Proiect Sisteme de Gestiune a Bazelor de Date

Bază de date pentru o companie aeriană

Nume student: Hodivoianu Anamaria

Grupă: 232

Cuprins

[1. Prezentarea bazei de date 2](#_Toc1728952519)

[2. Diagrama entitate-relație 2](#_Toc1894048925)

[3. Diagrama conceptuală 3](#_Toc315597793)

[4. Crearea tabelelor 4](#_Toc1739924857)

[5. Inserarea datelor 13](#_Toc325095394)

[6. Colecții 44](#_Toc828568371)

[7. Cursoare 48](#_Toc1816169129)

[8. Funcție 51](#_Toc265597719)

[9. Procedură 57](#_Toc1901209591)

[10. Trigger LMD la nivel de comandă 64](#_Toc936692341)

[11. Trigger LMD la nivel de linie 64](#_Toc1498275869)

[12. Trigger LDD 68](#_Toc487150810)

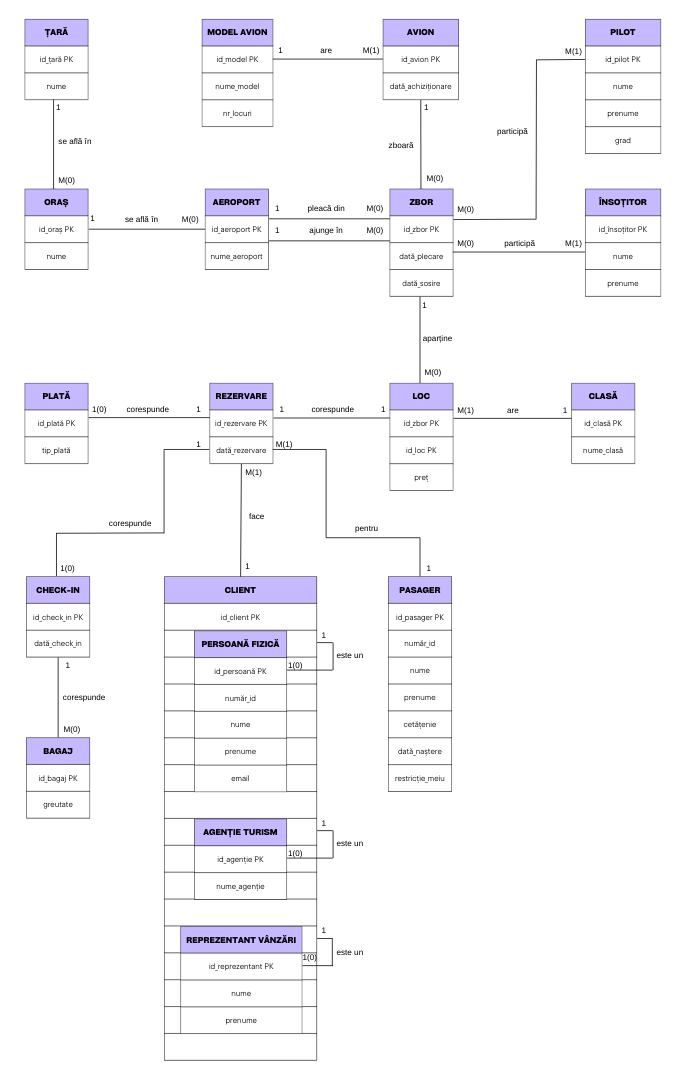
[13. Pachet obiecte proiect 70](#_Toc205289177)

[14. Pachet flux de acțiuni integrate 81](#_Toc1595006141)

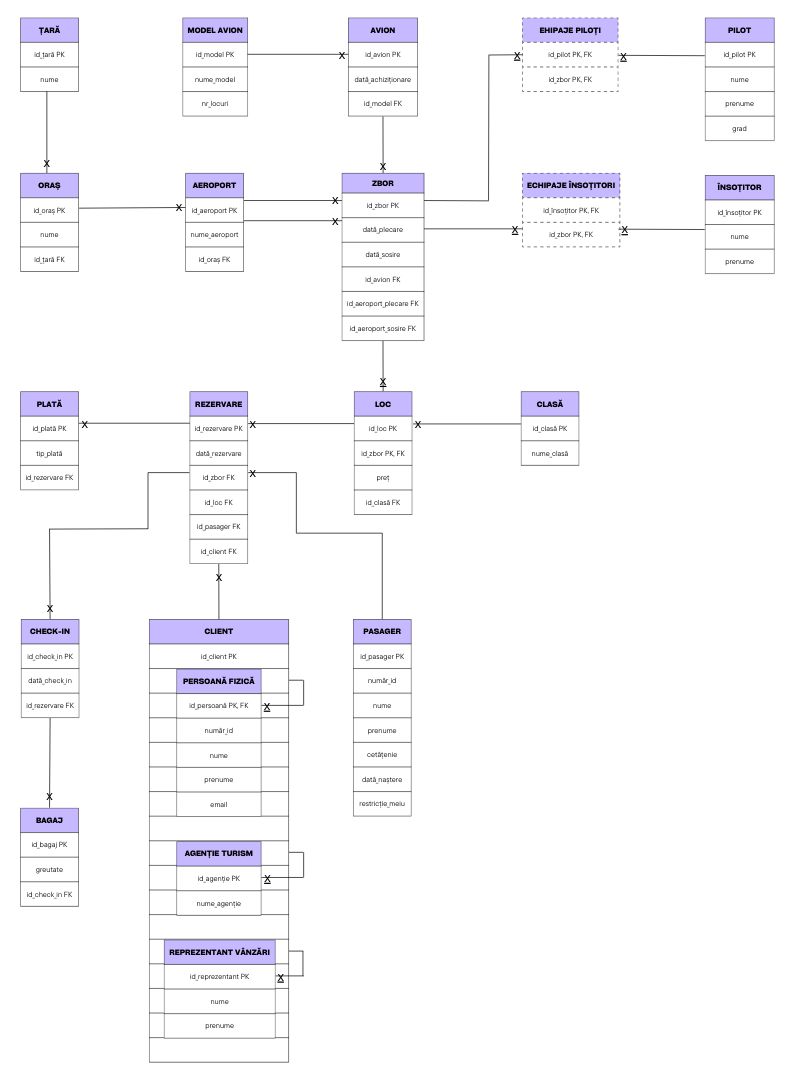
# Prezentarea bazei de date

Acest model descrie o bază de date pentru o companie aeriană. Compania aeriană deține avioane și efectuează zboruri în care vinde locuri. Fiecare avion este de un anumit model. Zborurile pleacă din și ajung în aeroporturi. Un aeroport se află într-un oraș, care se află într-o țară. La zboruri participă piloți și însoțitori de zbor. Un loc într-un zbor este rezervat de către un client (care poate fi o persoană fizică, o agenție de turism sau un reprezentant de vânzări). Locul este rezervat pentru un pasager. O rezervare este pentru un singur loc, pentru un singur pasager (pentru două locuri trebuie două rezervări). Pasagerii care au o rezervare și se prezintă la zbor trec prin check-in. La check-in își lasă bagajul / bagajele de cală.

# Diagrama entitate-relație



# Diagrama conceptuală



# Crearea tabelelor

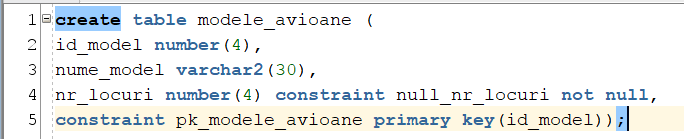
* Modele avioane

create table modele\_avioane (

id\_model number(4),

nume\_model varchar2(30),

nr\_locuri number(4) constraint null\_nr\_locuri not null,

constraint pk\_modele\_avioane primary key(id\_model));

* Avioane

create table avioane (

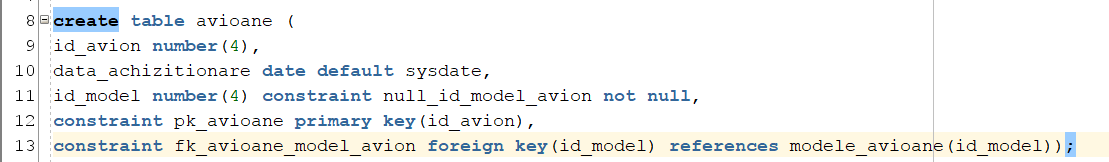
id\_avion number(4),

data\_achizitionare date default sysdate,

id\_model number(4) constraint null\_id\_model\_avion not null,

constraint pk\_avioane primary key(id\_avion),

constraint fk\_avioane\_model\_avion foreign key(id\_model) references modele\_avioane(id\_model));



* Țări

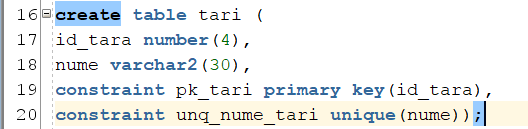
create table tari (

id\_tara number(4),

nume varchar2(30),

constraint pk\_tari primary key(id\_tara),

constraint unq\_nume\_tari unique(nume));



* Orașe

create table orase (

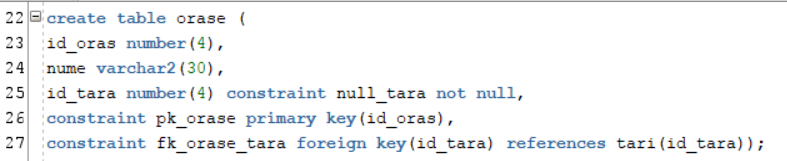
id\_oras number(4),

nume varchar2(30),

id\_tara number(4) constraint null\_tara not null,

constraint pk\_orase primary key(id\_oras),

constraint fk\_orase\_tara foreign key(id\_tara) references tari(id\_tara));



* Aeroporturi

create table aeroporturi (

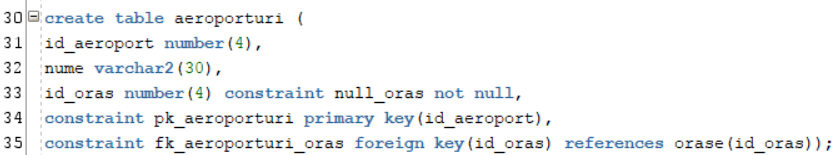
id\_aeroport number(4),

nume varchar2(30),

id\_oras number(4) constraint null\_oras not null,

constraint pk\_aeroporturi primary key(id\_aeroport),

constraint fk\_aeroporturi\_oras foreign key(id\_oras) references orase(id\_oras));



* Zboruri

create table zboruri (

id\_zbor number(4),

data\_plecare date,

data\_sosire date,

id\_avion number(4) constraint null\_avion not null,

id\_aeroport\_plecare number(4) constraint null\_plecare not null,

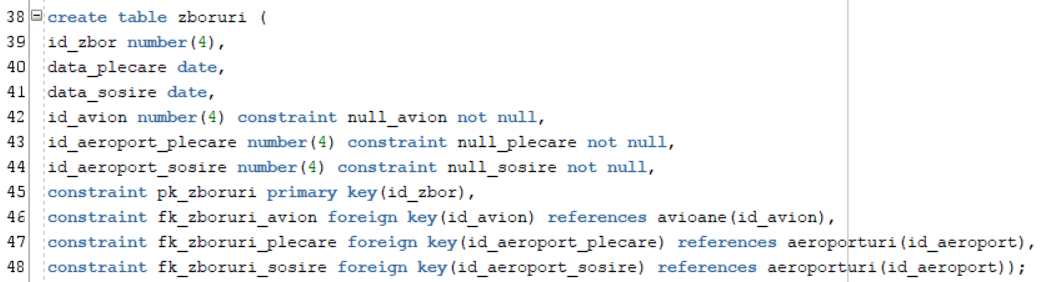
id\_aeroport\_sosire number(4) constraint null\_sosire not null,

constraint pk\_zboruri primary key(id\_zbor),

constraint fk\_zboruri\_avion foreign key(id\_avion) references avioane(id\_avion),

constraint fk\_zboruri\_plecare foreign key(id\_aeroport\_plecare) references aeroporturi(id\_aeroport),

constraint fk\_zboruri\_sosire foreign key(id\_aeroport\_sosire) references aeroporturi(id\_aeroport));



* Piloți

create table piloti (

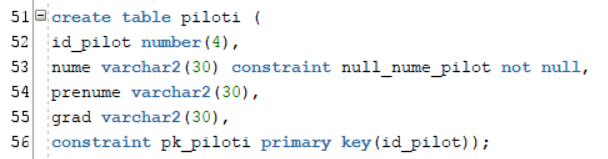
id\_pilot number(4),

nume varchar2(30) constraint null\_nume\_pilot not null,

prenume varchar2(30),

grad varchar2(30),

constraint pk\_piloti primary key(id\_pilot));



* Insoțitori zbor

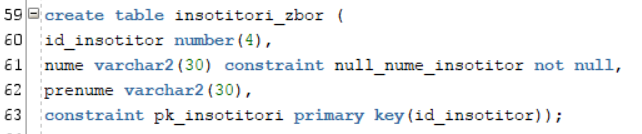
create table insotitori\_zbor (

id\_insotitor number(4),

nume varchar2(30) constraint null\_nume\_insotitor not null,

prenume varchar2(30),

constraint pk\_insotitori primary key(id\_insotitor));



* Echipaje piloți

create table echipaje\_piloti (

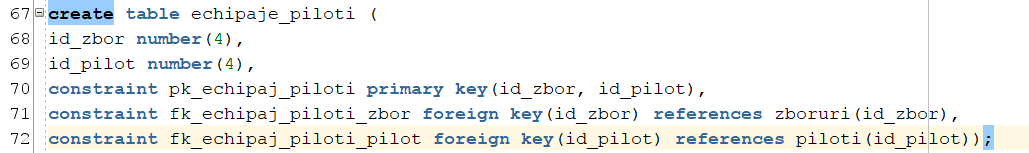
id\_zbor number(4),

id\_pilot number(4),

constraint pk\_echipaj\_piloti primary key(id\_zbor, id\_pilot),

constraint fk\_echipaj\_piloti\_zbor foreign key(id\_zbor) references zboruri(id\_zbor),

constraint fk\_echipaj\_piloti\_pilot foreign key(id\_pilot) references piloti(id\_pilot));



* Echipaje însoțitori

create table echipaje\_insotitori (

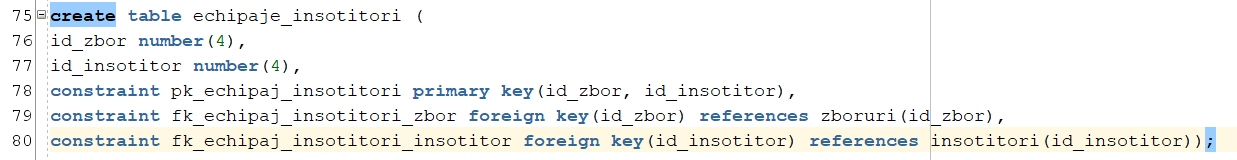
id\_zbor number(4),

id\_insotitor number(4),

constraint pk\_echipaj\_insotitori primary key(id\_zbor, id\_insotitor),

constraint fk\_echipaj\_insotitori\_zbor foreign key(id\_zbor) references zboruri(id\_zbor),

constraint fk\_echipaj\_insotitori\_insotitor foreign key(id\_insotitor) references insotitori(id\_insotitor));



* Clase

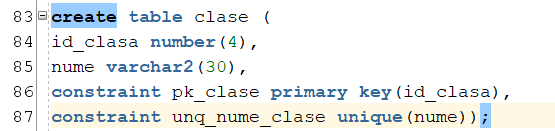
create table clase (

id\_clasa number(4),

nume varchar2(30),

constraint pk\_clase primary key(id\_clasa),

constraint unq\_nume\_clase unique(nume));



* Locuri

create table locuri (

id\_loc number(4),

id\_zbor number(4),

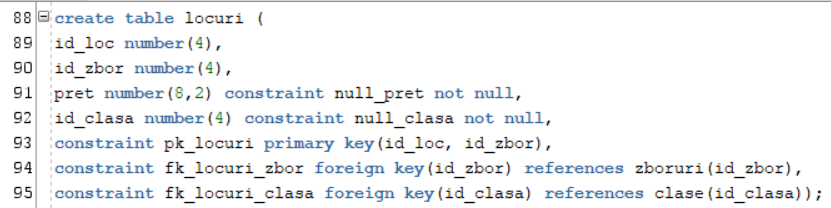
pret number(8,2) constraint null\_pret not null,

id\_clasa number(4) constraint null\_clasa not null,

constraint pk\_locuri primary key(id\_loc, id\_zbor),

constraint fk\_locuri\_zbor foreign key(id\_zbor) references zboruri(id\_zbor),

constraint fk\_locuri\_clasa foreign key(id\_clasa) references clase(id\_clasa));



* Pasageri

create table pasageri (

id\_pasager number(4),

numar\_id varchar2(20) constraint null\_nr\_id\_pasageri not null,

nume varchar2(30) constraint null\_nume\_pasageri not null,

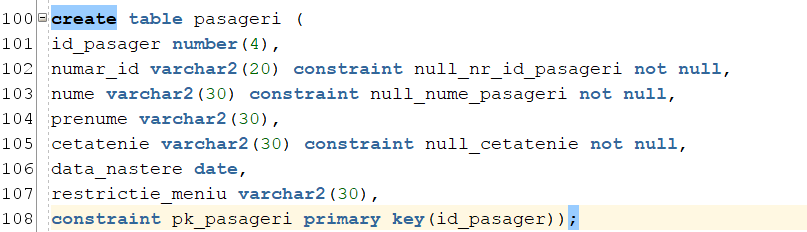
prenume varchar2(30),

cetatenie varchar2(30) constraint null\_cetatenie not null,

data\_nastere date,

restrictie\_meniu varchar2(30),

constraint pk\_pasageri primary key(id\_pasager));

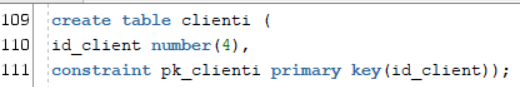


* Clienți

create table clienti (

id\_client number(4),

constraint pk\_clienti primary key(id\_client));



* Persoane fizice

create table persoane\_fizice (

id\_persoana number(4),

numar\_id varchar2(30) constraint null\_nr\_id\_pers not null,

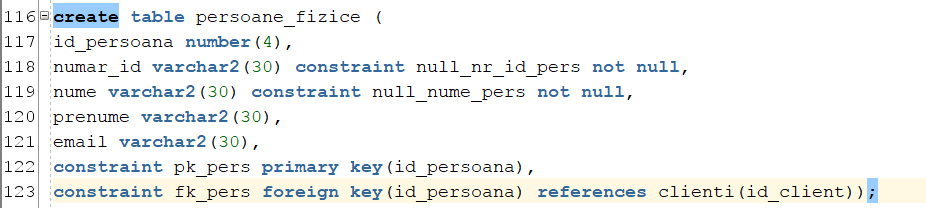
nume varchar2(30) constraint null\_nume\_pers not null,

prenume varchar2(30),

email varchar2(30),

constraint pk\_pers primary key(id\_persoana),

constraint fk\_pers foreign key(id\_persoana) references clienti(id\_client));



* Agenții turism

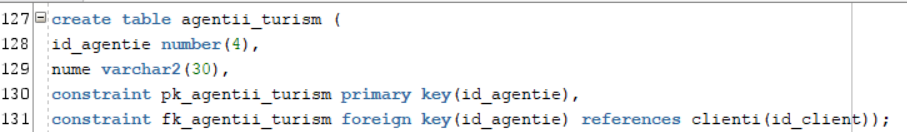
create table agentii\_turism (

id\_agentie number(4),

nume varchar2(30),

constraint pk\_agentii\_turism primary key(id\_agentie),

constraint fk\_agentii\_turism foreign key(id\_agentie) references clienti(id\_client));



* Reprezentanți vânzări

create table reprezentanti\_vanzari (

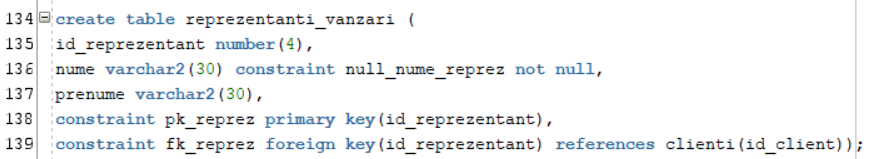
id\_reprezentant number(4),

nume varchar2(30) constraint null\_nume\_reprez not null,

prenume varchar2(30),

constraint pk\_reprez primary key(id\_reprezentant),

constraint fk\_reprez foreign key(id\_reprezentant) references clienti(id\_client));



* Rezervări

create table rezervari (

id\_rezervare number(4),

id\_loc number(4) constraint null\_loc not null,

id\_zbor number(4) constraint null\_zbor not null,

id\_pasager number(4) constraint null\_pasager not null,

id\_client number(4) constraint null\_client not null,

data\_rezervare date default sysdate constraint null\_data not null,

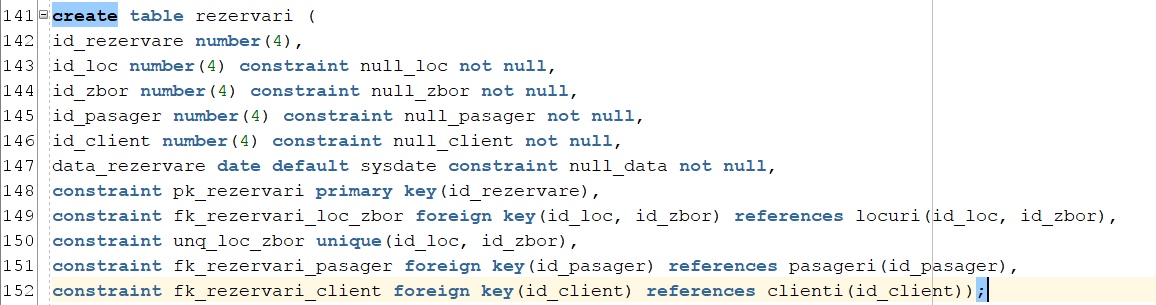
constraint pk\_rezervari primary key(id\_rezervare),

constraint fk\_rezervari\_loc\_zbor foreign key(id\_loc, id\_zbor) references locuri(id\_loc, id\_zbor),

constraint unq\_loc\_zbor unique(id\_loc, id\_zbor),

constraint fk\_rezervari\_pasager foreign key(id\_pasager) references pasageri(id\_pasager),

constraint fk\_rezervari\_client foreign key(id\_client) references clienti(id\_client));



* Plăți

create table plati (

id\_plata number(4),

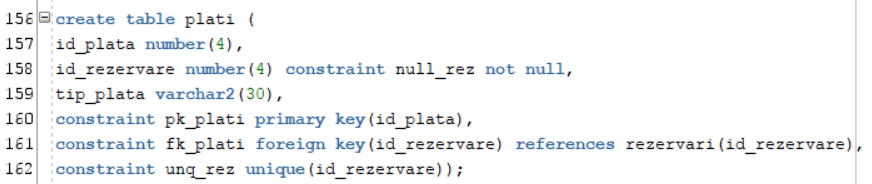
id\_rezervare number(4) constraint null\_rez not null,

tip\_plata varchar2(30),

constraint pk\_plati primary key(id\_plata),

constraint fk\_plati foreign key(id\_rezervare) references rezervari(id\_rezervare),

constraint unq\_rez unique(id\_rezervare));



* Check-in

create table check\_in (

id\_check\_in number(4),

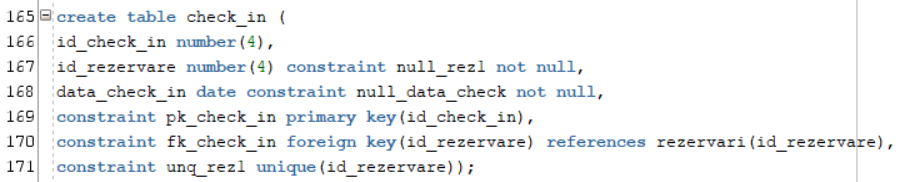
id\_rezervare number(4) constraint null\_rez1 not null,

data\_check\_in date constraint null\_data\_check not null,

constraint pk\_check\_in primary key(id\_check\_in),

constraint fk\_check\_in foreign key(id\_rezervare) references rezervari(id\_rezervare),

constraint unq\_rez1 unique(id\_rezervare));



* Bagaje

create table bagaje (

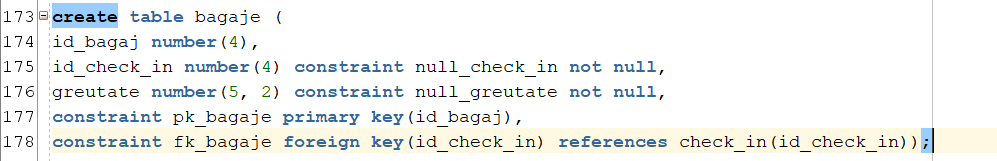
id\_bagaj number(4),

id\_check\_in number(4) constraint null\_check\_in not null,

greutate number(5, 2) constraint null\_greutate not null,

constraint pk\_bagaje primary key(id\_bagaj),

constraint fk\_bagaje foreign key(id\_check\_in) references check\_in(id\_check\_in));



# Inserarea datelor

* Modele avioane

create sequence secv\_modele\_avioane

start with 0

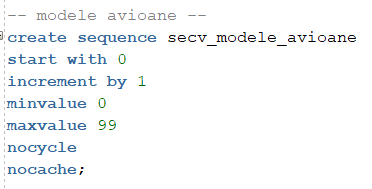
increment by 1

minvalue 0

maxvalue 99

nocycle

nocache;



insert into modele\_avioane

values (secv\_modele\_avioane.nextval, 'Boeing 777', 12);

insert into modele\_avioane

values (secv\_modele\_avioane.nextval, 'Airbus A340', 15);

insert into modele\_avioane

values (secv\_modele\_avioane.nextval, 'Airbus A380', 20);

insert into modele\_avioane

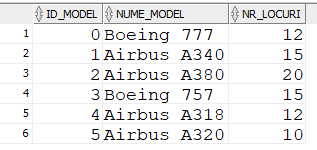
values (secv\_modele\_avioane.nextval, 'Boeing 757', 15);

insert into modele\_avioane

values (secv\_modele\_avioane.nextval, 'Airbus A318', 12);

insert into modele\_avioane

values (secv\_modele\_avioane.nextval, 'Airbus A320', 10);



* Avioane

create sequence secv\_avioane

start with 0

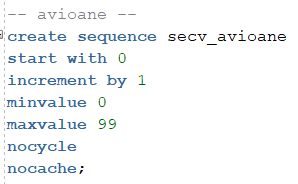
increment by 1

minvalue 0

maxvalue 99

nocycle

nocache;



insert into avioane

values (secv\_avioane.nextval, '08-jan-19', 3);

insert into avioane

values (secv\_avioane.nextval, '12-mar-15', 0);

insert into avioane

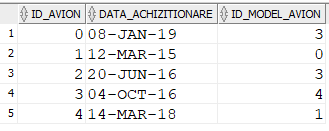
values (secv\_avioane.nextval, '20-jun-16', 3);

insert into avioane

values (secv\_avioane.nextval, '04-oct-16', 4);

insert into avioane

values (secv\_avioane.nextval, '14-mar-18', 1);



* Țări

insert into tari

values (0, 'Romania');

insert into tari

values (1, 'Germania');

insert into tari

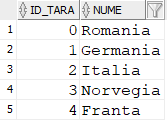
values (2, 'Italia');

insert into tari

values (3, 'Norvegia');

insert into tari

values (4, 'Franta');



* Orașe

insert into orase

values (0, 'Bucuresti', 0);

insert into orase

values (1, 'Cluj-Napoca', 0);

insert into orase

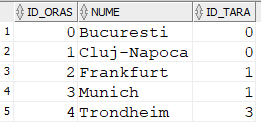
values (2, 'Frankfurt', 1);

insert into orase

values (3, 'Munich', 1);

insert into orase

values (4, 'Trondheim', 3);



* Aeroporturi

create sequence secv\_aeroporturi

start with 0

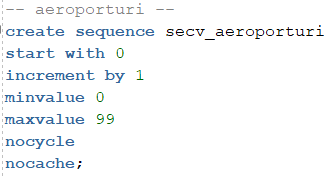
increment by 1

minvalue 0

maxvalue 99

nocycle

nocache;



insert into aeroporturi

values (secv\_aeroporturi.nextval, 'Henri Coanda', 0);

insert into aeroporturi

values (secv\_aeroporturi.nextval, 'Aurel Vlaicu', 0);

insert into aeroporturi

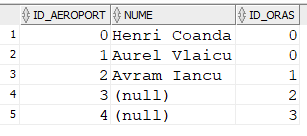
values (secv\_aeroporturi.nextval, 'Avram Iancu', 1);

insert into aeroporturi

values (secv\_aeroporturi.nextval, null, 2);

insert into aeroporturi

values (secv\_aeroporturi.nextval, null, 3);



* Zboruri

create sequence secv\_zboruri

start with 0

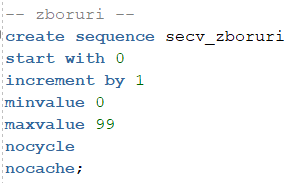
increment by 1

minvalue 0

maxvalue 99

nocycle

nocache;



insert into zboruri

values (secv\_zboruri.nextval, to\_date('2019-oct-12 15:30', 'yyyy-mon-dd hh24:mi'), to\_date('2019-oct-12 17:30', 'yyyy-mon-dd hh24:mi'), 1, 0, 3);

insert into zboruri

values (secv\_zboruri.nextval, to\_date('2019-dec-16 14:40', 'yyyy-mon-dd hh24:mi'), to\_date('2019-dec-16 15:20', 'yyyy-mon-dd hh24:mi'), 0, 1, 2);

insert into zboruri

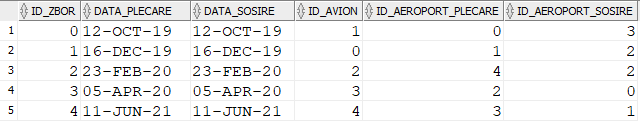
values (secv\_zboruri.nextval, to\_date('2020-feb-23 10:30', 'yyyy-mon-dd hh24:mi'), to\_date('2020-feb-23 12:15', 'yyyy-mon-dd hh24:mi'), 2, 4, 2);

insert into zboruri

values (secv\_zboruri.nextval, to\_date('2020-apr-5 20:00', 'yyyy-mon-dd hh24:mi'), to\_date('2020-apr-5 21:30', 'yyyy-mon-dd hh24:mi'), 3, 2, 0);

insert into zboruri

values (secv\_zboruri.nextval, to\_date('2021-jun-11 18:20', 'yyyy-mon-dd hh24:mi'), to\_date('2021-jun-11 20:00', 'yyyy-mon-dd hh24:mi'), 4, 3, 1);



* Piloți

create sequence secv\_piloti

start with 0

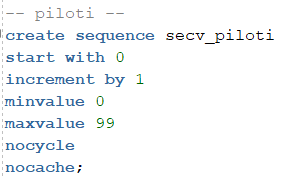
increment by 1

minvalue 0

maxvalue 99

nocycle

nocache;



insert into piloti

values (secv\_piloti.nextval, 'Albescu', 'Delia', 'pilot');

insert into piloti

values (secv\_piloti.nextval, 'Lupei', 'Dan', 'pilot');

insert into piloti

values (secv\_piloti.nextval, 'Albert', 'Andrei', 'copilot');

insert into piloti

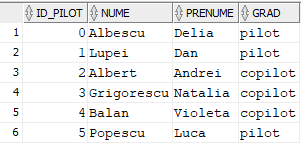
values (secv\_piloti.nextval, 'Grigorescu', 'Natalia', 'copilot');

insert into piloti

values (secv\_piloti.nextval, 'Balan', 'Violeta', 'copilot');

insert into piloti

values (secv\_piloti.nextval, 'Popescu', 'Luca', 'pilot');



* Insoțitori zbor

create sequence secv\_insotitori

start with 0

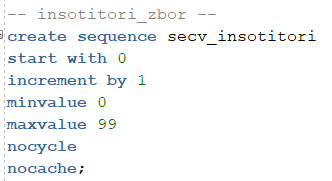
increment by 1

minvalue 0

maxvalue 99

nocycle

nocache;



insert into insotitori

values (secv\_insotitori.nextval, 'Vasile', 'Livia');

insert into insotitori

values (secv\_insotitori.nextval, 'Solomon', 'Isabella');

insert into insotitori

values (secv\_insotitori.nextval, 'Popescu', 'Ruxandra');

insert into insotitori

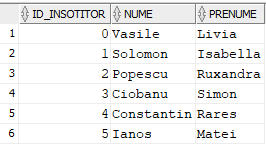
values (secv\_insotitori.nextval, 'Ciobanu', 'Simon');

insert into insotitori

values (secv\_insotitori.nextval, 'Constantin', 'Rares');

insert into insotitori

values (secv\_insotitori.nextval, 'Ianos', 'Matei');



* Echipaje piloți

insert into echipaje\_piloti

values (0, 0);

insert into echipaje\_piloti

values (0, 2);

insert into echipaje\_piloti

values (1, 1);

insert into echipaje\_piloti

values (1, 2);

insert into echipaje\_piloti

values (2, 1);

insert into echipaje\_piloti

values (2, 3);

insert into echipaje\_piloti

values (3, 0);

insert into echipaje\_piloti

values (3, 4);

insert into echipaje\_piloti

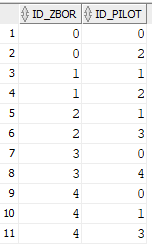
values (4, 0);

insert into echipaje\_piloti

values (4, 3);

insert into echipaje\_piloti

values (4, 1);



* Echipaje însoțitori

insert into echipaje\_insotitori

values (0, 0);

insert into echipaje\_insotitori

values (1, 2);

insert into echipaje\_insotitori

values (1, 3);

insert into echipaje\_insotitori

values (1, 0);

insert into echipaje\_insotitori

values (2, 4);

insert into echipaje\_insotitori

values (2, 5);

insert into echipaje\_insotitori

values (3, 1);

insert into echipaje\_insotitori

values (3, 2);

insert into echipaje\_insotitori

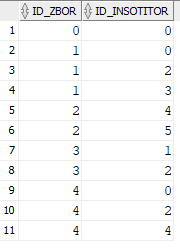
values (4, 0);

insert into echipaje\_insotitori

values (4, 2);

insert into echipaje\_insotitori

values (4, 4);



* Clase

insert into clase

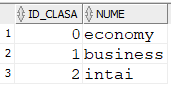
values (0, 'economy');

insert into clase

values (1, 'business');

insert into clase

values (2, 'intai');



* Locuri

-- zbor 0 - avion 1 - tip avion 0 - 10 locuri

insert into locuri(id\_loc, pret, id\_clasa, id\_zbor)

values (0, 40, 0, 0);

insert into locuri(id\_loc, pret, id\_clasa, id\_zbor)

values (1, 40, 0, 0);

insert into locuri(id\_loc, pret, id\_clasa, id\_zbor)

values (2, 50, 0, 0);

insert into locuri(id\_loc, pret, id\_clasa, id\_zbor)

values (3, 50, 0, 0);

insert into locuri(id\_loc, pret, id\_clasa, id\_zbor)

values (4, 50, 0, 0);

insert into locuri(id\_loc, pret, id\_clasa, id\_zbor)

values (5, 100, 1, 0);

insert into locuri(id\_loc, pret, id\_clasa, id\_zbor)

values (6, 100, 1, 0);

insert into locuri(id\_loc, pret, id\_clasa, id\_zbor)

values (7, 100, 1, 0);

insert into locuri(id\_loc, pret, id\_clasa, id\_zbor)

values (8, 110, 1, 0);

insert into locuri(id\_loc, pret, id\_clasa, id\_zbor)

values (9, 110, 1, 0);

-- zbor 1 - avion 0 - tip avion 3 - 12 locuri

insert into locuri(id\_loc, pret, id\_clasa, id\_zbor)

values (0, 40, 0, 1);

insert into locuri(id\_loc, pret, id\_clasa, id\_zbor)

values (1, 40, 0, 1);

insert into locuri(id\_loc, pret, id\_clasa, id\_zbor)

values (2, 40, 0, 1);

insert into locuri(id\_loc, pret, id\_clasa, id\_zbor)

values (3, 40, 0, 1);

insert into locuri(id\_loc, pret, id\_clasa, id\_zbor)

values (4, 45, 0, 1);

insert into locuri(id\_loc, pret, id\_clasa, id\_zbor)

values (5, 45, 0, 1);

insert into locuri(id\_loc, pret, id\_clasa, id\_zbor)

values (6, 90, 1, 1);

insert into locuri(id\_loc, pret, id\_clasa, id\_zbor)

values (7, 90, 1, 1);

insert into locuri(id\_loc, pret, id\_clasa, id\_zbor)

values (8, 95, 1, 1);

insert into locuri(id\_loc, pret, id\_clasa, id\_zbor)

values (9, 95, 1, 1);

insert into locuri(id\_loc, pret, id\_clasa, id\_zbor)

values (10, 200, 2, 1);

insert into locuri(id\_loc, pret, id\_clasa, id\_zbor)

values (11, 200, 2, 1);

-- zbor 2 - avion 2 - tip avion 3 - 12 locuri

insert into locuri(id\_loc, pret, id\_clasa, id\_zbor)

values (0, 60, 0, 2);

insert into locuri(id\_loc, pret, id\_clasa, id\_zbor)

values (1, 60, 0, 2);

insert into locuri(id\_loc, pret, id\_clasa, id\_zbor)

values (2, 60, 0, 2);

insert into locuri(id\_loc, pret, id\_clasa, id\_zbor)

values (3, 60, 0, 2);

insert into locuri(id\_loc, pret, id\_clasa, id\_zbor)

values (4, 70, 0, 2);

insert into locuri(id\_loc, pret, id\_clasa, id\_zbor)

values (5, 70, 0, 2);

insert into locuri(id\_loc, pret, id\_clasa, id\_zbor)

values (6, 180, 1, 2);

insert into locuri(id\_loc, pret, id\_clasa, id\_zbor)

values (7, 180, 1, 2);

insert into locuri(id\_loc, pret, id\_clasa, id\_zbor)

values (8, 180, 1, 2);

insert into locuri(id\_loc, pret, id\_clasa, id\_zbor)

values (9, 190, 1, 2);

insert into locuri(id\_loc, pret, id\_clasa, id\_zbor)

values (10, 350, 2, 2);

insert into locuri(id\_loc, pret, id\_clasa, id\_zbor)

values (11, 360, 2, 2);

-- zbor 3 - avion 3 - tip avion 4 - 15 locuri

insert into locuri(id\_loc, pret, id\_clasa, id\_zbor)

values (0, 35, 0, 3);

insert into locuri(id\_loc, pret, id\_clasa, id\_zbor)

values (1, 35, 0, 3);

insert into locuri(id\_loc, pret, id\_clasa, id\_zbor)

values (2, 35, 0, 3);

insert into locuri(id\_loc, pret, id\_clasa, id\_zbor)

values (3, 35, 0, 3);

insert into locuri(id\_loc, pret, id\_clasa, id\_zbor)

values (4, 35, 0, 3);

insert into locuri(id\_loc, pret, id\_clasa, id\_zbor)

values (5, 40, 0, 3);

insert into locuri(id\_loc, pret, id\_clasa, id\_zbor)

values (6, 40, 0, 3);

insert into locuri(id\_loc, pret, id\_clasa, id\_zbor)

values (7, 40, 0, 3);

insert into locuri(id\_loc, pret, id\_clasa, id\_zbor)

values (8, 70, 1, 3);

insert into locuri(id\_loc, pret, id\_clasa, id\_zbor)

values (9, 70, 1, 3);

insert into locuri(id\_loc, pret, id\_clasa, id\_zbor)

values (10, 70, 1, 3);

insert into locuri(id\_loc, pret, id\_clasa, id\_zbor)

values (11, 80, 1, 3);

insert into locuri(id\_loc, pret, id\_clasa, id\_zbor)

values (12, 180, 2, 3);

insert into locuri(id\_loc, pret, id\_clasa, id\_zbor)

values (13, 180, 2, 3);

insert into locuri(id\_loc, pret, id\_clasa, id\_zbor)

values (14, 190, 2, 3);

-- zbor 4 - avion 4 - tip avion 1 - 20 locuri

insert into locuri(id\_loc, pret, id\_clasa, id\_zbor)

values (0, 40, 0, 4);

insert into locuri(id\_loc, pret, id\_clasa, id\_zbor)

values (1, 40, 0, 4);

insert into locuri(id\_loc, pret, id\_clasa, id\_zbor)

values (2, 40, 0, 4);

insert into locuri(id\_loc, pret, id\_clasa, id\_zbor)

values (3, 40, 0, 4);

insert into locuri(id\_loc, pret, id\_clasa, id\_zbor)

values (4, 40, 0, 4);

insert into locuri(id\_loc, pret, id\_clasa, id\_zbor)

values (5, 40, 0, 4);

insert into locuri(id\_loc, pret, id\_clasa, id\_zbor)

values (6, 40, 0, 4);

insert into locuri(id\_loc, pret, id\_clasa, id\_zbor)

values (7, 45, 0, 4);

insert into locuri(id\_loc, pret, id\_clasa, id\_zbor)

values (8, 45, 0, 4);

insert into locuri(id\_loc, pret, id\_clasa, id\_zbor)

values (9, 45, 0, 4);

insert into locuri(id\_loc, pret, id\_clasa, id\_zbor)

values (10, 100, 1, 4);

insert into locuri(id\_loc, pret, id\_clasa, id\_zbor)

values (11, 100, 1, 4);

insert into locuri(id\_loc, pret, id\_clasa, id\_zbor)

values (12, 100, 1, 4);

insert into locuri(id\_loc, pret, id\_clasa, id\_zbor)

values (13, 110, 1, 4);

insert into locuri(id\_loc, pret, id\_clasa, id\_zbor)

values (14, 110, 1, 4);

insert into locuri(id\_loc, pret, id\_clasa, id\_zbor)

values (15, 300, 2, 4);

insert into locuri(id\_loc, pret, id\_clasa, id\_zbor)

values (16, 300, 2, 4);

insert into locuri(id\_loc, pret, id\_clasa, id\_zbor)

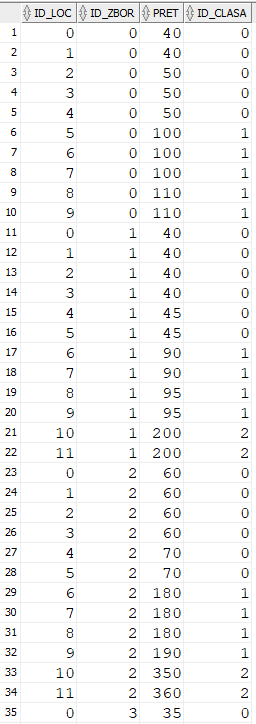
values (17, 300, 2, 4);

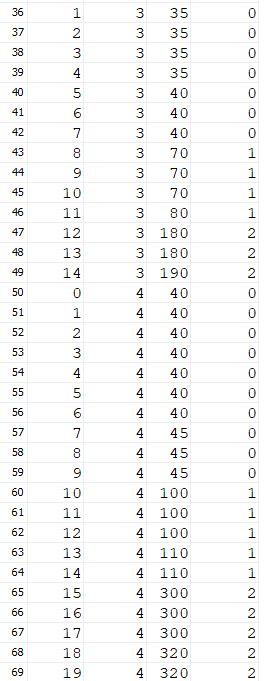
insert into locuri(id\_loc, pret, id\_clasa, id\_zbor)

values (18, 320, 2, 4);

insert into locuri(id\_loc, pret, id\_clasa, id\_zbor)

values (19, 320, 2, 4);





* Pasageri

insert into pasageri

values (0, '2940619092726', 'Cojocaru', 'Bianca', 'Romania', '19-jun-1994', null);

insert into pasageri

values (1, '2911119199676', 'Petrescu', 'Mihaela', 'Romania', '19-nov-1991', null);

insert into pasageri

values (2, '6040302227314', 'Avram', 'Emma', 'Romania', '02-mar-2004', 'vegetarian');

insert into pasageri

values (3, '6030321346602', 'Marin', 'Laura', 'Romania', '21-mar-2003', null);

insert into pasageri

values (4, '1951017453233', 'Dumitru', 'Mihai', 'Romania', '17-oct-1995', 'diabet');

insert into pasageri

values (5, '1940706334322', 'Neagu', 'Teodor', 'Romania', '06-jul-1994', null);

insert into pasageri

values (6, '5030309072901', 'Aliu', 'Matei', 'Romania', '09-mar-2003', null);

insert into pasageri

values (7, '5959JJNLC2', 'Rosenberger', 'Ivonne', 'Germania', '11-apr-1992', null);

insert into pasageri

values (8, '1327M3YY51', 'Kuntz', 'Hilbert', 'Germania', '23-oct-1992', null);

insert into pasageri

values (9, 'F8624KW3J6', 'Leitz', 'Josef', 'Germania', '01-aug-2005', 'vegetarian');

insert into pasageri

values (10, 'GE3082692', 'Trevisani', 'Damiana', 'Italia', '02-jan-1981', null);

insert into pasageri

values (11, 'OY37373564', 'Milani', 'Uranio', 'Italia', '30-nov-1996', null);

insert into pasageri

values (12, '6000130251658', 'Pop', 'Viorela', 'Romania', '30-jan-2000', null);

insert into pasageri

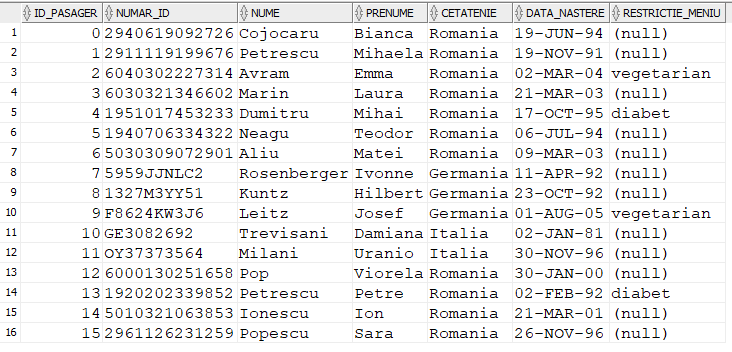
values (13, '1920202339852', 'Petrescu', 'Petre', 'Romania', '02-feb-1992', 'diabet');

insert into pasageri

values (14, '5010321063853', 'Ionescu', 'Ion', 'Romania', '21-mar-2001', null);

insert into pasageri

values (15, '2961126231259', 'Popescu', 'Sara', 'Romania', '26-nov-1996', null);



* Clienți

create sequence secv\_clienti

start with 0

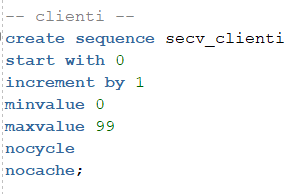
increment by 1

minvalue 0

maxvalue 99

nocycle

nocache;



insert into clienti

values (secv\_clienti.nextval);

insert into clienti

values (secv\_clienti.nextval);

insert into clienti

values (secv\_clienti.nextval);

insert into clienti

values (secv\_clienti.nextval);

insert into clienti

values (secv\_clienti.nextval);

insert into clienti

values (secv\_clienti.nextval);

insert into clienti

values (secv\_clienti.nextval);

insert into clienti

values (secv\_clienti.nextval);

insert into clienti

values (secv\_clienti.nextval);

insert into clienti

values (secv\_clienti.nextval);

insert into clienti

values (secv\_clienti.nextval);

insert into clienti

values (secv\_clienti.nextval);

insert into clienti

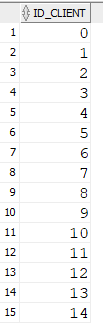
values (secv\_clienti.nextval);

insert into clienti

values (secv\_clienti.nextval);

insert into clienti

values (secv\_clienti.nextval);



* Persoane fizice

insert into persoane\_fizice

values (0, '6000130251658', 'Pop', 'Viorela', 'pop.viorela@gmail.com');

insert into persoane\_fizice

values (1, '1920202339852', 'Petrescu', 'Petre', 'petrescu.petre@gmail.com');

insert into persoane\_fizice

values (2, '5010321063853', 'Ionescu', 'Ion', 'ionescu.ion@gmail.com');

insert into persoane\_fizice

values (3, '2961126231259', 'Popescu', 'Sara', 'popescu.sara@gmail.com');

insert into persoane\_fizice

values (7, '2940619092726', 'Cojocaru', 'Bianca', 'cojocaru.bianca@gmail.com');

insert into persoane\_fizice

values (8, '2911119199676', 'Petrescu', 'Mihaela', 'petrescu.mihaela@gmail.com');

insert into persoane\_fizice

values (9, '6040302227314', 'Avram', 'Emma', 'avram.emma@gmail.com');

insert into persoane\_fizice

values (10, '6030321346602', 'Marin', 'Laura', 'marin.laura@gmail.com');

insert into persoane\_fizice

values (11, '1951017453233', 'Dumitru', 'Mihai', 'dumitru.mihai@gmail.com');

insert into persoane\_fizice

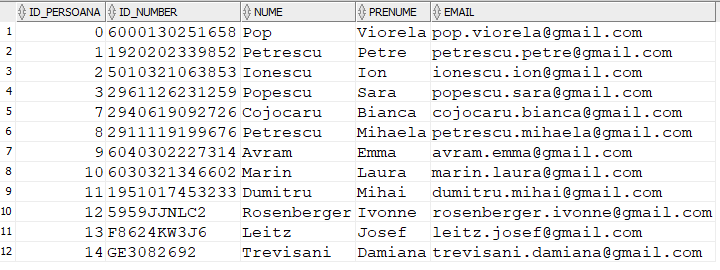
values (12, '5959JJNLC2', 'Rosenberger', 'Ivonne', 'rosenberger.ivonne@gmail.com');

insert into persoane\_fizice

values (13, 'F8624KW3J6', 'Leitz', 'Josef', 'leitz.josef@gmail.com');

insert into persoane\_fizice

values (14, 'GE3082692', 'Trevisani', 'Damiana', 'trevisani.damiana@gmail.com');



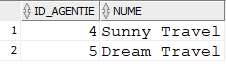
* Agenții turism

insert into agentii\_turism

values (4, 'Sunny Travel');

insert into agentii\_turism

values (5, 'Dream Travel');



* Reprezentanți vânzări

insert into reprezentanti\_vanzari

values (6, 'Stan', 'Carmen');



* Rezervări

-- zbor 0

insert into rezervari(id\_rezervare, data\_rezervare, id\_loc, id\_zbor, id\_pasager, id\_client)

values (0, to\_date('2019-oct-05 15:00', 'yyyy-mon-dd hh24:mi'), 3, 0, 15, 3);

insert into rezervari(id\_rezervare, data\_rezervare, id\_loc, id\_zbor, id\_pasager, id\_client)

values (1, to\_date('2019-sep-12 10:40', 'yyyy-mon-dd hh24:mi'), 5, 0, 14, 2);

insert into rezervari(id\_rezervare, data\_rezervare, id\_loc, id\_zbor, id\_pasager, id\_client)

values (2, to\_date('2019-aug-03 19:50', 'yyyy-mon-dd hh24:mi'), 9, 0, 13, 1);

insert into rezervari(id\_rezervare, data\_rezervare, id\_loc, id\_zbor, id\_pasager, id\_client)

values (3, to\_date('2019-oct-10 23:20', 'yyyy-mon-dd hh24:mi'), 2, 0, 12, 0);

insert into rezervari(id\_rezervare, data\_rezervare, id\_loc, id\_zbor, id\_pasager, id\_client)

values (4, to\_date('2019-sep-24 04:10', 'yyyy-mon-dd hh24:mi'), 4, 0, 10, 4);

-- zbor 1

insert into rezervari(id\_rezervare, data\_rezervare, id\_loc, id\_zbor, id\_pasager, id\_client)

values (5, to\_date('2019-dec-15 04:10', 'yyyy-mon-dd hh24:mi'), 0, 1, 0, 0);

insert into rezervari(id\_rezervare, data\_rezervare, id\_loc, id\_zbor, id\_pasager, id\_client)

values (6, to\_date('2019-dec-15 05:00', 'yyyy-mon-dd hh24:mi'), 2, 1, 9, 5);

insert into rezervari(id\_rezervare, data\_rezervare, id\_loc, id\_zbor, id\_pasager, id\_client)

values (7, to\_date('2019-dec-02 05:00', 'yyyy-mon-dd hh24:mi'), 3, 1, 10, 5);

insert into rezervari(id\_rezervare, data\_rezervare, id\_loc, id\_zbor, id\_pasager, id\_client)

values (8, to\_date('2019-dec-02 05:00', 'yyyy-mon-dd hh24:mi'), 5, 1, 4, 5);

insert into rezervari(id\_rezervare, data\_rezervare, id\_loc, id\_zbor, id\_pasager, id\_client)

values (9, to\_date('2019-nov-06 23:50', 'yyyy-mon-dd hh24:mi'), 11, 1, 4, 11);

insert into rezervari(id\_rezervare, data\_rezervare, id\_loc, id\_zbor, id\_pasager, id\_client)

values (10, to\_date('2019-nov-02 14:00', 'yyyy-mon-dd hh24:mi'), 7, 1, 2, 6);

-- zbor 2

insert into rezervari(id\_rezervare, data\_rezervare, id\_loc, id\_zbor, id\_pasager, id\_client)

values (11, to\_date('2020-feb-23 01:00', 'yyyy-mon-dd hh24:mi'), 0, 2, 1, 6);

insert into rezervari(id\_rezervare, data\_rezervare, id\_loc, id\_zbor, id\_pasager, id\_client)

values (12, to\_date('2020-jan-20 04:00', 'yyyy-mon-dd hh24:mi'), 8, 2, 3, 6);

insert into rezervari(id\_rezervare, data\_rezervare, id\_loc, id\_zbor, id\_pasager, id\_client)

values (13, to\_date('2020-jan-04 20:00', 'yyyy-mon-dd hh24:mi'), 11, 2, 6, 5);

insert into rezervari(id\_rezervare, data\_rezervare, id\_loc, id\_zbor, id\_pasager, id\_client)

values (14, to\_date('2020-jan-04 20:00', 'yyyy-mon-dd hh24:mi'), 10, 2, 5, 5);

-- zbor 3

insert into rezervari(id\_rezervare, data\_rezervare, id\_loc, id\_zbor, id\_pasager, id\_client)

values (15, to\_date('2020-apr-01 20:00', 'yyyy-mon-dd hh24:mi'), 9, 3, 11, 14);

insert into rezervari(id\_rezervare, data\_rezervare, id\_loc, id\_zbor, id\_pasager, id\_client)

values (16, to\_date('2020-apr-01 20:20', 'yyyy-mon-dd hh24:mi'), 10, 3, 12, 14);

insert into rezervari(id\_rezervare, data\_rezervare, id\_loc, id\_zbor, id\_pasager, id\_client)

values (17, to\_date('2020-apr-01 20:00', 'yyyy-mon-dd hh24:mi'), 3, 3, 2, 9);

insert into rezervari(id\_rezervare, data\_rezervare, id\_loc, id\_zbor, id\_pasager, id\_client)

values (18, to\_date('2020-feb-20 15:00', 'yyyy-mon-dd hh24:mi'), 5, 3, 3, 10);

insert into rezervari(id\_rezervare, data\_rezervare, id\_loc, id\_zbor, id\_pasager, id\_client)

values (19, to\_date('2020-mar-12 16:00', 'yyyy-mon-dd hh24:mi'), 12, 3, 7, 12);

insert into rezervari(id\_rezervare, data\_rezervare, id\_loc, id\_zbor, id\_pasager, id\_client)

values (20, to\_date('2020-mar-12 18:00', 'yyyy-mon-dd hh24:mi'), 13, 3, 8, 12);

insert into rezervari(id\_rezervare, data\_rezervare, id\_loc, id\_zbor, id\_pasager, id\_client)

values (21, to\_date('2020-feb-14 21:00', 'yyyy-mon-dd hh24:mi'), 7, 3, 0, 7);

-- zbor 4

insert into rezervari(id\_rezervare, data\_rezervare, id\_loc, id\_zbor, id\_pasager, id\_client)

values (22, to\_date('2021-jun-11 00:00', 'yyyy-mon-dd hh24:mi'), 5, 4, 6, 0);

insert into rezervari(id\_rezervare, data\_rezervare, id\_loc, id\_zbor, id\_pasager, id\_client)

values (23, to\_date('2021-apr-20 16:00', 'yyyy-mon-dd hh24:mi'), 10, 4, 5, 5);

insert into rezervari(id\_rezervare, data\_rezervare, id\_loc, id\_zbor, id\_pasager, id\_client)

values (24, to\_date('2021-apr-20 16:30', 'yyyy-mon-dd hh24:mi'), 11, 4, 7, 5);

insert into rezervari(id\_rezervare, data\_rezervare, id\_loc, id\_zbor, id\_pasager, id\_client)

values (25, to\_date('2021-apr-20 18:00', 'yyyy-mon-dd hh24:mi'), 12, 4, 8, 5);

insert into rezervari(id\_rezervare, data\_rezervare, id\_loc, id\_zbor, id\_pasager, id\_client)

values (26, to\_date('2021-may-03 12:00', 'yyyy-mon-dd hh24:mi'), 3, 4, 14, 4);

insert into rezervari(id\_rezervare, data\_rezervare, id\_loc, id\_zbor, id\_pasager, id\_client)

values (27, to\_date('2021-may-03 13:00', 'yyyy-mon-dd hh24:mi'), 4, 4, 15, 4);

insert into rezervari(id\_rezervare, data\_rezervare, id\_loc, id\_zbor, id\_pasager, id\_client)

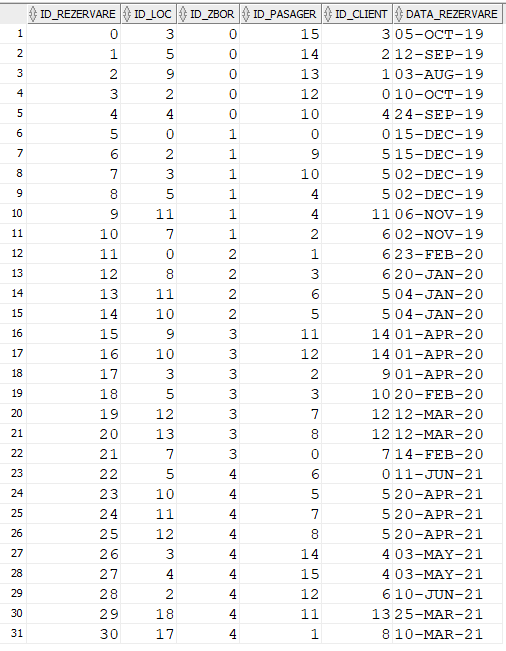
values (28, to\_date('2021-jun-10 11:00', 'yyyy-mon-dd hh24:mi'), 2, 4, 12, 6);

insert into rezervari(id\_rezervare, data\_rezervare, id\_loc, id\_zbor, id\_pasager, id\_client)

values (29, to\_date('2021-mar-25 19:00', 'yyyy-mon-dd hh24:mi'), 18, 4, 11, 13);

insert into rezervari(id\_rezervare, data\_rezervare, id\_loc, id\_zbor, id\_pasager, id\_client)

values (30, to\_date('2021-mar-10 13:00', 'yyyy-mon-dd hh24:mi'), 17, 4, 1, 8);



* Plăți

create sequence secv\_plati

start with 0

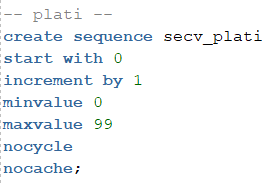
increment by 1

minvalue 0

maxvalue 99

nocycle

nocache;



insert into plati(id\_plata, tip\_plata, id\_rezervare)

values (secv\_plati.nextval, 'online', 0);

insert into plati(id\_plata, tip\_plata, id\_rezervare)

values (secv\_plati.nextval, 'online', 1);

insert into plati(id\_plata, tip\_plata, id\_rezervare)

values (secv\_plati.nextval, 'transfer bancar', 2);

insert into plati(id\_plata, tip\_plata, id\_rezervare)

values (secv\_plati.nextval, 'online', 3);

insert into plati(id\_plata, tip\_plata, id\_rezervare)

values (secv\_plati.nextval, 'online', 4);

insert into plati(id\_plata, tip\_plata, id\_rezervare)

values (secv\_plati.nextval, 'online', 5);

insert into plati(id\_plata, tip\_plata, id\_rezervare)

values (secv\_plati.nextval, 'online', 6);

insert into plati(id\_plata, tip\_plata, id\_rezervare)

values (secv\_plati.nextval, 'online', 7);

insert into plati(id\_plata, tip\_plata, id\_rezervare)

values (secv\_plati.nextval, 'transfer bancar', 8);

insert into plati(id\_plata, tip\_plata, id\_rezervare)

values (secv\_plati.nextval, 'transfer bancar', 9);

insert into plati(id\_plata, tip\_plata, id\_rezervare)

values (secv\_plati.nextval, 'transfer bancar', 10);

insert into plati(id\_plata, tip\_plata, id\_rezervare)

values (secv\_plati.nextval, 'transfer bancar', 11);

insert into plati(id\_plata, tip\_plata, id\_rezervare)

values (secv\_plati.nextval, 'online', 12);

insert into plati(id\_plata, tip\_plata, id\_rezervare)

values (secv\_plati.nextval, 'online', 13);

insert into plati(id\_plata, tip\_plata, id\_rezervare)

values (secv\_plati.nextval, 'online', 14);

insert into plati(id\_plata, tip\_plata, id\_rezervare)

values (secv\_plati.nextval, 'online', 15);

insert into plati(id\_plata, tip\_plata, id\_rezervare)

values (secv\_plati.nextval, 'online', 16);

insert into plati(id\_plata, tip\_plata, id\_rezervare)

values (secv\_plati.nextval, 'online', 17);

insert into plati(id\_plata, tip\_plata, id\_rezervare)

values (secv\_plati.nextval, 'online', 18);

insert into plati(id\_plata, tip\_plata, id\_rezervare)

values (secv\_plati.nextval, 'online', 19);

insert into plati(id\_plata, tip\_plata, id\_rezervare)

values (secv\_plati.nextval, 'transfer bancar', 20);

insert into plati(id\_plata, tip\_plata, id\_rezervare)

values (secv\_plati.nextval, 'transfer bancar', 21);

insert into plati(id\_plata, tip\_plata, id\_rezervare)

values (secv\_plati.nextval, 'transfer bancar', 22);

insert into plati(id\_plata, tip\_plata, id\_rezervare)

values (secv\_plati.nextval, 'online', 23);

insert into plati(id\_plata, tip\_plata, id\_rezervare)

values (secv\_plati.nextval, 'online', 24);

insert into plati(id\_plata, tip\_plata, id\_rezervare)

values (secv\_plati.nextval, 'online', 25);

insert into plati(id\_plata, tip\_plata, id\_rezervare)

values (secv\_plati.nextval, 'online', 26);

insert into plati(id\_plata, tip\_plata, id\_rezervare)

values (secv\_plati.nextval, 'online', 27);

insert into plati(id\_plata, tip\_plata, id\_rezervare)

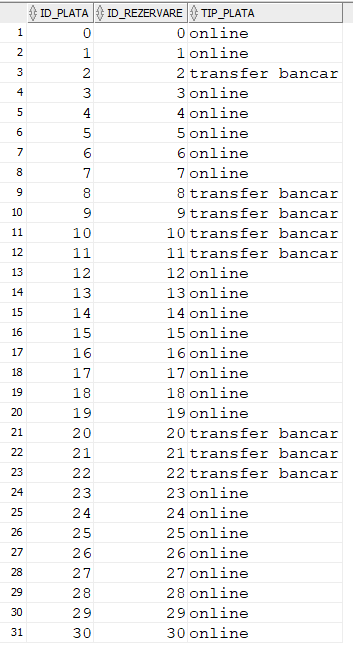
values (secv\_plati.nextval, 'online', 28);

insert into plati(id\_plata, tip\_plata, id\_rezervare)

values (secv\_plati.nextval, 'online', 29);

insert into plati(id\_plata, tip\_plata, id\_rezervare)

values (secv\_plati.nextval, 'online', 30);



* Check-in

create sequence secv\_check

start with 0

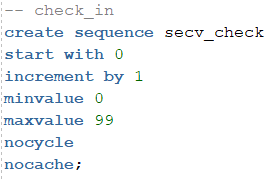
increment by 1

minvalue 0

maxvalue 99

nocycle

nocache;



-- zbor 0

insert into check\_in(id\_check\_in, data\_check\_in, id\_rezervare)

values (secv\_check.nextval, to\_date('2019-oct-19 15:10', 'yyyy-mon-dd hh24:mi'), 0);

insert into check\_in(id\_check\_in, data\_check\_in, id\_rezervare)

values (secv\_check.nextval, to\_date('2019-oct-19 14:30', 'yyyy-mon-dd hh24:mi'), 1);

insert into check\_in(id\_check\_in, data\_check\_in, id\_rezervare)

values (secv\_check.nextval, to\_date('2019-oct-19 12:00', 'yyyy-mon-dd hh24:mi'), 2);

insert into check\_in(id\_check\_in, data\_check\_in, id\_rezervare)

values (secv\_check.nextval, to\_date('2019-oct-19 13:30', 'yyyy-mon-dd hh24:mi'), 3);

insert into check\_in(id\_check\_in, data\_check\_in, id\_rezervare)

values (secv\_check.nextval, to\_date('2019-oct-19 15:00', 'yyyy-mon-dd hh24:mi'), 4);

-- zbor 1

insert into check\_in(id\_check\_in, data\_check\_in, id\_rezervare)

values (secv\_check.nextval, to\_date('2019-dec-16 14:30', 'yyyy-mon-dd hh24:mi'), 5);

insert into check\_in(id\_check\_in, data\_check\_in, id\_rezervare)

values (secv\_check.nextval, to\_date('2019-dec-16 13:30', 'yyyy-mon-dd hh24:mi'), 7);

insert into check\_in(id\_check\_in, data\_check\_in, id\_rezervare)

values (secv\_check.nextval, to\_date('2019-dec-16 12:30', 'yyyy-mon-dd hh24:mi'), 8);

insert into check\_in(id\_check\_in, data\_check\_in, id\_rezervare)

values (secv\_check.nextval, to\_date('2019-dec-16 13:00', 'yyyy-mon-dd hh24:mi'), 10);

-- zbor 2

insert into check\_in(id\_check\_in, data\_check\_in, id\_rezervare)

values (secv\_check.nextval, to\_date('2020-feb-23 10:10', 'yyyy-mon-dd hh24:mi'), 11);

insert into check\_in(id\_check\_in, data\_check\_in, id\_rezervare)

values (secv\_check.nextval, to\_date('2020-feb-23 08:00', 'yyyy-mon-dd hh24:mi'), 13);

insert into check\_in(id\_check\_in, data\_check\_in, id\_rezervare)

values (secv\_check.nextval, to\_date('2020-feb-23 09:30', 'yyyy-mon-dd hh24:mi'), 14);

-- zbor 3

insert into check\_in(id\_check\_in, data\_check\_in, id\_rezervare)

values (secv\_check.nextval, to\_date('2020-apr-5 19:00', 'yyyy-mon-dd hh24:mi'), 15);

insert into check\_in(id\_check\_in, data\_check\_in, id\_rezervare)

values (secv\_check.nextval, to\_date('2020-apr-5 19:30', 'yyyy-mon-dd hh24:mi'), 16);

insert into check\_in(id\_check\_in, data\_check\_in, id\_rezervare)

values (secv\_check.nextval, to\_date('2020-apr-5 18:00', 'yyyy-mon-dd hh24:mi'), 18);

insert into check\_in(id\_check\_in, data\_check\_in, id\_rezervare)

values (secv\_check.nextval, to\_date('2020-apr-5 16:20', 'yyyy-mon-dd hh24:mi'), 19);

insert into check\_in(id\_check\_in, data\_check\_in, id\_rezervare)

values (secv\_check.nextval, to\_date('2020-apr-5 19:00', 'yyyy-mon-dd hh24:mi'), 20);

insert into check\_in(id\_check\_in, data\_check\_in, id\_rezervare)

values (secv\_check.nextval, to\_date('2020-apr-5 19:20', 'yyyy-mon-dd hh24:mi'), 21);

-- zbor 4

insert into check\_in(id\_check\_in, data\_check\_in, id\_rezervare)

values (secv\_check.nextval, to\_date('2021-jun-11 18:00', 'yyyy-mon-dd hh24:mi'), 22);

insert into check\_in(id\_check\_in, data\_check\_in, id\_rezervare)

values (secv\_check.nextval, to\_date('2021-jun-11 15:00', 'yyyy-mon-dd hh24:mi'), 23);

insert into check\_in(id\_check\_in, data\_check\_in, id\_rezervare)

values (secv\_check.nextval, to\_date('2021-jun-11 14:00', 'yyyy-mon-dd hh24:mi'), 25);

insert into check\_in(id\_check\_in, data\_check\_in, id\_rezervare)

values (secv\_check.nextval, to\_date('2021-jun-11 17:30', 'yyyy-mon-dd hh24:mi'), 26);

insert into check\_in(id\_check\_in, data\_check\_in, id\_rezervare)

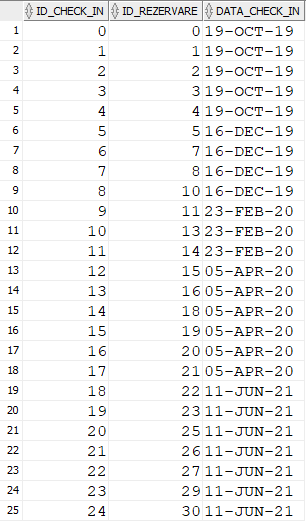
values (secv\_check.nextval, to\_date('2021-jun-11 18:10', 'yyyy-mon-dd hh24:mi'), 27);

insert into check\_in(id\_check\_in, data\_check\_in, id\_rezervare)

values (secv\_check.nextval, to\_date('2021-jun-11 14:30', 'yyyy-mon-dd hh24:mi'), 29);

insert into check\_in(id\_check\_in, data\_check\_in, id\_rezervare)

values (secv\_check.nextval, to\_date('2021-jun-11 15:00', 'yyyy-mon-dd hh24:mi'), 30);



* Bagaje

create sequence secv\_bagaje

start with 0

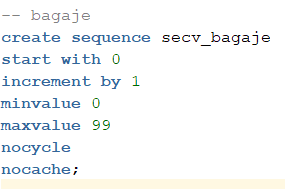
increment by 1

minvalue 0

maxvalue 99

nocycle

nocache;



insert into bagaje(id\_bagaj, greutate, id\_check\_in)

values (secv\_bagaje.nextval, 20, 1);

insert into bagaje(id\_bagaj, greutate, id\_check\_in)

values (secv\_bagaje.nextval, 10, 2);

insert into bagaje(id\_bagaj, greutate, id\_check\_in)

values (secv\_bagaje.nextval, 25, 2);

insert into bagaje(id\_bagaj, greutate, id\_check\_in)

values (secv\_bagaje.nextval, 20, 3);

insert into bagaje(id\_bagaj, greutate, id\_check\_in)

values (secv\_bagaje.nextval, 15, 12);

insert into bagaje(id\_bagaj, greutate, id\_check\_in)

values (secv\_bagaje.nextval, 10, 17);

insert into bagaje(id\_bagaj, greutate, id\_check\_in)

values (secv\_bagaje.nextval, 10, 18);

insert into bagaje(id\_bagaj, greutate, id\_check\_in)

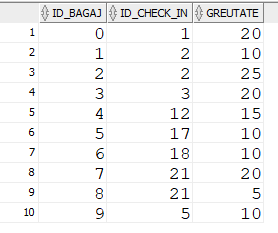
values (secv\_bagaje.nextval, 20, 21);

insert into bagaje(id\_bagaj, greutate, id\_check\_in)

values (secv\_bagaje.nextval, 5, 21);

insert into bagaje(id\_bagaj, greutate, id\_check\_in)

values (secv\_bagaje.nextval, 10, 5);



# Colecții

Formulați în limbaj natural o problemă pe care să o rezolvați folosind un subprogram stocat independent care să utilizeze toate cele 3 tipuri de colecții studiate. Apelați subprogramul.

Cerință: Luați primele v\_nr zboruri (în ordinea crescătoare a id-urilor) și puneți id-urile lor într-un vector. Pentru aceste zboruri, fie tabelul indexat de tabele imbricate care reține pentru fiecare zbor ce bagaje are. Modificați greutatea bagajelor care e mai mare decât v\_max în v\_max. Afișați înainte și după actualizare.

create or replace procedure p6(v\_nr in number, v\_max in number) is

type t\_imbricat is table of bagaje%rowtype;

t\_bagaje t\_imbricat := t\_imbricat();

type t\_indexat is table of t\_imbricat index by pls\_integer;

t\_bagaje\_zboruri t\_indexat;

type vector is varray(10) of zboruri.id\_zbor%type;

t\_zboruri vector := vector();

begin

for i in 1..v\_nr loop

t\_zboruri.extend;

end loop;

select id\_zbor

bulk collect into t\_zboruri

from zboruri

where rownum <= v\_nr

order by id\_zbor;

dbms\_output.put\_line('Inainte de actualizare: ');

for i in 1..v\_nr loop

dbms\_output.put\_line('Id zbor: ' || t\_zboruri(i));

select b.id\_bagaj, b.id\_check\_in, b.greutate

bulk collect into t\_bagaje

from bagaje b, check\_in c, rezervari r

where b.id\_check\_in = c.id\_check\_in

and c.id\_rezervare = r.id\_rezervare

and r.id\_zbor = t\_zboruri(i);

t\_bagaje\_zboruri(t\_zboruri(i)) := t\_bagaje;

if sql%rowcount = 0 then

dbms\_output.put\_line('Nu are bagaje');

else

for j in t\_bagaje\_zboruri(t\_zboruri(i)).first .. t\_bagaje\_zboruri(t\_zboruri(i)).last loop

dbms\_output.put\_line('Id bagaj: ' || t\_bagaje\_zboruri(t\_zboruri(i))(j).id\_bagaj ||

', Id check-in: ' || t\_bagaje\_zboruri(t\_zboruri(i))(j).id\_check\_in ||

', Greutate: ' || t\_bagaje\_zboruri(t\_zboruri(i))(j).greutate);

end loop;

end if;

t\_bagaje.delete;

end loop;

for i in 1..v\_nr loop

if t\_bagaje\_zboruri(t\_zboruri(i)).count > 0 then

for j in t\_bagaje\_zboruri(t\_zboruri(i)).first .. t\_bagaje\_zboruri(t\_zboruri(i)).last loop

if t\_bagaje\_zboruri(t\_zboruri(i))(j).greutate > v\_max then

t\_bagaje\_zboruri(t\_zboruri(i))(j).greutate := v\_max;

update bagaje

set greutate = v\_max

where id\_bagaj = t\_bagaje\_zboruri(t\_zboruri(i))(j).id\_bagaj;

end if;

end loop;

end if;

end loop;

dbms\_output.put\_line('Dupa actualizare: ');

for i in 1..v\_nr loop

dbms\_output.put\_line('Id zbor: ' || t\_zboruri(i));

if t\_bagaje\_zboruri(t\_zboruri(i)).count > 0 then

for j in t\_bagaje\_zboruri(t\_zboruri(i)).first .. t\_bagaje\_zboruri(t\_zboruri(i)).last loop

dbms\_output.put\_line('Id bagaj: ' || t\_bagaje\_zboruri(t\_zboruri(i))(j).id\_bagaj ||

', Id check-in: ' || t\_bagaje\_zboruri(t\_zboruri(i))(j).id\_check\_in ||

', Greutate: ' || t\_bagaje\_zboruri(t\_zboruri(i))(j).greutate);

end loop;

else

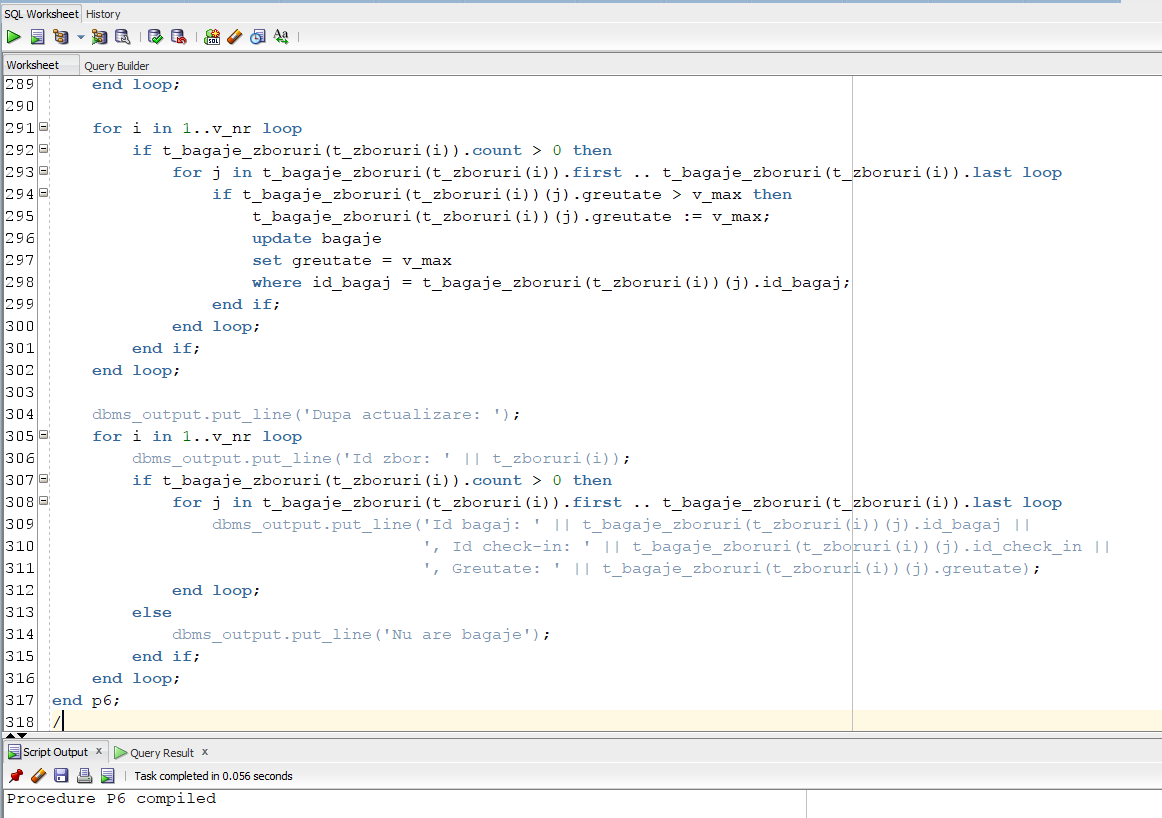
dbms\_output.put\_line('Nu are bagaje');

end if;

end loop;

end p6;

/



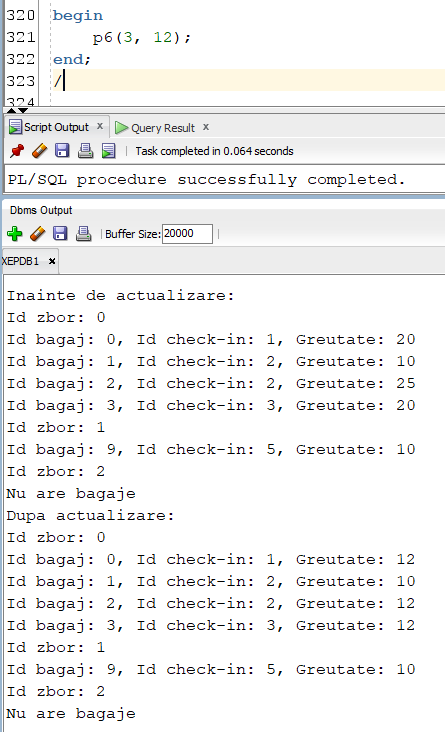
Apelare:

begin

p6(3, 12);

end;

/



# Cursoare

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

Cerință: Pentru piloții care au participat la cel puțin v\_nr\_min zboruri, să se afișeze numele lor și pentru fiecare zbor la care au participat să se afișeze modelul de avion cu care a fost efectuat zborul respectiv.

create or replace procedure p7(v\_nr\_min in number) is

cursor c\_piloti is

select p.id\_pilot, p.nume, count(\*)

from piloti p, echipaje\_piloti e

where p.id\_pilot = e.id\_pilot

group by p.id\_pilot, p.nume

having count(\*) >= v\_nr\_min;

cursor c\_zboruri(v\_id\_pilot piloti.id\_pilot%type) is

select z.id\_zbor id\_zbor, m.nume\_model nume\_model

from piloti p, echipaje\_piloti e, zboruri z, avioane a, modele\_avioane m

where p.id\_pilot = v\_id\_pilot

and p.id\_pilot = e.id\_pilot

and e.id\_zbor = z.id\_zbor

and z.id\_avion = a.id\_avion

and a.id\_model = m.id\_model;

v\_id\_pilot piloti.id\_pilot%type;

v\_nume piloti.nume%type;

v\_nr\_zboruri number;

begin

open c\_piloti;

loop

fetch c\_piloti into v\_id\_pilot, v\_nume, v\_nr\_zboruri;

exit when c\_piloti%notfound;

dbms\_output.put\_line('Pilotul cu id-ul ' || v\_id\_pilot || ' si numele ' || v\_nume || 'a participat la ' || v\_nr\_zboruri || ' zboruri:');

for c in c\_zboruri(v\_id\_pilot) loop

dbms\_output.put\_line('- zborul ' || c.id\_zbor || ' cu modelul de avion ' || c.nume\_model);

end loop;

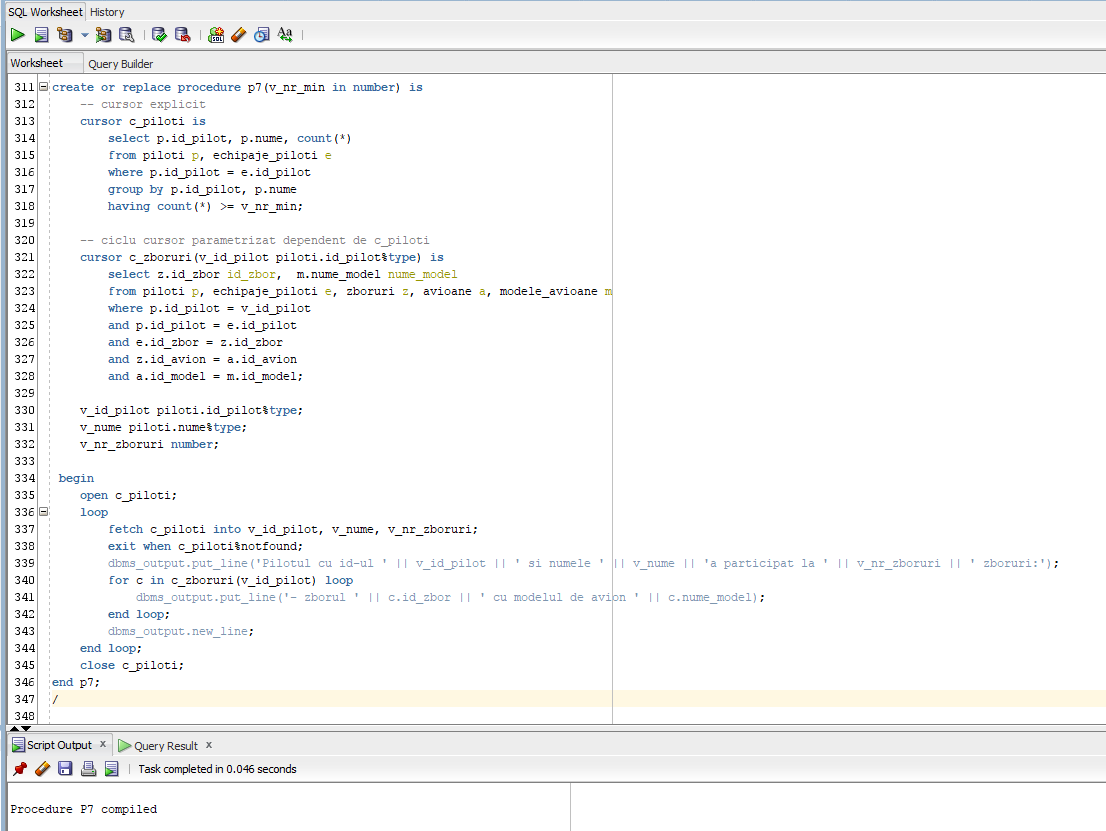
dbms\_output.new\_line;

end loop;

close c\_piloti;

end p7;

/



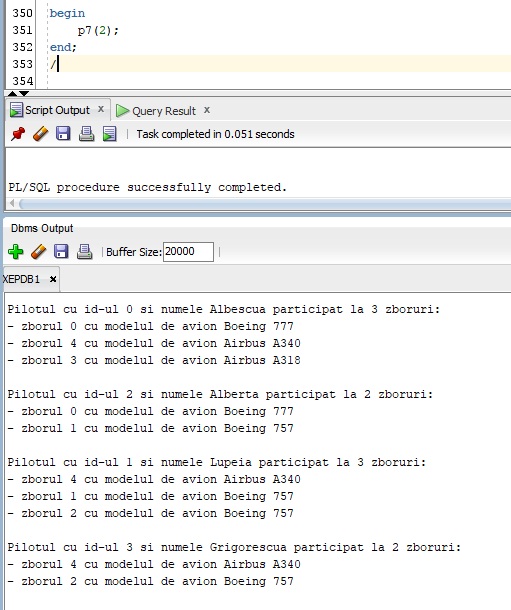
Apelare:

begin

p7(2);

end;

/



# Funcție

Formulați în limbaj natural o problemă pe care să o rezolvați folosind un subprogram stocat independent de tip funcție care să utilizeze într-o singură comandă SQL 3 dintre tabelele definite. Definiți minim 2 excepții proprii. Apelați subprogramul astfel încât să evidențiați toate cazurile definite și tratate.

Cerință: Pentru zborul care ajunge într-un oraș dat (v\_nume\_oraș), micșorați cu un procent v\_proc prețurile locurilor care nu s-au vândut mai scumpe decât un preț v\_preț și returnați numărul de prețuri schimbate.

create or replace function f8(v\_nume\_oras orase.nume%type, v\_proc number, v\_pret locuri.pret%type) return number is

v\_id\_zbor zboruri.id\_zbor%type;

v\_nr number;

e\_procent\_invalid exception;

e\_pret\_invalid exception;

e\_fara\_locuri exception;

begin

if v\_proc <= 0 or v\_proc >= 100 then

raise e\_procent\_invalid;

end if;

if v\_pret <= 0 then

raise e\_pret\_invalid;

end if;

select z.id\_zbor

into v\_id\_zbor

from zboruri z, aeroporturi a, orase o

where o.nume = v\_nume\_oras

and o.id\_oras = a.id\_oras

and a.id\_aeroport = z.id\_aeroport\_sosire;

begin

v\_nr := 0;

for c in (select id\_loc loc

from locuri l

where l.id\_zbor = v\_id\_zbor

and l.pret >= v\_pret

and not exists (select 1

from rezervari r

where r.id\_zbor = l.id\_zbor

and r.id\_loc = l.id\_loc)) loop

update locuri

set pret = pret - v\_proc/100 \* pret

where id\_loc = c.loc;

v\_nr := v\_nr + 1;

end loop;

if v\_nr = 0 then

raise e\_fara\_locuri;

end if;

return v\_nr;

end;

exception

when no\_data\_found then raise\_application\_error(-20000, 'Nu exista niciun zbor care ajunge in acest oras');

when too\_many\_rows then raise\_application\_error(-20001, 'Exista mai multe zboruri care ajung in acest oras');

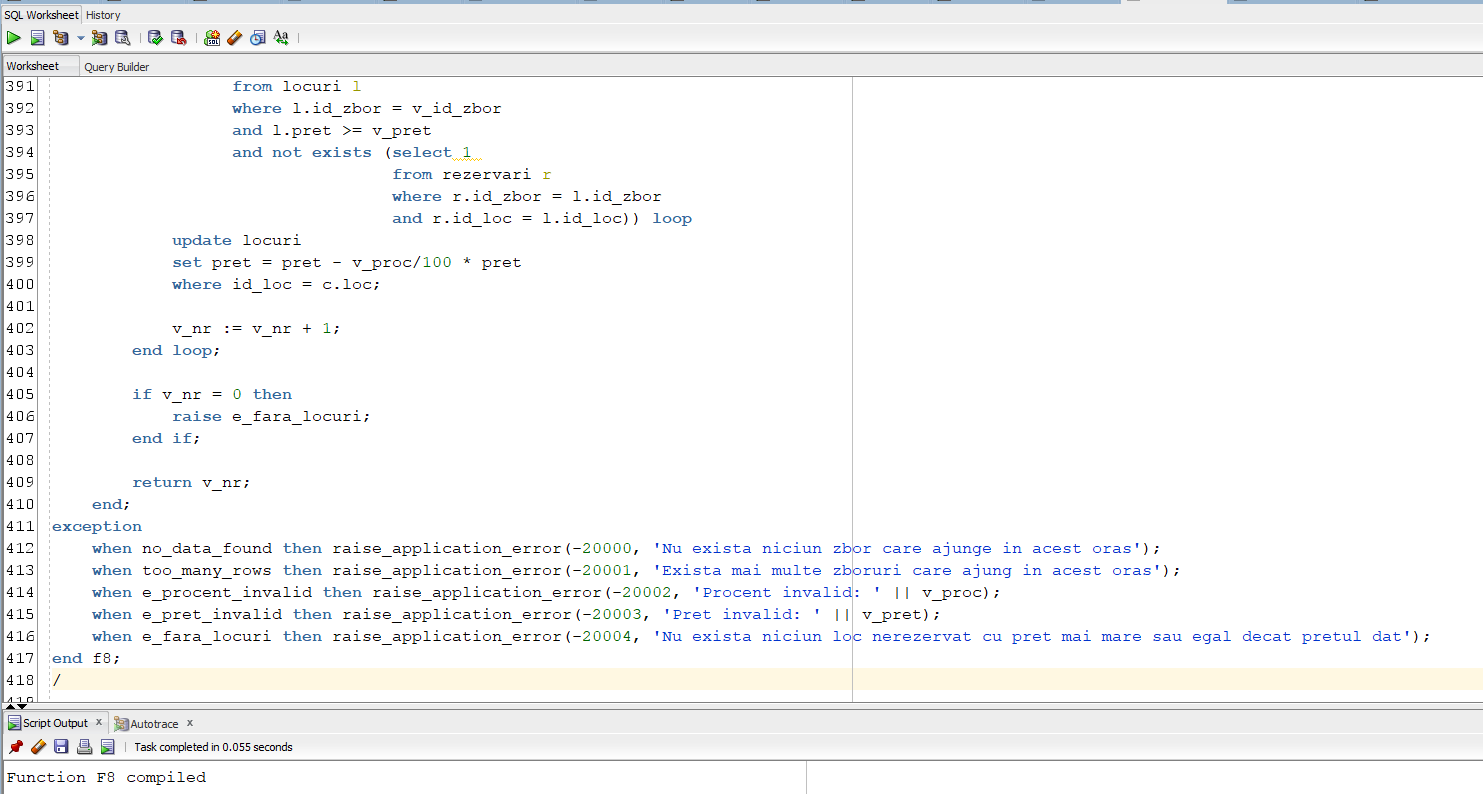
when e\_procent\_invalid then raise\_application\_error(-20002, 'Procent invalid: ' || v\_proc);

when e\_pret\_invalid then raise\_application\_error(-20003, 'Pret invalid: ' || v\_pret);

when e\_fara\_locuri then raise\_application\_error(-20004, 'Nu exista niciun loc nerezervat cu pret mai mare sau egal decat pretul dat');

end f8;

/



Apelare:

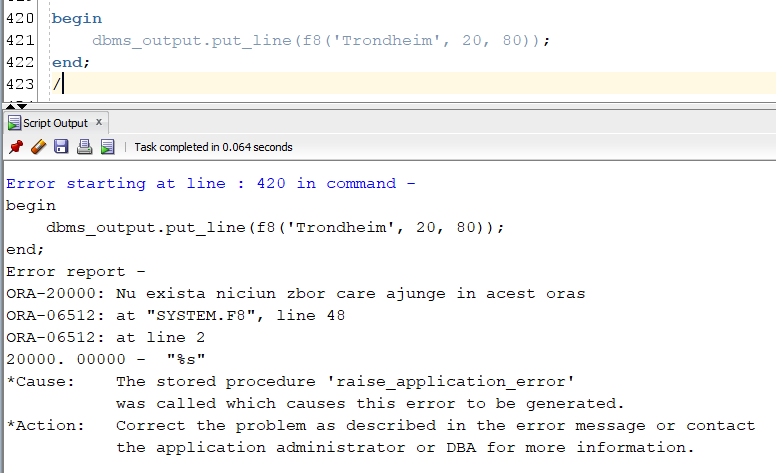
1. Când nu există niciun zbor care ajunge în acel oraș.

begin

dbms\_output.put\_line(f8('Trondheim', 20, 80));

end;

/



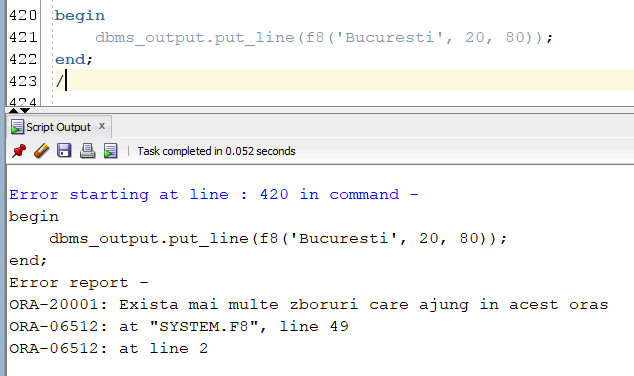
1. Când sunt mai multe zboruri care ajung în acel oraș.

begin

dbms\_output.put\_line(f8('Bucuresti', 20, 80));

end;

/



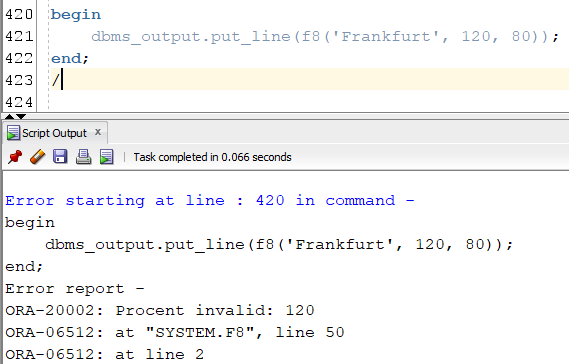
1. Când procentul este invalid.

begin

dbms\_output.put\_line(f8('Frankfurt', 120, 80));

end;

/



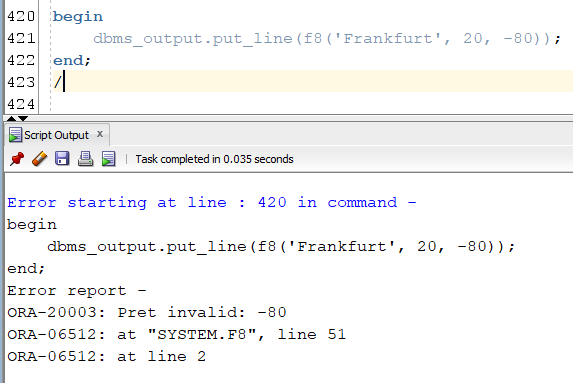
1. Când prețul este invalid.

begin

dbms\_output.put\_line(f8('Frankfurt', 20, -80));

end;

/



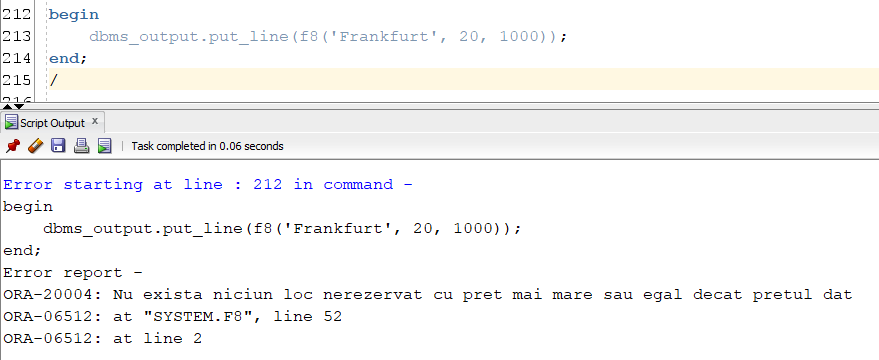
1. Când nu există niciun loc nerezervat cu prețul mai mare sau egal cu prețul dat.

begin

dbms\_output.put\_line(f8('Frankfurt', 20, 1000));

end;

/

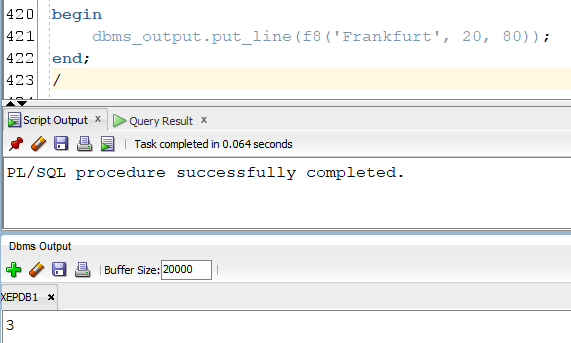


1. Când nu există nicio eroare.

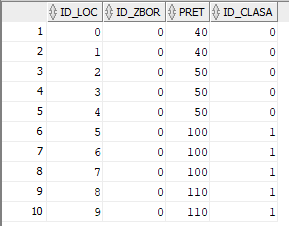
begin

dbms\_output.put\_line(f8('Frankfurt', 20, 80));

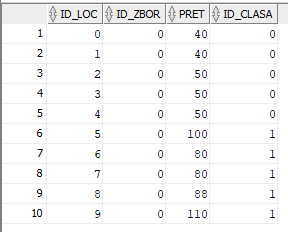
end;



Locurile zborului care ajunge în Frankfurt înainte de actualizare:



Locurile zborului care ajunge în Frankfurt după actualizare:



# Procedură

Formulați în limbaj natural o problemă pe care să o rezolvați folosind un subprogram stocat independent de tip procedură care să utilizeze într-o singură comandă SQL 5 dintre tabelele definite. Tratați toate excepțiile care pot apărea, incluzând excepțiile NO\_DATA\_FOUND și TOO\_MANY\_ROWS. Apelați subprogramul astfel încât să evidențiați toate cazurile tratate.

Cerință: Pentru un pasager dat (v\_nume), să se afle câte zboruri din cele în care are rezervare pleacă dintr-o țară dată v\_țară și locul rezervat este la o clasă dată v\_clasă.

create or replace procedure p9 (v\_nume in pasageri.nume%type, v\_tara in tari.nume%type, v\_clasa in clase.nume%type, v\_nr out number) is

v\_id\_pasager pasageri.id\_pasager%type;

v\_ok number;

begin

begin

select id\_pasager

into v\_id\_pasager

from pasageri

where lower(nume) = lower(v\_nume);

exception

when no\_data\_found then raise\_application\_error(-20000, 'Nu exista niciun pasager cu acest nume');

when too\_many\_rows then raise\_application\_error(-20001, 'Exista mai multi pasageri cu acest nume');

end;

begin

select 1

into v\_ok

from tari

where lower(nume) = lower(v\_tara);

exception

when no\_data\_found then raise\_application\_error(-20002, 'Nu exista nicio tara cu acest nume');

end;

begin

select 1

into v\_ok

from clase

where lower(nume) = lower(v\_clasa);

exception

when no\_data\_found then raise\_application\_error(-20003, 'Nu exista nicio clasa cu acest nume');

end;

select count(r.id\_pasager)

into v\_nr

from rezervari r, locuri l, clase c, zboruri z, aeroporturi a, orase o, tari t

where r.id\_pasager = v\_id\_pasager

and lower(c.nume) = lower(v\_clasa)

and lower(t.nume) = lower(v\_tara)

and r.id\_zbor = l.id\_zbor

and r.id\_loc = l.id\_loc

and l.id\_clasa = c.id\_clasa

and l.id\_zbor = z.id\_zbor

and z.id\_aeroport\_plecare = a.id\_aeroport

and a.id\_oras = o.id\_oras

and o.id\_tara = t.id\_tara

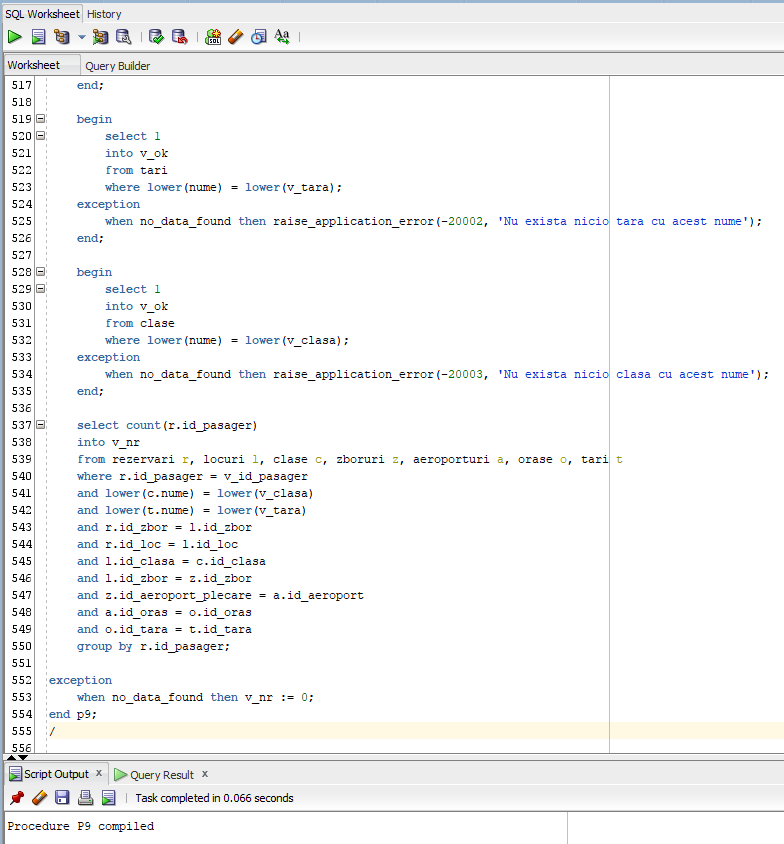
group by r.id\_pasager;

exception

when no\_data\_found then v\_nr := 0;

end p9;

/



Apelare:

1. Când există mai mulți pasageri cu numele dat.

declare

v\_nr number;

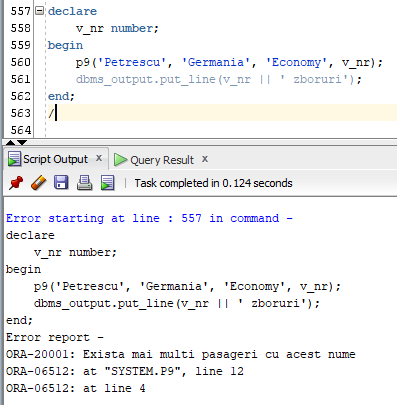
begin

p9('Petrescu', 'Germania', 'Economy', v\_nr);

dbms\_output.put\_line(v\_nr || ' zboruri');

end;

/



1. Când nu există nicio țară cu numele dat.

declare

v\_nr number;

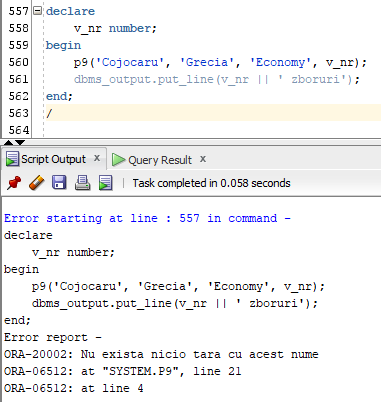
begin

p9('Cojocaru', 'Grecia', 'Economy', v\_nr);

dbms\_output.put\_line(v\_nr || ' zboruri');

end;

/



1. Când nu există nicio clasă cu numele dat.

declare

v\_nr number;

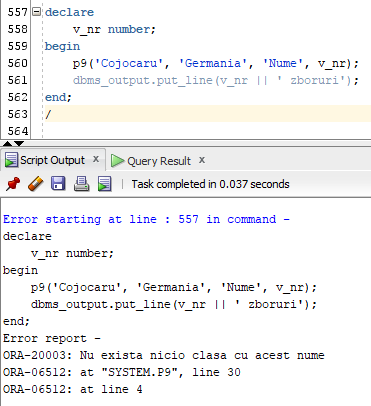
begin

p9('Cojocaru', 'Germania', 'Nume', v\_nr);

dbms\_output.put\_line(v\_nr || ' zboruri');

end;

/



1. Când sunt 0 zboruri.

declare

v\_nr number;

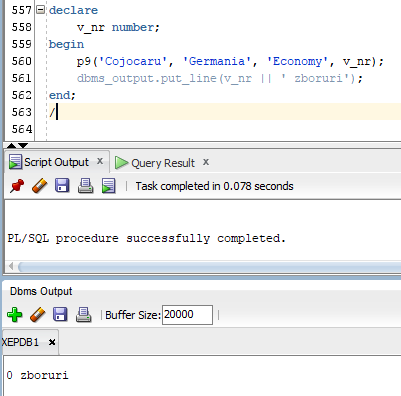
begin

p9('Cojocaru', 'Germania', 'Economy', v\_nr);

dbms\_output.put\_line(v\_nr || ' zboruri');

end;

/



1. Când nu sunt excepții.

declare

v\_nr number;

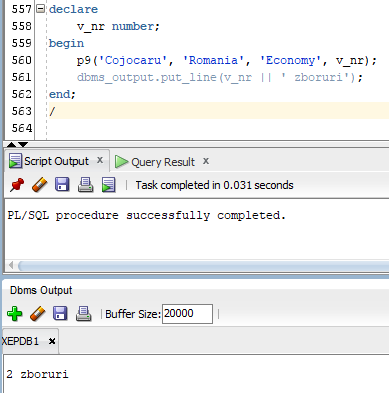
begin

p9('Cojocaru', 'Romania', 'Economy', v\_nr);

dbms\_output.put\_line(v\_nr || ' zboruri');

end;

/



# Trigger LMD la nivel de comandă

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

Cerință: Tabelul pasageri să nu poată fi modificat decât de luni până vineri în intervalul orar 9-18.

create or replace trigger trig10

before insert or update or delete on pasageri

begin

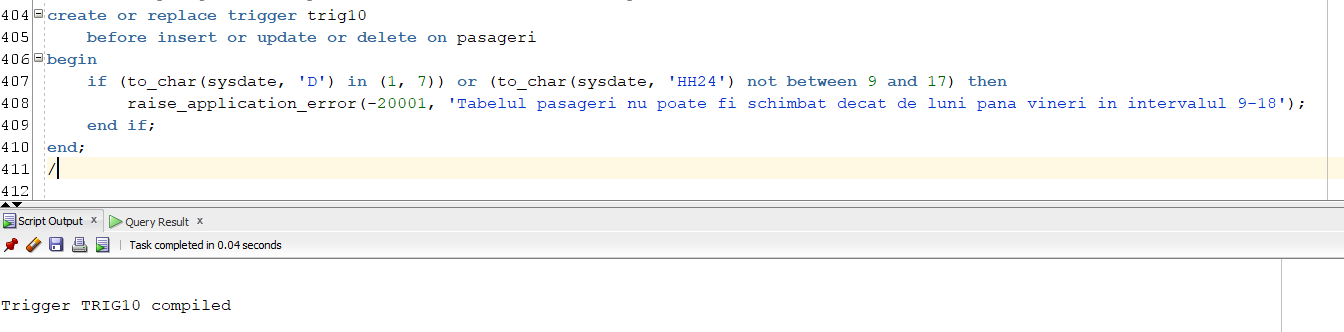
if (to\_char(sysdate, 'D') in (1, 7)) or (to\_char(sysdate, 'HH24') not between 9 and 17) then

raise\_application\_error(-20001, 'Tabelul pasageri nu poate fi schimbat decat de luni pana vineri in intervalul 9-18');

end if;

end;

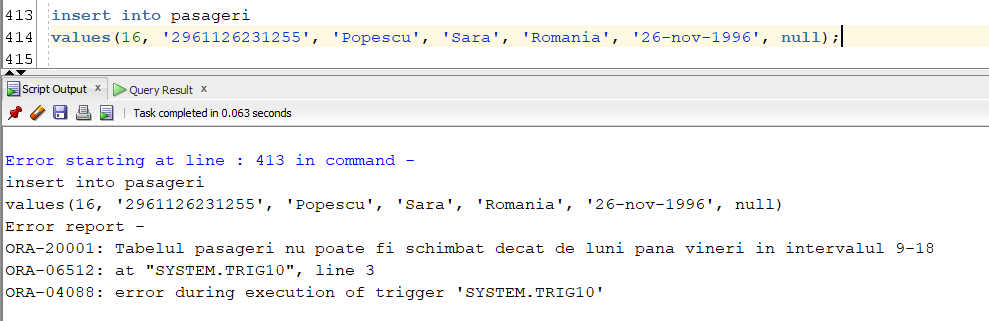
/



Declanșare:

insert into pasageri

values(16, '2961126231255', 'Popescu', 'Sara', 'Romania', '26-nov-1996', null);



# Trigger LMD la nivel de linie

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

Cerință: Fie tabelul info\_zboruri care are coloanele id\_zbor si pret\_total, care reprezintă suma prețurilor locurilor din acel zbor. Definiți un trigger care să actualizeze automat tabelul info\_zboruri atunci când se adaugă, se modifică sau se șterge un loc.

create table info\_zboruri (

id\_info number(4),

pret\_total number(10),

constraint pk\_info\_zboruri primary key(id\_info)

);

create or replace procedure modific\_info (v\_id\_zbor info\_zboruri.id\_info%type, v\_pret locuri.pret%type) as

begin

update info\_zboruri

set pret\_total = nvl(pret\_total, 0) + v\_pret

where id\_info = v\_id\_zbor;

end;

/

create or replace trigger trig11

after delete or update or insert of pret on locuri

for each row

begin

if deleting then

modific\_info(:old.id\_zbor, -1 \* :old.pret);

elsif updating then

modific\_info(:old.id\_zbor, :new.pret - :old.pret);

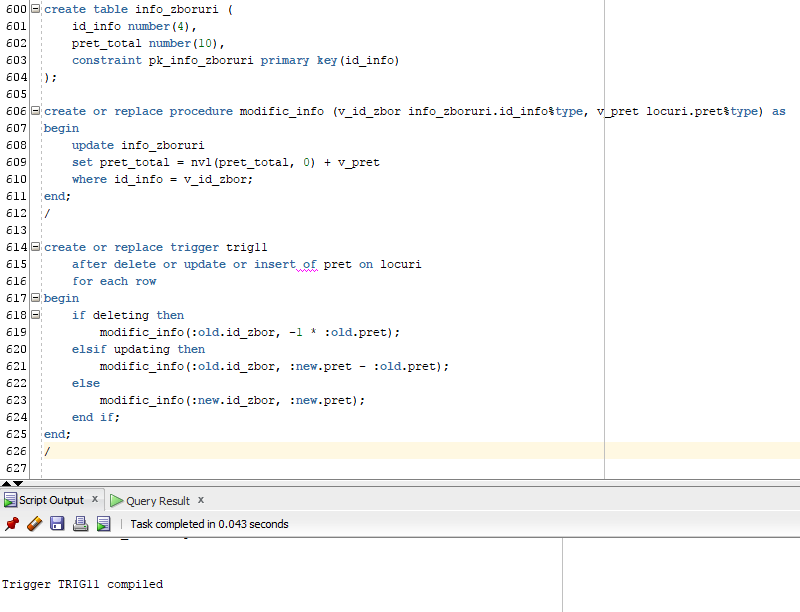
else

modific\_info(:new.id\_zbor, :new.pret);

end if;

end;

/



insert into info\_zboruri

values (0, 719);

insert into info\_zboruri

values (1, 1020);

insert into info\_zboruri

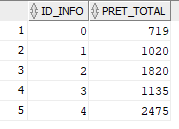
values (2, 1820);

insert into info\_zboruri

values (3, 1135);

insert into info\_zboruri

values (4, 2475);



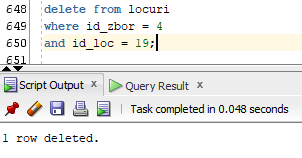
Declanșare:

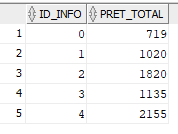
1. La ștergerea unui loc.

delete from locuri

where id\_zbor = 4

and id\_loc = 19;





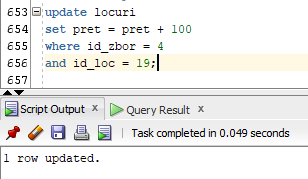
1. La modificarea unui loc.

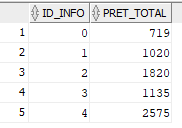
update locuri

set pret = pret + 100

where id\_zbor = 4

and id\_loc = 19;

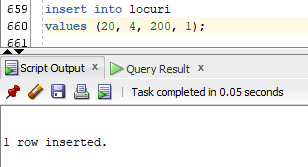


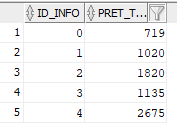


1. La adăugarea unui loc.

insert into locuri

values (20, 4, 200, 1);





# Trigger LDD

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

Cerință: Definiți un trigger care sa nu permită ștergerea tabelelor rezervări2, check\_in și plăți.

create table rezervari2 (

id number(4)

);

create or replace type lista\_tabele as table of varchar2(30);

create or replace trigger trig12

before drop on schema

declare

v\_tabele lista\_tabele := lista\_tabele('REZERVARI2', 'CHECK\_IN', 'PLATI');

begin

for i in 1..v\_tabele.count loop

if ora\_dict\_obj\_type = 'TABLE' and ora\_dict\_obj\_name = v\_tabele(i) then

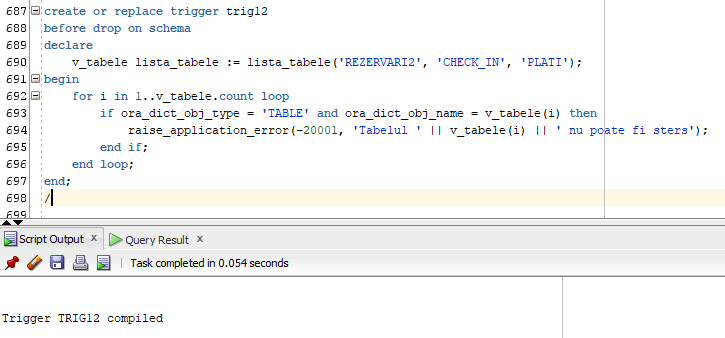
raise\_application\_error(-20001, 'Tabelul ' || v\_tabele(i) || ' nu poate fi sters');

end if;

end loop;

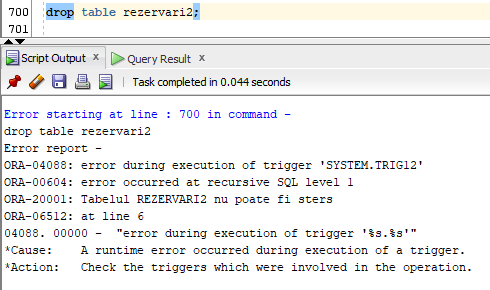
end;

/



Declanșare:

drop table rezervari2;



# Pachet obiecte proiect

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

create or replace package pachet13 as

procedure p6(v\_nr in number, v\_max in number);

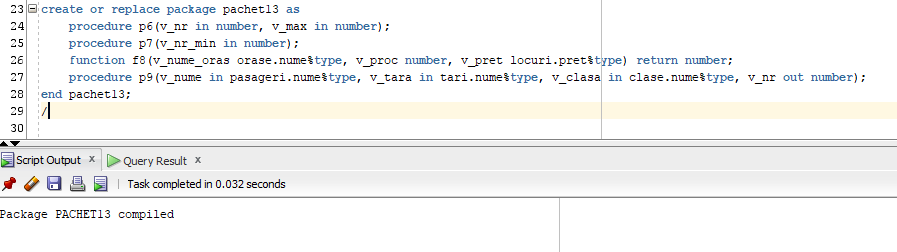
procedure p7(v\_nr\_min in number);

function f8(v\_nume\_oras orase.nume%type, v\_proc number, v\_pret locuri.pret%type) return number;

procedure p9(v\_nume in pasageri.nume%type, v\_tara in tari.nume%type, v\_clasa in clase.nume%type, v\_nr out number);

end pachet13;

/



create or replace package body pachet13 as

-- 6 ---------------------------------------------------------------------------------

procedure p6(v\_nr in number, v\_max in number) is

type t\_imbricat is table of bagaje%rowtype;

t\_bagaje t\_imbricat := t\_imbricat();

type t\_indexat is table of t\_imbricat index by pls\_integer;

t\_bagaje\_zboruri t\_indexat;

type vector is varray(10) of zboruri.id\_zbor%type;

t\_zboruri vector := vector();

begin

for i in 1..v\_nr loop

t\_zboruri.extend;

end loop;

select id\_zbor

bulk collect into t\_zboruri

from zboruri

where rownum <= v\_nr

order by id\_zbor;

dbms\_output.put\_line('Inainte de actualizare: ');

for i in 1..v\_nr loop

dbms\_output.put\_line('Id zbor: ' || t\_zboruri(i));

select b.id\_bagaj, b.id\_check\_in, b.greutate

bulk collect into t\_bagaje

from bagaje b, check\_in c, rezervari r

where b.id\_check\_in = c.id\_check\_in

and c.id\_rezervare = r.id\_rezervare

and r.id\_zbor = t\_zboruri(i);

t\_bagaje\_zboruri(t\_zboruri(i)) := t\_bagaje;

if sql%rowcount = 0 then

dbms\_output.put\_line('Nu are bagaje');

else

for j in t\_bagaje\_zboruri(t\_zboruri(i)).first .. t\_bagaje\_zboruri(t\_zboruri(i)).last loop

dbms\_output.put\_line('Id bagaj: ' || t\_bagaje\_zboruri(t\_zboruri(i))(j).id\_bagaj ||

', Id check-in: ' || t\_bagaje\_zboruri(t\_zboruri(i))(j).id\_check\_in ||

', Greutate: ' || t\_bagaje\_zboruri(t\_zboruri(i))(j).greutate);

end loop;

end if;

t\_bagaje.delete;

end loop;

for i in 1..v\_nr loop

if t\_bagaje\_zboruri(t\_zboruri(i)).count > 0 then

for j in t\_bagaje\_zboruri(t\_zboruri(i)).first .. t\_bagaje\_zboruri(t\_zboruri(i)).last loop

if t\_bagaje\_zboruri(t\_zboruri(i))(j).greutate > v\_max then

t\_bagaje\_zboruri(t\_zboruri(i))(j).greutate := v\_max;

update bagaje

set greutate = v\_max

where id\_bagaj = t\_bagaje\_zboruri(t\_zboruri(i))(j).id\_bagaj;

end if;

end loop;

end if;

end loop;

dbms\_output.put\_line('Dupa actualizare: ');

for i in 1..v\_nr loop

dbms\_output.put\_line('Id zbor: ' || t\_zboruri(i));

if t\_bagaje\_zboruri(t\_zboruri(i)).count > 0 then

for j in t\_bagaje\_zboruri(t\_zboruri(i)).first .. t\_bagaje\_zboruri(t\_zboruri(i)).last loop

dbms\_output.put\_line('Id bagaj: ' || t\_bagaje\_zboruri(t\_zboruri(i))(j).id\_bagaj ||

', Id check-in: ' || t\_bagaje\_zboruri(t\_zboruri(i))(j).id\_check\_in ||

', Greutate: ' || t\_bagaje\_zboruri(t\_zboruri(i))(j).greutate);

end loop;

else

dbms\_output.put\_line('Nu are bagaje');

end if;

end loop;

end p6;

-- 7 ---------------------------------------------------------------------------------

procedure p7(v\_nr\_min in number) is

cursor c\_piloti is

select p.id\_pilot, p.nume, count(\*)

from piloti p, echipaje\_piloti e

where p.id\_pilot = e.id\_pilot

group by p.id\_pilot, p.nume

having count(\*) >= v\_nr\_min;

cursor c\_zboruri(v\_id\_pilot piloti.id\_pilot%type) is

select z.id\_zbor id\_zbor, m.nume\_model nume\_model

from piloti p, echipaje\_piloti e, zboruri z, avioane a, modele\_avioane m

where p.id\_pilot = v\_id\_pilot

and p.id\_pilot = e.id\_pilot

and e.id\_zbor = z.id\_zbor

and z.id\_avion = a.id\_avion

and a.id\_model = m.id\_model;

v\_id\_pilot piloti.id\_pilot%type;

v\_nume piloti.nume%type;

v\_nr\_zboruri number;

begin

open c\_piloti;

loop

fetch c\_piloti into v\_id\_pilot, v\_nume, v\_nr\_zboruri;

exit when c\_piloti%notfound;

dbms\_output.put\_line('Pilotul cu id-ul ' || v\_id\_pilot || ' si numele ' || v\_nume || 'a participat la ' || v\_nr\_zboruri || ' zboruri:');

for c in c\_zboruri(v\_id\_pilot) loop

dbms\_output.put\_line('- zborul ' || c.id\_zbor || ' cu modelul de avion ' || c.nume\_model);

end loop;

dbms\_output.new\_line;

end loop;

close c\_piloti;

end p7;

-- 8 ---------------------------------------------------------------------------------

function f8(v\_nume\_oras orase.nume%type, v\_proc number, v\_pret locuri.pret%type) return number is

v\_id\_zbor zboruri.id\_zbor%type;

v\_nr number;

e\_procent\_invalid exception;

e\_pret\_invalid exception;

e\_fara\_locuri exception;

begin

if v\_proc <= 0 or v\_proc >= 100 then

raise e\_procent\_invalid;

end if;

if v\_pret <= 0 then

raise e\_pret\_invalid;

end if;

select z.id\_zbor

into v\_id\_zbor

from zboruri z, aeroporturi a, orase o

where o.nume = v\_nume\_oras

and o.id\_oras = a.id\_oras

and a.id\_aeroport = z.id\_aeroport\_sosire;

begin

v\_nr := 0;

for c in (select id\_loc loc

from locuri l

where l.id\_zbor = v\_id\_zbor

and l.pret >= v\_pret

and not exists (select 1

from rezervari r

where r.id\_zbor = l.id\_zbor

and r.id\_loc = l.id\_loc)) loop

update locuri

set pret = pret - v\_proc/100 \* pret

where id\_loc = c.loc;

v\_nr := v\_nr + 1;

end loop;

if v\_nr = 0 then

raise e\_fara\_locuri;

end if;

return v\_nr;

end;

exception

when no\_data\_found then raise\_application\_error(-20000, 'Nu exista niciun zbor care ajunge in acest oras');

when too\_many\_rows then raise\_application\_error(-20001, 'Exista mai multe zboruri care ajung in acest oras');

when e\_procent\_invalid then raise\_application\_error(-20002, 'Procent invalid: ' || v\_proc);

when e\_pret\_invalid then raise\_application\_error(-20003, 'Pret invalid: ' || v\_pret);

when e\_fara\_locuri then raise\_application\_error(-20004, 'Nu exista niciun loc nerezervat cu pret mai mare sau egal decat pretul dat');

end f8;

-- 9 ---------------------------------------------------------------------------------

procedure p9

(v\_nume in pasageri.nume%type, v\_tara in tari.nume%type, v\_clasa in clase.nume%type, v\_nr out number) is

v\_id\_pasager pasageri.id\_pasager%type;

v\_ok number;

begin

begin

select id\_pasager

into v\_id\_pasager

from pasageri

where lower(nume) = lower(v\_nume);

exception

when no\_data\_found then raise\_application\_error(-20000, 'Nu exista niciun pasager cu acest nume');

when too\_many\_rows then raise\_application\_error(-20001, 'Exista mai multi pasageri cu acest nume');

end;

begin

select 1

into v\_ok

from tari

where lower(nume) = lower(v\_tara);

exception

when no\_data\_found then raise\_application\_error(-20002, 'Nu exista nicio tara cu acest nume');

end;

begin

select 1

into v\_ok

from clase

where lower(nume) = lower(v\_clasa);

exception

when no\_data\_found then raise\_application\_error(-20003, 'Nu exista nicio clasa cu acest nume');

end;

select count(r.id\_pasager)

into v\_nr

from rezervari r, locuri l, clase c, zboruri z, aeroporturi a, orase o, tari t

where r.id\_pasager = v\_id\_pasager

and lower(c.nume) = lower(v\_clasa)

and lower(t.nume) = lower(v\_tara)

and r.id\_zbor = l.id\_zbor

and r.id\_loc = l.id\_loc

and l.id\_clasa = c.id\_clasa

and l.id\_zbor = z.id\_zbor

and z.id\_aeroport\_plecare = a.id\_aeroport

and a.id\_oras = o.id\_oras

and o.id\_tara = t.id\_tara

group by r.id\_pasager;

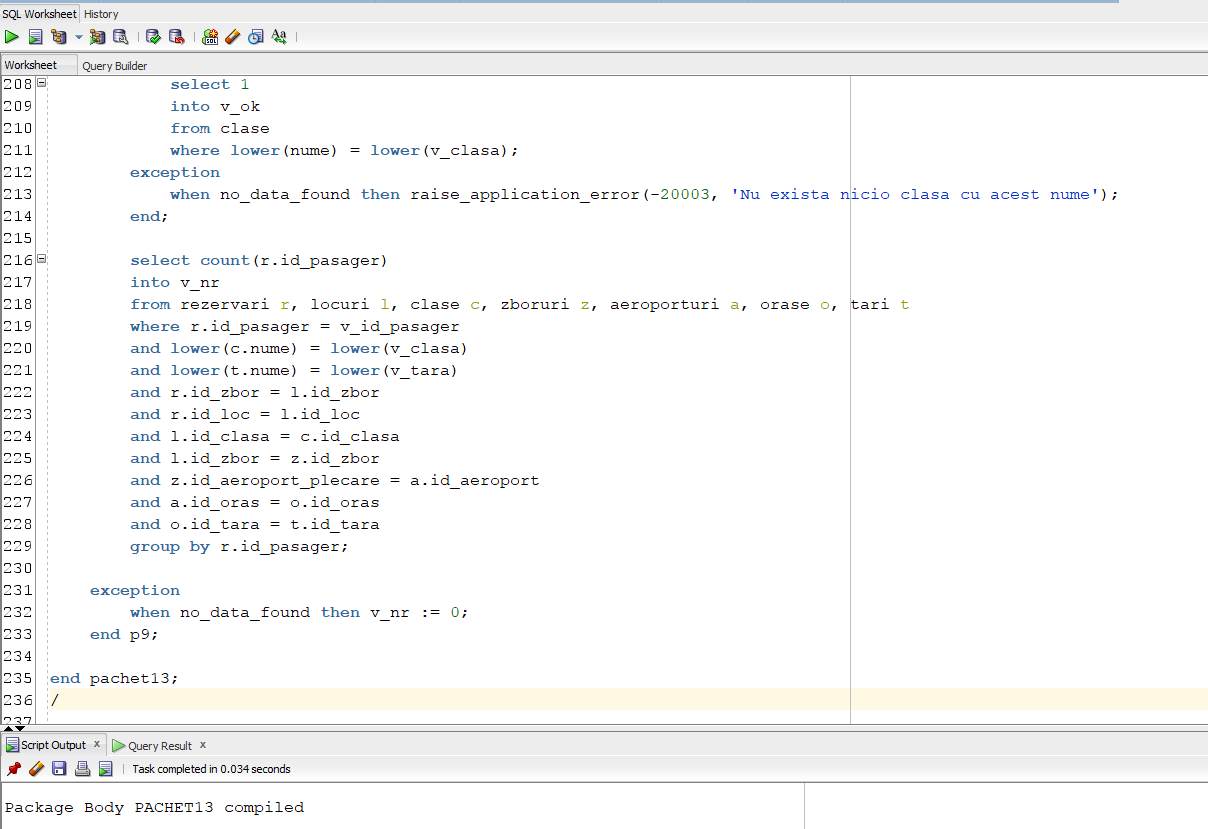
exception

when no\_data\_found then v\_nr := 0;

end p9;

end pachet13;

/



Apelare:

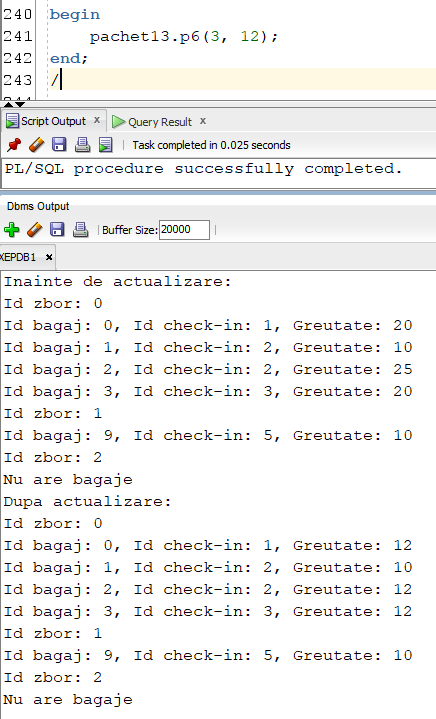
1. 6

begin

pachet13.p6(3, 12);

end;

/



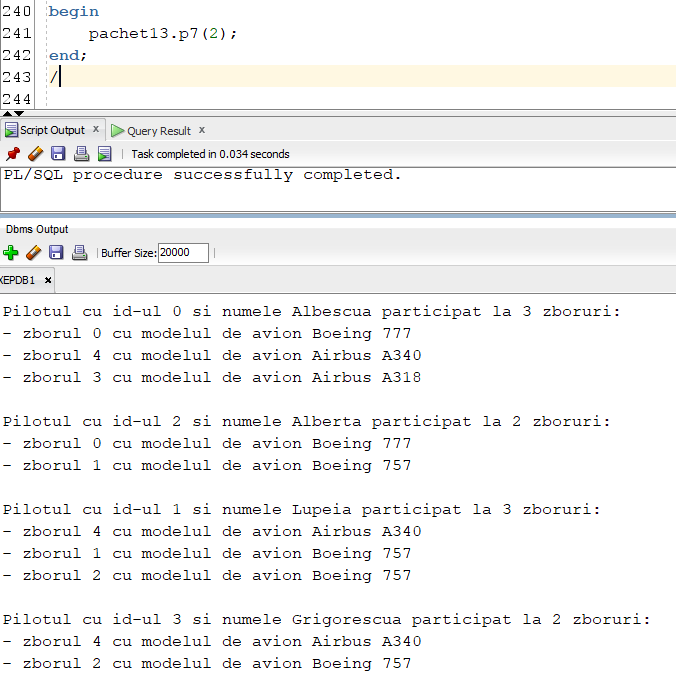
1. 7

begin

pachet13.p7(2);

end;

/



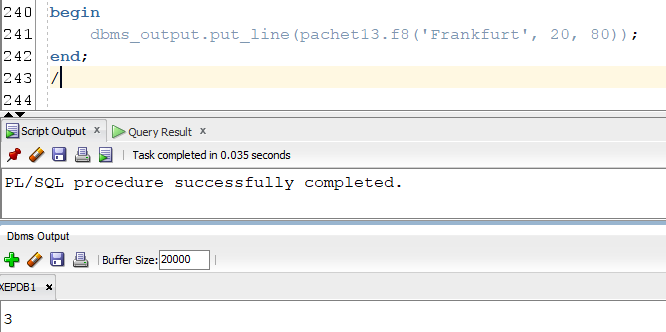
1. 8

begin

dbms\_output.put\_line(pachet13.f8('Frankfurt', 20, 80));

end;

/



1. 9

declare

v\_nr number;

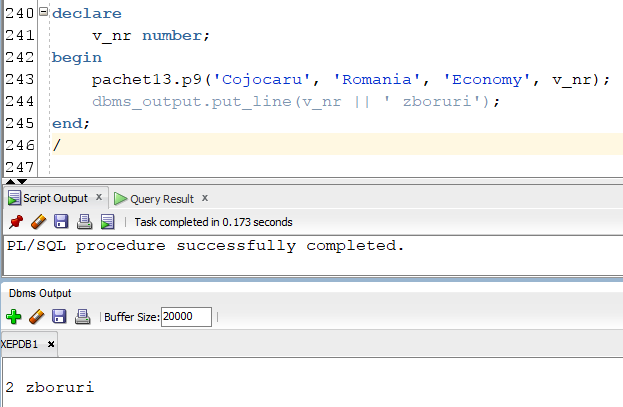
begin

pachet13.p9('Cojocaru', 'Romania', 'Economy', v\_nr);

dbms\_output.put\_line(v\_nr || ' zboruri');

end;

/



# Pachet flux de acțiuni integrate

Definiți un pachet care să includă tipuri de date complexe și obiecte necesare unui flux de acțiuni integrate, specifice bazei de date definite (minim 2 tipuri de date, minim 2 funcții, minim 2 proceduri).

Cerință: Un pachet pentru adăugarea unui zbor. Parametri: dată plecare, dată sosire, id avion, id aeroport plecare, id aeroport sosire. Id-ul zborului să fie generat automat de o secvență. Să se insereze automat informații în tabela locuri (numărul de inserări = numărul de locuri din modelul de avion) cu prețul media prețurilor locurilor din cel mai recent zbor și clasa cu id-ul 1. Să se insereze automat în echipaje piloți un număr dat de piloți pentru acest zbor (cei care au participat la cele mai puține zboruri).

create or replace package pachet14 as

type zbor\_record is record (

data\_plecare zboruri.data\_plecare%type,

data\_sosire zboruri.data\_sosire%type,

id\_avion zboruri.id\_avion%type,

id\_aeroport\_plecare zboruri.id\_aeroport\_plecare%type,

id\_aeroport\_sosire zboruri.id\_aeroport\_sosire%type

);

cursor c\_piloti(v\_nr number) return piloti%rowtype;

type tablou\_imbricat is table of echipaje\_piloti%rowtype;

t\_echipaj\_piloti tablou\_imbricat := tablou\_imbricat();

function f\_inserare\_zbor(v\_zbor in zbor\_record) return zboruri.id\_zbor%type;

function f\_ultimul\_zbor return zboruri.id\_zbor%type;

function f\_pret(v\_id\_zbor zboruri.id\_zbor%type) return locuri.pret%type;

function f\_nr\_locuri(v\_id\_avion zboruri.id\_avion%type) return modele\_avioane.nr\_locuri%type;

procedure p\_inserare\_locuri(v\_id\_zbor in zboruri.id\_zbor%type, v\_pret in locuri.pret%type,

v\_id\_clasa in locuri.id\_clasa%type, v\_nr\_locuri in modele\_avioane.nr\_locuri%type);

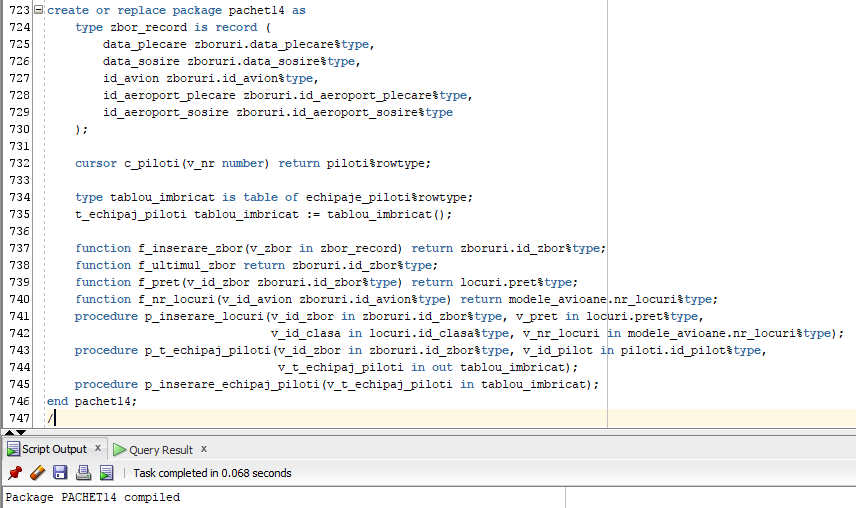
procedure p\_t\_echipaj\_piloti(v\_id\_zbor in zboruri.id\_zbor%type, v\_id\_pilot in piloti.id\_pilot%type,

v\_t\_echipaj\_piloti in out tablou\_imbricat);

procedure p\_inserare\_echipaj\_piloti(v\_t\_echipaj\_piloti in tablou\_imbricat);

end pachet14;

/



create or replace package body pachet14 as

cursor c\_piloti(v\_nr number) return piloti%rowtype is

select \*

from piloti

where id\_pilot in (select \*

from (select id\_pilot

from echipaje\_piloti

group by id\_pilot

order by count(\*), nume)

where rownum <= v\_nr);

function f\_inserare\_zbor(v\_zbor in zbor\_record) return zboruri.id\_zbor%type is

v\_id\_zbor zboruri.id\_zbor%type;

begin

v\_id\_zbor := secv\_zboruri.nextval;

insert into zboruri

values (v\_id\_zbor, v\_zbor.data\_plecare, v\_zbor.data\_sosire, v\_zbor.id\_avion, v\_zbor.id\_aeroport\_plecare, v\_zbor.id\_aeroport\_sosire);

return v\_id\_zbor;

end f\_inserare\_zbor;

function f\_ultimul\_zbor return zboruri.id\_zbor%type is

v\_id\_zbor zboruri.id\_zbor%type;

begin

select id\_zbor

into v\_id\_zbor

from zboruri

where data\_plecare = (select max(data\_plecare)

from zboruri);

return v\_id\_zbor;

end f\_ultimul\_zbor;

function f\_pret(v\_id\_zbor zboruri.id\_zbor%type) return locuri.pret%type is

v\_pret locuri.pret%type;

begin

select avg(pret)

into v\_pret

from locuri

where id\_zbor = v\_id\_zbor;

return v\_pret;

end f\_pret;

function f\_nr\_locuri(v\_id\_avion zboruri.id\_avion%type) return modele\_avioane.nr\_locuri%type is

v\_nr modele\_avioane.nr\_locuri%type;

begin

select m.nr\_locuri

into v\_nr

from modele\_avioane m, avioane a

where m.id\_model = a.id\_model

and a.id\_avion = v\_id\_avion;

return v\_nr;

end f\_nr\_locuri;

procedure p\_inserare\_locuri(v\_id\_zbor in zboruri.id\_zbor%type, v\_pret in locuri.pret%type,

v\_id\_clasa in locuri.id\_clasa%type, v\_nr\_locuri in modele\_avioane.nr\_locuri%type) is

begin

for i in 0..v\_nr\_locuri-1 loop

insert into locuri

values(i, v\_id\_zbor, v\_pret, v\_id\_clasa);

end loop;

end p\_inserare\_locuri;

procedure p\_t\_echipaj\_piloti(v\_id\_zbor in zboruri.id\_zbor%type, v\_id\_pilot in piloti.id\_pilot%type, v\_t\_echipaj\_piloti in out tablou\_imbricat) is

v\_echipaj echipaje\_piloti%rowtype;

begin

v\_echipaj.id\_zbor := v\_id\_zbor;

v\_echipaj.id\_pilot := v\_id\_pilot;

v\_t\_echipaj\_piloti.extend;

v\_t\_echipaj\_piloti(v\_t\_echipaj\_piloti.count) := v\_echipaj;

end p\_t\_echipaj\_piloti;

procedure p\_inserare\_echipaj\_piloti(v\_t\_echipaj\_piloti in tablou\_imbricat) is

begin

for i in 1..v\_t\_echipaj\_piloti.count loop

insert into echipaje\_piloti

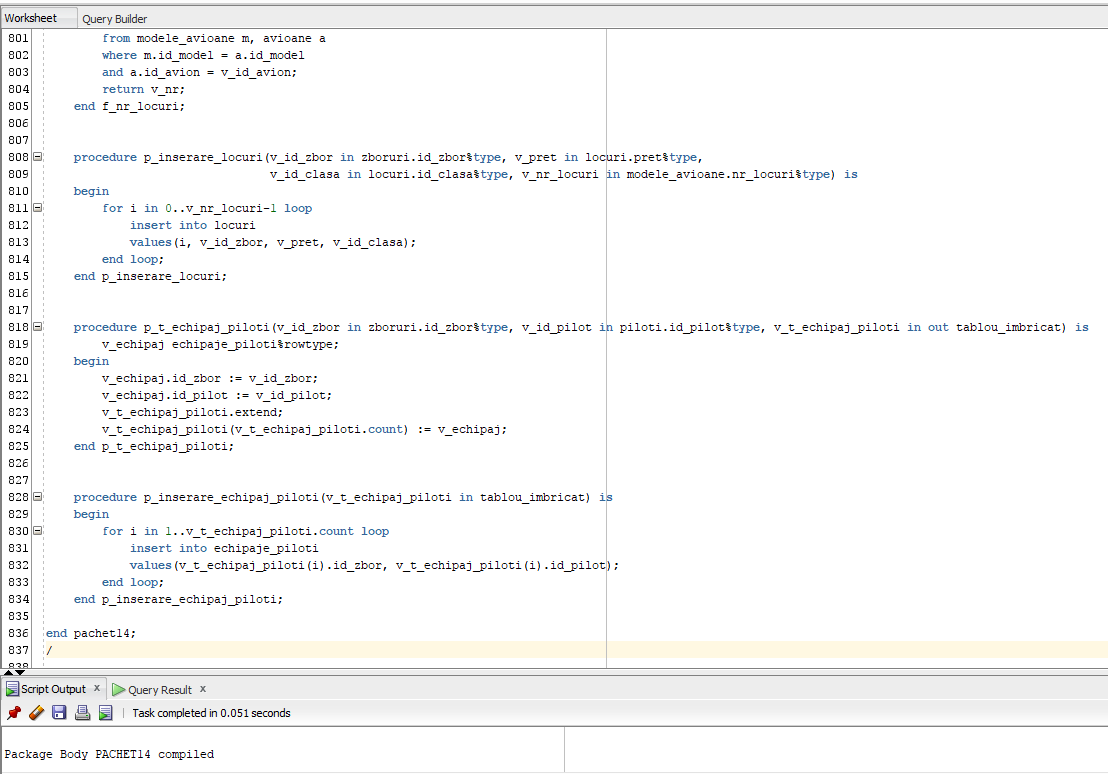
values(v\_t\_echipaj\_piloti(i).id\_zbor, v\_t\_echipaj\_piloti(i).id\_pilot);

end loop;

end p\_inserare\_echipaj\_piloti;

end pachet14;

/



Folosire pachet:

declare

v\_zbor pachet14.zbor\_record;

v\_id\_zbor zboruri.id\_zbor%type;

v\_id\_ult\_zbor zboruri.id\_zbor%type;

v\_pret locuri.pret%type;

v\_nr\_locuri modele\_avioane.nr\_locuri%type;

v\_echipaj\_piloti pachet14.tablou\_imbricat;

begin

v\_zbor.data\_plecare := to\_date('2023-feb-03 12:30', 'yyyy-mon-dd hh24:mi');

v\_zbor.data\_sosire := to\_date('2023-feb-03 14:30', 'yyyy-mon-dd hh24:mi');

v\_zbor.id\_avion := 2;

v\_zbor.id\_aeroport\_plecare := 1;

v\_zbor.id\_aeroport\_sosire := 3;

v\_id\_ult\_zbor := pachet14.f\_ultimul\_zbor;

v\_pret := pachet14.f\_pret(v\_id\_ult\_zbor);

v\_nr\_locuri := pachet14.f\_nr\_locuri(v\_zbor.id\_avion);

v\_id\_zbor := pachet14.f\_inserare\_zbor(v\_zbor);

pachet14.p\_inserare\_locuri(v\_id\_zbor, v\_pret, 1, v\_nr\_locuri);

v\_echipaj\_piloti := pachet14.tablou\_imbricat();

for v\_cursor in pachet14.c\_piloti(2) loop

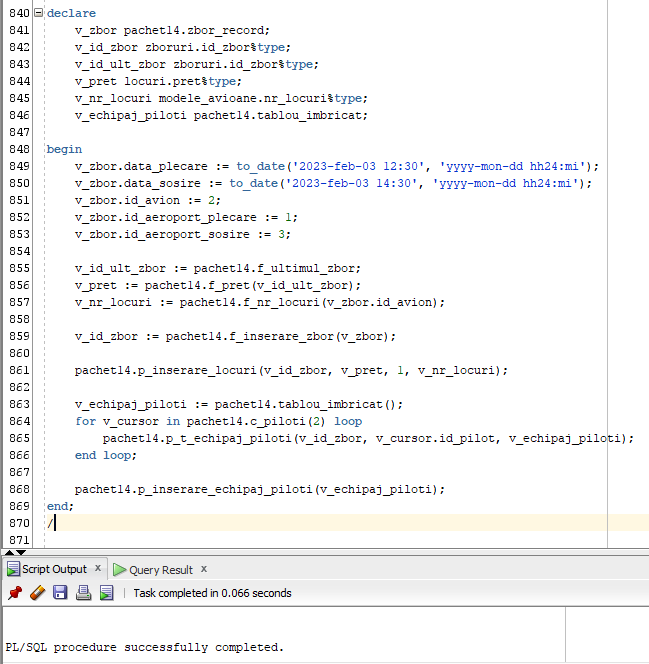
pachet14.p\_t\_echipaj\_piloti(v\_id\_zbor, v\_cursor.id\_pilot, v\_echipaj\_piloti);

end loop;

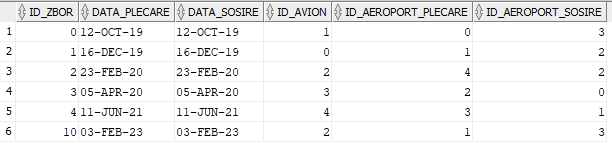
pachet14.p\_inserare\_echipaj\_piloti(v\_echipaj\_piloti);

end;

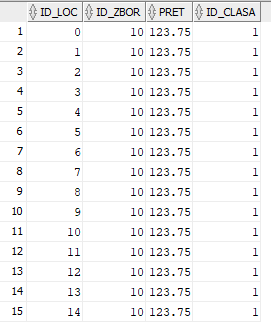
/



Tabelul zboruri după inserare:



Tabelul locuri după inserare (cele din zborul tocmai inserat):



Tabelul echipaje piloți după inserare:

