

Proiect – SGBD

Stafie Călin

Ianuarie 2022

Cuprins

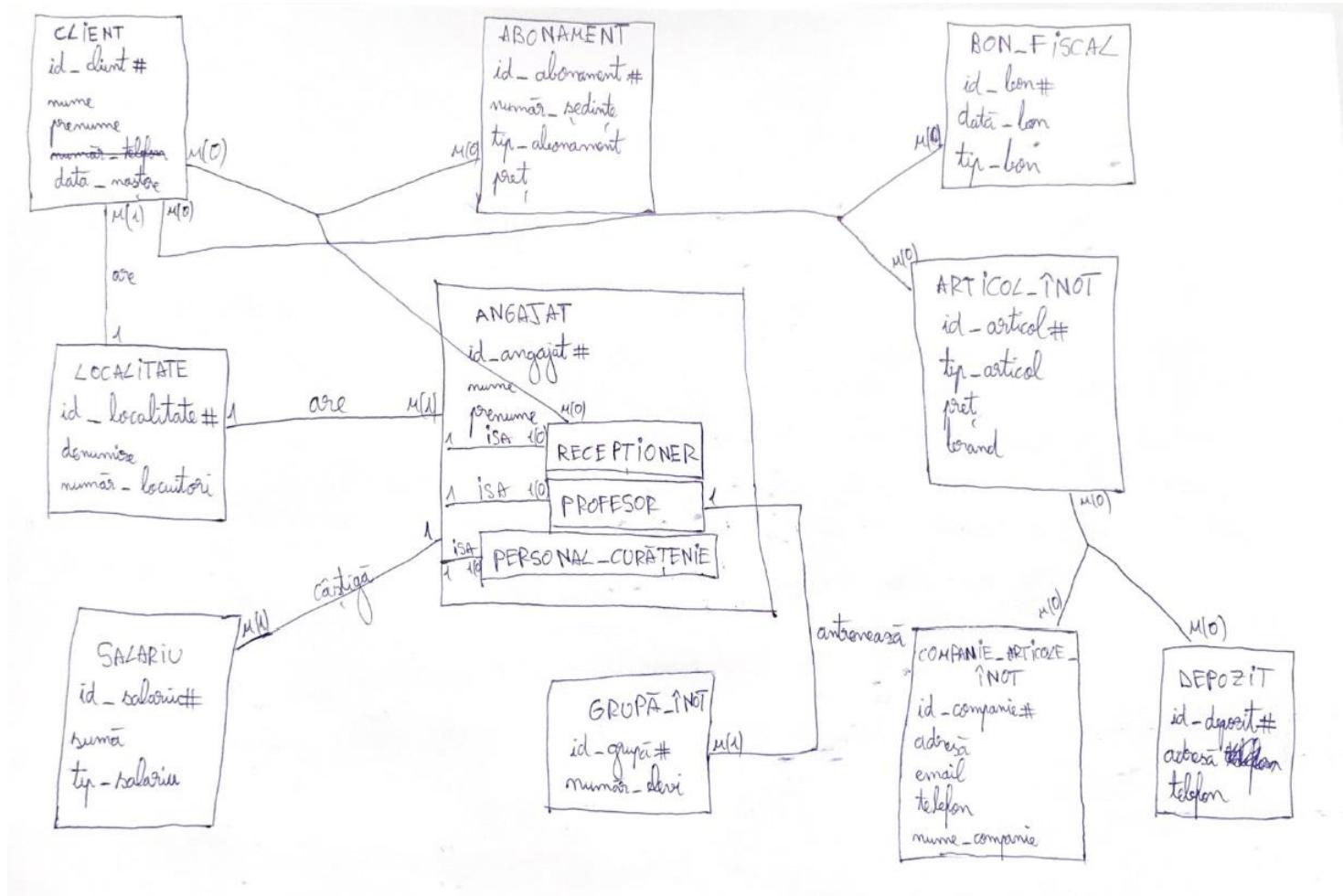
1 Descrierea modelului real, a regulilor de funcționare	2
2 Realizarea diagramei entitate-relație	3
3 Realizarea diagramei conceptuale	4
4 Crearea tabelelor și implementarea constrângerilor în SQL	7
5 Inserarea de date	12
6 Subprogram stocat cu două tipuri de colecții	20
7 Subprogram stocat cu un tip de cursor	24
8 Subprogram stocat de tip funcție – să utilizeze într-o singură comandă SQL 3 tabele + tratarea excepțiilor	27
9 Subprogram stocat de tip procedură – să utilizeze într-o singură comandă SQL 5 tabele + tratarea excepțiilor	33
10 Trigger de tip LMD la nivel de comandă	39
11 Trigger de tip LMD la nivel de linie	43
12 Trigger de tip LDD	48
13 Pachet ce conține toate obiectele definite în proiect	51
14 Pachet ce conține tipuri de date complexe și obiecte necesare unui flux de acțiuni integrate	64

1 Descrierea modelului real, a utilității acestuia și a regulilor de funcționare

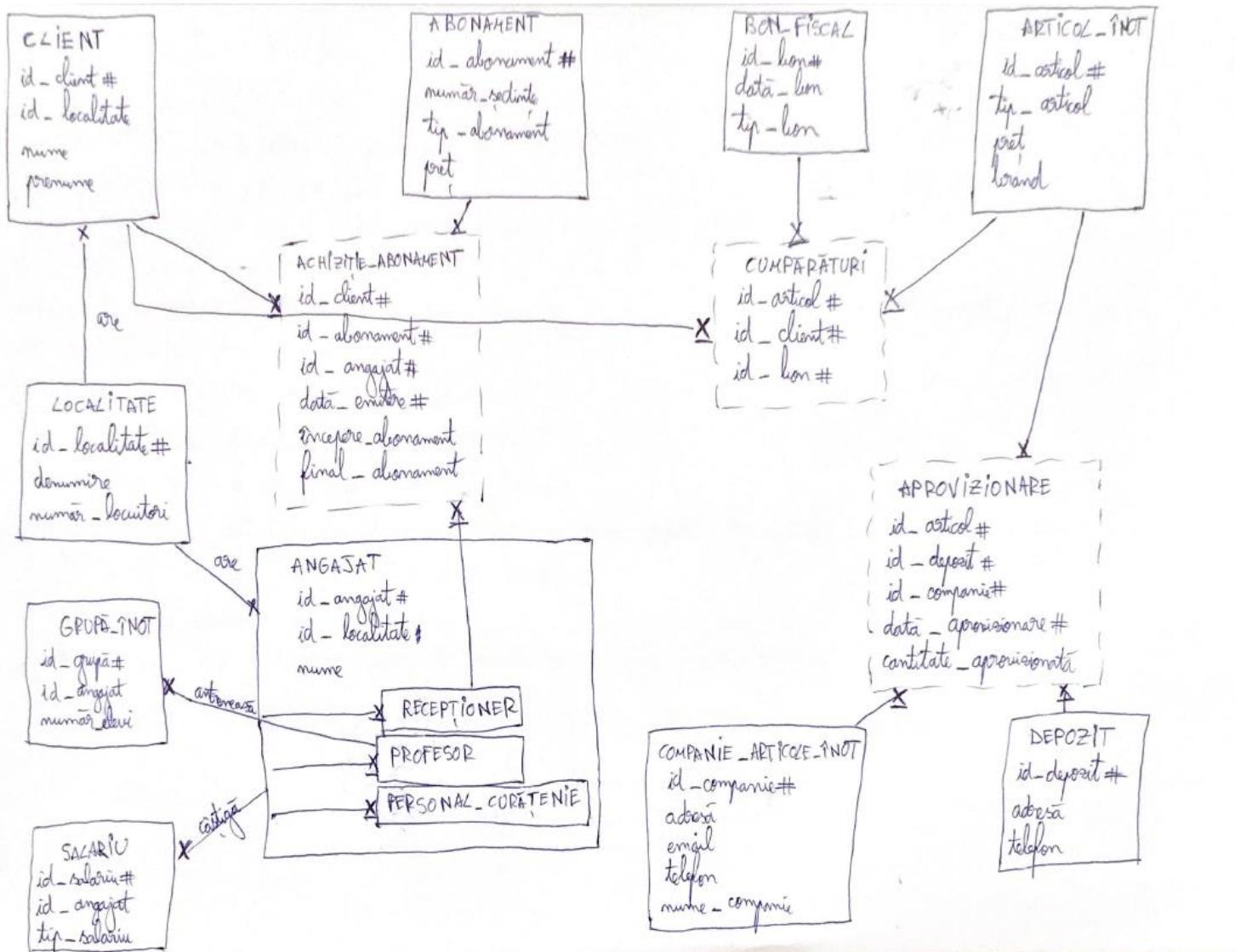
În acest proiect, am ales să descriu modelul unui bazin de înot care are în incinta sa și un magazine de articole pentru înot. Cei pasionați de acest sport pot să își achiziționeze un abonament la recepția bazinului, prețul variind în funcție de numărul de sedințe de înot. La parter se va afla un receptioner care se va ocupa cu eliberarea abonamentele clienților. Dacă o persoană este dornică să învețe să înoate, profesorii noștri specializați se vor ocupa de fiecare elev în parte care va ocupa un loc într-o grupă. De asemenea, orice client care duce lipsa de cască, ochelari, șlapi etc. este binevenit să își achiziționeze orice articol sportiv. Produsele noastre sunt aduse de la diferite depozite din țară și sunt de cea mai înaltă calitate, fiind aduse de la cele mai mari firme din lume.

Un client poate să își cumpere minim un abonament de orice fel dorește, fiind eliberat de un receptioner care lucrează în ziua respectivă. De asemenea, clientul poate să își achiziționeze mai multe abonamente în aceeași zi, dacă dorește, fiind eliberat de același receptioner. Fiecare angajat are atribuit câte un salarior lunar, iar angajații care au lucrat mai mult și au dat dovedă de mai mult devotement vor primi un bonus. Un profesor de înot poate fi atașat la mai multe grupe cu elevi, dar o grupă poate avea doar un singur profesor. Fiecare client poate să achiziționeze un produs din magazinul nostru. Pentru fiecare comandă, se va elibera un bon fiscal unic. Când se dă o comandă de la o anumită firmă pentru un anume articol sportiv, acesta va fi dus la unul din depozite. Se poate da comandă doar de produsele care se află în tabela ARTICOL_INOT. Tabele asociativă APROVIZIONARE va ține informațiile despre comanda respectivă: articolul comandat, respectiv cantitatea comandată, firma de unde s-a comandat și depozitul în care se va duce comanda. (la care se adaugă și data la care s-a dat comanda). În tabela CLIENT vor fi trecuti și clientii care au cumpărat de la magazin și nu și-au achiziționat un abonament și invers.

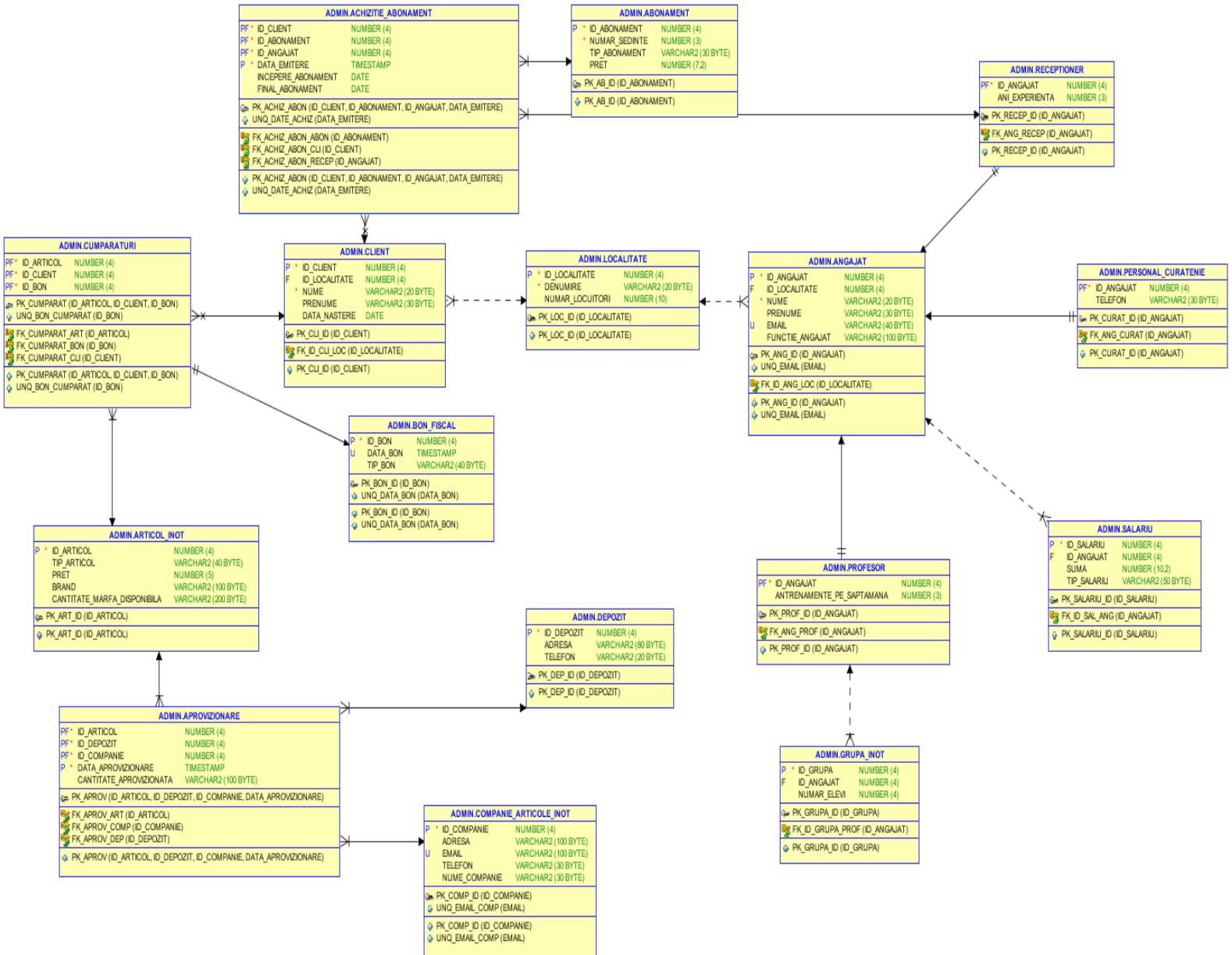
2 Realizarea diagramei entitate-relație



3 Realizarea diagramei conceptuale



*Diagrama conceptuală din SQL Developer



*Schema relatională

CLIENT(id-client#, id-localitate, nume, prenume, data-nastere)
LOCALITATE(id-localitate#, denumire, numar-locuitori)
ABONAMENT(id-abonament#, numar-sedinte, tip-abonament, pret)
BON-FISCAL(id-bon#, data-bon, tip-bon)
ANGAJAT(id-angajat#, id-localitate, nume, prenume, email, functie-angajat)
RECEPTIONER(id-angajat#, ani-experienta)
ACHIZITIE-ABONAMENT(id-client#, id-abonament#, id-angajat#, data-emiter#), incepe-abonament, final-abonament
PERSONAL-CURANTENIIE(id-angajat#, telefon)
PROFESOR(id-angajat#, antrenamente-pe-saptamana)
GRUPA-INTOT{id-grupa#, id-angajat, numar-dervi)
SALARIU(id-salariu#, id-angajat, suma, tip-salariu)
ARTICOL-INTOT{id-articol#, tip-articol, pret, brand, cantitate-marcfa-disponibila)
COMPARATORI{id-articol#, id-client#, id-bon#})
DEPOZIT{id-deposit#, adresa, telefon)
COMPANIE-ARTICOLE-INTOT{id-companie#, adresa, email, telefon, nume-companie)
APROVIZIONARE{id-articol#, id-deposit#, id-companie#, data-aprovizionare#, cantitate-aprovizionata)

4 Crearea tabelelor și implementarea constrângerilor în SQL

La crearea tabelelor am implementat constrângerile de cheie primară, de cheie externă, *NOT NULL*, *CHECK* și *UNIQUE*.

-- Generare tabele

```
create table LOCALITATE(
    id_localitate number(4),
    denumire varchar2(20) constraint null_denum_loc not null,
    numar_locuitori number(10),
    constraint pk_loc_id primary key(id_localitate)
);
```

```
create table CLIENT(
```

```
    id_client number(4) constraint pk_cli_id primary key,
    id_localitate number(4) constraint fk_id_cli_loc references
localitate(id_localitate),
    nume varchar2(20) constraint null_nume_cli not null,
    prenume varchar2(30),
    data_nastere date
);
```

```
create table ABONAMENT(
```

```
    id_abonament number(4) constraint pk_ab_id primary key,
    numar_sedinte number(3) constraint null_nr_sed_ab not null,
    tip_abonament varchar2(30),
    pret number (7, 2)
);
```

```
create table ACHIZITIE_ABONAMENT(
```

```
    id_client number(4),
    id_abonament number(4),
    id_angajat number(4),
```

```
data_emitere timestamp constraint unq_date_achiz unique,  
incepere_abonament date,  
final_abonament date,  
constraint pk_achiz_abon primary key(id_client, id_abonament,  
id_angajat, data_emitere),  
constraint fk_achiz_abon_cli foreign key(id_client) references  
client(id_client),  
constraint fk_achiz_abon_abon foreign key(id_abonament)  
references abonament(id_abonament),  
constraint fk_achiz_abon_recep foreign key(id_angajat) references  
receptioner(id_angajat)  
);
```

```
create table ANGAJAT (  
    id_angajat number(4) constraint pk_ang_id primary key,  
    id_localitate number(4) constraint fk_id_ang_loc references  
localitate(id_localitate),  
    nume varchar2(20) constraint null_numel_ang not null,  
    prenume varchar2(30),  
    email varchar2(40) constraint unq_email unique,  
    functie_angajat varchar2(100)  
);
```

```
create table RECEPTIONER (  
    id_angajat number(4) constraint pk_recep_id primary key,  
    ani_experienta number(3),  
    constraint fk_ang_recep foreign key(id_angajat) references  
angajat(id_angajat)  
);
```

```
create table PROFESOR (  
    id_angajat number(4) constraint pk_prof_id primary key,  
    antrenamente_pe_saptamana number(3),
```

```
constraint fk_ang_prof foreign key(id_angajat) references
angajat(id_angajat)
);
```

```
create table PERSONAL_CURATENIE (
    id_angajat number(4) constraint pk_curat_id primary key,
    telefon varchar2(30),
    constraint fk_ang_curat foreign key(id_angajat) references
    angajat(id_angajat)
);
```

```
create table SALARIU (
    id_salariu number(4) constraint pk_salariu_id primary key,
    id_angajat number(4) constraint fk_id_sal_ang references
    angajat(id_angajat),
    suma number(10,2),
    tip_salariu varchar2(50)
);
```

```
create table GRUPA_INOT (
    id_grupa number(4) constraint pk_grupa_id primary key,
    id_angajat number(4) constraint fk_id_grupa_prof references
    profesor(id_angajat),
    numar_elevi number(4)
);
```

```
create table BON_FISCAL (
    id_bon number(4) constraint pk_bon_id primary key,
    data_bon timestamp constraint unq_data_bon unique,
    tip_bon varchar2(40)
);
```

```
create table ARTICOL_INOT (
```

```

id_articol number(4) constraint pk_art_id primary key,
tip_articol varchar2(40),
pret number(5) constraint ck_pret_art check(pret > 0),
brand varchar2(100),
cantitate_marfa_disponibila varchar2(200)
);

create table DEPOZIT (
    id_depozit number(4) constraint pk_dep_id primary key,
    adresa varchar2(80),
    telefon varchar2(20)
);

create table CUMPARATURI (
    id_articol number(4),
    id_client number(4),
    id_bon number(4) constraint unq_bon_cumparat unique,
    constraint pk_cumparat primary key(id_articol, id_client, id_bon),
    constraint fk_cumparat_art foreign key(id_articol) references
    articol_inot(id_articol),
    constraint fk_cumparat_cli foreign key(id_client) references
    client(id_client),
    constraint fk_cumparat_bon foreign key(id_bon) references
    bon_fiscal(id_bon)
);

create table COMPANIE_ARTICOLE_INOT (
    id_companie number(4) constraint pk_comp_id primary key,
    adresa varchar2(100),
    email varchar2(100) constraint unq_email_comp unique,
    telefon varchar2(30),
    nume_companie varchar2(30)
);

```

```
create table APROVIZIONARE(
    id_articol number(4),
    id_depozit number(4),
    id_companie number(4),
    data_aprovizionare timestamp,
    tip_articol_aprovizionat varchar2(100),
    cantitate_aprovizionata varchar2(100),
    constraint pk_aprov primary key(id_articol, id_depozit,
    id_companie, data_aprovizionare),
    constraint fk_aprov_art foreign key(id_articol) references
    articol_inot(id_articol),
    constraint fk_aprov_dep foreign key(id_depozit) references
    depozit(id_depozit),
    constraint fk_aprov_comp foreign key(id_companie) references
    companie_articole_inot(id_companie)
);
```

5 Inserarea de date

```
-- Inserare date in tabele
-- Inserare in tabela LOCALITATE
insert into localitate values(100, 'Suceava', 743645);
insert into localitate values(120, 'Bucuresti', 2496895);
insert into localitate values(130, 'Iasi', 919049);
insert into localitate values(150, 'Brasov', 630807);
insert into localitate values(160, 'Arad', 473946);

-- Inserare in tabela CLIENT
insert into client values(20, 100, 'Popescu', 'Marcel', to_date('12-02-1995', 'dd-mm-yyyy'));
insert into client values(30, 130, 'Croitoru', 'Daniel', to_date('23-10-1989', 'dd-mm-yyyy'));
insert into client values(40, 160, 'Georgescu', 'Paul', to_date('04-05-2000', 'dd-mm-yyyy'));
insert into client values(60, 120, 'Munteanu', 'Iulian', to_date('15-08-2004', 'dd-mm-yyyy'));
insert into client values(70, 100, 'Gaitan', 'Maria', to_date('25-04-1992', 'dd-mm-yyyy'));

-- Inserare in tabela ABONAMENT
insert into abonament values(1, 10, 'Cursuri inot', 100);
insert into abonament values(2, 15, 'Cursuri inot', 135);
insert into abonament values(3, 10, 'Agrement', 75);
insert into abonament values(4, 20, 'Agrement', 140);
insert into abonament values(5, 20, 'Cursuri inot', 180);

-- Inserare in tabela ANGAJAT
insert into angajat values(50, 150, 'Moscaliuc', 'Cezar',
'cezar.moscaliuc@gmail.com', 'PROFESOR');
insert into angajat values(60, 120, 'Oprea', 'Iustin',
```

```
'oprea.iustin@gmail.com', 'RECEPTIONER');  
insert into angajat values(70, 130, 'Maxim', 'Vlad',  
'vlad_maxim@gmail.com', 'PROFESOR');  
insert into angajat values(80, 160, 'Andries', 'Tudor',  
'andries.tudor@yahoo.com"RECEPTIONER');  
insert into angajat values(90, 100, 'Carpineanu', 'Vlad',  
'carpineanu.vlad@gmail.com"RECEPTIONER');  
insert into angajat values(100, 120, 'Airinei', 'Mirela',  
'mirela.airinei@yahoo.com', 'PROFESOR');  
insert into angajat values(110, 130, 'Bucaciuc', 'Andreea',  
'andreea.bucaciuc@gmail.com', 'PERSONAL CURATENIE');  
insert into angajat values(120, 100, 'Cernaut', 'Angela',  
'angela.cernaut@yahoo.com', 'PERSONAL CURATENIE');  
insert into angajat values(130, 120, 'Costache', 'Viorel',  
'viorel.costache@yahoo.com', 'PAZNIC');  
insert into angajat values(140, 130, 'Nistor', 'Alexandru',  
'alexandru.nistor@gmail.com', 'ADMINISTRATOR BAZIN');  
insert into angajat values(150, 160, 'Axinte', 'Mirel',  
'axinte.mirel@gmail.com', 'RECEPTIONER');  
insert into angajat values(160, 130, 'Leonte', 'Vlad',  
'leonte.vlad@gmail.com', 'RECEPTIONER');
```

-- Inserare in tabela PROFESOR

```
insert into profesor values(50, 35);  
insert into profesor values(70, 25);  
insert into profesor values(100, 18);
```

-- Inserare in tabela RECEPTIONER

```
insert into receptioner values(60, null);  
insert into receptioner values(90, 2);  
insert into receptioner values(80, 1);  
insert into receptioner values(150, 2);  
insert into receptioner values(160, 2);
```

```
-- Inserare in tabela PERSONAL_CURATENIE
insert into personal_curatenie values(110, '0724857239');
insert into personal_curatenie values(120, '0745825967');

-- Inserare in tabela BON_FISCAL
insert into bon_fiscal values(200, '18-06-2020 13:26:43', 'CASH');
insert into bon_fiscal values(300, '20-06-2020 09:51:20', 'CARD');
insert into bon_fiscal values(400, '28-06-2020 10:10:20', 'CASH');
insert into bon_fiscal values(500, '04-07-2020 19:04:06', 'CASH');
insert into bon_fiscal values(600, '28-07-2020 08:30:34', 'CARD');
insert into bon_fiscal values(700, '29-07-2020 14:36:35', 'CARD');
insert into bon_fiscal values(800, '13-08-2020 12:30:23', 'CASH');
insert into bon_fiscal values(900, '10-07-2020 17:23:12', 'CASH');
insert into bon_fiscal values(1000, '08-07-2020 11:05:09', 'CARD');
insert into bon_fiscal values(1100, '17-07-2020 19:10:56', 'CARD');

-- Inserare in tabela GRUPA_INOT
insert into grupa_inot values(132, 70, 15);
insert into grupa_inot values(133, 50, 20);
insert into grupa_inot values(134, 100, 10);
insert into grupa_inot values(141, 50, 25);
insert into grupa_inot values(142, 70, 20);

-- Inserare in tabela SALARIU
insert into salariu values(10, 50, 4000, 'LUNAR');
insert into salariu values(20, 60, 5000, 'LUNAR');
insert into salariu values(30, 110, 2000, 'LUNAR');
insert into salariu values(40, 100, 4400, 'LUNAR');
insert into salariu values(50, 80, 6500, 'LUNAR');
insert into salariu values(60, 90, 7000, 'LUNAR');
insert into salariu values(70, 120, 2200, 'LUNAR');
insert into salariu values(80, 80, 200, 'BONUS');
insert into salariu values(90, 70, 5200, 'LUNAR');
```

```
insert into salariu values(100, 130, 1600, 'LUNAR');
insert into salariu values(110, 130, 400, 'BONUS');
insert into salariu values(120, 140, 12000, 'LUNAR');
insert into salariu values(130, 150, 7000, 'LUNAR');
insert into salariu values(140, 160, 4200, 'LUNAR');

-- Inserare in tabela ACHIZITIE_ABONAMENT
insert into achizitie_abonament values(20, 1, 60, '19-06-2020
12:26:43', '19-06-2020', '10-07-2020');
insert into achizitie_abonament values(30, 3, 60, '20-06-2020
16:23:52', '20-06-2020', '05-07-2020');
insert into achizitie_abonament values(40, 4, 80, '19-06-2020
12:26:01', '19-06-2020', '20-07-2020');
insert into achizitie_abonament values(60, 2, 90, '02-07-2020
15:43:43', '02-07-2020', '29-07-2020');
insert into achizitie_abonament values(70, 5, 80, '20-06-2020
12:21:41', '20-06-2020', '15-07-2020');
insert into achizitie_abonament values(40, 3, 60, '01-08-2020
18:21:12', '01-08-2020', '24-08-2020');
insert into achizitie_abonament values(30, 3, 60, '07-07-2020
10:30:31', '07-07-2020', '27-07-2020');
insert into achizitie_abonament values(70, 3, 90, '19-07-2020
16:26:43', '19-07-2020', '03-08-2020');
insert into achizitie_abonament values(40, 3, 80, '26-08-2020
18:43:59', '27-08-2020', '20-09-2020');
insert into achizitie_abonament values(60, 1, 60, '01-08-2020
12:29:40', '01-08-2020', '31-08-2020');

-- Inserare in tabela ARTICOL_INOT
insert into articol_inot values(1, 'Slipi', 120, 'Arena', '70 perechi');
insert into articol_inot values(2, 'Slapi', 90, 'Speedo', '30 perechi');
insert into articol_inot values(3, 'Ochelari', 160, 'TYR', '110
perechi');
```

```
insert into articol_inot values(4, 'Slipi', 145, 'Speedo', '40 perechi');
insert into articol_inot values(5, 'Casca', 40, 'Jaked', '25 perechi');
insert into articol_inot values(6, 'Ochelari', 200, 'Arena', '90
perechi');
insert into articol_inot values(7, 'Ochelari', 170, 'Nautica', null);
insert into articol_inot values(8, 'Slapi', 250, 'Arena', '20 perechi');
```

-- Inserare in tabela CUMPARATURI

```
insert into cumparaturi values (1, 20, 400);
insert into cumparaturi values (4, 60, 300);
insert into cumparaturi values (3, 40, 200);
insert into cumparaturi values (6, 70, 500);
insert into cumparaturi values (2, 30, 600);
insert into cumparaturi values (5, 20, 700);
insert into cumparaturi values (1, 60, 800);
insert into cumparaturi values (3, 30, 900);
insert into cumparaturi values (4, 70, 1000);
insert into cumparaturi values (2, 40, 1100);
```

-- Inserare in tabela DEPOZIT

```
insert into depozit values(10, 'Strada Alexandru cel Bun, nr. 4,
Suceava', '0724685357');
insert into depozit values(20, 'Strada Lalelelor, nr. 21, Iasi',
'0756981234');
insert into depozit values(30, 'Strada Grigore Ghica, nr. 9,
Bucuresti', '0783457631');
insert into depozit values(40, 'Strada Mihai Viteazu, nr. 13,
Botosani', '0756914574');
insert into depozit values(50, 'Strada Petru Musat, nr. 10, Vaslui',
'0742313568');
```

-- Inserare in tabela COMPANIE_ARTICOLE_INOT

```
insert into companie_articole_inot values(100, 'Strada Dimitrie
```

Cantemir, nr. 40, Oradea', 'arena@arenaromania.ro', '0040 259416324', 'Arena');

insert into companie_articole_inot values(200, 'Strada Spiru Haret, nr. 31, Bucuresti', 'speedo@speedoromania.ro', '0040 256435675', 'Speedo');

insert into companie_articole_inot values(300, 'Strada Stefan cel Mare, nr. 20, Cluj-Napoca', 'tyr@tyrromania.ro', '0040 278445948', 'TYR');

insert into companie_articole_inot values(400, 'Strada Emil Racovita, nr. 17, Bistrita', 'jaked@jakedromania.ro', '0040 289424535', 'Jaked');

insert into companie_articole_inot values(500, 'Strada Constantin Grigorescu, nr. 29, Suceava', 'nautica@nauticaromania.ro', '0040 268463832', 'Nautica');

```
select id_articol, id_depozit, id_companie, articol_inot.tip_articol,
companie_articole_inot.nume_companie
from articol_inot, depozit, companie_articole_inot;
```

-- Inserare in tabela APROVIZIONARE

insert into aprovisionare values(1, 20, 100, '12.05.2020 09:20:43', '40 perechi');

insert into aprovisionare values(5, 40, 400, '20.05.2020 11:31:20', '12 perechi');

insert into aprovisionare values(2, 50, 200, '02.06.2020 07:37:23', '30 perechi');

insert into aprovisionare values(3, 30, 300, '10.06.2020 09:37:23', '50 perechi');

insert into aprovisionare values(4, 10, 200, '28.05.2020 12:21:45', '35 perechi');

insert into aprovisionare values(1, 30, 100, '04.07.2020 08:40:57', '20 perechi');

insert into aprovisionare values(6, 20, 100, '10.06.2020 12:29:21',

```

'15 perechi');
insert into aprovisionare values(5, 10, 400, '17.06.2020 09:12:12',
'20 perechi');
insert into aprovisionare values(2, 30, 200, '06.08.2020 18:07:43',
'26 perechi');
insert into aprovisionare values(3, 50, 300, '24.07.2020 11:14:56',
'16 perechi');

commit;

```

select * from client;

	ID_CLIENT	ID_LOCALITATE	NUME	PRENUME	DATA_NASTERE
1	70	100	Gaitan	Maria	25-04-1992
2	80	120	Anghel	Marius	23-05-1985
3	90	150	Ionita	Serban	19-02-1987
4	100	160	Rusu	Andrei	29-06-1999
5	110	160	Ichim	Radu	12-10-1996
6	120	130	Manea	Robert	20-12-2000
7	140	100	Cirlan	George	10-03-1997
8	20	100	Popescu	Marcel	12-02-1995
9	30	130	Croitoru	Daniel	23-10-1989
10	40	160	Georgescu	Paul	04-05-2000
11	60	120	Munteanu	Iulian	15-08-2004

select * from client;

select * from achizitie_abonament;

select * from angajat;

	ID_CLIENT	ID_ABONAMENT	ID_ANGAJAT	DATA_EMITERE	INCEPERE_ABONAMENT	FINAL_ABONAMENT
1	100	3	8002-09-2020	15:29:40	02-09-2020 05-10-2020	
2	100	5	9010-11-2020	19:32:10	10-11-2020 12-12-2020	
3	120	5	6012-08-2020	12:23:12	12-08-2020 12-09-2020	
4	120	3	9020-09-2020	20:20:02	20-09-2020 22-10-2020	
5	70	5	8020-06-2020	12:21:41	20-06-2020 15-07-2020	
6	70	3	9019-07-2020	16:26:43	19-07-2020 03-08-2020	
7	20	1	6019-06-2020	12:26:43	19-06-2020 10-07-2020	
8	30	3	6020-06-2020	16:23:52	20-06-2020 05-07-2020	
9	40	4	8019-06-2020	12:26:01	19-06-2020 20-07-2020	
10	60	2	9002-07-2020	15:43:43	02-07-2020 29-07-2020	
11	40	3	6001-08-2020	18:21:12	01-08-2020 24-08-2020	
12	30	3	6007-07-2020	10:30:31	07-07-2020 27-07-2020	
13	40	3	8026-08-2020	18:43:59	27-08-2020 20-09-2020	
14	60	1	6001-08-2020	12:29:40	01-08-2020 31-08-2020	

```

select * from client;
select * from achizitie_abonament;
select * from angajat;

commit;

```

Script Output | Query Result | SQL | All Rows Fetched: 10 in 0.006 seconds

ID_ANGAJAT	ID_LOCALITATE	NUME	PRENUME	EMAIL	FUNCTIE_ANGAJAT
1	50	150Moscaliuc	Cezar	cezar.moscaliuc@gmail.com	PROFESOR
2	60	120Oprea	Iustin	oprea.iustin@gmail.com	RECEPTIONER
3	70	130Maxim	Vlad	vlad maxim@gmail.com	PROFESOR
4	80	160Andries	Tudor	andries.tudor@yahoo.com	RECEPTIONER
5	90	100Carpineanu	Vlad	carpineanu.vlad@gmail.com	RECEPTIONER
6	100	120Airinei	Mirela	mirela.airinei@yahoo.com	PROFESOR
7	110	130Bucaciuc	Andreea	andreea.bucaciuc@gmail.com	PERSONAL CURATENIE
8	120	100Cernaut	Angela	angela.cernaut@yahoo.com	PERSONAL CURATENIE
9	130	120Costache	Viorel	viorel.costache@yahoo.com	PAZNIC
10	140	130Nistor	Alexandru	nistor@gmail.com	ADMINISTRATOR BAZIN

```

select * from client;
select * from achizitie_abonament;
select * from angajat;
select * from localitate;

```

Script Output | Query Result | SQL | All Rows Fetched: 7 in 0.002 seconds

ID_LOCALITATE	DENUMIRE	NUMAR_LOCUITORI
1	100Suceava	743645
2	120Bucuresti	2496895
3	130Iasi	919049
4	150Brasov	630807
5	160Arad	473946
6	170Pitesti	778294
7	180Vaslui	324564

6 Subprogram stocat cu două tipuri de colecții

Cerinta:

Din cauza economiei scăzute produsa de pandemia COVID, seful bazinei de inot dorește să micsoreze salariile acestor receptioneri care au cel mai mare salar din între toți receptionerii angajați, scaderea facându-se cu 10%. Se vor afisa toate datele despre acești angajați.(Va fi luat în considerare doar salarul lunar, fără bonus). De asemenea, afisati si numarul acestora.

Cod SQL:

```
CREATE OR REPLACE PROCEDURE ex_6 IS
    TYPE t_imbricat IS TABLE OF salariu.suma%TYPE;
    TYPE t_imb_angaj IS TABLE OF angajat.id_angajat%TYPE;
    t_salarii t_imbricat;
    t_receptioneri t_imb_angaj;
    TYPE detalii_record IS RECORD (v_id_angajat
        angajat.id_angajat%TYPE,
        v_nume_localitate
        localitate.denumire%TYPE,
        v_nume_angajat.nume%TYPE,
        v_prenume_angajat.nume%TYPE,
        v_email_angajat.email%TYPE,
        v_sal_nou salariu.suma%TYPE);
    TYPE t_date IS TABLE OF detalii_record INDEX BY
    PLS_INTEGER;
    tablou_date_angajat t_date;
    maxx_sal salariu.suma%TYPE := 0;
    maxx_old_sal salariu.suma%TYPE := 0;
BEGIN
    SELECT sal.suma
    BULK COLLECT INTO t_salarii
```

```
FROM receptioner r, salariu sal
WHERE LOWER(sal.tip_salariu) = 'lunar' AND r.id_angajat =
sal.id_angajat;
```

```
FOR i IN t_salarii.first..t_salarii.last LOOP
  IF t_salarii(i) > maxx_sal THEN
    maxx_sal := t_salarii(i);
  END IF;
END LOOP;
```

```
SELECT r.id_angajat
BULK COLLECT INTO t_receptioneri
FROM receptioner r, salariu sal
WHERE LOWER(sal.tip_salariu) = 'lunar' AND r.id_angajat =
sal.id_angajat AND sal.suma = maxx_sal;
```

```
FORALL i in t_receptioneri.first..t_receptioneri.last
  UPDATE salariu
  SET salariu.suma = salariu.suma - 0.1 * salariu.suma
  WHERE salariu.id_angajat = t_receptioneri(i);
```

```
maxx_old_sal := maxx_sal;
```

```
SELECT MAX(sal.suma)
INTO maxx_sal
FROM salariu sal, angajat a
WHERE LOWER(a.functie_angajat) = 'receptioner' AND
sal.id_angajat = a.id_angajat;
```

```
SELECT a.id_angajat, l.denumire, a.nume, a.prenume, a.email,
sal.suma
BULK COLLECT INTO tablou_date_angajat
```

```

    FROM angajat a, salariu sal, localitate l
    WHERE a.id_angajat = sal.id_angajat AND
    LOWER(sal.tip_salariu) = 'lunar' AND
        LOWER(a.functie_angajat) = 'receptioner' AND a.id_localitate
        = l.id_localitate
        AND sal.suma = maxx_sal;

    FOR i IN tablou_date_angajat.first..tablou_date_angajat.last
    LOOP
        dbms_output.put_line('Receptionerul cu id-ul ' ||
        tablou_date_angajat(i).v_id_angajat || ' cu numele complet ' ||
        tablou_date_angajat(i).v_nume || ' ' ||
        tablou_date_angajat(i).v_prenume || ' din localitatea ' ||
        tablou_date_angajat(i).v_numelocalitate || ' cu email-ul ' ||
        tablou_date_angajat(i).v_email
        || chr(10) || 'a avut salariul de ' || maxx_old_sal || ', acum
        avand ' || tablou_date_angajat(i).v_sal_nou
        || '.' || chr(10));
    END LOOP;
    dbms_output.put_line('Numarul de angajati al caror salariu s-a
    modificat este: ' || tablou_date_angajat.count);

END ex_6;
/

```

Screenshot rulare procedura:

BEFORE:

```

351
352     END ex_6;
353
354
355 BEGIN
356     ex_6;
357 END;
358 /
359
360 ROLLBACK;
361
362 -- Exercitiul 7

```

PL/SQL procedure successfully completed.

Rollback complete.

ID_SALARIU	ID_ANGAZAT	SUMA	TIP_SALARIU
1	10	50	4000 LUNAR
2	20	60	5000 LUNAR
3	30	110	2000 LUNAR
4	40	100	4400 LUNAR
5	50	80	5850 LUNAR
6	60	70	7000 LUNAR
7	70	120	2200 LUNAR
8	80	80	180 BONUS
9	90	50	5200 LUNAR
10	100	130	1600 LUNAR
11	110	130	400 BONUS
12	120	140	12000 LUNAR
13	130	150	7000 LUNAR
14	140	160	4200 LUNAR

AFTER:

```

351
352     END ex_6;
353
354
355 BEGIN
356     ex_6;
357 END;
358 /
359
360 ROLLBACK;
361
362 -- Exercitiul 7

```

Rollback complete.

Receptionerul cu id-ul 90 cu numele complet Carpineanu Vlad din localitatea Suceava a avut salariul de 7000, acum avand 6300.

Receptionerul cu id-ul 150 cu numele complet Axinte Mirel din localitatea Arad cu a avut salariul de 7000, acum avand 6300.

Numarul de angajati al caror salariu s-a modificarat este: 2

ID_SALARIU	ID_ANGAZAT	SUMA	TIP_SALARIU
1	10	50	4000 LUNAR
2	20	60	5000 LUNAR
3	30	110	2000 LUNAR
4	40	100	4400 LUNAR
5	50	80	5850 LUNAR
6	60	90	6300 LUNAR
7	70	120	2200 LUNAR
8	80	60	800 BONUS
9	90	70	5200 LUNAR
10	100	130	1600 LUNAR
11	110	130	400 BONUS
12	120	140	12000 LUNAR
13	130	150	6300 LUNAR
14	140	160	4200 LUNAR

7 Subprogram stocat cu un tip de cursor

Cerinta:

Afisati pentru un receptioner, al carui id este dat ca parametru al unei proceduri, toate detaliile despre clientii carora le-a emis un abonament, cat si cateva detalii despre el(receptioner). Daca nu a emis niciun abonament niciunei persoane(deoarece este nou la munca) se va afisa un mesaj corespunzator.

Daca acelasi receptioner a emis un abonament de mai multe ori unui client la interval de timp diferit, se va afisa doar o data acel client, NU de cate ori i-a emis.

Cod SQL:

```
CREATE OR REPLACE PROCEDURE ex_7(v_id
receptioner.id_angajat%TYPE) IS
CURSOR c IS
    SELECT a.id_angajat, a.nume, a.prenume, l.denumire,
    CURSOR(SELECT DISTINCT c.id_client, c.nume,
c.prenume, c.data_nastere--cursor imbricat
        FROM client c, achizitie_abonament ach
        where ach.id_client = c.id_client AND a.id_angajat =
ach.id_angajat)
    FROM angajat a, localitate l
    WHERE LOWER(a.functie_angajat) = 'receptioner' AND
a.id_angajat = v_id AND a.id_localitate = l.id_localitate;
TYPE c_dinamic IS REF CURSOR;
info_clienti c_dinamic;
v_id_ang angajat.id_angajat%TYPE;
v_nume_ang angajat.nume%TYPE;
v_prenume_ang angajat.prenume%TYPE;
v_localitate_ang localitate.denumire%TYPE;
```

```

v_id_cli client.id_client%TYPE;
v_nume_cli client.nume%TYPE;
v_prenume_cli client.prenume%TYPE;
v_data_cli client.data_nastere%TYPE;
clienti_prezenti BOOLEAN := FALSE;
BEGIN
    OPEN c;
    LOOP
        FETCH c INTO v_id_ang, v_nume_ang, v_prenume_ang,
v_localitate_ang, info_clienti;
        EXIT WHEN c%NOTFOUND;

        dbms_output.put_line('Receptionerul ' || v_nume_ang || ' ' ||
v_prenume_ang || ', cu id-ul ' || v_id_ang
        || ', din orasul ' || v_localitate_ang || ' a emis abonamente
urmatorilor clienti: ');

        LOOP
            FETCH info_clienti into v_id_cli, v_nume_cli,
v_prenume_cli, v_data_cli;
            EXIT WHEN info_clienti%NOTFOUND;

            clienti_prezenti := TRUE;
            dbms_output.put_line('Clientul ' || v_nume_cli || ' ' ||
v_prenume_cli || ', cu id-ul ' || v_id_cli
            || ', nascut la data de ' || v_data_cli);
        END LOOP;
        IF clienti_prezenti = FALSE THEN
            dbms_output.put_line('Nu exista clienti ajutati de acest
receptioner.');
        END IF;
--    dbms_output.new_line();
    END LOOP;

```

```
END ex_7;
```

```
/
```

```
BEGIN  
    ex_7(150);  
END;  
/
```

Screenshot-uri rulare procedura:

The screenshot shows the Oracle SQL Developer interface. The central area displays the PL/SQL code for procedure ex_7. The code includes a loop that calls ex_7(60) 150 times. The output window shows the successful completion of the procedure and lists five client details: Oprea Iustin, Georgescu Paul, Munteanu Julian, Popescu Marcel, Manea Robert, and Croitoru Daniel.

```
408     dbms_output.new_line();  
409     END LOOP;  
410  END ex_7;  
411 /  
412  
413 BEGIN  
414     ex_7(60);  
415 END;  
416 /  
417  
418
```

PL/SQL procedure successfully completed.
Receptionerul Oprea Iustin, cu id-ul 60, din orasul Bucuresti a emis abonamente urmatorilor clienti:
Clientul Georgescu Paul, cu id-ul 40, nascut la data de 04-05-2000
Clientul Munteanu Julian, cu id-ul 60, nascut la data de 15-08-2004
Clientul Popescu Marcel, cu id-ul 20, nascut la data de 12-02-1995
Clientul Manea Robert, cu id-ul 120, nascut la data de 20-12-2000
Clientul Croitoru Daniel, cu id-ul 30, nascut la data de 23-10-1989

```

408      dbms_output.new_line();
409  END LOOP;
410 END ex_7;
411 /
412
413 BEGIN
414   ex_7(150);
415 END;
416 /
417
418

```

PL/SQL procedure successfully completed.

Receptionerul Axinte Mirel, cu id-ul 150, din orasul Arad a emis abonamente urmatorilor clienti:
Nu exista clienti ajutati de acest receptioner.

PL/SQL procedure successfully completed.

8 Subprogram stocat de tip funcție – să utilizeze într-o singură comandă SQL 3 tabele + tratarea excepțiilor

Cerinta:

Pentru un anumit articol sportiv de la un anumit brand, date amandoua ca parametrii ai functiei, sa se afiseze lista bonurilor fiscale care au fost emise, data emiterii si metoda de achitare, si returneaza raportul dintre numarul de bucati cumparate inmultite cu pretul articoului si numarul total de perechi disponibile fara a scadea din numarul acestora perechile cumparate. Se vor trata toate exceptiile care pot aparea.

Cod SQL:

```
CREATE OR REPLACE FUNCTION ex_8(v_brand
articol_inot.brand%TYPE, v_tip articol_inot.tip_articol%TYPE)
RETURN VARCHAR2 IS
    TYPE detalii_record IS RECORD (v_id_bon
bon_fiscal.id_bon%TYPE,
                                v_data bon_fiscal.data_bon%TYPE,
                                v_metoda bon_fiscal.tip_bon%TYPE);
    TYPE t_date IS TABLE OF detalii_record;
    info_tranzactii t_date;
    TYPE art_rec IS RECORD (v_id_art
articol_inot.id_articol%TYPE,
                        v_pret articol_inot.pret%TYPE,
                        v_cant
articol_inot.cantitate_marfa_disponibila%TYPE);
    articole_sportive art_rec;
    numar_art_disp NUMBER(5);
    medie NUMBER(8, 3) := 0;
    exceptie exception;
BEGIN
    SELECT id_articol, pret, cantitate_marfa_disponibila
    INTO articole_sportive
    FROM articol_inot
    WHERE brand = v_brand AND tip_articol = v_tip;

    SELECT b.*
    BULK COLLECT INTO info_tranzactii
    FROM articol_inot a, cumparaturi c, bon_fiscal b
    WHERE a.brand = v_brand AND a.tip_articol = v_tip AND
    c.id_articol = a.id_articol AND c.id_bon = b.id_bon;

    IF articole_sportive.v_cant IS null THEN
```

```

        medie := (info_tranzactii.count * articole_sportive.v_pret) /
NVL(articole_sportive.v_cant, 0);
        ELSE
            numar_art_disp :=
TO_NUMBER(SUBSTR(articole_sportive.v_cant, 1, 2));
            medie := (info_tranzactii.count * articole_sportive.v_pret) /
numar_art_disp;
        END IF;

        dbms_output.put_line('Pentru articolul ' || v_tip || ' de la brand-ul '
|| v_brand ||
        ' s-au emis urmatoarele bonuri: ' || chr(10));
        IF info_tranzactii.count = 0 THEN
            RAISE exceptie;
        ELSE
            FOR i IN 1..info_tranzactii.count LOOP
                dbms_output.put_line('Bonul cu id-ul ' ||
info_tranzactii(i).v_id_bon || ' a fost emis la data si ora '
                || info_tranzactii(i).v_data || ' si plata s-a achitat ' ||
info_tranzactii(i).v_metoda);
            END LOOP;
        END IF;

        RETURN ('Raportul este: ' || medie);
    
```

EXCEPTION

```

        WHEN exceptie THEN
            RETURN 'Acest produs nu a fost inca achizitionat de nimeni,
deci media va fi 0!';
        WHEN no_data_found THEN
            RETURN 'Nu a fost cumparat niciodata acest produs de la
brandul acesta!';
        WHEN zero_divide THEN
    
```

RETURN 'Imi pare rau, dar nu avem produsul momentan in stoc!';

```
WHEN others THEN
    RETURN 'Au intervenit alte erori!';
END ex_8;
/
BEGIN
    dbms_output.put_line(ex_8('Speedo', 'Slipi'));
--    dbms_output.put_line(ex_8('Nabaiji', 'Ochelari'));
--    dbms_output.put_line(ex_8('Arena', 'Slapi'));
--    dbms_output.put_line(ex_8('Nautica', 'Ochelari'));
END;
/
```

Screenshot-uri fiecare caz in parte:

1. Cazul OK, fara eroare:

The screenshot shows the Oracle SQL Developer interface. In the top navigation bar, the path is C:\Users\calin\Desktop\iteme facultate VAN \SGBD\Project final\project.sql. The main workspace shows a PL/SQL script being run. The code is as follows:

```
479 END ex_8;
480 /
481 BEGIN
482     dbms_output.put_line(ex_8('Speedo', 'Slipi'));
483 --    dbms_output.put_line(ex_8('Nabaiji', 'Ochelari'));
484 --    dbms_output.put_line(ex_8('Arena', 'Slapi'));
485 --    dbms_output.put_line(ex_8('Nautica', 'Ochelari'));
486 END;
487 /
488
489
```

The output pane at the bottom displays the message "PL/SQL procedure successfully completed." Below it, there is some additional text about a customer named Slipi.

2. Eroarea no_data_found:

```

475 END ex_8;
480 /
481 BEGIN
482 --      dbms_output.put_line(ex_8('Speedo', 'Slipi'));
483 --      dbms_output.put_line(ex_8('Nabajji', 'Ochelari'));
484 --      dbms_output.put_line(ex_8('Arena', 'Slapi'));
485 --      dbms_output.put_line(ex_8('Nautica', 'Ochelari'));
486 END;
487 /
488
489

```

PL/SQL procedure successfully completed.

Nu a fost cumparat niciodata acest produs de la brandul acesta!

PL/SQL procedure successfully completed.

3. O exceptie proprie, cand lista de achizitionari este vida, tratam aceasta exceptie:

```

475 END ex_8;
480 /
481 BEGIN
482 --      dbms_output.put_line(ex_8('Speedo', 'Slipi'));
483 --      dbms_output.put_line(ex_8('Nabajji', 'Ochelari'));
484 --      dbms_output.put_line(ex_8('Arena', 'Slapi'));
485 --      dbms_output.put_line(ex_8('Nautica', 'Ochelari'));
486 END;
487 /
488
489

```

NU A FOST CUMPARAT NICIODATA ACEST PRODUS DE LA BRANDUL ACESTA!

PL/SQL procedure successfully completed.

Pentru articoul Slapi de la brand-ul Arena s-au emis urmatoarele bonuri:

Acest produs nu a fost inca achizitionat de nimeni, deci media va fi 0!

4. Exceptia zero_divide:

The screenshot shows the Oracle SQL Developer interface. The top menu bar includes File, Edit, View, Navigate, Run, Source, Text, Tools, Window, and Help. The left sidebar displays a tree view of database objects under 'Connections' (Tabeli), including tables like PUNCTE, AF_TECN_LIN, BON_FISCAL, CLASA, PROFESSOR, RECEPTIONER, and SALARIU, along with Views and Indices.

The main workspace contains a 'Query builder' window with the following PL/SQL code:

```
479 END ex_8;
480 /
481 BEGIN
482 --    dbms_output.put_line(ex_8('Speedo', 'Slipi'));
483 --    dbms_output.put_line(ex_8('Nabaiji', 'Ochelari'));
484 --    dbms_output.put_line(ex_8('Arena', 'Slapi'));
485 dbms_output.put_line(ex_8('Nautica', 'Ochelari'));
486 END;
```

The code is highlighted in yellow, indicating it has been run. Below the code, the output pane shows:

```
PL/SQL procedure successfully completed.

Imi pare rau, dar nu avem produsul momentan in stoc!
```

Another message follows:

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

The bottom pane, titled 'Messages - Log', shows:

```
EX_11 Compiled (with errors)
```

A status bar at the bottom indicates 'Click on an identifier with the Control key down to perform "Go to Declaration"'.

9 Subprogram stocat de tip procedură – să utilizeze într-o singură comandă SQL 5 tabele + tratarea excepțiilor

Cerinta:

Sa se construiasca o procedura care primeste ca parametru ani de experienta ai unui receptioner si prenumele sau si afiseaza toate detaliile despre acesta, cat si despre clientii carora le-a emis un abonament, incluzand si detaliile abonamentului. Pentru fiecare client servit, i se va aplica o marire de salariu lunar de 2% respectivului receptioner.

Sa trateze exceptiile cand nu exista receptionerul cu numarul respectiv de ani de experienta, cand doi receptioneri au acelasi numar de ani si cand un receptioner nu a emis niciun abonament.

Cod SQL:

```
CREATE OR REPLACE PROCEDURE ex_9(nr_ani
receptioner.ani_experienta%TYPE, v_prenume
angajat.prenume%TYPE) IS
    v_id_receptioner receptioner.id_angajat%TYPE;
    CURSOR c IS
        SELECT c.id_client id, c.nume nume, c.prenume prenume,
l.denumire oras, a.numar_sedinte nr_sed,
        a.tip_abonament tip, a.pret pret, ach.data_emitere data
        FROM receptioner r, achizitie_abonament ach, abonament a,
client c, localitate l
        WHERE r.id_angajat = v_id_receptioner AND r.id_angajat =
ach.id_angajat AND
```

```

        ach.id_abonament = a.id_abonament AND ach.id_client =
c.id_client AND c.id_localitate = l.id_localitate;

        TYPE detalii_record IS RECORD (v_nume
angajat.nume%TYPE,
                                v_prenume angajat.prenume%TYPE,
                                v_email angajat.email%TYPE,
                                v_local localitate.denumire%TYPE,
                                v_sal salariu.suma%TYPE);

t_date detalii_record;
nr_clienti NUMBER := 0;
exceptie exception;
BEGIN
    SELECT r.id_angajat
    INTO v_id_receptioner
    FROM angajat a, receptioner r
    WHERE NVL(r.ani_experienta, 0) = NVL(nr_ani, 0) AND
v_prenume = a.prenume AND a.id_angajat = r.id_angajat;

    SELECT a.nume, a.prenume, a.email, l.denumire, s.suma
    INTO t_date
    FROM angajat a, localitate l, salariu s
    WHERE a.id_angajat = v_id_receptioner AND a.id_localitate =
l.id_localitate AND a.id_angajat = s.id_angajat
        AND lower(s.tip_salariu) = 'lunar';

        dbms_output.put_line('Receptionerul ' || t_date.v_nume || ' ' ||
t_date.v_prenume || ', cu email-ul '
        || t_date.v_email || ', din orasul ' || t_date.v_local || ' ce castiga
un salariu de '
        || t_date.v_sal || ' RON a eliberat urmatorilor clienti
abonamente: ' || chr(10));

```

```

FOR i in c LOOP
    dbms_output.put_line(nr_clienti + 1 || '.Clientul ' || i.nume || ' ' ||
i.prenume || ', cu id-ul '
        || i.id || ', din orasul ' || i.oras || ' si-a achizitionat un
abonament cu un numar de sedinte de '
        || i.nr_sed || ' pentru ' || i.tip || ' la pretul de ' || i.pret || ' RON si
a fost emis la data si
        ora ' || i.data);
    nr_clienti := nr_clienti + 1;
END LOOP;

IF nr_clienti = 0 THEN
    RAISE exceptie;
ELSE
    t_date.v_sal := t_date.v_sal + 0.02 * nr_clienti * t_date.v_sal;

    UPDATE salariu
    SET suma = t_date.v_sal
    WHERE LOWER(tip_salariu) = 'lunar' AND salariu.id_angajat
= v_id_receptioner;

    dbms_output.put_line(chr(10) || 'Noul salariul al
receptionerului este de ' || t_date.v_sal || ' RON.');
END IF;

EXCEPTION
    WHEN exceptie THEN
        dbms_output.put_line('Acest receptioner nu a emis niciun
abonament inca, deoarece de abia a fost angajat!');
    WHEN no_data_found THEN
        dbms_output.put_line('Nu lucreaza niciun receptioner cu acest
prenume sau ani de experienta!');

```

```

WHEN too_many_rows THEN
    dbms_output.put_line('Sunt mai mult de 2 receptioneri care au
acelasi prenume si aceiasi ani de experienta!');
WHEN others THEN
    dbms_output.put_line('Au intervenit alte erori!');

END EX_9;
/

BEGIN
    ex_9(1, 'Tudor');
--    ex_9(2, 'Vlad');
--    ex_9(null, 'Iustin');
--    ex_9(2, 'Mirel');
--    ex_9(10, 'Andrei');
END;
/

```

Screenshot-uri fiecare caz in parte:

1. Cazul OK

Oracle SQL Developer : C:\Users\calin\Desktop\teme facultate\AN 2\SGBD\Project final\project.sql

```

562    END EX_9;
563  /
564
565 BEGIN
566   ex_9(1, 'Tudor');
567  -- ex_9(2, 'Vlad');

PL/SQL procedure successfully completed.

Receptionerul Andries Tudor, cu email-ul andries.tudor@yahoo.com, din orasul Arad ce castiga un salariu de 5850 RON a elibera:
1.Clientul Gaitan Maria, cu id-ul 70, din orasul Suceava si-a achizitionat un abonament cu un numar de sedinte de 20 pentru
ora 20-06-2020 12:21:41
2.Clientul Georgescu Paul, cu id-ul 40, din orasul Arad si-a achizitionat un abonament cu un numar de sedinte de 10 pentru
ora 26-08-2020 18:43:59
3.Clientul Georgescu Paul, cu id-ul 40, din orasul Arad si-a achizitionat un abonament cu un numar de sedinte de 20 pentru
ora 19-06-2020 12:26:01
4.Clientul Rusu Andrei, cu id-ul 100, din orasul Arad si-a achizitionat un abonament cu un numar de sedinte de 10 pentru
ora 02-09-2020 15:29:40

Noul salariul al receptionerului este de 6318 RON.

EX_11 Compiled (with errors)

```

Oracle SQL Developer : C:\Users\calin\Desktop\teme facultate\AN 2\SGBD\Project final\project.sql

```

562    END EX_9;
563  /
564
565 BEGIN
566   ex_9(1, 'Tudor');
567  -- ex_9(2, 'Vlad');

ROLLBACK complete.

Receptionerul Oprea Iustin, cu email-ul oprea.iustin@gmail.com, din orasul Bucuresti ce castiga un salariu de 5000 RON a elibera:
1.Clientul Popescu Marcel, cu id-ul 20, din orasul Suceava si-a achizitionat un abonament cu un numar de sedinte de 10 pentru
ora 19-06-2020 12:26:43
2.Clientul Munteanu Iulian, cu id-ul 60, din orasul Bucuresti si-a achizitionat un abonament cu un numar de sedinte de 10 pentru
ora 01-08-2020 12:29:40
3.Clientul Croitoru Daniel, cu id-ul 30, din orasul Iasi si-a achizitionat un abonament cu un numar de sedinte de 10 pentru
ora 07-07-2020 10:30:31
4.Clientul Croitoru Daniel, cu id-ul 30, din orasul Iasi si-a achizitionat un abonament cu un numar de sedinte de 10 pentru
ora 20-06-2020 16:23:52
5.Clientul Manea Robert, cu id-ul 120, din orasul Iasi si-a achizitionat un abonament cu un numar de sedinte de 20 pentru
ora 12-08-2020 12:23:12
6.Clientul Georgescu Paul, cu id-ul 40, din orasul Arad si-a achizitionat un abonament cu un numar de sedinte de 10 pentru
ora 01-08-2020 18:21:12

Noul salariul al receptionerului este de 5600 RON.

EX_11 Compiled (with errors)

```

2. Exceptia no_data_found

```
562 END EX_9;
563 /
564
565 BEGIN
566   -- ex_9(1, 'Tudor');
567   -- ex_9(2, 'Vlad');
568   -- ex_9(null, 'Iustin');
569   -- ex_9(2, 'Mirel');
570   ex_9(10, 'Andrei');
571
572 /
573
```

Reports

- All Reports
- Analytics View Reports
- Data Dictionary Reports
- Data Modeler Reports
- OLAP Reports
- Times Ten Reports
- User Defined Reports

Script Output: A [Query Result] x [Query Result 1] x [Query Result 2] x Task completed in 0.069 seconds

Rollback complete.

Nu lucreaza niciun receptioner cu acest prenume sau ani de experienta!

PL/SQL procedure successfully completed.

3. Exceptia too_many_rows:

The screenshot shows the Oracle SQL Developer interface with the following details:

- File Edit View Navigator Run Source Tools Window Help** - The top menu bar.
- Connections** - A tree view showing database connections and schemas, including ARTECOL_IN, BOL_FISCAL, CLERCI, COMPANIE_AI, COMPARATIR, DEPOZITI, GRUPA_AMT, LOCATEATE, PERSONAL_C, PROFESIONI, RECEPTIONER, SALARII, and TIP_SALAR.
- SQL Worksheet - Query Builder** - The main workspace where the PL/SQL code is entered.
- Code:**

```
562 END EX_9;
563 /
564
565 BEGIN
566   -- ex_9(1, 'Tudor');
567   ex_9(2, 'Vlad');
568   -- ex_9(null, 'Iustin');
569   -- ex_9(2, 'Mirel');
570   -- ex_9(10, 'Andrei');
571 END;
572 /
573
```
- Reports** - A sidebar with options like All Reports, Analytics, View Reports, Data Dictionary Reports, Data Modeler Reports, CLR Reports, TimeLine Reports, and User Defined Reports.
- Script Output** - A tab showing the result of the execution: "PL/SQL procedure successfully completed."
- Messages - Log** - A log window showing the message: "EX_9 compiled with errors!"
- Navigation Bar:** Messages, Logging Page, Extensions, Statements, Compiler.
- Status Bar:** Click on an identifier with the Control key down to perform "Go To Declaration". Line 573 Column 2 Insert Modified Windows.

4. O exceptie proprie, cand lista de emiteri de abonamente a unui receptioner este vida, tratam aceasta exceptie:

```

562 END EX_9;
563 /
564
565 BEGIN
566 --    ex_9(1, 'Tudor');
567 --    ex_9(2, 'Vlad');
568 --    ex_9(null, 'Iustin');
569     ex_9(2, 'Mirel');
570 --    ex_9(10, 'Andrei');
571 END;
572 /
573

```

Receptionerul Axinte Mirel, cu email-ul axinte.mirel@gmail.com, din orasul Arad ce castiga un salariu de 7000 RON a elibera.

Acest receptioner nu a emis niciun abonament inca, deoarece de abia a fost angajat!

PL/SQL procedure successfully completed.

10 Trigger de tip LMD la nivel de comandă

Cerinta:

*Administratorul bazarului a decis ca accesul la baza de date a bazarului sa fie de DUMINICA de la 10 - 12.
De asemenea el doreste ca inainte de sarbatori, daca pica Craciunul sau Anul Nou duminica, modificarea bazei de date nu se va face in duminica respectiv.
Creati un trigger care sa nu permita inserarea in tablele, tinand cont de lucrurile mentionate de administrator.*

Cod SQL:

```
CREATE OR REPLACE TRIGGER trigger_ex_10
    BEFORE INSERT OR UPDATE OR DELETE ON
        achizitie_abonament
    BEGIN

        IF (TO_CHAR(SYSDATE, 'd') != 7) OR
        (TO_CHAR(SYSDATE, 'hh24') NOT BETWEEN 10 AND 12)
            OR (TO_CHAR(SYSDATE, 'DD') = 25 AND
        TO_CHAR(SYSDATE, 'MM') = 12)
            OR (TO_CHAR(SYSDATE, 'DD') = 01 AND
        TO_CHAR(SYSDATE, 'MM') = 01)
        THEN
            IF INSERTING THEN
                RAISE_APPLICATION_ERROR(-20001, 'Nu se poate
insera in acest moment in baza de date!');
            ELSIF UPDATING THEN
                RAISE_APPLICATION_ERROR(-20002, 'Nu se poate
modifica in acest moment in baza de date!');
            ELSIF DELETING THEN
                RAISE_APPLICATION_ERROR(-20003, 'Nu se poate
sterge in acest moment in baza de date!');
            END IF;
        END IF;
    END;
    /
```

```
ALTER TRIGGER trigger_ex_10 DISABLE;
ALTER TRIGGER trigger_ex_10 ENABLE;
```

```
insert into achizitie_abonament values(20, 3, 80, '19-12-2021
12:26:43', '19-12-2021', '10-01-2022');
```

```
update achizitie_abonament  
set id_angajat = 100  
where id_client = 20 and id_abonament = 1;
```

```
delete from achizitie_abonament  
where id_abonament = 1;
```

Screenshot-uri :

The screenshot shows the Oracle SQL Developer interface. The left sidebar displays a database schema with various tables like ARTICOLE, ANGAJATI, PROFESORI, and RECEPTORI. The main window contains a query editor with the following code:

```
500  
501 ALTER TRIGGER trigger_ex_10 DISABLE;  
502 ALTER TRIGGER trigger_ex_10 ENABLE;  
503  
504 insert into achizitie_abonament values(20, 3, 80, '19-12-2021 12:26:43', '19-12-2021', '10-01-2022');  
505  
506 update achizitie_abonament  
507 set id_angajat = 100  
508 where id_client = 20 and id_abonament = 1;  
509  
510 delete from achizitie_abonament  
511 where id_abonament = 1;
```

Below the code, an error message is displayed:

```
Error starting at line : 604 in command -  
insert into achizitie_abonament values(20, 3, 80, '19-12-2021 12:26:43', '19-12-2021', '10-01-2022')  
Error report -  
ORA-20001: Nu se poate insera in acest moment in baza de date!  
ORA-06512: at "ADMIN.TRIGGER_EX_10", line 8  
ORA-04088: error during execution of trigger 'ADMIN.TRIGGER_EX_10'
```

The bottom status bar indicates the message log.

Oracle SQL Developer : C:\Users\calin\Desktop\teme facultate\AN 2\SGBD\Project final\project.sql

```

ALTER TRIGGER trigger_ex_10 DISABLE;
ALTER TRIGGER trigger_ex_10 ENABLE;

insert into achizitie_abonament values(20, 3, 80, '19-12-2021 12:26:43', '19-12-2021', '10-01-2022');

update achizitie_abonament
set id_angajat = 100
where id_client = 20 and id_abonament = 1;

delete from achizitie_abonament
where id_abonament = 1;

set id_angajat = 100
where id_client = 20 and id_abonament = 1
Error report -
ORA-20002: Nu se poate modifica in acest moment in baza de date!
ORA-06512: at "ADMIN.TRIGGER_EX_10", line 10
ORA-04088: error during execution of trigger 'ADMIN.TRIGGER_EX_10'

601
602
603
604
605
606
607
608
609
610
611
612
613
614

```

Messages-Log | Breakpoints |
B6_11 Compiled (with errors)

Messages | Logging Page | Extensions | Statements | Compiler |

Click on an identifier with the Control key down to perform 'Go to Declaration'

Oracle SQL Developer : C:\Users\calin\Desktop\teme facultate\AN 2\SGBD\Project final\project.sql

```

ALTER TRIGGER trigger_ex_10 ENABLE;

insert into achizitie_abonament values(20, 3, 80, '19-12-2021 12:26:43', '19-12-2021', '10-01-2022);

update achizitie_abonament
set id_angajat = 100
where id_client = 20 and id_abonament = 1;

delete from achizitie_abonament
where id_abonament = 1;

-- Exercitiul 11

delete from achizitie_abonament
where id_abonament = 1
Error report -
ORA-20003: Nu se poate sterge in acest moment in baza de date!
ORA-06512: at "ADMIN.TRIGGER_EX_10", line 12
ORA-04088: error during execution of trigger 'ADMIN.TRIGGER_EX_10'

601
602
603
604
605
606
607
608
609
610
611
612
613
614

```

Messages-Log | Breakpoints |
B6_11 Compiled (with errors)

Messages | Logging Page | Extensions | Statements | Compiler |

Click on an identifier with the Control key down to perform 'Go to Declaration'

11 Trigger de tip LMD la nivel de linie

Cerinta:

Administratorul bazarului a decis ca salariul unui nou angajat sa fie cel minim pe economii(1524 de lei) si pe acel post care se angajeaza salariatul sa se scada cate 100 de lei din salariile celorlalri angajati de pe postul respectiv. Se vor actualiza si sub tabelele daca functia noului angajat este RECEPȚIONER/PROFESOR/PERSONAL_CURATENIE astfel:

- pentru RECEPȚIONER se va adauga ca ani de experienta NULL*
- pentru PROFESOR se vor lasa 10 antrenamente pe saptamana*
- pentru PERSONAL_CURATENIE se va genera un nr de telefon random.*

Cod SQL:

```
CREATE TABLE angajat_copie AS  
SELECT * FROM angajat;
```

```
DROP TABLE angajat_copie;
```

```
CREATE OR REPLACE PROCEDURE ex_11(v_id  
angajat.id_angajat%TYPE, v_func angajat.functie_angajat%TYPE)  
IS  
    TYPE t_id IS TABLE OF angajat.id_angajat%TYPE;  
    id_s t_id;  
    max_id_sal salariu.id_salariu%TYPE;  
BEGIN  
    SELECT MAX(id_salariu)  
    INTO max_id_sal
```

```

FROM salariu;

IF LOWER(v_func) = 'receptioner' THEN
    INSERT INTO receptioner VALUES(v_id, null);
ELSIF LOWER(v_func) = 'profesor' THEN
    INSERT INTO profesor VALUES(v_id, 10);
ELSIF LOWER(v_func) = 'personal_curatenie' THEN
    INSERT INTO personal_curatenie VALUES(v_id, '07' ||
TO_CHAR(ROUND(dbms_random.value(23424553, 99999999))));
```

END IF;

```

SELECT a.id_angajat
BULK COLLECT INTO id_s
FROM angajat_copie a
WHERE LOWER(a.functie_angajat) = LOWER(v_func);
```

```

FORALL i in id_s.first..id_s.last
    UPDATE salariu
        SET salariu.suma = salariu.suma - 100
        WHERE salariu.id_angajat = id_s(i) and salariu.tip_salariu =
'LUNAR';
```

```

    INSERT INTO salariu VALUES(max_id_sal + 10, v_id, 1524,
'LUNAR');
```

```

END ex_11;
/
CREATE OR REPLACE TRIGGER trigger_ex_11
AFTER INSERT ON angajat
FOR EACH ROW
BEGIN
    ex_11(:NEW.id_angajat, :NEW.functie_angajat);

```

END;

/

```
insert into angajat values(170, 100,'Mara', 'Ioana',
'mara.ioana@gmail.com', 'RECEPTIONER');
ROLLBACK;
```

Screenshot-uri:

BEFORE:

ANGAJAT

ID_ANGAJAT	ID_LOCALITATE	NUME	PRENUME	EMAIL	FUNCTIE_ANGAJAT
1	50	150Moscaliuc	Cezar	cezar.moscaliuc@gmail.com	PROFESOR
2	60	120Oprea	Iustin	oprea.iustin@gmail.com	RECEPTIONER
3	70	130Maxim	Vlad	vlad maxim@gmail.com	PROFESOR
4	80	160Andries	Tudor	andries.tudor@yahoo.com	RECEPTIONER
5	90	100Carpineanu	Vlad	carpineanu.vlad@gmail.com	RECEPTIONER
6	100	120Airinei	Mirela	mirela.airinei@yahoo.com	PROFESOR
7	110	130Bucaciuc	Andreea	andreea.bucaciuc@gmail.com	PERSONAL CURATENIE
8	120	100Cernaut	Angela	angela.cernaut@yahoo.com	PERSONAL CURATENIE
9	130	120Costache	Viorel	viorel.costache@yahoo.com	PAZNIC
10	140	130Nistor	Alexandru	alexandru.nistor@gmail.com	ADMINISTRATOR BAZIN
11	150	160Axinte	Mirel	axinte.mirel@gmail.com	RECEPTIONER
12	160	130Leonte	Vlad	leonte.vlad@gmail.com	RECEPTIONER

RECEPTIONER

ID_ANGAJAT	ANI_EXPERIENTA
1	60 (null)
2	90 2
3	80 1
4	150 2
5	160 2

SALARIU

	ID_SALARIU	ID_ANGAJAT	SUMA	TIP_SALARIU
1	10	50	4000	LUNAR
2	20	60	5000	LUNAR
3	30	110	2000	LUNAR
4	40	100	4400	LUNAR
5	50	80	5850	LUNAR
6	60	90	7000	LUNAR
7	70	120	2200	LUNAR
8	80	80	180	BONUS
9	90	70	5200	LUNAR
10	100	130	1600	LUNAR
11	110	130	400	BONUS
12	120	140	12000	LUNAR
13	130	150	7000	LUNAR
14	140	160	4200	LUNAR

1 row inserted.

AFTER:

ANGAJAT

	ID_ANGAJAT	ID_LOCALITATE	NUME	PRENUME	EMAIL	FUNCTIE_ANGAJAT
1	50	150	Moscaliuc	Cezar	cezar.moscaliuc@gmail.com	PROFESOR
2	60	120	Oprea	Iustin	oprea.iustin@gmail.com	RECEPTIONER
3	70	130	Maxim	Vlad	vlad.maxim@gmail.com	PROFESOR
4	80	160	Andries	Tudor	andries.tudor@yahoo.com	RECEPTIONER
5	90	100	Carpineanu	Vlad	carpineanu.vlad@gmail.com	RECEPTIONER
6	100	120	Airinei	Mirela	mirela.airinei@yahoo.com	PROFESOR
7	110	130	Bucaciuc	Andreea	andreea.bucaciuc@gmail.com	PERSONAL CURATENIE
8	120	100	Cernaut	Angela	angela.cernaut@yahoo.com	PERSONAL CURATENIE
9	130	120	Costache	Viorel	viorel.costache@yahoo.com	PAZNIC
10	140	130	Nistor	Alexandru	alexandru.nistor@gmail.com	ADMINISTRATOR BAZIN
11	150	160	Axinte	Mirel	axinte.mirel@gmail.com	RECEPTIONER
12	160	130	Leonte	Vlad	leonte.vlad@gmail.com	RECEPTIONER
13	170	100	Mara	Ioana	mara.ioana@gmail.com	RECEPTIONER

RECEPTIONER

	ID_ANGAJAT	ANI_EXPERIENTA
1	60	(null)
2	90	2
3	80	1
4	150	2
5	160	2
6	170	(null)

SALARIU

	ID_SALARIU	ID_ANGAJAT	SUMA	TIP_SALARIU
1	10	50	4000	LUNAR
2	20	60	4900	LUNAR
3	30	110	2000	LUNAR
4	40	100	4400	LUNAR
5	50	80	5750	LUNAR
6	60	90	6900	LUNAR
7	70	120	2200	LUNAR
8	80	80	180	BONUS
9	90	70	5200	LUNAR
10	100	130	1600	LUNAR
11	110	130	400	BONUS
12	120	140	12000	LUNAR
13	130	150	6900	LUNAR
14	140	160	4100	LUNAR
15	150	170	1524	LUNAR

12 Trigger de tip LDD

Cerinta:

Administratorul doreste sa nu fie facuta nicio operatie pe baza lui de date in zilele de Sambata si Duminica in afara intervalului 16-20. De asemenea, daca este intr-o zi si un interval orar favorabil pentru modificare, acesta ar dori sa cunoasca ce operatii s-au facut pe aceasta.

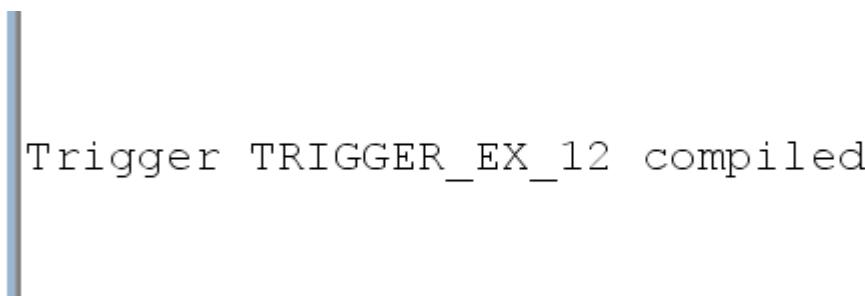
Cod SQL:

```
CREATE TABLE istoric_modificari_baza(  
    nume_baza_de_date VARCHAR2(80),  
    utilizator VARCHAR2(50),  
    operatie VARCHAR2(30),  
    data DATE,  
    tip_object VARCHAR2(40),  
    nume_object VARCHAR2(40)  
)
```

```
CREATE OR REPLACE TRIGGER trigger_ex_12  
    BEFORE CREATE OR DROP OR ALTER ON  
        SCHEMA  
    BEGIN  
        IF TO_CHAR(SYSDATE, 'd') = 6 OR  
            TO_CHAR(SYSDATE, 'd') = 7 OR  
            (TO_CHAR(SYSDATE, 'hh24') NOT BETWEEN 16  
            AND 20)  
        THEN
```

```
        RAISE_APPLICATION_ERROR(-20001, 'De ce vrei  
sa modific baza de date in weekend?');  
ELSE  
    INSERT INTO istoric_modificari_baza VALUES  
(SYS DATABASE_NAME, SYS LOGIN_USER,  
           SYS SYSEVENT, SYSDATE,  
           SYS DICTIONARY_OBJ_TYPE,  
           SYS DICTIONARY_OBJ_NAME);  
END IF;  
END;  
/  
DROP TABLE angajat_copie;  
  
SELECT * FROM istoric_modificari_baza;
```

Screenshot-uri:



```
Trigger TRIGGER_EX_12 compiled
```

```

Worksheet  Query Builder
689 IF TO_CHAR(SYSDATE, 'd') = 6 OR TO_CHAR(SYSDATE, 'd') = 7 OR (TO_CHAR(SYSDATE, 'hh24') NOT BETWEEN 19 AND 20)
690 THEN
691     RAISE_APPLICATION_ERROR(-20001, 'De ce vrei sa modifici baza de date in weekend?');
692 ELSE
693     INSERT INTO istoric_modificari_baza VALUES (SYS.DATABASE_NAME, SYS.LOGIN_USER,
694                                         SYS.SYSEVENT, SYSDATE, SYS.DICTIONARY_OBJ_TYPE, SYS.DICTIONARY_OBJ_NAM
695     END IF;
696 END;
697 /
698 DROP TABLE angajat_copie;
699
700 SELECT * FROM istoric_modificari_baza;
701

```

Script Output | Query Result | Query Result 1 | Query Result 2 | Task completed in 0.036 seconds

DROP TABLE angajat_copie
Error report -
ORA-00604: error occurred at recursive SQL level 1
ORA-20001: De ce vrei sa modifici baza de date in weekend?
ORA-06512: at line 4
00604. 00000 - "error occurred at recursive SQL level %s"
*Cause: An error occurred while processing a recursive SQL statement
(a statement applying to internal dictionary tables).

Messages - Log | Breakpoints |

```

Worksheet  Query Builder
695 END IF;
696 END;
697 /
698 DROP TABLE angajat_copie;
699
700 SELECT * FROM istoric_modificari_baza;
701
702 -- Exercitiul 13
703
704 CREATE OR REPLACE PACKAGE pacchet_cerinte AS
705     PROCEDURE ex_6;
706     PROCEDURE ex_7(v_id receptioner.id_angajat%TYPE);
707     FUNCTION ex_8(v_brand_articol inof.brand%TYPE, v_tip_articol inof.tip_articol%TYPE) RETURN VARCHAR2;

```

Script Output | Query Result | Query Result 1 | Query Result 2 | All Rows Fetched: 10 in 0.001 seconds

NUME_BAZA_DE_DATE	UTILIZATOR	OPERATIE	DATA	TIP_OBJECT	NUME_OBJECT
1 XE	ADMIN	DROP	07-01-2022	TABLE	ANGAJAT COPIE
2 XE	ADMIN	CREATE	07-01-2022	TABLE	ANGAJAT COPIE
3 XE	ADMIN	DROP	07-01-2022	TABLE	ANGAJAT COPIE
4 XE	ADMIN	CREATE	07-01-2022	TABLE	ANGAJAT COPIE
5 XE	ADMIN	ALTER	07-01-2022	TRIGGER	TRIGGER EX 10
6 XE	ADMIN	CREATE	07-01-2022	TRIGGER	TRIGGER EX 10
7 XE	ADMIN	ALTER	07-01-2022	TRIGGER	TRIGGER EX 11
8 XE	ADMIN	ALTER	07-01-2022	PROCEDURE	EX 11
9 XE	ADMIN	DROP	07-01-2022	TABLE	ANGAJAT COPIE
10 XE	ADMIN	CREATE	07-01-2022	TABLE	ANGAJAT COPIE

13 Pachet ce conține toate obiectele definite în proiect

```
CREATE OR REPLACE PACKAGE pachet_cerinte AS
    PROCEDURE ex_6;
    PROCEDURE ex_7(v_id receptioner.id_angajat%TYPE);
    FUNCTION ex_8(v_brand articol_inot.brand%TYPE,
                  v_tip articol_inot.tip_articol%TYPE) RETURN
                  VARCHAR2;
    PROCEDURE ex_9(nr_ani
                   receptioner.ani_experienta%TYPE, v_prenume
                   angajat.prenume%TYPE);
    PROCEDURE ex_11(v_id angajat.id_angajat%TYPE,
                    v_func angajat.functie_angajat%TYPE);
END pachet_cerinte;
/
```

```
CREATE OR REPLACE PACKAGE BODY pachet_cerinte
    AS
        PROCEDURE ex_6 IS
            TYPE t_imbricat IS TABLE OF salariu.suma%TYPE;
            TYPE t_imb_angaj IS TABLE OF
                angajat.id_angajat%TYPE;
            t_salarii t_imbricat;
            t_receptioneri t_imb_angaj;
            TYPE detalii_record IS RECORD (v_id_angajat
                angajat.id_angajat%TYPE,
                v_nume_localitate
                localitate.denumire%TYPE,
                v_nume_angajat.nume%TYPE,
```

```

        v_prenume angajat.nume%TYPE,
        v_email angajat.email%TYPE,
        v_sal_nou salariu.suma%TYPE);

        TYPE t_date IS TABLE OF detalii_record INDEX BY
        PLS_INTEGER;
        tablou_date_angajat t_date;
        maxx_sal salariu.suma%TYPE := 0;
        maxx_old_sal salariu.suma%TYPE := 0;

        BEGIN
            SELECT sal.suma
            BULK COLLECT INTO t_salarii
            FROM receptioner r, salariu sal
            WHERE LOWER(sal.tip_salariu) = 'lunar' AND
            r.id_angajat = sal.id_angajat;

            FOR i IN t_salarii.first..t_salarii.last LOOP
                IF t_salarii(i) > maxx_sal THEN
                    maxx_sal := t_salarii(i);
                END IF;
            END LOOP;

            SELECT r.id_angajat
            BULK COLLECT INTO t_receptioneri
            FROM receptioner r, salariu sal
            WHERE LOWER(sal.tip_salariu) = 'lunar' AND
            r.id_angajat = sal.id_angajat AND sal.suma = maxx_sal;

            FORALL i in t_receptioneri.first..t_receptioneri.last
                UPDATE salariu

```

```
SET salariu.suma = salariu.suma - 0.1 * salariu.suma  
WHERE salariu.id_angajat = t_receptioneri(i);
```

```
maxx_old_sal := maxx_sal;
```

```
SELECT MAX(sal.suma)  
INTO maxx_sal  
FROM salariu sal, angajat a  
WHERE LOWER(a.functie_angajat) = 'receptioner'  
AND sal.id_angajat = a.id_angajat;
```

```
SELECT a.id_angajat, l.denumire, a.nume, a.prenume,  
a.email, sal.suma  
BULK COLLECT INTO tablou_date_angajat  
FROM angajat a, salariu sal, localitate l  
WHERE a.id_angajat = sal.id_angajat AND  
LOWER(sal.tip_salariu) = 'lunar' AND  
LOWER(a.functie_angajat) = 'receptioner' AND  
a.id_localitate = l.id_localitate  
AND sal.suma = maxx_sal;
```

```
FOR i IN  
tablou_date_angajat.first..tablou_date_angajat.last LOOP  
    dbms_output.put_line('Receptionerul cu id-ul ' ||  
        tablou_date_angajat(i).v_id_angajat || ' cu numele  
        complet '  
        || tablou_date_angajat(i).v_nume || ' ' ||  
        tablou_date_angajat(i).v_prenume || ' din localitatea '  
        || tablou_date_angajat(i).v_nume_localitate || ' cu  
        email-ul ' || tablou_date_angajat(i).v_email
```

```

    || chr(10) || 'a avut salariul de ' || maxx_old_sal || ',
acum avand ' || tablou_date_angajat(i).v_sal_nou
    || '.' || chr(10));
END LOOP;
dbms_output.put_line('Numarul de angajati al caror
salariu s-a modificat este: ' || tablou_date_angajat.count);
END ex_6;

```

```

PROCEDURE ex_7(v_id receptioner.id_angajat%TYPE)
IS
    CURSOR c IS
        SELECT a.id_angajat, a.nume, a.prenume,
l.denumire,
        CURSOR(SELECT DISTINCT c.id_client,
c.nume, c.prenume, c.data_nastere--cursor imbricat
        FROM client c, achizitie_abonament ach
        where ach.id_client = c.id_client AND
a.id_angajat = ach.id_angajat)
        FROM angajat a, localitate l
        WHERE LOWER(a.functie_angajat) =
'receptioner' AND a.id_angajat = v_id AND
a.id_localitate = l.id_localitate;
    TYPE c_dinamic IS REF CURSOR;
    info_clienti c_dinamic;
    v_id_ang angajat.id_angajat%TYPE;
    v_nume_ang angajat.nume%TYPE;
    v_prenume_ang angajat.prenume%TYPE;
    v_localitate_ang localitate.denumire%TYPE;
    v_id_cli client.id_client%TYPE;
    v_nume_cli client.nume%TYPE;

```

```

    v_prenume_cli client.prenume%TYPE;
    v_data_cli client.data_nastere%TYPE;
    clienti_prezenti BOOLEAN := FALSE;

BEGIN
    OPEN c;
    LOOP
        FETCH c INTO v_id_ang, v_nume_ang,
        v_prenume_ang, v_localitate_ang, info_clienti;
        EXIT WHEN c%NOTFOUND;

        dbms_output.put_line('Receptionerul ' || v_nume_ang
        || ' ' || v_prenume_ang || ', cu id-ul ' || v_id_ang
        || ', din orasul ' || v_localitate_ang || ' a emis
        abonamente urmatorilor clienti: ');

    LOOP
        FETCH info_clienti into v_id_cli, v_nume_cli,
        v_prenume_cli, v_data_cli;
        EXIT WHEN info_clienti%NOTFOUND;

        clienti_prezenti := TRUE;
        dbms_output.put_line('Clientul ' || v_nume_cli || ''
        || v_prenume_cli || ', cu id-ul ' || v_id_cli
        || ', nascut la data de ' || v_data_cli);
    END LOOP;
    IF clienti_prezenti = FALSE THEN
        dbms_output.put_line('Nu exista clienti ajutati de
        acest receptioner.');
    END IF;
--    dbms_output.new_line();

```

```

        END LOOP;
END ex_7;

FUNCTION ex_8(v_brand articol_inot.brand%TYPE,
              v_tip articol_inot.tip_articol%TYPE)
RETURN VARCHAR2 IS
    TYPE detalii_record IS RECORD (v_id_bon
                                     bon_fiscal.id_bon%TYPE,
                                     v_data bon_fiscal.data_bon%TYPE,
                                     v_metoda
                                     bon_fiscal.tip_bon%TYPE);
    TYPE t_date IS TABLE OF detalii_record;
    info_tranzactii t_date;
    TYPE art_rec IS RECORD (v_id_art
                           articol_inot.id_articol%TYPE,
                           v_pret articol_inot.pret%TYPE,
                           v_cant
                           articol_inot.cantitate_marfa_disponibila%TYPE);
    articole_sportive art_rec;
    numar_art_disp NUMBER(5);
    medie NUMBER(8, 3) := 0;
    exceptie exception;
BEGIN
    SELECT id_articol, pret, cantitate_marfa_disponibila
    INTO articole_sportive
    FROM articol_inot
    WHERE brand = v_brand AND tip_articol = v_tip;

    SELECT b.*
```

```

BULK COLLECT INTO info_tranzactii
  FROM articol_inot a, cumparaturi c, bon_fiscal b
  WHERE a.brand = v_brand AND a.tip_articol = v_tip
  AND c.id_articol = a.id_articol AND c.id_bon =
  b.id_bon;

IF articole_sportive.v_cant IS null THEN
  medie := (info_tranzactii.count *
articole_sportive.v_pret) / NVL(articole_sportive.v_cant,
0);
ELSE
  numar_art_disp :=
TO_NUMBER(SUBSTR(articole_sportive.v_cant, 1,
2));
  medie := (info_tranzactii.count *
articole_sportive.v_pret) / numar_art_disp;
END IF;

dbms_output.put_line('Pentru articolul ' || v_tip || ' de la
brand-ul ' || v_brand ||
' s-au emis urmatoarele bonuri: ' || chr(10));
IF info_tranzactii.count = 0 THEN
  RAISE exceptie;
ELSE
  FOR i IN 1..info_tranzactii.count LOOP
    dbms_output.put_line('Bonul cu id-ul ' ||
info_tranzactii(i).v_id_bon || ' a fost emis la data si ora '
|| info_tranzactii(i).v_data || ' si plata s-a achitat '
|| info_tranzactii(i).v_metoda);
  END LOOP;

```

END IF;

RETURN ('Raportul este: ' || medie);

EXCEPTION

WHEN exceptie THEN

RETURN 'Acest produs nu a fost inca achizitionat de nimeni, deci media va fi 0!';

WHEN no_data_found THEN

RETURN 'Nu a fost cumparat niciodata acest produs de la brandul acesta!';

WHEN zero_divide THEN

RETURN 'Imi pare rau, dar nu avem produsul momentan in stoc!';

WHEN others THEN

RETURN 'Au intervenit alte erori!';

END ex_8;

PROCEDURE ex_9(nr_ani
receptioner.ani_experienta%TYPE, v_prenume
angajat.prenume%TYPE) IS
v_id_receptioner receptioner.id_angajat%TYPE;
CURSOR c IS
SELECT c.id_client id, c.nume nume, c.prenume
prenume, l.denumire oras, a.numar_sedinte nr_sed,
a.tip_abonament tip, a.pret pret,
ach.data_emitere data
FROM receptioner r, achizitie_abonament ach,
abonament a, client c, localitate l
WHERE r.id_angajat = v_id_receptioner AND

```

r.id_angajat = ach.id_angajat AND
    ach.id_abonament = a.id_abonament AND
    ach.id_client = c.id_client AND c.id_localitate =
    l.id_localitate;

TYPE detalii_record IS RECORD (v_numere
angajat.nume%TYPE,
                                v_prenume
angajat.prenume%TYPE,
                                v_email angajat.email%TYPE,
                                v_local localitate.denumire%TYPE,
                                v_sal salariu.suma%TYPE);
t_date detalii_record;
nr_clienti NUMBER := 0;
exceptie exception;
BEGIN
    SELECT r.id_angajat
    INTO v_id_receptioner
    FROM angajat a, receptioner r
    WHERE NVL(r.ani_experienta, 0) = NVL(nr_ani, 0)
    AND v_prenume = a.prenume AND a.id_angajat =
    r.id_angajat;

    SELECT a.nume, a.prenume, a.email, l.denumire,
s.suma
    INTO t_date
    FROM angajat a, localitate l, salariu s
    WHERE a.id_angajat = v_id_receptioner AND
    a.id_localitate = l.id_localitate AND a.id_angajat =
    s.id_angajat

```

AND lower(s.tip_salariu) = 'lunar';

```
dbms_output.put_line('Receptionerul ' || t_date.v_nume
|| '' || t_date.v_prenume || ', cu email-ul '
|| t_date.v_email || ', din orasul ' || t_date.v_local || ' ce
castiga un salariu de '
|| t_date.v_sal || ' RON a eliberat urmatorilor clienti
abonamente: ' || chr(10));
```

FOR i in c LOOP

```
dbms_output.put_line(nr_clienti + 1 || '.Clientul ' ||
i.nume || '' || i.prenume || ', cu id-ul '
|| i.id || ', din orasul ' || i.oras || ' si-a achizitionat un
abonament cu un numar de sedinte de '
|| i.nr_sed || ' pentru ' || i.tip || ' la pretul de ' || i.pret
|| ' RON si a fost emis la data si
ora ' || i.data);
nr_clienti := nr_clienti + 1;
END LOOP;
```

IF nr_clienti = 0 THEN

RAISE exceptie;

ELSE

```
t_date.v_sal := t_date.v_sal + 0.02 * nr_clienti *
t_date.v_sal;
```

UPDATE salariu

SET suma = t_date.v_sal

WHERE LOWER(tip_salariu) = 'lunar' AND

```

        salariu.id_angajat = v_id_receptioner;

        dbms_output.put_line(chr(10) || 'Noul salariul al
receptionerului este de ' || t_date.v_sal || ' RON.');

        END IF;

EXCEPTION
    WHEN exceptie THEN
        dbms_output.put_line('Acest receptioner nu a emis
niciun abonament inca, deoarece de abia a fost angajat!');

    WHEN no_data_found THEN
        dbms_output.put_line('Nu lucreaza niciun
receptioner cu acest prenume sau ani de experienta!');

    WHEN too_many_rows THEN
        dbms_output.put_line('Sunt mai mult de 2
receptioneri care au acelasi prenume si aceiasi ani de
experienta!');

    WHEN others THEN
        dbms_output.put_line('Au intervenit alte erori!');

END EX_9;

```

```

-- procedura TRIGGER 11
PROCEDURE ex_11(v_id_angajat.id_angajat%TYPE,
v_func_angajat.functie_angajat%TYPE) IS
    TYPE t_id IS TABLE OF angajat.id_angajat%TYPE;
    id_s t_id;
    max_id_sal salariu.id_salariu%TYPE;
BEGIN
    SELECT MAX(id_salariu)

```

```

    INTO max_id_sal
    FROM salariu;

    IF LOWER(v_func) = 'receptioner' THEN
        INSERT INTO receptioner VALUES(v_id, null);
    ELSIF LOWER(v_func) = 'profesor' THEN
        INSERT INTO profesor VALUES(v_id, 10);
    ELSIF LOWER(v_func) = 'personal_curatenie' THEN
        INSERT INTO personal_curatenie VALUES(v_id,
        '07' ||
        TO_CHAR(ROUND(dbms_random.value(23424553,
        99999999))));

    END IF;

    SELECT a.id_angajat
    BULK COLLECT INTO id_s
    FROM angajat_copie a
    WHERE LOWER(a.functie_angajat) =
    LOWER(v_func);

    FORALL i in id_s.first..id_s.last
        UPDATE salariu
        SET salariu.suma = salariu.suma - 100
        WHERE salariu.id_angajat = id_s(i) and
        salariu.tip_salariu = 'LUNAR';

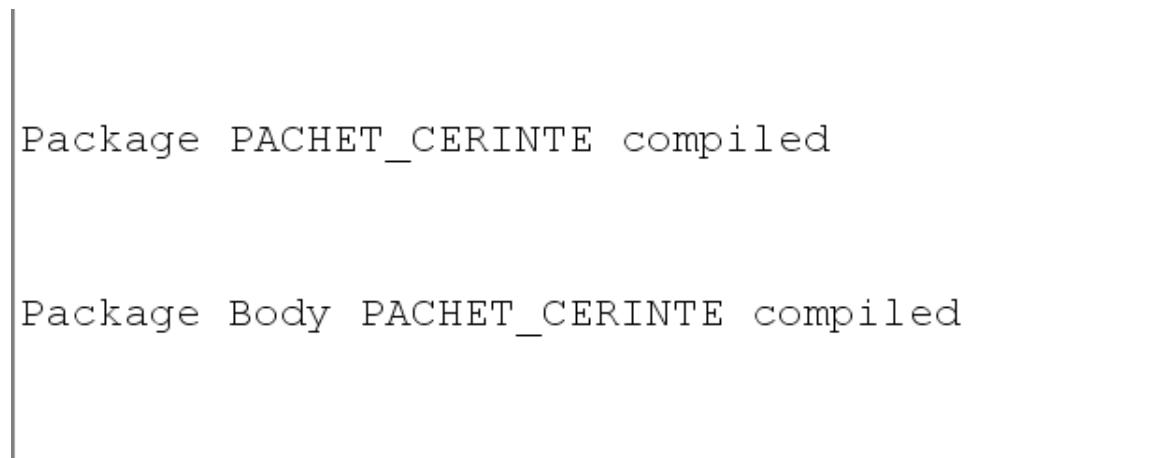
        INSERT INTO salariu VALUES(max_id_sal + 10,
        v_id, 1524, 'LUNAR');

    END ex_11;

```

```
END pachet_cerinte;  
/  
;
```

Screenshot-uri:



The screenshot shows the Oracle SQL Developer interface. The 'Worksheet' tab is active and contains the following PL/SQL code:

```
972 END pachet_cerinte;  
973 /  
974  
975 BEGIN  
976   pachet_cerinte.ex_7(60);  
977 END;  
978
```

Below the code, the status bar shows: 'Script Output X | Query Result X | Query Result 1 X | Query Result 2 X | Task completed in 0.037 seconds'.

PL/SQL procedure successfully completed.

Receptionerul Oprea Iustin, cu id-ul 60, din orasul Bucuresti a emis abonamente urmatorilor clienti:
Clientul Georgescu Paul, cu id-ul 40, nascut la data de 04-05-2000
Clientul Munteanu Iulian, cu id-ul 60, nascut la data de 15-08-2004
Clientul Popescu Marcel, cu id-ul 20, nascut la data de 12-02-1995
Clientul Manea Robert, cu id-ul 120, nascut la data de 20-12-2000
Clientul Croitoru Daniel, cu id-ul 30, nascut la data de 23-10-1989

PL/SQL procedure successfully completed.

The screenshot shows a SQL Worksheet window in SSMS. The script pane contains the following code:

```

974
975 BEGIN
976 -- pachet_cerinte.ex_7(60);
977 pachet_cerinte.ex_9(1, 'Tudor');
978 END;

```

The results pane displays the output of the stored procedure, which includes a message about a receptionist's salary increase and a list of clients' names and their corresponding abonament counts.

Receptionerul Andries Tudor, cu email-ul andries.tudor@yahoo.com, din orasul Arad ce castiga un salariu de 5850 RON a elibera o noua cerinta.

1.Clientul Gaitan Maria, cu id-ul 70, din orasul Suceava si-a achizitionat un abonament cu un numar de sedinte de 20 pentru ora 20-06-2020 12:21:41

2.Clientul Georgescu Paul, cu id-ul 40, din orasul Arad si-a achizitionat un abonament cu un numar de sedinte de 10 pentru ora 26-08-2020 18:43:59

3.Clientul Georgescu Paul, cu id-ul 40, din orasul Arad si-a achizitionat un abonament cu un numar de sedinte de 20 pentru ora 19-06-2020 12:26:01

4.Clientul Rusu Andrei, cu id-ul 100, din orasul Arad si-a achizitionat un abonament cu un numar de sedinte de 10 pentru ora 02-09-2020 15:29:40

Noul salariul al receptionerului este de 6318 RON.

14 Pachet ce conține tipuri de date complexe și obiecte necesare unui flux de acțiuni integrate

Cerinta:

Intrand in 2022, administratorul bazinei doreste sa faca o lista pentru fiecare receptioner in parte, unde sa puna id-ul clientilor care au fost ajutati de ei. Deoarece administratorul este un sef darnic, el a promis ca va mari salariul cu 30% acelui receptioner care a vandut cele mai multe abonamente. (daca un client a cumparat de mai multe ori un abonament de la acelasi receptioner, se va contoriza doar o data)

(Daca sunt mai multi receptioneri care au acelasi numar maxim de clienti serviti atunci se va lua unul din ei la intamplare...se pare ca seful nu e chiar asa darnic..).

In plus, administratorul doreste sa stie cate perechi de echipamente sportive a achizitionat de la fiecare mare brand sportiv. De asemenea, el doreste sa stie si cati clienti au achizitionat cele mai multe abonamente dintre toti.

Cod SQL:

```
CREATE OR REPLACE PACKAGE pachet_complex AS  
  TYPE vect_cli IS VARRAY(100) OF  
    client.id_client%TYPE;  
  TYPE matrice_recep IS VARRAY(100) OF vect_cli;  
  TYPE vect_recep IS VARRAY(100) OF  
    angajat.id_angajat%TYPE;  
  
  TYPE detalii_record IS RECORD (v_tip  
    articol_inot.tip_articol%TYPE,  
    v_cant  
    articol_inot.cantitate_marfa_disponibila%TYPE);  
  TYPE vect_echip IS TABLE OF detalii_record;  
  TYPE matrice_comp IS VARRAY(100) OF vect_echip;  
  TYPE vect_comp IS TABLE OF  
    companie_articole_inot.id_companie%TYPE;  
  
  TYPE detalii_record_cli IS RECORD (v_id  
    client.id_client%TYPE,  
    v_nr NUMBER);  
  TYPE vect_clienti IS TABLE OF detalii_record_cli
```

```

INDEX BY PLS_INTEGER;

PROCEDURE creare_matrice_recep;
FUNCTION calculeaza_maxim_recep RETURN
    receptioner.id_angajat%TYPE;
PROCEDURE marire_salariu(id IN
    receptioner.id_angajat%TYPE);
PROCEDURE afisare_recep;

PROCEDURE creare_matrice_echip;
PROCEDURE afisare_firme;

PROCEDURE creare_colectie;
PROCEDURE afisare_clienti;
FUNCTION calculeaza_maxim_cli RETURN NUMBER;
FUNCTION numar_max_cli RETURN NUMBER;

END pachet_complex;

CREATE OR REPLACE PACKAGE BODY
    pachet_complex AS
        m_recep matrice_recep := matrice_recep();
        m_comp matrice_comp := matrice_comp();
        t_cli vect_clienti;
        indice NUMBER := 1;
        jndice NUMBER := 1;

```

```

PROCEDURE creare_matrice_recep IS
    CURSOR c1 IS
        SELECT id_angajat
        FROM receptioner;
    BEGIN
        FOR i IN c1 LOOP
            m_recep.extend();
            SELECT DISTINCT c.id_client
            BULK COLLECT INTO m_recep(indice)
            FROM client c, achizitie_abonament a
            WHERE a.id_angajat = i.id_angajat AND a.id_client
            = c.id_client;
            indice := indice + 1;
        END LOOP;
    END creare_matrice_recep;

```

```

PROCEDURE afisare_recep IS
    t vect_recep;
BEGIN
    SELECT id_angajat
    BULK COLLECT INTO t
    FROM receptioner;

    FOR i IN 1..indice-1 LOOP
        IF m_recep(i).count = 0 THEN

```

```

        dbms_output.put_line('Receptionerul cu id-ul ' ||
t(i) || ' NU A ELIBERAT NICIUN ABONAMENT!! ');
        ELSE
            dbms_output.put_line('Receptionerul cu id-ul ' ||
t(i) || ' a eliberat abonamente clientilor cu id-ul: ');
            FOR j in 1..m_recep(i).count LOOP
                dbms_output.put_line(m_recep(i)(j));
            END LOOP;
        END IF;
    END LOOP;

```

```

dbms_output.put_line('Receptionerul care a emis cele
mai multe abonamente este cel cu id-ul: ' ||
calculeaza_maxim_recep);
marire_salariu(calculeaza_maxim_recep);
END afisare_recep;

```

```

FUNCTION calculeaza_maxim_recep
RETURN receptioner.id_angajat%TYPE IS
    maxx NUMBER := 0;
    t vect_recep;
    id_ret receptioner.id_angajat%TYPE;
BEGIN
    SELECT id_angajat
    BULK COLLECT INTO t
    FROM receptioner;

```

```

FOR i IN 1..indice-1 LOOP
    IF m_recep(i).count > maxx THEN

```

```
    maxx := m_recep(i).count;
    id_ret := t(i);
END IF;
END LOOP;
```

```
RETURN id_ret;
END calculeaza_maxim_recep;
```

```
PROCEDURE marire_salariu(id IN
                           receptioner.id_angajat%TYPE) IS
BEGIN
    UPDATE salariu
    SET suma = suma + 0.3 * suma
    WHERE salariu.id_angajat = id;
END marire_salariu;
```

```
PROCEDURE creare_matrice_echip IS
    CURSOR c2 IS
        SELECT nume_companie
        FROM companie_articole_inot;
BEGIN
    FOR i IN c2 LOOP
        m_comp.extend();
        SELECT a.tip_articol,
nvl(a.cantitate_marfa_disponibila, 0)
```

```

        BULK COLLECT INTO m_comp(jndice)
        FROM articol_inot a
        WHERE LOWER(a.brand) =
        LOWER(i.nume_companie);

        jndice := jndice + 1;

    END LOOP;
END creare_matrice_echip;

PROCEDURE afisare_firme IS
    t vect_comp;
    suma NUMBER;
BEGIN
    SELECT id_companie
    BULK COLLECT INTO t
    FROM companie_articole_inot;

    FOR i IN 1..jndice-1 LOOP
        IF m_comp(i).count = 0 THEN
            dbms_output.put_line('Compania cu id-ul ' || t(i) || '
NU A FACUT CONTRACT CU NOI!! ');
        ELSE
            dbms_output.put_line('Compania cu id-ul ' || t(i) || '
a vandut: ');
            suma := 0;
            FOR j in 1..m_comp(i).count LOOP
                dbms_output.put_line(m_comp(i)(j).v_cant || '
de ' || m_comp(i)(j).v_tip);
        END LOOP;
    END IF;
END;

```

```

        suma := suma +
TO_NUMBER(regexp_replace(m_comp(i)(j).v_cant,
'[^0-9]'));
        END LOOP;
    END IF;
    dbms_output.put_line('Compania a trimis ' || suma || '
de produse(slipi, slapi, ochelari etc.)' || chr(10));
    END LOOP;
END afisare_firme;

```

```

PROCEDURE creare_colectie IS
BEGIN
    SELECT c.id_client, COUNT(a.id_abonament)
    BULK COLLECT INTO t_cli
    FROM client c, achizitie_abonament a
    WHERE c.id_client = a.id_client
    GROUP BY c.id_client;
END creare_colectie;

```

```

FUNCTION calculeaza_maxim_cli
RETURN NUMBER IS
    maxx NUMBER := 0;
BEGIN
    creare_colectie;
    FOR i in t_cli.first..t_cli.last LOOP

```

```

        IF maxx < t_cli(i).v_nr THEN
            maxx := t_cli(i).v_nr;
        END IF;
    END LOOP;

    RETURN maxx;
END calculeaza_maxim_cli;

FUNCTION numar_max_cli
RETURN NUMBER IS
    nr NUMBER := 0;
    maxim NUMBER := calculeaza_maxim_cli;
BEGIN
    creare_colectie;
    FOR i in t_cli.first..t_cli.last LOOP
        IF maxim = t_cli(i).v_nr THEN
            nr := nr + 1;
        END IF;
    END LOOP;

    RETURN nr;
END numar_max_cli;

PROCEDURE afisare_clienti IS
BEGIN
    creare_colectie;
    FOR i in t_cli.first..t_cli.last LOOP
        dbms_output.put_line('Clientul cu id-ul ' ||
t_cli(i).v_id || ' a achizitionat ' || t_cli(i).v_nr

```

```

        || ' abonamente.');
    END LOOP;
END afisare_clienti;
END pachet_complex;
/

BEGIN
--  pachet_complex.creare_matrice_recep;
--  pachet_complex.afisare_recep;
--  pachet_complex.creare_matrice_echip;
--  pachet_complex.afisare_firme;
--  pachet_complex.afisare_clienti;
--  dbms_output.put_line('Numarul de clienti care a
achizitionat cele mai multe abonamente este de: '
--      || pachet_complex.numar_max_cli);
END;

```

Screenshot-uri:

```

SQL Worksheet: History
Worksheet | Query Builder | Baza de date locală
1195
1196 BEGIN
1197     pachet_complex.creare_matrice_recep;
1198     pachet_complex.afisare_recep;
1199
1200
1201
1202 Receptionerul cu id-ul 80 a eliberat abonamente clientilor cu id-ul:
1203
1204
1205 Receptionerul cu id-ul 90 a eliberat abonamente clientilor cu id-ul:
1206
1207
1208 Receptionerul cu id-ul 150 NU A ELIBERAT NICIUN ABONAMENT!!
1209 Receptionerul cu id-ul 160 NU A ELIBERAT NICIUN ABONAMENT!!
1210 Receptionerul care a emis cele mai multe abonamente este cel cu id-ul: 60

```

SQL Worksheet | History

Worksheet Query Builder

```

1196      pachet_complex.create_matrice_echip;
1197      pachet_complex.afisare_firme;

```

Script Output | Query Result 1 | Query Result 2 | Task completed in 0.074 seconds

```

Compania cu id-ul 100 a vandut:
70 perechi de Slapi
90 perechi de Ochelari
20 perechi de Slapi
Compania a trimis 180 de produse(slapi, slapi, ochelari etc.)

Compania cu id-ul 200 a vandut:
30 perechi de Slapi
40 perechi de Slapi
Compania a trimis 70 de produse(slapi, slapi, ochelari etc.)

Compania cu id-ul 300 a vandut:
110 perechi de Ochelari
Compania a trimis 110 de produse(slapi, slapi, ochelari etc.)

Compania cu id-ul 400 a vandut:
25 perechi de Casca

```

Worksheet Query Builder

```

1196 BEGIN
1197   -- pachet_complex.creare_matrice_recep;
1198   -- pachet_complex.afisare_recep;
1199   -- pachet_complex.creare_matrice_echip;
1200   -- pachet_complex.afisare_firme;
1201   pachet_complex.afisare_clienti;
1202   dbms_output.put_line('Numarul de clienti care a achizitionat cele mai multe abonamente este de: '
1203                      || pachet_complex.numar_max_cli);
1204 END;
1205
1206 commit;

```

Script Output | Query Result 1 | Query Result 2 | Task completed in 0.072 seconds

```

Clientul cu id-ul 20 a achizitionat 1 abonamente.
Clientul cu id-ul 30 a achizitionat 2 abonamente.
Clientul cu id-ul 40 a achizitionat 3 abonamente.
Clientul cu id-ul 60 a achizitionat 2 abonamente.
Clientul cu id-ul 70 a achizitionat 2 abonamente.
Clientul cu id-ul 100 a achizitionat 2 abonamente.
Clientul cu id-ul 120 a achizitionat 2 abonamente.
Numarul de clienti care a achizitionat cele mai multe abonamente este de: 1

```