-yyyy'), 1);
INSERT INTO FURNIZOR VALUES(40, 15000, SYSDATE, NULL);
INSERT INTO FURNIZOR VALUES(50, 9000, TO_DATE('23-02-2021', 'dd-mm-yyyy'),
NULL);

```

COMMIT;

SELECT * FROM FURNIZOR;

	COD_FURNIZOR	SUMA	DATA_CONTRACT	DURATA
1	10	5000	01-MAY-21	0.25
2	20	7500	10-APR-21	0.5
3	30	10000	12-MAR-21	1
4	40	15000	03-JAN-22	(null)
5	50	9000	23-FEB-21	(null)

9. Tabela independentă ANGAJAT

CREATE TABLE ANGAJAT

```
(cod_angajat number(6),
nume_angajat varchar2(20) constraint nume_ang not null,
prenume_angajat varchar2(20) constraint prenume_ang not null,
salariu_angajat number(6),
data_angajarii date default sysdate,
cod_job number(6),
constraint pk_angajat primary key(cod_angajat),
constraint fk_angajat_job foreign key(cod_job) references jobs(cod_job)
);
```

INSERT INTO ANGAJAT VALUES (100, 'Georgescu', 'Anastasia', 2115, TO_DATE('01-05-2018', 'dd-mm-yyyy'), 1);

INSERT INTO ANGAJAT VALUES (101, 'Calinescu', 'Andreea', 2786, TO_DATE('01-07-2017', 'dd-mm-yyyy'), 2);

INSERT INTO ANGAJAT VALUES (102, 'Voineasa', 'Bianca', 2450, TO_DATE('03-08-2019', 'dd-mm-yyyy'), 3);

INSERT INTO ANGAJAT VALUES (103, 'Alexandrescu', 'Diana', 1978, TO_DATE('13-12-2018', 'dd-mm-yyyy'), 4);

INSERT INTO ANGAJAT VALUES (104, 'Anghel', 'Florentina', 3200, TO_DATE('14-10-2017', 'dd-mm-yyyy'), 5);

INSERT INTO ANGAJAT VALUES (105, 'Andreescu', 'Ana', 2455, TO_DATE('10-06-2017', 'dd-mm-yyyy'), 1);

INSERT INTO ANGAJAT VALUES (106, 'Pop', 'Dorina', 3126, TO_DATE('01-07-2016', 'dd-mm-yyyy'), 2);

INSERT INTO ANGAJAT VALUES (107, 'Ion', 'Raluca', 2350, TO_DATE('13-11-2019', 'dd-mm-yyyy'), 3);

INSERT INTO ANGAJAT VALUES (108, 'Mircescu', 'Maria', 2078, TO_DATE('12-01-2018', 'dd-mm-yyyy'), 4);

INSERT INTO ANGAJAT VALUES (109, 'Petrescu', 'Horia', 3300, TO_DATE('13-10-2017', 'dd-mm-yyyy'), 5);

INSERT INTO ANGAJAT VALUES (110, 'Paun', 'Madalina', 2615, TO_DATE('01-12-2018', 'dd-mm-yyyy'), 1);

INSERT INTO ANGAJAT VALUES (111, 'Dumitru', 'Daria', 2886, TO_DATE('23-07-2017', 'dd-mm-yyyy'), 2);

INSERT INTO ANGAJAT VALUES (112, 'Dumitriu', 'Antonia', 2150, TO_DATE('13-10-2019', 'dd-mm-yyyy'), 3);

INSERT INTO ANGAJAT VALUES (113, 'Balan', 'Adriana', 2378, TO_DATE('12-11-2018', 'dd-mm-yyyy'), 4);

INSERT INTO ANGAJAT VALUES (114, 'Trandafir', 'Toma', 3400, TO_DATE('12-04-2017', 'dd-mm-yyyy'), 5);

INSERT INTO ANGAJAT VALUES (115, 'Calin', 'Beatrice', 2345, TO_DATE('01-03-2018', 'dd-mm-yyyy'), 1);

INSERT INTO ANGAJAT VALUES (116, 'Geamanu', 'Laura', 2986, TO_DATE('14-07-2017', 'dd-mm-yyyy'), 2);

INSERT INTO ANGAJAT VALUES (117, 'Diacon', 'Larisa', 1950, TO_DATE('03-10-2019', 'dd-mm-yyyy'), 3);

INSERT INTO ANGAJAT VALUES (118, 'Breazu', 'Diana', 1878, TO_DATE('15-12-2018', 'dd-mm-yyyy'), 4);

INSERT INTO ANGAJAT VALUES (119, 'Savu', 'Andrei', 2800, TO_DATE('14-12-2017', 'dd-mm-yyyy'), 5);

INSERT INTO ANGAJAT VALUES (120, 'Alecui', 'Alexandra', 2905, TO_DATE('11-11-2018', 'dd-mm-yyyy'), 1);

```
INSERT INTO ANGAJAT VALUES (121, 'Florea', 'Delia', 3186, TO_DATE('18-02-2017', 'dd-mm-yyyy'), 2);
```

```
INSERT INTO ANGAJAT VALUES (122, 'Popescu', 'Cornelia', 2220, TO_DATE('12-06-2019', 'dd-mm-yyyy'), 3);
```

```
INSERT INTO ANGAJAT VALUES (123, 'Nae', 'Victoria', 2378, TO_DATE('04-12-2018', 'dd-mm-yyyy'), 4);
```

```
INSERT INTO ANGAJAT VALUES (124, 'Florescu', 'Catalin', 3000, TO_DATE('25-10-2017', 'dd-mm-yyyy'), 5);
```

```
INSERT INTO ANGAJAT VALUES (125, 'Antonescu', 'Ioana', 2340, TO_DATE('13-10-2019', 'dd-mm-yyyy'), 2);
```

```
COMMIT;
```

```
SELECT * FROM ANGAJAT;
```

	COD_ANGAJAT	NUME_ANGAJAT	PRENUME_ANGAJAT	SALARIU_ANGAJAT	DATA_ANGAJARII	COD_JOB
1	100	Georgescu	Anastasia	2115	01-MAY-18	1
2	101	Calinescu	Andreea	2786	01-JUL-17	2
3	102	Voineasa	Bianca	2450	03-AUG-19	3
4	103	Alexandrescu	Diana	1978	13-DEC-18	4
5	104	Anghel	Florentina	3200	14-OCT-17	5
6	105	Andreescu	Ana	2455	10-JUN-17	1
7	106	Pop	Dorina	3126	01-JUL-16	2
8	107	Ion	Raluca	2350	13-NOV-19	3
9	108	Mircescu	Maria	2078	12-JAN-18	4
10	109	Petrescu	Horia	3300	13-OCT-17	5
11	110	Paun	Madalina	2615	01-DEC-18	1
12	111	Dumitru	Daria	2886	23-JUL-17	2
13	112	Dumitriu	Antonia	2150	13-OCT-19	3
14	113	Balan	Adriana	2378	12-NOV-18	4
15	114	Trandafir	Toma	3400	12-APR-17	5
16	115	Calin	Beatrice	2345	01-MAR-18	1
17	116	Geamanu	Laura	2986	14-JUL-17	2
18	117	Diacon	Larisa	1950	03-OCT-19	3
19	119	Savu	Andrei	2800	14-DEC-17	5
20	120	Alecu	Alexandra	2905	11-NOV-18	1
21	121	Florea	Delia	3186	18-FEB-17	2
22	122	Popescu	Cornelia	2220	12-JUN-19	3
23	123	Nae	Victoria	2378	04-DEC-18	4
24	124	Florescu	Catalin	3000	25-OCT-17	5
25	125	Antonescu	Ioana	2340	13-OCT-19	2
26	118	Breazu	Diana	1878	15-DEC-18	4

10. Tabela asociativă APROVIZIONAT_DE

```
CREATE TABLE APROVIZIONAT_DE
```

```
(cod_furnizor number(6),
```

```
cod_salon number(6),
```

```
constraint pk_aprovizionat_de primary key(cod_salon, cod_furnizor),
```

```
constraint fk_aprovizionat_de_salon foreign key (cod_salon) references salon(cod_salon),
```

```
constraint fk_aprovizionat_de_furnizor foreign key (cod_furnizor) references  
furnizor(cod_furnizor)
```

```
);
```

```
INSERT INTO APROVIZIONAT_DE VALUES(50,3);
```

```
INSERT INTO APROVIZIONAT_DE VALUES(30,4);
```

```
INSERT INTO APROVIZIONAT_DE VALUES(20,1);
```

```
INSERT INTO APROVIZIONAT_DE VALUES(40,2);
```

```
INSERT INTO APROVIZIONAT_DE VALUES(10,5);
```

```
INSERT INTO APROVIZIONAT_DE VALUES(20,3);
```

```
INSERT INTO APROVIZIONAT_DE VALUES(10,4);
```

```
INSERT INTO APROVIZIONAT_DE VALUES(50,1);
```

```
INSERT INTO APROVIZIONAT_DE VALUES(30,2);
```

```
INSERT INTO APROVIZIONAT_DE VALUES(40,5);
```

```
INSERT INTO APROVIZIONAT_DE VALUES(50,6);
```

```
INSERT INTO APROVIZIONAT_DE VALUES(20,6);
```

```
COMMIT;
```

```
SELECT * FROM APROVIZIONAT_DE;
```


	COD_FURNIZOR	COD_SALON
1	50	3
2	30	4
3	20	1
4	40	2
5	10	5
6	20	3
7	10	4
8	50	1
9	30	2
10	40	5
11	50	6
12	20	6

11. Tabela asociativă CURATAT_DE

```
CREATE TABLE CURATAT_DE
```

```
(cod_ingrijitor number(6),
```

```
cod_salon number(6),
```

```
constraint pk_curatat_de primary key(cod_salon, cod_ingrijitor),
```

```
constraint fk_curatat_de_salon foreign key (cod_salon) references salon(cod_salon),
```

```
constraint fk_curatat_de_ingrijitor foreign key (cod_ingrijitor) references  
ingrijitor(cod_ingrijitor)
```

```
);
```

```
INSERT INTO CURATAT_DE VALUES (1, 3);
```

```
INSERT INTO CURATAT_DE VALUES (2, 4);
```

```
INSERT INTO CURATAT_DE VALUES (3, 2);
```

```
INSERT INTO CURATAT_DE VALUES (4, 5);
```

```
INSERT INTO CURATAT_DE VALUES (5, 1);
```

```
INSERT INTO CURATAT_DE VALUES (1, 5);
```

```
INSERT INTO CURATAT_DE VALUES (2, 3);
```

```
INSERT INTO CURATAT_DE VALUES (3, 4);
```

```
INSERT INTO CURATAT_DE VALUES (4, 1);
```

```
INSERT INTO CURATAT_DE VALUES (5, 2);
```

```
COMMIT;
```

```
SELECT * FROM CURATAT_DE;
```

	COD_INGRIJITOR	COD_SALON
1	1	3
2	2	4
3	3	2
4	4	5
5	5	1
6	1	5
7	2	3
8	3	4
9	4	1
10	5	2

12. Tabela asociativă MERGE_LA

```
CREATE TABLE MERGE_LA
```

```
(cod_angajat number(6),
```

```
cod_client number(6),
```

```
cod_salon number(6),
```

```
constraint pk_merge_la primary key(cod_angajat, cod_client, cod_salon),
```

```
constraint fk_merge_la_angajat foreign key (cod_angajat) references angajat(cod_angajat),
```

```
constraint fk_merge_la_client foreign key (cod_client) references client(cod_client),
```

```
constraint fk_merge_la_salon foreign key (cod_salon) references salon(cod_salon)
```

```
);
```

```
INSERT INTO MERGE_LA VALUES (124, 300, 5);
```

```
INSERT INTO MERGE_LA VALUES (112, 100, 3);
```

```
INSERT INTO MERGE_LA VALUES (101, 200, 1);
```

```
INSERT INTO MERGE_LA VALUES (114, 500, 3);
```

```
INSERT INTO MERGE_LA VALUES (115 , 400, 4);
```

```

INSERT INTO MERGE_LA VALUES (121, 800, 5);
INSERT INTO MERGE_LA VALUES (102, 1000, 1);
INSERT INTO MERGE_LA VALUES (108, 900, 2);
INSERT INTO MERGE_LA VALUES (105, 900, 2);
INSERT INTO MERGE_LA VALUES (100, 200, 1);
INSERT INTO MERGE_LA VALUES (125, 200, 3);
COMMIT;

```

```
SELECT * FROM MERGE_LA;
```

	COD_ANGAJAT	COD_CLIENT	COD_SALON
1	124	300	5
2	112	100	3
3	101	200	1
4	114	500	3
5	115	400	4
6	121	800	5
7	102	1000	1
8	108	900	2
9	105	900	2
10	100	200	1
11	125	200	3

Cerința 6.

Afișați clienții care au programare la un serviciu cu id-ul dat și care merg la angajații care au salariul mai mare decât media salariilor angajaților.

```

CREATE OR REPLACE PROCEDURE p6 (id_serviciu serviciu.cod_serviciu%TYPE)
IS
TYPE vector IS VARRAY(30) OF angajat.cod_angajat%TYPE;
TYPE t_imbricat IS TABLE OF client.cod_client%TYPE;
v_ang vector := vector();
v_cli t_imbricat := t_imbricat();
medie angajat.salariu_angajat%TYPE;
contor NUMBER(3);

```

```

CURSOR c IS
    (SELECT cod_angajat, cod_client
    FROM merge_la);
BEGIN
    SELECT AVG(salariu_angajat), COUNT(*)
    INTO medie, contor
    FROM angajat;

    FOR i IN 1..contor LOOP
        v_ang.extend;
    END LOOP;

    SELECT cod_angajat
    BULK COLLECT INTO v_ang
    FROM angajat
    WHERE salariu_angajat > medie
    ORDER BY salariu_angajat DESC;

    SELECT cod_client
    BULK COLLECT INTO v_cli
    FROM client c JOIN programare p USING (cod_client)
    WHERE p.cod_serviciu = id_serviciu;

    FOR i in c LOOP
        FOR j in 1..v_cli.count LOOP
            IF v_cli(j) = i.cod_client THEN
                FOR k in 1..v_ang.count LOOP
                    IF v_ang(k) = i.cod_angajat THEN

```

```

        DBMS_OUTPUT.PUT_LINE('Clientul cu id-ul '||i.cod_client||' merge la angajatul
cu id-ul '||i.cod_angajat);
    END IF;
END LOOP;
END IF;
END LOOP;
END LOOP;
END p6;
/
BEGIN
    p6(5);
END;
```

In urma rulării, s-a obținut următorul rezultat:

```

Clientul cu id-ul 300 merge la angajatul cu id-ul 124
Clientul cu id-ul 500 merge la angajatul cu id-ul 114
```

Cerința 7.

Afișați angajații angajați începând cu data de 1 iulie 2018 care au job-ul al cărui cod este dat ca parametru.

In cazul în care nu există, afișați mesajul 'Nu există'.

```

CREATE OR REPLACE PROCEDURE p7 (id_job angajat.cod_job%TYPE)
IS
    contor NUMBER(2) := 0;
    v_numa angajat.numa_angajat%TYPE;
    v_pnuma angajat.pnuma_angajat%TYPE;
    v_data angajat.data_angajarii%TYPE;
```

```
CURSOR c IS (SELECT nume_angajat, prenume_angajat, data_angajarii
              FROM angajat
              WHERE data_angajarii > '01-JUL-2018' AND cod_job = id_job);

BEGIN
    OPEN c;
    LOOP
        FETCH c INTO v_nume, v_prenume, v_data;
        EXIT WHEN c%NOTFOUND;
        contor := contor + 1;
        DBMS_OUTPUT.PUT_LINE(v_nume||' '||v_prenume||' '||v_data);
    END LOOP;
    IF contor = 0 THEN
        DBMS_OUTPUT.PUT_LINE('Nu exista');
    END IF;
    CLOSE c;
END p7;

/

BEGIN
    p7(4);
    DBMS_OUTPUT.PUT_LINE('-----');
    p7(5);
END;
```

În urma rulării, s-a obținut următorul rezultat:

```
Alexandrescu Diana 13-DEC-18
Balan Adriana 12-NOV-18
Nae Victoria 04-DEC-18
Breazu Diana 15-DEC-18
-----
Nu exista
```

Cerinta 8.

Afișați orașul în care se află salonul cu numele dat ca parametru care este aprovizionat de un furnizor cu o anumită sumă dată ca parametru.

```
CREATE OR REPLACE FUNCTION f8 (v_suma furnizor.suma%TYPE,
                                v_nume_salon salon.nume_salon%TYPE)
RETURN locatie.oras%TYPE
IS
    v_oras locatie.oras%TYPE;
BEGIN
    SELECT l.oras
    INTO v_oras
    FROM locatie l JOIN salon s USING (cod_locatie)
        JOIN aprovizionat_de a USING (cod_salon)
        JOIN furnizor f USING (cod_furnizor)
    WHERE v_suma = suma and upper(v_nume_salon) = upper(s.nume_salon);
    RETURN v_oras;

EXCEPTION
    WHEN NO_DATA_FOUND THEN
        RAISE_APPLICATION_ERROR(-20000, 'Nu exista astfel de oras!');
```

```

WHEN TOO_MANY_ROWS THEN

    RAISE_APPLICATION_ERROR(-20001, 'Exista mai multe orase care indeplinesc
conditia!');

WHEN OTHERS THEN

    RAISE_APPLICATION_ERROR(-20002, 'Alta eroare!');

END f8;

/

BEGIN

    DBMS_OUTPUT.PUT_LINE(f8(5000,'Rstyle')); --nu exista

END;

```

```

    DBMS_OUTPUT.PUT_LINE(pachet_proiect.f8(5000,'Rstyle')); --nu exista
END;
Error report -
ORA-20000: Nu exista astfel de oras!
ORA-06512: at "TESTUSER.PACHET_PROIECT", line 85
ORA-06512: at line 2
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.

```

```

BEGIN

    DBMS_OUTPUT.PUT_LINE(f8(5000, 'Alisa'));

END;

```

Cluj-Napoca


```
BEGIN
```

```
    DBMS_OUTPUT.PUT_LINE(f8(7500, 'Why Not')); --prea multe
```

```
END;
```

```
Error starting at line : 484 in command -
```

```
BEGIN
```

```
    DBMS_OUTPUT.PUT_LINE(f8(7500, 'Why Not')); --prea multe
```

```
END;
```

```
Error report -
```

```
ORA-20001: Exista mai multe orase care indeplinesc conditia!
```

```
ORA-06512: at "TESTUSER.F8", line 18
```

```
ORA-06512: at line 2
```

Cerința 9.

Afișați data angajării și salariul minim de pe job-ul celui mai recent angajat care lucrează într-un oraș dat ca parametru.

```
CREATE OR REPLACE PROCEDURE p9 (v_oras locatie.oras%TYPE)
```

```
IS
```

```
    v_salariu jobs.salariu_min%TYPE;
```

```
    v_data angajat.data_angajarii%TYPE;
```

```
BEGIN
```

```
    SELECT max(data_angajarii)
```

```
    INTO v_data
```

```
    FROM jobs j JOIN angajat a USING (cod_job)
```

```
         JOIN merge_la m USING (cod_angajat)
```

```
         JOIN salon s USING (cod_salon)
```

```
         JOIN locatie l USING (cod_locatie)
```

```
    WHERE upper(l.oras) = upper(v_oras);
```

```
SELECT j.salariu_min
INTO v_salariu
FROM jobs j JOIN angajat a USING (cod_job)
      JOIN merge_la m USING (cod_angajat)
      JOIN salon s USING (cod_salon)
      JOIN locatie l USING (cod_locatie)
WHERE upper(l.oras) = upper(v_oras)
      and data_angajarii = v_data;

DBMS_OUTPUT.PUT_LINE('Salariul cautat este '||v_salariu||' si data angajarii: '||v_data);

EXCEPTION

  WHEN NO_DATA_FOUND THEN
    RAISE_APPLICATION_ERROR(-20000, 'Nu exista astfel de intrari in baza de date!');

  WHEN TOO_MANY_ROWS THEN
    RAISE_APPLICATION_ERROR(-20001, 'Exista mai multe intrari care indeplinesc
conditia!');

  WHEN OTHERS THEN
    RAISE_APPLICATION_ERROR(-20002, 'Alta eroare!');

END p9;
```

```
BEGIN
```

```
    p9('bucuresti'); --prea multi
```

```
END;
```

```
Error starting at line : 535 in command -
```

```
BEGIN
```

```
    p9('bucuresti'); --prea multi
```

```
END;
```

```
Error report -
```

```
ORA-20002: Exista mai multe intrari care indeplinesc conditia!
```

```
ORA-06512: at "TESTUSER.P9", line 30
```

```
ORA-06512: at line 2
```

```
BEGIN
```

```
    p9('bacau'); --nu exista
```

```
END;
```

```
Error starting at line : 539 in command -
```

```
BEGIN
```

```
    p9('bacau'); --nu exista
```

```
END;
```

```
Error report -
```

```
ORA-20001: Nu exista astfel de intrari in baza de date!
```

```
ORA-06512: at "TESTUSER.P9", line 28
```

```
ORA-06512: at line 2
```

```
BEGIN
```

```
    p9('cluj-napoca');
```

```
END;
```

```
Salariul cautat este 3000 si data angajarii: 25-OCT-17
```

Cerința 10.

Definiți un trigger care să nu permită modificări asupra tabelului programare în afara programului saloanelor. Orar saloane: L-V: 8-20, S: 8-14.

```
CREATE OR REPLACE TRIGGER trig_ex10
```

```
    BEFORE INSERT OR UPDATE OR DELETE ON programare
```

```
BEGIN
```

```
    IF (TO_CHAR(SYSDATE, 'd') = 1) OR ((TO_CHAR(SYSDATE, 'd') = 7) AND  
    (TO_CHAR(SYSDATE, 'HH24') NOT BETWEEN 8 AND 14)) OR
```

```
        (TO_CHAR(SYSDATE, 'HH24') NOT BETWEEN 8 AND 20)
```

```
    THEN
```

```
        IF INSERTING THEN
```

```
            RAISE_APPLICATION_ERROR(-20001, 'Nu puteti face o programare in afara orelor de  
lucru!');
```

```
        ELSIF UPDATING THEN
```

```
            RAISE_APPLICATION_ERROR(-20002, 'Nu puteti modifica o programare in afara  
orelor de lucru!');
```

```
        ELSE
```

```
            RAISE_APPLICATION_ERROR(-20003, 'Nu puteti sterge o programare in afara orelor  
de lucru!');
```

```
        END IF;
```

```
    END IF;
```

```
END;
```

Am cuprins în captura de ecran și ora la care a fost rulată instrucțiunea de INSERT.

```

INSERT INTO PROGRAMARE VALUES (11, TO_DATE('07-07-2021 12:30', 'dd-mm-yyyy hh24:mi'), 200, 1);

```

Task completed in 0.21 seconds

Error starting at line : 575 in command -
 INSERT INTO PROGRAMARE VALUES (11, TO_DATE('07-07-2021 12:30', 'dd-mm-yyyy hh24:mi'), 200, 1)
 Error report -
 ORA-20001: Nu puteti face o programare in afara orelor de lucru!
 ORA-06512: at "TESTUSER.TRIG_EX10", line 6
 ORA-04088: error during execution of trigger 'TESTUSER.TRIG_EX10'

Cerința 11.

Definiți un trigger care să nu permită modificarea salariului unui angajat dacă se face o micșorare de mai mult de 10% sau dacă se depășesc limitele de salariu pe job-ul angajatului respectiv.

CREATE OR REPLACE TRIGGER trig_ex11

BEFORE UPDATE OF salariu_angajat ON angajat

FOR EACH ROW

DECLARE

sal_min jobs.salariu_min%TYPE;

sal_max jobs.salariu_max%TYPE;

BEGIN

SELECT salariu_min, salariu_max

INTO sal_min, sal_max

FROM jobs

WHERE cod_job = :NEW.cod_job;

```

IF :NEW.salariu_angajat < 0.9 * :OLD.salariu_angajat THEN

    RAISE_APPLICATION_ERROR(-20001, 'Nu puteti micsora salariul cu mai mult de 10%!');

ELSIF :NEW.salariu_angajat < sal_min THEN

    RAISE_APPLICATION_ERROR(-20002, 'Nu puteti micsora salariul sub limita minima de
'||sal_min||' pe acest job!');

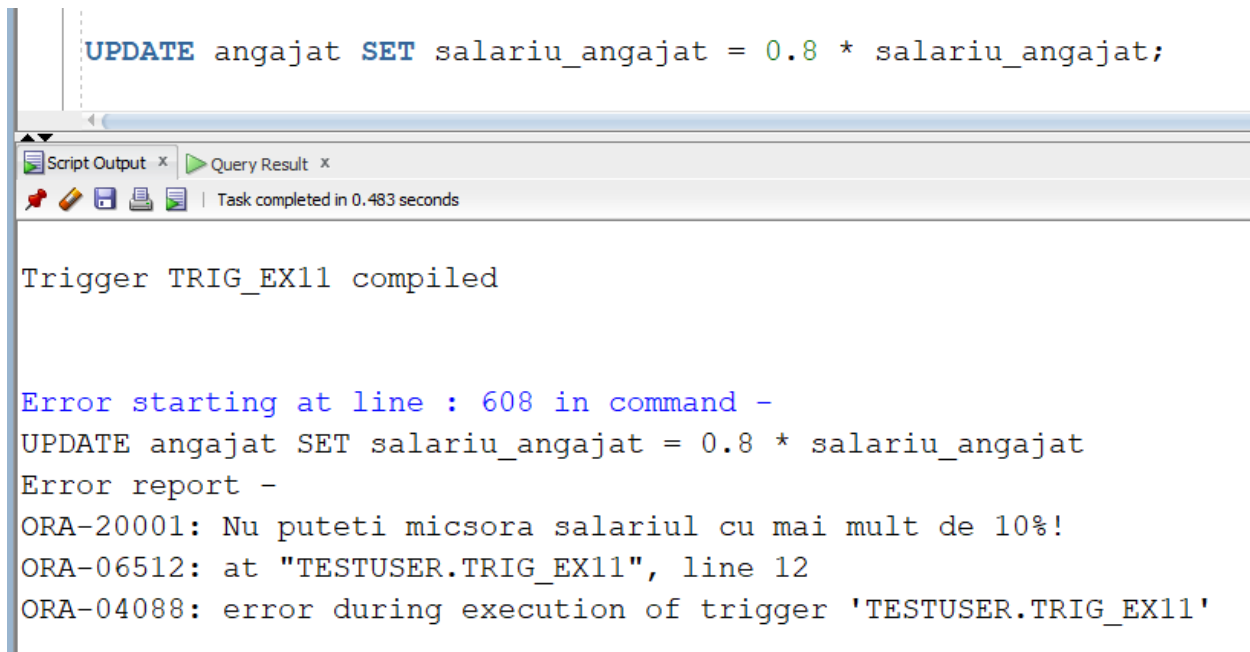
ELSIF :NEW.salariu_angajat > sal_max THEN

    RAISE_APPLICATION_ERROR(-20003, 'Nu puteti mari salariul peste limita maxima de
'||sal_max||' pe acest job!');

END IF;

END;

```



The screenshot shows a SQL development tool interface. At the top, a SQL statement is entered: `UPDATE angajat SET salariu_angajat = 0.8 * salariu_angajat;`. Below the editor, a status bar indicates "Task completed in 0.483 seconds". The main output area displays the following text:

```

Trigger TRIG_EX11 compiled

Error starting at line : 608 in command -
UPDATE angajat SET salariu_angajat = 0.8 * salariu_angajat
Error report -
ORA-20001: Nu puteti micsora salariul cu mai mult de 10%!
ORA-06512: at "TESTUSER.TRIG_EX11", line 12
ORA-04088: error during execution of trigger 'TESTUSER.TRIG_EX11'

```

Cerința 12.

Definiți un trigger care inserează în tabelul control_db date despre comenzile LDD ale utilizatorilor asupra bazei de date.

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

```
CREATE OR REPLACE TRIGGER trig_ex12
AFTER CREATE OR DROP OR ALTER ON SCHEMA
BEGIN
    INSERT INTO control_db
    VALUES (SYS.LOGIN_USER, SYS.DATABASE_NAME, SYS.SYSEVENT,
    SYS.DICTIONARY_OBJ_NAME, SYSDATE);
END;
```

Am rulat câteva comenzi de ALTER, CREATE și DROP pentru a popula tabelul.

```
ALTER TRIGGER trig_ex10 DISABLE;
ALTER TRIGGER trig_ex11 DISABLE;
ALTER TRIGGER trig_ex10 ENABLE;
ALTER TRIGGER trig_ex11 ENABLE;
```

```
CREATE SEQUENCE SEQ_TEST
```

```
START WITH 1
```

```
INCREMENT BY 1
```

```
MINVALUE 0
```

```
MAXVALUE 10
```

```
NOCYCLE;
```

```
DROP SEQUENCE SEQ_TEST;
```

```
SELECT * FROM control_db;
```

```
COMMIT;
```

	UTILIZATOR	NUME_BD	EVENTIMENT	NUME_OBIECT	DATA
1	TESTUSER XE		ALTER	TRIG EX10	03-JAN-22
2	TESTUSER XE		ALTER	TRIG EX11	03-JAN-22
3	TESTUSER XE		ALTER	TRIG EX10	03-JAN-22
4	TESTUSER XE		ALTER	TRIG EX11	03-JAN-22
5	TESTUSER XE		CREATE	SEQ TEST	03-JAN-22
6	TESTUSER XE		DROP	SEQ TEST	03-JAN-22

Cerința 13.

Crearea unui pachet care să conțină toate funcțiile și procedurile din proiect.

```
CREATE OR REPLACE PACKAGE pachet_proiect AS
```

```
    PROCEDURE p6(id_serviciu serviciu.cod_serviciu%TYPE);
```

```
    PROCEDURE p7 (id_job angajat.cod_job%TYPE);
```

```
    FUNCTION f8 (v_suma furnizor.suma%TYPE, v_numa_salon salon.numa_salon%TYPE)
```

```
        RETURN locatie.oras%TYPE;
```

```
    PROCEDURE p9 (v_oras locatie.oras%TYPE);
```

```
END pachet_proiect;
```


CREATE OR REPLACE PACKAGE BODY pachet_proiect AS

```
PROCEDURE p6(id_serviciu serviciu.cod_serviciu%TYPE)
IS
    TYPE vector IS VARRAY(30) OF angajat.cod_angajat%TYPE;
    TYPE t_imbricat IS TABLE OF client.cod_client%TYPE;
    v_ang vector := vector();
    v_cli t_imbricat := t_imbricat();
    medie angajat.salariu_angajat%TYPE;
    contor NUMBER(3);
    CURSOR c IS
        (SELECT cod_angajat, cod_client
         FROM merge_la);
BEGIN
    SELECT AVG(salariu_angajat), COUNT(*)
    INTO medie, contor
    FROM angajat;

    FOR i IN 1..contor LOOP
        v_ang.extend;
    END LOOP;

    SELECT cod_angajat
    BULK COLLECT INTO v_ang
    FROM angajat
    WHERE salariu_angajat > medie
    ORDER BY salariu_angajat DESC;
```

```

SELECT cod_client
BULK COLLECT INTO v_cli
FROM client c JOIN programare p USING (cod_client)
WHERE p.cod_serviciu = id_serviciu;

FOR i in c LOOP
  FOR j in 1..v_cli.count LOOP
    IF v_cli(j) = i.cod_client THEN
      FOR k in 1..v_ang.count LOOP
        IF v_ang(k) = i.cod_angajat THEN
          DBMS_OUTPUT.PUT_LINE('Clientul cu id-ul '||i.cod_client||' merge la
angajatul cu id-ul '||i.cod_angajat);
        END IF;
      END LOOP;
    END IF;
  END LOOP;
END LOOP;
END p6;
-----

PROCEDURE p7 (id_job angajat.cod_job%TYPE)
IS
  contor NUMBER(2) := 0;
  v_numa angajat.numa_angajat%TYPE;
  v_prename angajat.prename_angajat%TYPE;
  v_data angajat.data_angajarii%TYPE;
  CURSOR c IS (SELECT numa_angajat, prename_angajat, data_angajarii
    FROM angajat
    WHERE data_angajarii > '01-JUL-2018' AND cod_job = id_job);

```

```

BEGIN
    OPEN c;
    LOOP
        FETCH c INTO v_num, v_prenume, v_data;
        EXIT WHEN c%NOTFOUND;
        contor := contor + 1;
        DBMS_OUTPUT.PUT_LINE(v_num||' '||v_prenume||' '||v_data);
    END LOOP;
    IF contor = 0 THEN
        DBMS_OUTPUT.PUT_LINE('Nu exista');
    END IF;
    CLOSE c;
END p7;

```

```

-----
FUNCTION f8 (v_suma furnizor.suma%TYPE,
             v_num_salon salon.num_salon%TYPE)
    RETURN locatie.oras%TYPE
IS
    v_oras locatie.oras%TYPE;
BEGIN
    SELECT l.oras
    INTO v_oras
    FROM locatie l JOIN salon s USING (cod_locatie)
        JOIN aprovizionat_de a USING (cod_salon)
        JOIN furnizor f USING (cod_furnizor)
    WHERE v_suma = suma and upper(v_num_salon) = upper(s.num_salon);
    RETURN v_oras;

```

EXCEPTION

WHEN NO_DATA_FOUND THEN

RAISE_APPLICATION_ERROR(-20000, 'Nu exista astfel de oras!');

WHEN TOO_MANY_ROWS THEN

RAISE_APPLICATION_ERROR(-20001, 'Exista mai multe orase care indeplinesc conditia!');

WHEN OTHERS THEN

RAISE_APPLICATION_ERROR(-20002, 'Alta eroare!');

END f8;

PROCEDURE p9 (v_oras locatie.oras%TYPE)

IS

v_salariu jobs.salariu_min%TYPE;

v_data angajat.data_angajarii%TYPE;

BEGIN

SELECT max(data_angajarii)

INTO v_data

FROM jobs j JOIN angajat a USING (cod_job)

JOIN merge_la m USING (cod_angajat)

JOIN salon s USING (cod_salon)

JOIN locatie l USING (cod_locatie)

WHERE upper(l.oras) = upper(v_oras);

SELECT j.salariu_min

INTO v_salariu

FROM jobs j JOIN angajat a USING (cod_job)

JOIN merge_la m USING (cod_angajat)

JOIN salon s USING (cod_salon)

```
        JOIN locatie l USING (cod_locatie)
    WHERE upper(l.oras) = upper(v_oras)
        and data_angajarii = v_data;

    DBMS_OUTPUT.PUT_LINE('Salariul cautat este '||v_salariu||' si data angajarii: '||v_data);

EXCEPTION

    WHEN NO_DATA_FOUND THEN

        RAISE_APPLICATION_ERROR(-20000, 'Nu exista astfel de intrari in baza de date!');

    WHEN TOO_MANY_ROWS THEN

        RAISE_APPLICATION_ERROR(-20001, 'Exista mai multe intrari care indeplinesc
conditia!');

    WHEN OTHERS THEN

        RAISE_APPLICATION_ERROR(-20002, 'Alta eroare!');

    END p9;

END;
```