

Proiect SGBD - Companie Aeriana

Contents

1. Prezentati pe scurt baza de date (utilitatea ei).....	7
2. Realizati diagrama entitate-relatie (ERD): entitatile, relatiile și atributele trebuie definite în limba romana (vezi curs SGBD / model de diagrama ERD; nu se va accepta alt format).....	9
3. Pornind de la diagrama entitate-relatie realizati diagrama conceptuala a modelului propus, integrand toate atributele necesare: entitatile, relatiile si atributele trebuie definite în limba română.....	10
4. Implementati în Oracle diagrama conceptuala realizata: definiti toate tabelele, definind toate constrângerile de integritate necesare (chei primare, cheile externe etc).....	12
5. Adaugati informatii coerente în tabelele create (minim 5 inregistrari pentru fiecare entitate independenta; minim 10 inregistrari pentru tabela asociativa).....	12
6. Formulati în limbaj natural o problema pe care sa o rezolvati folosind un subprogram stocat independent care sa utilizeze toate cele 3 tipuri de colectii studiate. Apelati subprogramul.....	92
7. Formulati in limbaj natural o problema pe care sa o rezolvati folosind un subprogram stocat independent care sa utilizeze 2 tipuri diferite de cursoare studiate, unul dintre acestea fiind cursor parametrizat, dependent de celalalt cursor. Apelati subprogramul.....	96
8. Formulați în limbaj natural o problemă pe care să o rezolvați folosind un subprogram stocat independent de tip funcție care să utilizeze într-o singură comandă SQL 3 dintre tabelele definite. Definiți minim 2 excepții proprii. Apelați subprogramul astfel încât să evidențiați toate cazurile definite și tratate.....	102
9. Formulați în limbaj natural o problemă pe care să o rezolvați folosind un subprogram stocat independent de tip procedură care să utilizeze într-o singură comandă SQL 5 dintre tabelele definite. Tratați toate excepțiile care pot apărea, inclusiv excepțiile NO_DATA_FOUND și TOO_MANY_ROWS. Apelați subprogramul astfel încât să evidențiați toate cazurile tratate.....	108
10. Definiți un trigger de tip LMD la nivel de comandă. Declanșați trigger-ul.....	115
11. Definiți un trigger de tip LMD la nivel de linie. Declanșați trigger-ul.....	120
12. Definiți un trigger de tip LDD. Declanșați trigger-ul.....	130
13. Definiți un pachet care să conțină toate obiectele definite în cadrul proiectului.....	132
14. 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).....	157

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

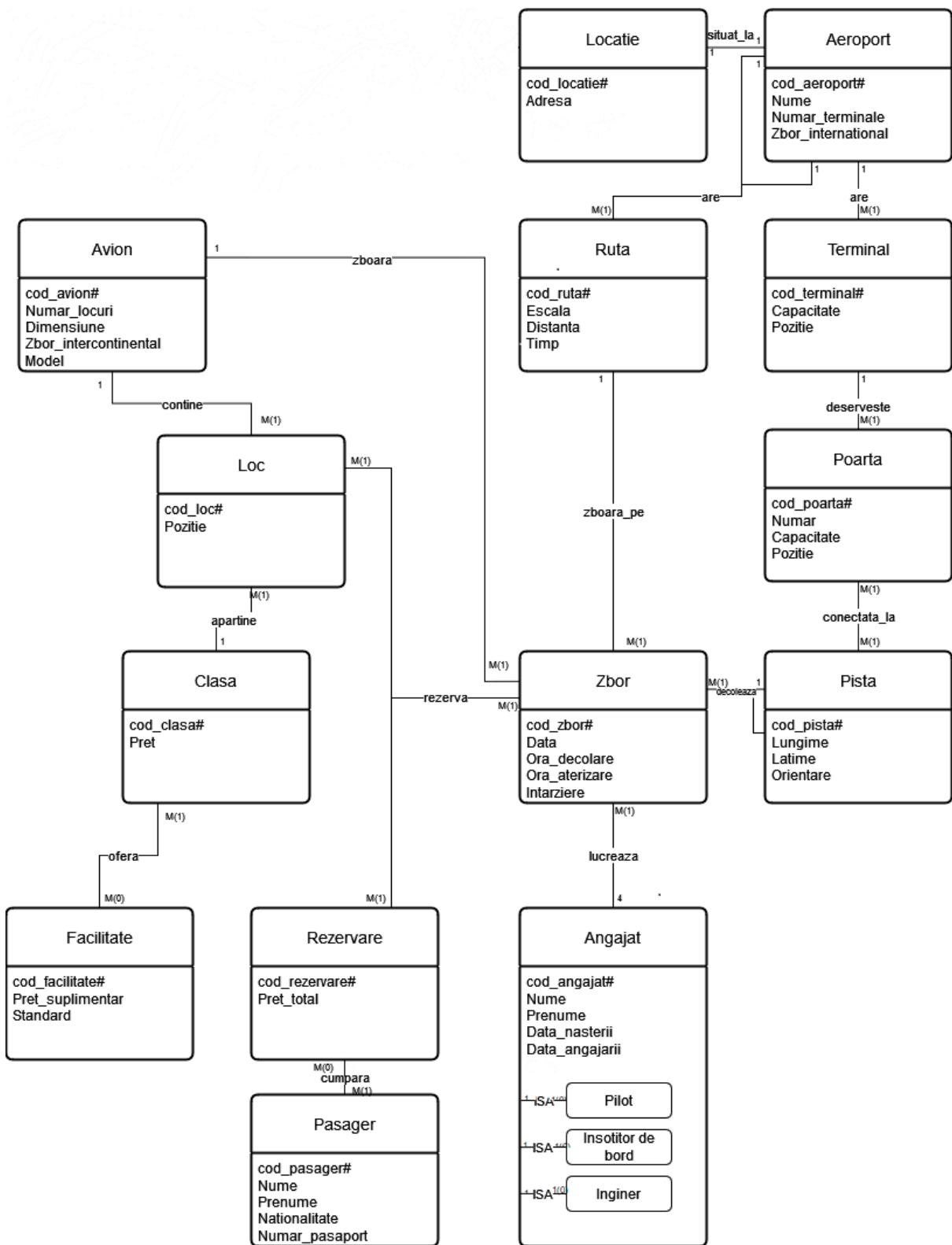
Modelul de date are rolul de a gestiona informatiile esentiale referitoare la organizarea si functionarea zborurilor in cadrul unei companii aeriene. Compania desfasoara operatiuni de transport aerian intre aeroporturi situate in diverse locatii din intreaga lume. Fiecare aeroport are o locatie unica si este legat de alte aeroporturi prin rute stabilite.

In cadrul acestor rute, avioanele companiei efectueaza zboruri, decoland de la un aeroport si aterizand la altul. Pentru a desfasura aceste operatiuni, fiecare aeroport dispune de piste si porti, care sunt utilizate de avioane in timpul decolarii, aterizarii si debarcarii pasagerilor. Pentru fiecare zbor, sunt atribuiti un pilot calificat, un insotitor de bord si un inginer care se ocupa de efectuarea reviziei tehnice a aeronavei inainte de decolare. Acesti angajati sunt membri ai echipei companiei aeriene si asigura desfasurarea fiecarui zbor in conditii optime.

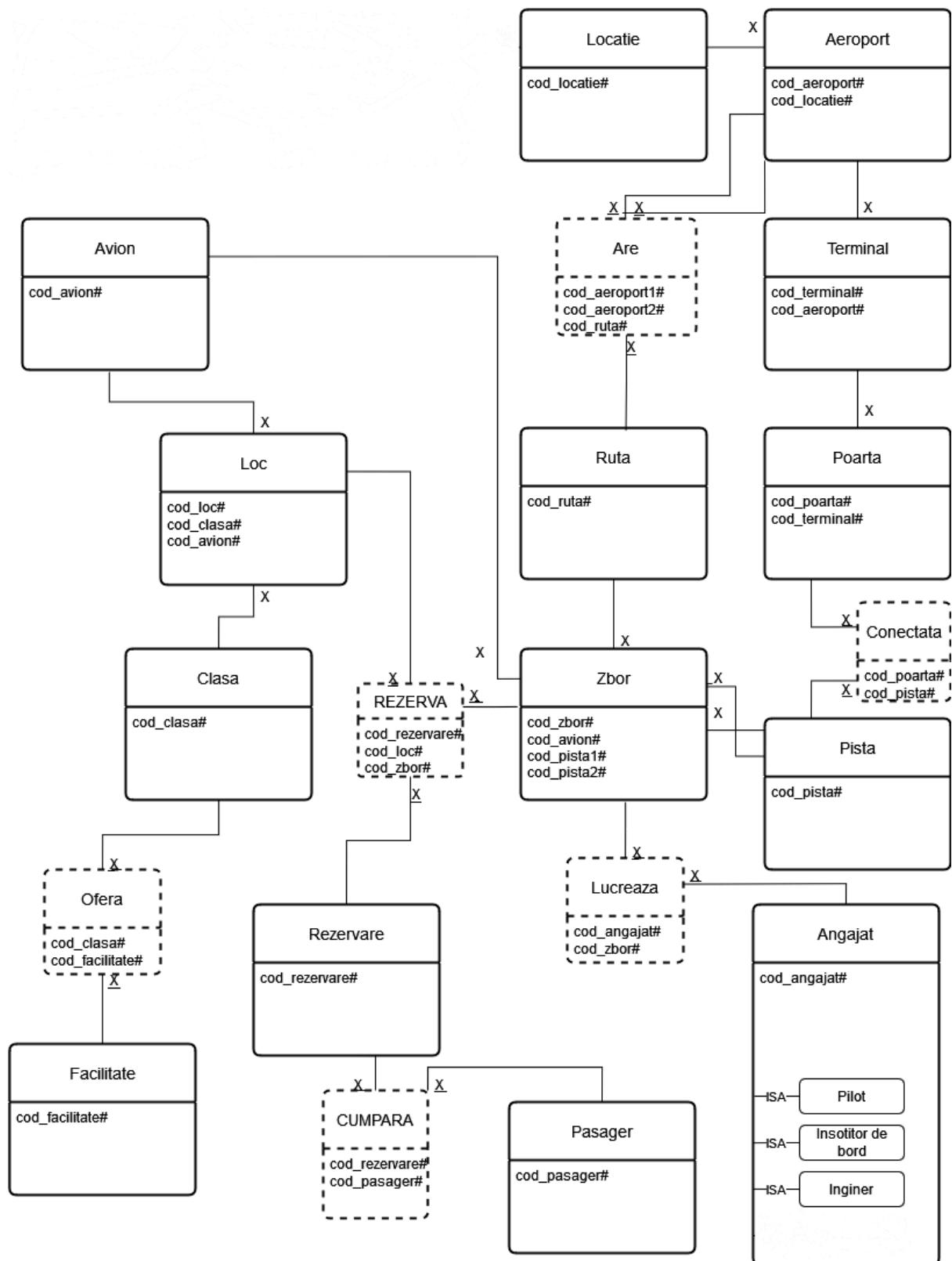
Compania aeriana ofera servicii pasagerilor. Acestia pot achizitiona bilete pentru a calatori in cadrul unui zbor si pot ocupa unul sau mai multe locuri ale aeronavei. Pentru a satisface nevoile si preferintele pasagerilor, locurile sunt impartite in cinci clase distincte, fiecare cu propriile caracteristici si facilitati. Pasagerii au posibilitatea de a alege facilitatile suplimentare, care pot include, de exemplu, privilegii speciale, optiuni de divertisment sau servicii de catering.

Prin gestionarea acestor informatii intr-un model de date eficient, compania aeriana poate planifica si coordona zborurile, asigurandu-se ca fiecare zbor este atribuit personalului potrivit, ca pasagerii beneficiaza de servicii adecvate si ca toate operatiunile se desfasoara in conformitate cu standardele de siguranta si reglementarile din industria aviatica.

2. Realizati diagrama entitate-relatie (ERD):
 entitatile, relatiile și atributele trebuie definite
 în limba română (vezi curs SGBD / model de diagrama
 ERD; nu se va accepta alt format).



3. Pornind de la diagrama entitate-relatie realizati diagrama conceptuala a modelului propus, integrand toate atributele necesare: entitatile, relatiile si atributele trebuie definite in limba romana.



4. Implementati în Oracle diagrama conceptuală realizată: definiti toate tabelele, definind toate constrângările de integritate necesare (chei primare, cheile externe etc).

```
CREATE TABLE LOCATIE(
    cod_locatie NUMBER(5),
    adresa VARCHAR2(50),
    tara VARCHAR2(30) NOT NULL,
    oras VARCHAR2(30) NOT NULL,
    limba_oficiala VARCHAR2(25),
    moneda_oficiala CHAR(3),
    CONSTRAINT pk_locatie PRIMARY KEY(cod_locatie)
);

CREATE TABLE AEROPORT(
    cod_aeroport NUMBER(5),
    cod_locatie NUMBER(5),
    nume VARCHAR2(50),
    zbor_international CHAR(2),
    CONSTRAINT pk_aeroport PRIMARY KEY(cod_aeroport),
    CONSTRAINT fk_aeroport FOREIGN KEY(cod_locatie) REFERENCES
LOCATIE(cod_locatie)
);

CREATE TABLE TERMINAL(
    cod_terminal NUMBER(5),
    cod_aeroport NUMBER(5),
    capacitate NUMBER(5),
    pozitie VARCHAR2(10),
    CONSTRAINT pk_terminal PRIMARY KEY(cod_terminal),
    CONSTRAINT fk_terminal FOREIGN KEY(cod_aeroport)
REFERENCES AEROPORT(cod_aeroport)
);

CREATE TABLE POARTA(
    cod_poarta NUMBER(5),
    cod_terminal NUMBER(5),
    capacitate NUMBER(5),
    pozitie VARCHAR2(50),
    CONSTRAINT pk_poarta PRIMARY KEY(cod_poarta),
    CONSTRAINT fk_poarta FOREIGN KEY(cod_terminal) REFERENCES
TERMINAL(cod_terminal)
);
```

```

CREATE TABLE PISTA(
    cod_pista NUMBER(5),
    lungime NUMBER(4),
    latime NUMBER(2),
    orientare VARCHAR2(30),
    CONSTRAINT pk_pista PRIMARY KEY(cod_pista)
);

CREATE TABLE RUTA(
    cod_ruta NUMBER(5),
    escala VARCHAR2(50),
    distanta NUMBER(5),
    timp NUMBER(4),
    CONSTRAINT pk_ruta PRIMARY KEY(cod_ruta)
);

CREATE TABLE AVION(
    cod_avion NUMBER(5),
    numar_locuri NUMBER(3),
    inaltime NUMBER(2),
    lungime NUMBER(2),
    zbor_intercontinental CHAR(2),
    model VARCHAR2(20),
    CONSTRAINT pk_avion PRIMARY KEY(cod_avion)
);

CREATE TABLE CLASA(
    cod_clasa NUMBER(5),
    pret NUMBER(4),
    CONSTRAINT pk_clasa PRIMARY KEY(cod_clasa)
);

CREATE TABLE LOC(
    cod_loc NUMBER(5),
    cod_avion NUMBER(5),
    pozitie VARCHAR2(9),
    numar_loc NUMBER(3),
    cod_clasa NUMBER(5),
    CONSTRAINT pk_loc PRIMARY KEY(cod_loc),
    CONSTRAINT fk_loc1 FOREIGN KEY(cod_avion) REFERENCES
AVION(cod_avion),
    CONSTRAINT fk_loc2 FOREIGN KEY(cod_clasa) REFERENCES
CLASA(cod_clasa)
);

```

```

CREATE TABLE FACILITATE(
    cod_facilitate NUMBER(5),
    nume_facilitate VARCHAR2(25),
    pret_suplimentar NUMBER(3),
    standard CHAR(2),
    CONSTRAINT pk_facilitate PRIMARY KEY(cod_facilitate)
);

CREATE TABLE PASAGER(
    cod_pasager NUMBER(5),
    nume VARCHAR2(25),
    prenume VARCHAR2(25),
    nationalitate VARCHAR2(25),
    numar_pasaport NUMBER(8),
    CONSTRAINT pk_pasager PRIMARY KEY(cod_pasager)
);

CREATE TABLE REZERVARE(
    cod_rezervare NUMBER(5),
    pret_total NUMBER(5),
    CONSTRAINT pk_rezervare PRIMARY KEY(cod_rezervare)
);
;

CREATE TABLE ANGAJAT(
    cod_angajat NUMBER(5),
    nume VARCHAR2(25),
    salariu NUMBER(5),
    prenume VARCHAR2(25),
    data_nasterii DATE,
    data_angajarii DATE,
    nationalitate VARCHAR2(25),
    CONSTRAINT pk_angajat PRIMARY KEY(cod_angajat)
);

CREATE TABLE PILOT(
    cod_angajat NUMBER(5),
    educatie VARCHAR2(50),
    experienta NUMBER(2),
    CONSTRAINT pk_pilot FOREIGN KEY (cod_angajat) REFERENCES
ANGAJAT(cod_angajat)
);
;

CREATE TABLE INSOTITOR_DE_BORD(
    cod_angajat NUMBER(5),
    experienta NUMBER(2),
    limba_straina VARCHAR2(25),
    CONSTRAINT pk_insotitor FOREIGN KEY (cod_angajat)
REFERENCES
    ANGAJAT(cod_angajat)
);
;
```

```

CREATE TABLE INGINER(
    cod_angajat NUMBER(5),
    educatie VARCHAR2(50),
    CONSTRAINT fk_inginer FOREIGN KEY (cod_angajat) REFERENCES
ANGAJAT(cod_angajat)
);

CREATE TABLE ZBOR(
    cod_zbor NUMBER(5),
    data DATE,
    intarziere NUMBER(5),
    cod_avion NUMBER(5),
    cod_pista1 NUMBER(5),
    cod_pista2 NUMBER(5),
    cod_ruta NUMBER(5),
    CONSTRAINT pk_zbor PRIMARY KEY(cod_zbor),
    CONSTRAINT fk_zbor4 FOREIGN KEY(cod_avion) REFERENCES
AVION(cod_avion),
    CONSTRAINT fk_zbor1 FOREIGN KEY(cod_pista1) REFERENCES
PISTA(cod_pista),
    CONSTRAINT fk_zbor2 FOREIGN KEY(cod_pista2) REFERENCES
PISTA(cod_pista),
    CONSTRAINT fk_zbor3 FOREIGN KEY(cod_ruta) REFERENCES
RUTA(cod_ruta)
);

CREATE TABLE ARE(
    cod_aeroport1 NUMBER(5),
    cod_aeroport2 NUMBER(5),
    cod_ruta NUMBER(5),
    CONSTRAINT pk_are PRIMARY KEY(cod_aeroport1,
cod_aeroport2, cod_ruta),
    CONSTRAINT fk_are1 FOREIGN KEY(cod_aeroport1) REFERENCES
AEROPORT(cod_aeroport),
    CONSTRAINT fk_are2 FOREIGN KEY(cod_aeroport2) REFERENCES
AEROPORT(cod_aeroport),
    CONSTRAINT fk_are3 FOREIGN KEY(cod_ruta) REFERENCES
RUTA(cod_ruta)
);
CREATE TABLE OFERA(
    cod_clasa NUMBER(5),
    cod_facilitate NUMBER(5),
    CONSTRAINT pk_ofera PRIMARY KEY(cod_clasa,
cod_facilitate),
    CONSTRAINT fk_oferal FOREIGN KEY(cod_clasa) REFERENCES
CLASA(cod_clasa),
    CONSTRAINT fk_ofera2 FOREIGN KEY(cod_facilitate)
REFERENCES FACILITATE(cod_facilitate)
);

CREATE TABLE CUMPARA(
    cod_rezervare NUMBER(5),

```

```

cod_pasager NUMBER(5),
numar_locuri NUMBER(2),
CONSTRAINT pk_cumpara PRIMARY KEY(cod_rezervare,
cod_pasager),
CONSTRAINT fk_cumpara1 FOREIGN KEY(cod_rezervare)
REFERENCES REZERVARE(cod_rezervare),
CONSTRAINT fk_cumpara2 FOREIGN KEY(cod_pasager) REFERENCES
PASAGER(cod_pasager)
);

CREATE TABLE REZERVA(
cod_rezervare NUMBER(5),
cod_loc NUMBER(5),
cod_zbor NUMBER(5),
data_rezervare DATE,
CONSTRAINT pk_rezerva PRIMARY KEY(cod_rezervare, cod_loc,
cod_zbor),
CONSTRAINT fk_rezerva1 FOREIGN KEY(cod_rezervare)
REFERENCES REZERVARE(cod_rezervare),
CONSTRAINT fk_rezerva2 FOREIGN KEY(cod_loc) REFERENCES
LOC(cod_loc),
CONSTRAINT fk_rezerva3 FOREIGN KEY(cod_zbor) REFERENCES
ZBOR(cod_zbor)
);

CREATE TABLE LUCREAZA(
cod_angajat NUMBER(5),
cod_zbor NUMBER(5),
ore NUMBER(2),
spor NUMBER(4),
CONSTRAINT pk_lucreaza PRIMARY KEY(cod_angajat, cod_zbor),
CONSTRAINT fk_lucreaza1 FOREIGN KEY(cod_angajat)
REFERENCES ANGAJAT(cod_angajat),
CONSTRAINT fk_lucreaza2 FOREIGN KEY(cod_zbor) REFERENCES
ZBOR(cod_zbor)
);

CREATE TABLE CONECTATA(
cod_poarta NUMBER(5),
cod_pista NUMBER(5),
CONSTRAINT pk_CONECTATA PRIMARY KEY(cod_poarta,
cod_pista),
CONSTRAINT fk_CONECTATA1 FOREIGN KEY(cod_poarta)
REFERENCES POARTA(cod_poarta),
CONSTRAINT fk_CONECTATA2 FOREIGN KEY(cod_pista) REFERENCES
PISTA(cod_pista)
);

```

```
Oracle SQL Developer : C:\Users\40773\project.sql
File Edit View Navigate Run Source Team Tools Window Help
SQL History project.sql Welcome Page
SQL Worksheet History
c_nutri(p_tara); 1 of 1 -> mariap19
Worksheet Query Builder
  CONSTRAINT pk_terminal PRIMARY KEY(cod_terminal),
  CONSTRAINT fk_terminal FOREIGN KEY(cod_aeroport) REFERENCES AEROPORT(cod_aeroport)
);

CREATE TABLE POARTA(
  cod_poarta NUMBER(5),
  cod_terminal NUMBER(5),
  capacitate NUMBER(5),
  pozitie VARCHAR2(50),
  CONSTRAINT pk_poarta PRIMARY KEY(cod_poarta),
  CONSTRAINT fk_poarta FOREIGN KEY(cod_terminal) REFERENCES TERMINAL(cod_terminal)
);

CREATE TABLE PISTA(
  cod_pista NUMBER(5),
  lungime NUMBER(4),
  latime NUMBER(2),
  orientare VARCHAR2(30),
  CONSTRAINT pk_pista PRIMARY KEY(cod_pista)
);

CREATE TABLE RUTA(
  cod_ruta NUMBER(5),
  escafa VARCHAR2(50),
  dist_pista NUMBER(5),
  timp NUMBER(4),
  CONSTRAINT pk_ruta PRIMARY KEY(cod_ruta)
);
Script Output Query Result
Task completed in 0.165 seconds
Table RUTA created.
```

```
Line 62 Column 11 Insert Modified: Windows CR
1 -4°C Cloudy
Search ENG US 8:08 PM 1/12/2024
Oracle SQL Developer : C:\Users\40773\project.sql
File Edit View Navigate Run Source Team Tools Window Help
SQL History project.sql Welcome Page
SQL Worksheet History
c_nutri(p_tara); 1 of 1 -> mariap19
Worksheet Query Builder
  CONSTRAINT pk_terminal PRIMARY KEY(cod_terminal),
  CONSTRAINT fk_terminal FOREIGN KEY(cod_aeroport) REFERENCES AEROPORT(cod_aeroport)
);

CREATE TABLE POARTA(
  cod_poarta NUMBER(5),
  cod_terminal NUMBER(5),
  capacitate NUMBER(5),
  pozitie VARCHAR2(50),
  CONSTRAINT pk_poarta PRIMARY KEY(cod_poarta),
  CONSTRAINT fk_poarta FOREIGN KEY(cod_terminal) REFERENCES TERMINAL(cod_terminal)
);

CREATE TABLE PISTA(
  cod_pista NUMBER(5),
  lungime NUMBER(4),
  latime NUMBER(2),
  orientare VARCHAR2(30),
  CONSTRAINT pk_pista PRIMARY KEY(cod_pista)
);

CREATE TABLE RUTA(
  cod_ruta NUMBER(5),
  escafa VARCHAR2(50),
  dist_pista NUMBER(5),
  timp NUMBER(4),
  CONSTRAINT pk_ruta PRIMARY KEY(cod_ruta)
);
Script Output Query Result
Task completed in 0.204 seconds
Table PISTA created.
```

```
Oracle SQL Developer : C:\Users\40773\project.sql
File Edit View Navigate Run Source Team Tools Window Help
SQL History project.sql Welcome Page
SQL Worksheet History
c_nutri(p_tara); 1 of 1 > > > + - < <
Worksheet Query Builder
CREATE TABLE RUTA(
    cod_ruta NUMBER(5),
    escala VARCHAR2(50),
    distancia NUMBER(5),
    tiempo NUMBER(),
    CONSTRAINT pk_ruta PRIMARY KEY(cod_ruta)
);

CREATE TABLE AVION(
    cod_avion NUMBER(5),
    numero_locuri NUMBER(3),
    inaltime NUMBER(2),
    lungime NUMBER(2),
    zbor_intercontinental CHAR(2),
    model VARCHAR2(20),
    CONSTRAINT pk_avion PRIMARY KEY(cod_avion)
);

CREATE TABLE CLASA(
    cod_clasa NUMBER(5),
    pret NUMBER(4),
    CONSTRAINT pk_clasa PRIMARY KEY(cod_clasa)
);

Script Output > Query Result >
Task completed in 0.059 seconds
Table CLASA created.
```

```
Line 83 Column 11 | Insert | Modified: Windows CR
1 -4°C Cloudy
Search ENG US 8:08 PM 1/12/2024
Oracle SQL Developer : C:\Users\40773\project.sql
File Edit View Navigate Run Source Team Tools Window Help
SQL History project.sql Welcome Page
SQL Worksheet History
c_nutri(p_tara); 1 of 1 > > > + - < <
Worksheet Query Builder
    inaltime NUMBER(2),
    lungime NUMBER(2),
    zbor_intercontinental CHAR(2),
    model VARCHAR2(20),
    CONSTRAINT pk_avion PRIMARY KEY(cod_avion)
);

CREATE TABLE CLASA(
    cod_clasa NUMBER(5),
    pret NUMBER(4),
    CONSTRAINT pk_clasa PRIMARY KEY(cod_clasa)
);

CREATE TABLE LOC(
    cod_loc NUMBER(5),
    cod_avion NUMBER(5),
    pozitie VARCHAR2(50),
    numar_loc NUMBER(3),
    cod_clasa NUMBER(5),
    CONSTRAINT pk_loc PRIMARY KEY(cod_loc),
    CONSTRAINT fk_loc FOREIGN KEY(cod_avion) REFERENCES AVION(cod_avion),
    CONSTRAINT fk_loc2 FOREIGN KEY(cod_clasa) REFERENCES CLASA(cod_clasa)
);

Script Output > Query Result >
Task completed in 0.208 seconds
Table LOC created.
```

```
Oracle SQL Developer : C:\Users\40773\project.sql
File Edit View Navigate Run Source Team Tools Window Help
SQL History project.sql Welcome Page
SQL Worksheet History
c_nutri(p_tara); 1 of 1
Worksheet: Query Builder
CONSTRAINT pk_clasa PRIMARY KEY(cod_clasa)
);

CREATE TABLE LOC(
    cod_loc NUMBER(5),
    cod_avion NUMBER(5),
    pozitie VARCHAR2(5),
    numar_loc NUMBER(3),
    cod_clasa NUMBER(5),
    CONSTRAINT pk_loc PRIMARY KEY(cod_loc),
    CONSTRAINT fk_loc1 FOREIGN KEY(cod_avion) REFERENCES AVION(cod_avion),
    CONSTRAINT fk_loc2 FOREIGN KEY(cod_clasa) REFERENCES CLASA(cod_clasa)
);

CREATE TABLE FACILITATE(
    cod_facilitate NUMBER(5),
    nume_facilitate VARCHAR2(25),
    pret_suplimentar NUMBER(3),
    standard CHAR(2),
    CONSTRAINT pk_facilitate PRIMARY KEY(cod_facilitate)
);

CREATE TABLE PASAGER(
    cod_pasager NUMBER(5),
    nume VARCHAR2(25),
    prenume VARCHAR2(25),
    nationalitate VARCHAR2(25),
    numar_pasaport NUMBER(8),
    CONSTRAINT pk_pasager PRIMARY KEY(cod_pasager)
);

Script Output: Query Result
Task completed in 0.558 seconds
Table FACILITATE created.
```

```
Oracle SQL Developer : C:\Users\40773\project.sql
File Edit View Navigate Run Source Team Tools Window Help
SQL History project.sql Welcome Page
SQL Worksheet History
c_nutri(p_tara); 1 of 1
Worksheet: Query Builder
CONSTRAINT pk_loc PRIMARY KEY(cod_loc),
    cod_avion NUMBER(5),
    pozitie VARCHAR2(5),
    numar_loc NUMBER(3),
    cod_clasa NUMBER(5),
    CONSTRAINT pk_loc PRIMARY KEY(cod_loc),
    CONSTRAINT fk_loc1 FOREIGN KEY(cod_avion) REFERENCES AVION(cod_avion),
    CONSTRAINT fk_loc2 FOREIGN KEY(cod_clasa) REFERENCES CLASA(cod_clasa)
);

CREATE TABLE FACILITATE(
    cod_facilitate NUMBER(5),
    nume_facilitate VARCHAR2(25),
    pret_suplimentar NUMBER(3),
    standard CHAR(2),
    CONSTRAINT pk_facilitate PRIMARY KEY(cod_facilitate)
);

CREATE TABLE PASAGER(
    cod_pasager NUMBER(5),
    nume VARCHAR2(25),
    prenume VARCHAR2(25),
    nationalitate VARCHAR2(25),
    numar_pasaport NUMBER(8),
    CONSTRAINT pk_pasager PRIMARY KEY(cod_pasager)
);

CREATE TABLE REZERVARE(
    cod_rezervare NUMBER(5),
    cod_loc NUMBER(5),
    cod_pasager NUMBER(5),
    cod_facilitate NUMBER(5),
    numar_loc NUMBER(3),
    numar_rezervare NUMBER(3),
    numar_clasa NUMBER(2),
    numar_persoane NUMBER(2),
    CONSTRAINT pk_rezervare PRIMARY KEY(cod_rezervare)
);

Script Output: Query Result
Task completed in 0.197 seconds
Table PASAGER created.
```

```
CREATE TABLE FACILITATE(
    cod_facilitate NUMBER(5),
    nume_facilitate VARCHAR2(25),
    pret_suplimentar NUMBER(3),
    standard CHAR(2),
    CONSTRAINT pk_facilitate PRIMARY KEY(cod_facilitate)
);

CREATE TABLE PASAGER(
    cod_pasager NUMBER(5),
    nume VARCHAR2(25),
    prenume VARCHAR2(25),
    nationalitate VARCHAR2(25),
    numar_pasaport NUMBER(8),
    CONSTRAINT pk_pasager PRIMARY KEY(cod_pasager)
);

CREATE TABLE REZERVARE(
    cod_rezervare NUMBER(5),
    pret_total NUMBER(5),
    CONSTRAINT pk_rezervare PRIMARY KEY(cod_rezervare)
);

CREATE TABLE ANGAJAT(
    cod_angajat NUMBER(5),
    nume VARCHAR2(25),
    prenume VARCHAR2(25),
    salariu NUMBER(5),
    pret_materiale NUMBER(5),
    data_materiale DATE,
    data_angajarii DATE,
    tara VARCHAR2(25),
    CONSTRAINT pk_angajat PRIMARY KEY(cod_angajat)
);

Table REZERVARE created.

Table ANGAJAT created.
```

Script Output : Task completed in 0.163 seconds

Line 121 Column 11 | Insert | Modified: Windows CR

Oracle SQL Developer : C:\Users\40773\project.sql

File Edit View Navigate Run Source Tools Window Help

SQL History project.sql Welcome Page

SQL Worksheet History

c_nutri(p_tara); 1 of 1 --> 1 of 1

Worksheet: Query Builder

```
cod_locatie NUMBER(5),
nume VARCHAR2(50),
zbor_international CHAR(2),
CONSTRAINT pk_aeroport PRIMARY KEY(cod_aeroport),
CONSTRAINT fk_aeroport FOREIGN KEY(cod_locatie) REFERENCES LOCATIE(cod_locatie)
);

CREATE TABLE TERMINAL(
cod_terminal NUMBER(5),
cod_aeroport NUMBER(5),
capacitate NUMBER(5),
pozitie VARCHAR2(10),
CONSTRAINT pk_terminal PRIMARY KEY(cod_terminal),
CONSTRAINT fk_terminal FOREIGN KEY(cod_aeroport) REFERENCES AEROPORT(cod_aeroport)
);

CREATE TABLE POARTA(
cod_poarta NUMBER(5),
cod_terminal NUMBER(5),
capacitate NUMBER(5),
pozitie VARCHAR2(50),
CONSTRAINT pk_poarta PRIMARY KEY(cod_poarta),
CONSTRAINT fk_poarta FOREIGN KEY(cod_terminal) REFERENCES TERMINAL(cod_terminal)
);

CREATE TABLE PISTA(
cod_pista NUMBER(5),

```

Script Output : Query Result :

Task completed in 0.168 seconds

Table POARTA created.

Oracle SQL Developer : C:\Users\40773\project.sql

File Edit View Navigate Run Source Tools Window Help

SQL History project.sql Welcome Page

SQL Worksheet History

c_nutri(poarta); 1 of 1 --> 1 of 1

Worksheet: Query Builder

```
tara VARCHAR2(10) NOT NULL,
oras VARCHAR2(10) NOT NULL,
limba_oficiala VARCHAR2(5),
moneda_oficiala CHAR(3),
CONSTRAINT pk_locatie PRIMARY KEY(cod_locatie)
);

CREATE TABLE AEROPORTI(
cod_aeroport NUMBER(5),
cod_locatie NUMBER(5),
nume VARCHAR2(50),
zbor_international CHAR(2),
CONSTRAINT pk_aeroport PRIMARY KEY(cod_aeroport),
CONSTRAINT fk_aeroport FOREIGN KEY(cod_locatie) REFERENCES LOCATIE(cod_locatie)
);

CREATE TABLE TERMINALI(
cod_terminal NUMBER(5),
cod_aeroport NUMBER(5),
capacitate NUMBER(5),
pozitie VARCHAR2(10),
CONSTRAINT pk_terminal PRIMARY KEY(cod_terminal),
CONSTRAINT fk_terminal FOREIGN KEY(cod_aeroport) REFERENCES AEROPORT(cod_aeroport)
);

CREATE TABLE POARTAI(
cod_poarta NUMBER(5),
cod_terminal NUMBER(5),

```

Script Output : Query Result :

Task completed in 0.151 seconds

Table TERMINAL created.


```
Oracle SQL Developer : C:\Users\40773\project.sql
File Edit View Navigate Run Source Team Tools Window Help
SQL History project.sql Welcome Page
SQL Worksheet History
C:\Users\40773\project.sql
Worksheet - Query Builder
--Code: pentru crearea tabelelor
CREATE SEQUENCE CHIESE
INCREMENT by 10000
START WITH 10
MAXVALUE 20000
NOCYCLE;

CREATE TABLE LOCATIE(
cod_locatie NUMBER(10),
adresa VARCHAR(50),
tara VARCHAR(30) NOT NULL,
oraș VARCHAR(30) NOT NULL,
limba_oficiala VARCHAR(25),
moneda_oficiala CHAR(1),
CONSTRAINT pk_locatie PRIMARY KEY(cod_locatie)
);

CREATE TABLE AEROPORT(
cod_aeroport NUMBER(5),
cod_locatie NUMBER(10),
nume VARCHAR(50),
zbor_international CHAR(1),
CONSTRAINT pk_aeroport PRIMARY KEY(cod_aeroport),
CONSTRAINT fk_aeroport FOREIGN KEY(cod_locatie) REFERENCES LOCATIE(cod_locatie)
);

Script Output ->Query Result
Task completed in 0.18 seconds
Table LOCATIE created.
```

```
Line 52 Column 23 | Insert | Modified Windows CR
Oracle SQL Developer : C:\Users\40773\project.sql
File Edit View Navigate Run Source Team Tools Window Help
SQL History project.sql Welcome Page
SQL Worksheet History
C:\Users\40773\project.sql
Worksheet - Query Builder
--CREATE TABLE PISTA(
cod_pista NUMBER(5),
lungime NUMBER(4),
latime NUMBER(2),
orientare VARCHAR2(30),
CONSTRAINT pk_pista PRIMARY KEY(cod_pista)
);

CREATE TABLE RUTA(
cod_ruta NUMBER(5),
escala VARCHAR2(50),
distanță NUMBER(5),
timp NUMBER(4),
CONSTRAINT pk_ruta PRIMARY KEY(cod_ruta)
);

CREATE TABLE AVION(
cod_avion NUMBER(5),
numar_locuri NUMBER(3),
inaltime NUMBER(2),
lungime NUMBER(2),
zbor_international CHAR(2),
model VARCHAR2(20),
CONSTRAINT pk_avion PRIMARY KEY(cod_avion)
);

Script Output ->Query Result
Task completed in 0.229 seconds
Table AVION created.
```

5. Adaugati informatii coerente în tabelele create (minim 5 inregistrari pentru fiecare entitate independenta; minim 10 inregistrari pentru tabela asociativa).

```
CREATE SEQUENCE CHEIE
INCREMENT by 10
START WITH 10
MAXVALUE 10000
NOCYCLE;

--PENTRU LOCATIE
INSERT INTO LOCATIE(cod_locatie, adresa, tara, oras,
limba_oficiala, moneda_oficiala)

VALUES (CHEIE.NEXTVAL, 'JFK Access Rd, NY 11430', 'Statele Unite
ale Americii', 'New York', 'engleza', 'USD');

INSERT INTO LOCATIE(cod_locatie, adresa, tara, oras,
limba_oficiala, moneda_oficiala)

VALUES (CHEIE.NEXTVAL, 'Longford TW6', 'Regatul Unit', 'Londra',
'engleza', 'GBP');

INSERT INTO LOCATIE(cod_locatie, adresa, tara, oras,
limba_oficiala, moneda_oficiala)

VALUES (CHEIE.NEXTVAL, '1 Sky Plaza Rd', 'Hong Kong', 'Hong
Kong', null, 'HKD');

INSERT INTO LOCATIE(cod_locatie, adresa, tara, oras,
limba_oficiala, moneda_oficiala)
```

```
VALUES (CHEIE.NEXTVAL, '95700 Roissy-en-France', 'Franta',
'Paris', 'franceza', 'EUR');

INSERT INTO LOCATIE(cod_locatie, adresa, tara, oras,
limba_oficiala, moneda_oficiala)

VALUES (CHEIE.NEXTVAL, '60547 Frankfurt', 'Germania',
'Frankfurt', 'germania', 'EUR');

INSERT INTO LOCATIE(cod_locatie, adresa, tara, oras,
limba_oficiala, moneda_oficiala)

VALUES (CHEIE.NEXTVAL, '272 Gonghang-ro, Jung-gu, Incheon',
'Coreea de Sud', 'Seul', 'coreeana', 'KRW');

INSERT INTO LOCATIE(cod_locatie, adresa, tara, oras,
limba_oficiala, moneda_oficiala)

VALUES (CHEIE.NEXTVAL, 'Delhi 110037', 'India', 'New Delhi',
'hindi', 'INR');

INSERT INTO LOCATIE(cod_locatie, adresa, tara, oras,
limba_oficiala, moneda_oficiala)

VALUES (CHEIE.NEXTVAL, 'Mascot NSW 2020', 'Australia', 'Sydney',
'engleza', 'AUD');

INSERT INTO LOCATIE(cod_locatie, adresa, tara, oras,
limba_oficiala, moneda_oficiala)

VALUES (CHEIE.NEXTVAL, '6301 Silver Dart Dr, Mississauga, L5P
1B2', 'Canada', 'Toronto', 'engleza', 'CAD');
```

```
INSERT INTO LOCATIE(cod_locatie, adresa, tara, oras,
limba_oficiala, moneda_oficiala)

VALUES (CHEIE.NEXTVAL, 'Calea Bucurestilor', 'Romania',
'Bucuresti', 'romana', 'RON');

INSERT INTO LOCATIE(cod_locatie, adresa, tara, oras,
limba_oficiala, moneda_oficiala)

VALUES (CHEIE.NEXTVAL, 'Strada Aeroportului 1', 'Romania',
'Iasi', 'romana', 'RON');

INSERT INTO LOCATIE(cod_locatie, adresa, tara, oras,
limba_oficiala, moneda_oficiala)

VALUES (CHEIE.NEXTVAL, 'Strada Aeroportului', 'Romania',
'Craiova', 'romana', 'RON');

INSERT INTO LOCATIE(cod_locatie, adresa, tara, oras,
limba_oficiala, moneda_oficiala)

VALUES (CHEIE.NEXTVAL, 'Via dell Aeroporto di Fiumicino',
'Italia', 'Roma', 'italiana', 'EUR');

--INSERARI PENTRU AEROPORT

INSERT INTO AEROPORT(cod_aeroport, cod_locatie, nume,
zbor_international)

VALUES (CHEIE.NEXTVAL, 10, 'John F. Kennedy International
Airport', 'DA');

INSERT INTO AEROPORT(cod_aeroport, cod_locatie, nume,
zbor_international)
```

```
VALUES (CHEIE.NEXTVAL, 20, 'Aeroportul Internațional Heathrow', 'DA');
```

```
INSERT INTO AEROPORT(cod_aeroport, cod_locatie, nume,  
zbor_international)
```

```
VALUES (CHEIE.NEXTVAL, 30, 'Aeroportul Internațional de la Hong Kong', 'DA');
```

```
INSERT INTO AEROPORT(cod_aeroport, cod_locatie, nume,  
zbor_international)
```

```
VALUES (CHEIE.NEXTVAL, 40, 'Aeroportul Internațional Charles de Gaulle', 'DA');
```

```
INSERT INTO AEROPORT(cod_aeroport, cod_locatie, nume,  
zbor_international)
```

```
VALUES (CHEIE.NEXTVAL, 50, 'Aeroportul Internațional de la Frankfurt', 'DA');
```

```
INSERT INTO AEROPORT(cod_aeroport, cod_locatie, nume,  
zbor_international)
```

```
VALUES (CHEIE.NEXTVAL, 60, 'Aeroportul Internațional Incheon', 'DA');
```

```
INSERT INTO AEROPORT(cod_aeroport, cod_locatie, nume,  
zbor_international)
```

```
VALUES (CHEIE.NEXTVAL, 70, 'Aeroportul Internațional Indira Gandhi', 'DA');
```

```
INSERT INTO AEROPORT(cod_aeroport, cod_locatie, nume,  
zbor_international)
```

```
VALUES (CHEIE.NEXTVAL, 80, 'Aeroportul Internațional din Sydney', 'DA');
```

```
INSERT INTO AEROPORT(cod_aeroport, cod_locatie, nume,
zbor_international)

VALUES (CHEIE.NEXTVAL, 90, 'Aeroportul Internațional Pearson Toronto', 'DA');

INSERT INTO AEROPORT(cod_aeroport, cod_locatie, nume,
zbor_international)

VALUES (CHEIE.NEXTVAL, 100, 'Aeroportul Internațional Henri
Coandă București', 'DA');

INSERT INTO AEROPORT(cod_aeroport, cod_locatie, nume,
zbor_international)

VALUES (CHEIE.NEXTVAL, 110, 'Aeroportul Internațional Iași', 'DA');

INSERT INTO AEROPORT(cod_aeroport, cod_locatie, nume,
zbor_international)

VALUES (CHEIE.NEXTVAL, 120, 'Aeroportul Internațional Craiova', 'DA');

INSERT INTO AEROPORT(cod_aeroport, cod_locatie, nume,
zbor_international)

VALUES (CHEIE.NEXTVAL, 130, 'Aeroportul Internațional Leonardo da Vinci', 'DA');

--INSERTURI PENTRU TERMINAL

INSERT INTO TERMINAL(cod_terminal, cod_aeroport, capacitate,
pozitie)

VALUES (CHEIE.NEXTVAL, 140, 2500, 'SUD');
```

```
INSERT INTO TERMINAL(cod_terminal, cod_aeroport, capacitate,  
pozitie)  
  
VALUES (CHEIE.NEXTVAL, 140, 1800, 'NORD-VEST');  
INSERT INTO TERMINAL(cod_terminal, cod_aeroport, capacitate,  
pozitie)  
  
VALUES (CHEIE.NEXTVAL, 140, 3000, 'NORD-EST');  
  
INSERT INTO TERMINAL(cod_terminal, cod_aeroport, capacitate,  
pozitie)  
  
VALUES (CHEIE.NEXTVAL, 150, 3000, 'CENTRAL');  
  
INSERT INTO TERMINAL(cod_terminal, cod_aeroport, capacitate,  
pozitie)  
  
VALUES (CHEIE.NEXTVAL, 160, 2200, 'EST');  
  
INSERT INTO TERMINAL(cod_terminal, cod_aeroport, capacitate,  
pozitie)  
  
VALUES (CHEIE.NEXTVAL, 160, 1500, 'VEST');  
  
INSERT INTO TERMINAL(cod_terminal, cod_aeroport, capacitate,  
pozitie)  
  
VALUES (CHEIE.NEXTVAL, 170, 2800, 'NORD');  
  
INSERT INTO TERMINAL(cod_terminal, cod_aeroport, capacitate,  
pozitie)  
  
VALUES (CHEIE.NEXTVAL, 170, 2100, 'SUD');
```

```
INSERT INTO TERMINAL(cod_terminal, cod_aeroport, capacitate,  
pozitie)  
  
VALUES (CHEIE.NEXTVAL, 180, 3200, 'EST');  
  
INSERT INTO TERMINAL(cod_terminal, cod_aeroport, capacitate,  
pozitie)  
  
VALUES (CHEIE.NEXTVAL, 180, 2500, 'SUD-VEST');  
  
INSERT INTO TERMINAL(cod_terminal, cod_aeroport, capacitate,  
pozitie)  
  
VALUES (CHEIE.NEXTVAL, 190, 2400, 'NORD');  
  
INSERT INTO TERMINAL(cod_terminal, cod_aeroport, capacitate,  
pozitie)  
  
VALUES (CHEIE.NEXTVAL, 200, 2800, 'SUD-EST');  
  
INSERT INTO TERMINAL(cod_terminal, cod_aeroport, capacitate,  
pozitie)  
  
VALUES (CHEIE.NEXTVAL, 210, 3500, 'CENTRAL');  
  
INSERT INTO TERMINAL(cod_terminal, cod_aeroport, capacitate,  
pozitie)  
  
VALUES (CHEIE.NEXTVAL, 220, 1800, 'NORD-VEST');  
  
INSERT INTO TERMINAL(cod_terminal, cod_aeroport, capacitate,  
pozitie)
```

```
VALUES (CHEIE.NEXTVAL, 230, 2000, 'VEST');

INSERT INTO TERMINAL(cod_terminal, cod_aeroport, capacitate,
pozitie)

VALUES (CHEIE.NEXTVAL, 230, 1500, 'SUD-EST');

INSERT INTO TERMINAL(cod_terminal, cod_aeroport, capacitate,
pozitie)

VALUES (CHEIE.NEXTVAL, 240, 600, 'EST');

INSERT INTO TERMINAL(cod_terminal, cod_aeroport, capacitate,
pozitie)

VALUES (CHEIE.NEXTVAL, 250, 850, 'NORD');

INSERT INTO TERMINAL(cod_terminal, cod_aeroport, capacitate,
pozitie)

VALUES (CHEIE.NEXTVAL, 260, 3000, 'SUD');

INSERT INTO TERMINAL(cod_terminal, cod_aeroport, capacitate,
pozitie)

VALUES (CHEIE.NEXTVAL, 260, 2500, 'CENTRAL');

--INSERTURI PENTRU POARTA

INSERT INTO POARTA(cod_poarta, cod_terminal, capacitate, pozitie)
```

```
VALUES(CHEIE.NEXTVAL, 270, 400, 'NORD');

INSERT INTO POARTA(cod_poarta, cod_terminal, capacitate, pozitie)
VALUES(CHEIE.NEXTVAL, 270, 200, 'SUD');

INSERT INTO POARTA(cod_poarta, cod_terminal, capacitate, pozitie)
VALUES(CHEIE.NEXTVAL, 280, 350, 'NORD');

INSERT INTO POARTA(cod_poarta, cod_terminal, capacitate, pozitie)
VALUES(CHEIE.NEXTVAL, 290, 400, 'CENTRAL');

INSERT INTO POARTA(cod_poarta, cod_terminal, capacitate, pozitie)
VALUES(CHEIE.NEXTVAL, 290, 290, 'EST');

INSERT INTO POARTA(cod_poarta, cod_terminal, capacitate, pozitie)
VALUES(CHEIE.NEXTVAL, 290, 400, 'NORD');

INSERT INTO POARTA(cod_poarta, cod_terminal, capacitate, pozitie)
VALUES(CHEIE.NEXTVAL, 300, 380, 'VEST');

INSERT INTO POARTA(cod_poarta, cod_terminal, capacitate, pozitie)
VALUES(CHEIE.NEXTVAL, 310, 200, 'NORD');

INSERT INTO POARTA(cod_poarta, cod_terminal, capacitate, pozitie)
```

```
VALUES(CHEIE.NEXTVAL, 320, 400, 'NORD');

INSERT INTO POARTA(cod_poarta, cod_terminal, capacitate, pozitie)
VALUES(CHEIE.NEXTVAL, 320, 340, 'SUD');

INSERT INTO POARTA(cod_poarta, cod_terminal, capacitate, pozitie)
VALUES(CHEIE.NEXTVAL, 330, 100, 'CENTRAL');

INSERT INTO POARTA(cod_poarta, cod_terminal, capacitate, pozitie)
VALUES(CHEIE.NEXTVAL, 340, 180, 'NORD-EST');

INSERT INTO POARTA(cod_poarta, cod_terminal, capacitate, pozitie)
VALUES(CHEIE.NEXTVAL, 350, 400, 'NORD-VEST');

INSERT INTO POARTA(cod_poarta, cod_terminal, capacitate, pozitie)
VALUES(CHEIE.NEXTVAL, 360, 500, 'SUD-EST');

INSERT INTO POARTA(cod_poarta, cod_terminal, capacitate, pozitie)
VALUES(CHEIE.NEXTVAL, 370, 400, 'NORD');

INSERT INTO POARTA(cod_poarta, cod_terminal, capacitate, pozitie)
VALUES(CHEIE.NEXTVAL, 380, 300, 'CENTRAL');
```

```
INSERT INTO POARTA(cod_poarta, cod_terminal, capacitate, pozitie)
VALUES(CHEIE.NEXTVAL, 390, 400, 'NORD');

INSERT INTO POARTA(cod_poarta, cod_terminal, capacitate, pozitie)
VALUES(CHEIE.NEXTVAL, 400, 100, 'EST');

INSERT INTO POARTA(cod_poarta, cod_terminal, capacitate, pozitie)
VALUES(CHEIE.NEXTVAL, 410, 400, 'NORD');

INSERT INTO POARTA(cod_poarta, cod_terminal, capacitate, pozitie)
VALUES(CHEIE.NEXTVAL, 410, 400, 'SUD');

INSERT INTO POARTA(cod_poarta, cod_terminal, capacitate, pozitie)
VALUES(CHEIE.NEXTVAL, 420, 200, 'VEST');

INSERT INTO POARTA(cod_poarta, cod_terminal, capacitate, pozitie)
VALUES(CHEIE.NEXTVAL, 430, 256, 'NORD');

INSERT INTO POARTA(cod_poarta, cod_terminal, capacitate, pozitie)
VALUES(CHEIE.NEXTVAL, 440, 420, 'SUD-EST');

INSERT INTO POARTA(cod_poarta, cod_terminal, capacitate, pozitie)
```

```
VALUES(CHEIE.NEXTVAL, 450, 400, 'NORD');

INSERT INTO POARTA(cod_poarta, cod_terminal, capacitate, pozitie)
VALUES(CHEIE.NEXTVAL, 460, 200, 'EST');

--INSERTURI PENTRU PISTA

INSERT INTO PISTA(cod_pista, lungime, latime, orientare)
VALUES (CHEIE.NEXTVAL, 3200, 45, 'Nord');

INSERT INTO PISTA(cod_pista, lungime, latime, orientare)
VALUES (CHEIE.NEXTVAL, 3800, 60, 'Est');

INSERT INTO PISTA(cod_pista, lungime, latime, orientare)
VALUES (CHEIE.NEXTVAL, 3000, 50, 'Nord-Vest');

INSERT INTO PISTA(cod_pista, lungime, latime, orientare)
VALUES (CHEIE.NEXTVAL, 2800, 55, 'Sud');

INSERT INTO PISTA(cod_pista, lungime, latime, orientare)
VALUES (CHEIE.NEXTVAL, 3500, 40, 'Sud-Est');

INSERT INTO PISTA(cod_pista, lungime, latime, orientare)
```

```
VALUES (CHEIE.NEXTVAL, 4000, 70, 'Vest');

INSERT INTO PISTA(cod_pista, lungime, latime, orientare)
VALUES (CHEIE.NEXTVAL, 3200, 45, 'Nord-Est');

INSERT INTO PISTA(cod_pista, lungime, latime, orientare)
VALUES (CHEIE.NEXTVAL, 3700, 55, 'Sud-Vest');
INSERT INTO PISTA(cod_pista, lungime, latime, orientare)
VALUES (CHEIE.NEXTVAL, 3100, 60, 'Est-Vest');

INSERT INTO PISTA(cod_pista, lungime, latime, orientare)
VALUES (CHEIE.NEXTVAL, 2900, 50, 'Nord-Sud');

INSERT INTO PISTA(cod_pista, lungime, latime, orientare)
VALUES (CHEIE.NEXTVAL, 3300, 40, 'Sud-Est');

INSERT INTO PISTA(cod_pista, lungime, latime, orientare)
VALUES (CHEIE.NEXTVAL, 3600, 70, 'Vest-Est');

INSERT INTO PISTA(cod_pista, lungime, latime, orientare)
VALUES (CHEIE.NEXTVAL, 3200, 45, 'Nord-Vest');

INSERT INTO PISTA(cod_pista, lungime, latime, orientare)
VALUES (CHEIE.NEXTVAL, 3800, 60, 'Est-Nord');
```

```
INSERT INTO PISTA(cod_pista, lungime, latime, orientare)
VALUES (CHEIE.NEXTVAL, 3000, 50, 'Nord-Est');

INSERT INTO PISTA(cod_pista, lungime, latime, orientare)
VALUES (CHEIE.NEXTVAL, 2800, 55, 'Sud-Vest');

INSERT INTO PISTA(cod_pista, lungime, latime, orientare)
VALUES (CHEIE.NEXTVAL, 3500, 40, 'Sud-Est');

INSERT INTO PISTA(cod_pista, lungime, latime, orientare)
VALUES (CHEIE.NEXTVAL, 4000, 70, 'Vest-Sud');

INSERT INTO PISTA(cod_pista, lungime, latime, orientare)
VALUES (CHEIE.NEXTVAL, 3200, 45, 'Nord-Est');

INSERT INTO PISTA(cod_pista, lungime, latime, orientare)
VALUES (CHEIE.NEXTVAL, 3100, 60, 'Est-Sud');

INSERT INTO PISTA(cod_pista, lungime, latime, orientare)
VALUES (CHEIE.NEXTVAL, 2900, 50, 'Nord-Vest');

--INSERTURI PENTRU RUTA
```

```
INSERT INTO RUTA(cod_ruta, escala, distanta, timp)

VALUES(CHEIE.NEXTVAL, null, 2500, 250);

INSERT INTO RUTA(cod_ruta, escala, distanta, timp)

VALUES(CHEIE.NEXTVAL, null, 6000, 530);

INSERT INTO RUTA(cod_ruta, escala, distanta, timp)

VALUES(CHEIE.NEXTVAL, null, 1000, 210);

INSERT INTO RUTA(cod_ruta, escala, distanta, timp)

VALUES(CHEIE.NEXTVAL, null, 2000, 430);

INSERT INTO RUTA(cod_ruta, escala, distanta, timp)

VALUES(CHEIE.NEXTVAL, 'Aeroportul International Charles de Gaulle', 5500, 650);

INSERT INTO RUTA(cod_ruta, escala, distanta, timp)

VALUES(CHEIE.NEXTVAL, null, 800, 150);

INSERT INTO RUTA(cod_ruta, escala, distanta, timp)

VALUES(CHEIE.NEXTVAL, null, 5500, 530);
```

```
INSERT INTO RUTA(cod_ruta, escala, distanta, timp)

VALUES(CHEIE.NEXTVAL, null, 1800, 340);

INSERT INTO RUTA(cod_ruta, escala, distanta, timp)

VALUES(CHEIE.NEXTVAL, null, 1500, 370);

INSERT INTO RUTA(cod_ruta, escala, distanta, timp)

VALUES(CHEIE.NEXTVAL, 'Aeroportul International Henri Coanda
Bucuresti', 700, 300);

--INSERTURI PENTRU AVION

INSERT INTO AVION (cod_avion, numar_locuri, inaltime, lungime,
zbor_intercontinental, model)

VALUES (CHEIE.NEXTVAL, 150, 10, 15, 'DA', 'Boeing 747');

INSERT INTO AVION (cod_avion, numar_locuri, inaltime, lungime,
zbor_intercontinental, model)

VALUES (CHEIE.NEXTVAL, 100, 8, 12, 'NU', 'Airbus A320');

INSERT INTO AVION (cod_avion, numar_locuri, inaltime, lungime,
zbor_intercontinental, model)

VALUES (CHEIE.NEXTVAL, 200, 12, 18, 'DA', 'Boeing 787');

INSERT INTO AVION (cod_avion, numar_locuri, inaltime, lungime,
zbor_intercontinental, model)
```

```
VALUES (CHEIE.NEXTVAL, 80, 6, 10, 'NU', 'Embraer E190');

INSERT INTO AVION (cod_avion, numar_locuri, inaltime, lungime,
zbor_intercontinental, model)

VALUES (CHEIE.NEXTVAL, 250, 15, 20, 'DA', 'Airbus A380');

INSERT INTO AVION (cod_avion, numar_locuri, inaltime, lungime,
zbor_intercontinental, model)

VALUES (CHEIE.NEXTVAL, 120, 9, 14, 'NU', 'Boeing 737');

INSERT INTO AVION (cod_avion, numar_locuri, inaltime, lungime,
zbor_intercontinental, model)

VALUES (CHEIE.NEXTVAL, 180, 11, 16, 'DA', 'Airbus A350');

INSERT INTO AVION (cod_avion, numar_locuri, inaltime, lungime,
zbor_intercontinental, model)

VALUES (CHEIE.NEXTVAL, 90, 7, 11, 'NU', 'Embraer E195');

--INSERTURI PENTRU ZBOR

INSERT INTO ZBOR(cod_zbor, data, intarziere, cod_avion,
cod_pista1, cod_pista2, cod_ruta)

VALUES (CHEIE.NEXTVAL, TO_DATE('2023-05-17', 'YYYY-MM-DD'), 10,
1080, 820, 780, 930);
```

```
INSERT INTO ZBOR(cod_zbor, data, intarziere, cod_avion,
cod_pista1, cod_pista2, cod_ruta)

VALUES (CHEIE.NEXTVAL, TO_DATE('2023-05-18', 'YYYY-MM-DD'), 0,
1050, 720, 840, 940);

INSERT INTO ZBOR(cod_zbor, data, intarziere, cod_avion,
cod_pista1, cod_pista2, cod_ruta)

VALUES (CHEIE.NEXTVAL, TO_DATE('2023-05-19', 'YYYY-MM-DD'), 0,
1030, 790, 870, 950);

INSERT INTO ZBOR(cod_zbor, data, intarziere, cod_avion,
cod_pista1, cod_pista2, cod_ruta)

VALUES (CHEIE.NEXTVAL, TO_DATE('2023-05-19', 'YYYY-MM-DD'), 45,
1040, 720, 850, 960);

INSERT INTO ZBOR(cod_zbor, data, intarziere, cod_avion,
cod_pista1, cod_pista2, cod_ruta)

VALUES (CHEIE.NEXTVAL, TO_DATE('2023-05-21', 'YYYY-MM-DD'), 10,
1040, 750, 820, 970);

INSERT INTO ZBOR(cod_zbor, data, intarziere, cod_avion,
cod_pista1, cod_pista2, cod_ruta)

VALUES (CHEIE.NEXTVAL, TO_DATE('2023-05-22', 'YYYY-MM-DD'), 0,
1090, 750, 910, 980);

INSERT INTO ZBOR(cod_zbor, data, intarziere, cod_avion,
cod_pista1, cod_pista2, cod_ruta)
```

```
VALUES (CHEIE.NEXTVAL, TO_DATE('2023-05-23', 'YYYY-MM-DD'), 0,  
1030, 830, 750, 990);
```

```
INSERT INTO ZBOR(cod_zbor, data, intarziere, cod_avion,  
cod_pista1, cod_pista2, cod_ruta)
```

```
VALUES (CHEIE.NEXTVAL, TO_DATE('2023-05-24', 'YYYY-MM-DD'), 0,  
1050, 910, 770, 1000);
```

```
INSERT INTO ZBOR(cod_zbor, data, intarziere, cod_avion,  
cod_pista1, cod_pista2, cod_ruta)
```

```
VALUES (CHEIE.NEXTVAL, TO_DATE('2023-05-25', 'YYYY-MM-DD'), 0,  
1070, 750, 810, 1010);
```

```
INSERT INTO ZBOR(cod_zbor, data, intarziere, cod_avion,  
cod_pista1, cod_pista2, cod_ruta)
```

```
VALUES (CHEIE.NEXTVAL, TO_DATE('2023-05-26', 'YYYY-MM-DD'), 0,  
1060, 880, 800, 1020);
```

```
--INSERTURI PENTRU REZERVARE
```

```
INSERT INTO REZERVARE(cod_rezervare, pret_total)
```

```
VALUES(CHEIE.NEXTVAL,515);
```

```
INSERT INTO REZERVARE(cod_rezervare, pret_total)
```

```
VALUES(CHEIE.NEXTVAL,1650);
```

```
INSERT INTO REZERVARE(cod_rezervare, pret_total)  
VALUES(CHEIE.NEXTVAL,1255);
```

```
INSERT INTO REZERVARE(cod_rezervare, pret_total)  
VALUES(CHEIE.NEXTVAL,2420);
```

```
INSERT INTO REZERVARE(cod_rezervare, pret_total)  
VALUES(CHEIE.NEXTVAL,3120);
```

```
INSERT INTO REZERVARE(cod_rezervare, pret_total)  
VALUES(CHEIE.NEXTVAL,1115);
```

```
INSERT INTO REZERVARE(cod_rezervare, pret_total)  
VALUES(CHEIE.NEXTVAL,1010);
```

```
INSERT INTO REZERVARE(cod_rezervare, pret_total)  
VALUES(CHEIE.NEXTVAL,1100);
```

```
INSERT INTO REZERVARE(cod_rezervare, pret_total)  
VALUES(CHEIE.NEXTVAL,1100);
```

```
INSERT INTO REZERVARE(cod_rezervare, pret_total)
```

```
VALUES(CHEIE.NEXTVAL,1200);

--INSERTURI PENTRU CLASA

INSERT INTO CLASA(cod_clasa, pret)

VALUES (CHEIE.NEXTVAL, 500);

INSERT INTO CLASA(cod_clasa, pret)

VALUES (CHEIE.NEXTVAL, 800);

INSERT INTO CLASA(cod_clasa, pret)

VALUES (CHEIE.NEXTVAL, 1200);

INSERT INTO CLASA(cod_clasa, pret)

VALUES (CHEIE.NEXTVAL, 1100);

INSERT INTO CLASA(cod_clasa, pret)

VALUES (CHEIE.NEXTVAL, 1000);

--INSERTURI PENTRU LOC

INSERT INTO LOC(cod_loc, cod_avion, pozitie, numar_loc,
cod_clasa)

VALUES(CHEIE.NEXTVAL, 1030, 'Centru', 14, 1310);
```

```
INSERT INTO LOC(cod_loc, cod_avion, pozitie, numar_loc,  
cod_clasa)
```

```
VALUES(CHEIE.NEXTVAL, 1030, 'Geam', 18, 1310);
```

```
INSERT INTO LOC(cod_loc, cod_avion, pozitie, numar_loc,  
cod_clasa)
```

```
VALUES(CHEIE.NEXTVAL, 1040, 'Margine', 14, 1330);
```

```
INSERT INTO LOC(cod_loc, cod_avion, pozitie, numar_loc,  
cod_clasa)
```

```
VALUES(CHEIE.NEXTVAL, 1050, 'Central', 20, 1330);
```

```
INSERT INTO LOC(cod_loc, cod_avion, pozitie, numar_loc,  
cod_clasa)
```

```
VALUES(CHEIE.NEXTVAL, 1060, 'Geam', 21, 1340);
```

```
INSERT INTO LOC(cod_loc, cod_avion, pozitie, numar_loc,  
cod_clasa)
```

```
VALUES(CHEIE.NEXTVAL, 1070, 'Geam', 67, 1350);
```

```
INSERT INTO LOC(cod_loc, cod_avion, pozitie, numar_loc,  
cod_clasa)
```

```
VALUES(CHEIE.NEXTVAL, 1080, 'Central', 114, 1320);
```

```
INSERT INTO LOC(cod_loc, cod_avion, pozitie, numar_loc,  
cod_clasa)
```

```
VALUES(CHEIE.NEXTVAL, 1090, 'Geam', 22, 1350);
```

```
INSERT INTO LOC(cod_loc, cod_avion, pozitie, numar_loc,
cod_clasa)

VALUES(CHEIE.NEXTVAL, 1090, 'Margine', 22, 1320);

INSERT INTO LOC(cod_loc, cod_avion, pozitie, numar_loc,
cod_clasa)

VALUES(CHEIE.NEXTVAL, 1090, 'Margine', 12, 1320);

INSERT INTO LOC(cod_loc, cod_avion, pozitie, numar_loc,
cod_clasa)

VALUES(CHEIE.NEXTVAL, 1090, 'Geam', 35, 1340);

INSERT INTO LOC(cod_loc, cod_avion, pozitie, numar_loc,
cod_clasa)

VALUES(CHEIE.NEXTVAL, 1090, 'Geam', 1, 1310);

INSERT INTO LOC(cod_loc, cod_avion, pozitie, numar_loc,
cod_clasa)

VALUES(CHEIE.NEXTVAL, 1090, 'Central', 98, 1340);

INSERT INTO LOC(cod_loc, cod_avion, pozitie, numar_loc,
cod_clasa)

VALUES(CHEIE.NEXTVAL, 1070, 'Central', 2, 1330);
```

```
INSERT INTO LOC(cod_loc, cod_avion, pozitie, numar_loc,  
cod_clasa)
```

```
VALUES(CHEIE.NEXTVAL, 1050, 'Margine', 26, 1330);
```

```
--INSERTURI PENTRU FACILITATE
```

```
INSERT INTO FACILITATE(cod_facilitate, nume_facilitate,  
pret_suplimentar, standard)
```

```
VALUES(CHEIE.NEXTVAL, 'WiFi', 10, 'DA');
```

```
INSERT INTO FACILITATE(cod_facilitate, nume_facilitate,  
pret_suplimentar, standard)
```

```
VALUES(CHEIE.NEXTVAL, 'Masa calda', 15, 'DA');
```

```
INSERT INTO FACILITATE(cod_facilitate, nume_facilitate,  
pret_suplimentar, standard)
```

```
VALUES(CHEIE.NEXTVAL, 'Divertisment la bord', 5, 'DA');
```

```
INSERT INTO FACILITATE(cod_facilitate, nume_facilitate,  
pret_suplimentar, standard)
```

```
VALUES(CHEIE.NEXTVAL, 'Prioritate la Imbarcare', 8, 'NU');
```

```
INSERT INTO FACILITATE(cod_facilitate, nume_facilitate,  
pret_suplimentar, standard)
```

```
VALUES(CHEIE.NEXTVAL, 'Bagaj de cala', 12, 'NU');
```

```
INSERT INTO FACILITATE(cod_facilitate, nume_facilitate,
pret_suplimentar, standard)

VALUES(CHEIE.NEXTVAL, 'Acces la salonul VIP', 20, 'DA');

--INSERTURI PENTRU PASAGER

INSERT INTO PASAGER(cod_pasager, nume, prenume, nationalitate,
numar_pasaport)

VALUES(CHEIE.NEXTVAL, 'Popescu', 'Ion', 'Română', 12345678);

INSERT INTO PASAGER(cod_pasager, nume, prenume, nationalitate,
numar_pasaport)

VALUES(CHEIE.NEXTVAL, 'Ionescu', 'Maria', 'Română', 87654321);

INSERT INTO PASAGER(cod_pasager, nume, prenume, nationalitate,
numar_pasaport)

VALUES(CHEIE.NEXTVAL, 'Smith', 'John', 'Engleză', 98765432);

INSERT INTO PASAGER(cod_pasager, nume, prenume, nationalitate,
numar_pasaport)

VALUES(CHEIE.NEXTVAL, 'Müller', 'Hans', 'Germană', 54321678);

INSERT INTO PASAGER(cod_pasager, nume, prenume, nationalitate,
numar_pasaport)

VALUES(CHEIE.NEXTVAL, 'Rossi', 'Giovanni', 'Italiană', 87654321);
```

```
INSERT INTO PASAGER(cod_pasager, nume, prenume, nationalitate,
numar_pasaport)

VALUES(CHEIE.NEXTVAL, 'López', null, 'Spaniolă', 43218765);

INSERT INTO PASAGER(cod_pasager, nume, prenume, nationalitate,
numar_pasaport)

VALUES(CHEIE.NEXTVAL, 'Dupont', 'Jean', 'Franceză', 65432187);

INSERT INTO PASAGER(cod_pasager, nume, prenume, nationalitate,
numar_pasaport)

VALUES(CHEIE.NEXTVAL, 'Kovács', 'János', 'Maghiară', 87654321);

INSERT INTO PASAGER(cod_pasager, nume, prenume, nationalitate,
numar_pasaport)

VALUES(CHEIE.NEXTVAL, 'Sato', 'Takeshi', 'Japoneză', 12348765);

INSERT INTO PASAGER(cod_pasager, nume, prenume, nationalitate,
numar_pasaport)

VALUES(CHEIE.NEXTVAL, 'González', 'María', 'Mexicană', 87651234);

--INSERTURI PENTRU ANGAJAT

INSERT INTO ANGAJAT(cod_angajat, nume, prenume, salariu,
data_nasterii, data_angajarii, nationalitate)
VALUES(CHEIE.NEXTVAL, 'Popescu', 'Ion', 8000,
TO_DATE('01-01-1990', 'dd-mm-yyyy'), TO_DATE('01-01-2020',
'dd-mm-yyyy'), 'Romana');
```

```
INSERT INTO ANGAJAT(cod_angajat, nume, prenume, salariu,
data_nasterii, data_angajarii, nationalitate)
VALUES(CHEIE.NEXTVAL, 'Ionescu', 'Andrei', 12000,
TO_DATE('15-02-1985', 'dd-mm-yyyy'), TO_DATE('01-07-2015',
'dd-mm-yyyy'), 'Romana');

INSERT INTO ANGAJAT(cod_angajat, nume, prenume, salariu,
data_nasterii, data_angajarii, nationalitate)
VALUES(CHEIE.NEXTVAL, 'Popa', 'Maria', 15000,
TO_DATE('10-05-1992', 'dd-mm-yyyy'), TO_DATE('15-03-2021',
'dd-mm-yyyy'), 'Italiana');

INSERT INTO ANGAJAT(cod_angajat, nume, prenume, salariu,
data_nasterii, data_angajarii, nationalitate)
VALUES(CHEIE.NEXTVAL, 'Muller', 'Hans', 10000,
TO_DATE('20-06-1988', 'dd-mm-yyyy'), TO_DATE('10-09-2017',
'dd-mm-yyyy'), 'Germana');

INSERT INTO ANGAJAT(cod_angajat, nume, prenume, salariu,
data_nasterii, data_angajarii, nationalitate)
VALUES(CHEIE.NEXTVAL, 'Smith', 'John', 18000,
TO_DATE('03-12-1995', 'dd-mm-yyyy'), TO_DATE('25-05-2019',
'dd-mm-yyyy'), 'Engleza');

INSERT INTO ANGAJAT(cod_angajat, nume, prenume, salariu,
data_nasterii, data_angajarii, nationalitate)
VALUES(CHEIE.NEXTVAL, 'Lopez', 'Maria', 12000,
TO_DATE('08-09-1991', 'dd-mm-yyyy'), TO_DATE('18-11-2018',
'dd-mm-yyyy'), 'Spaniola');

INSERT INTO ANGAJAT(cod_angajat, nume, prenume, salariu,
data_nasterii, data_angajarii, nationalitate)
VALUES(CHEIE.NEXTVAL, 'Dupont', 'Jean', 9000,
TO_DATE('12-04-1987', 'dd-mm-yyyy'), TO_DATE('05-02-2016',
'dd-mm-yyyy'), 'Franceza');

INSERT INTO ANGAJAT(cod_angajat, nume, prenume, salariu,
data_nasterii, data_angajarii, nationalitate)
```

```
VALUES(CHEIE.NEXTVAL, 'Kim', 'Min-ji', 15000,
TO_DATE('29-07-1993', 'dd-mm-yyyy'), TO_DATE('12-08-2020',
'dd-mm-yyyy'), 'Coreeana');
```

```
INSERT INTO ANGAJAT(cod_angajat, nume, prenume, salariu,
data_nasterii, data_angajarii, nationalitate)
VALUES(CHEIE.NEXTVAL, 'Muhammad', 'Ahmed', 16000,
TO_DATE('17-11-1994', 'dd-mm-yyyy'), TO_DATE('30-07-2019',
'dd-mm-yyyy'), 'Pachistaneza');
```

```
INSERT INTO ANGAJAT(cod_angajat, nume, prenume, salariu,
data_nasterii, data_angajarii, nationalitate)
VALUES(CHEIE.NEXTVAL, 'James', 'Anna', 7000,
TO_DATE('17-10-1994', 'dd-mm-yyyy'), TO_DATE('20-07-2019',
'dd-mm-yyyy'), 'Engleza');
```

```
--INSERTURI PENTRU PILOT
```

```
INSERT INTO PILOT(cod_angajat, educatie, experienta)
VALUES(1670, 'Universitatea Politehnica Bucuresti', 5);
```

```
INSERT INTO PILOT(cod_angajat, educatie, experienta)
VALUES(1700, 'ETH Zurich', 8);
```

```
INSERT INTO PILOT(cod_angajat, educatie, experienta)
VALUES(1710, 'California Institute of Technology', 12);
```

```
INSERT INTO PILOT(cod_angajat, educatie, experienta)
VALUES(1690, 'Massachusetts Institute of Technology', 7);
```

```
--INSERTURI PENTRU INSOTITOR_DE_BORD

INSERT INTO INSOTITOR_DE_BORD(cod_angajat, experienta,
limba_straina)
VALUES(1740, 3, 'Engleza');

INSERT INTO INSOTITOR_DE_BORD(cod_angajat, experienta,
limba_straina)
VALUES(1750, 5, 'Franceza');

INSERT INTO INSOTITOR_DE_BORD(cod_angajat, experienta,
limba_straina)
VALUES(1720, 2, 'Germana');

INSERT INTO INSOTITOR_DE_BORD(cod_angajat, experienta,
limba_straina)
VALUES(1730, 4, 'Spaniola');

--INSERTURI PENTRU INGINER

INSERT INTO INGINER(cod_angajat, educatie)
VALUES(1680, 'Universitatea Politehnica Bucuresti');

INSERT INTO INGINER(cod_angajat, educatie)
VALUES(1760, 'Universitatea Politehnica Bucuresti');
```

```
--INSERTURI PENTRU ARE
```

```
INSERT INTO ARE(cod_aeroport1, cod_aeroport2, cod_ruta)  
VALUES(140,210,940);
```

```
INSERT INTO ARE(cod_aeroport1, cod_aeroport2, cod_ruta)  
VALUES(170,190,930);
```

```
INSERT INTO ARE(cod_aeroport1, cod_aeroport2, cod_ruta)  
VALUES(170,230,950);
```

```
INSERT INTO ARE(cod_aeroport1, cod_aeroport2, cod_ruta)  
VALUES(140,220,960);
```

```
INSERT INTO ARE(cod_aeroport1, cod_aeroport2, cod_ruta)  
VALUES(150,190,970);
```

```
INSERT INTO ARE(cod_aeroport1, cod_aeroport2, cod_ruta)  
VALUES(190,260,980);
```

```
INSERT INTO ARE(cod_aeroport1, cod_aeroport2, cod_ruta)  
VALUES(200,190,990);
```

```
INSERT INTO ARE(cod_aeroport1, cod_aeroport2, cod_ruta)
VALUES(260,160,1000);

INSERT INTO ARE(cod_aeroport1, cod_aeroport2, cod_ruta)
VALUES(150,180,1010);

INSERT INTO ARE(cod_aeroport1, cod_aeroport2, cod_ruta)
VALUES(240,180,1020);

--INSERTURI PENTRU CUMPARA

INSERT INTO CUMPARA(cod_rezervare, cod_pasager, numar_locuri)
VALUES(1210, 1570, 1);

INSERT INTO CUMPARA(cod_rezervare, cod_pasager, numar_locuri)
VALUES(1220, 1580, 2);

INSERT INTO CUMPARA(cod_rezervare, cod_pasager, numar_locuri)
VALUES(1230, 1590, 1);

INSERT INTO CUMPARA(cod_rezervare, cod_pasager, numar_locuri)
VALUES(1240, 1600, 2);
```

```
INSERT INTO CUMPARA(cod_rezervare, cod_pasager, numar_locuri)  
VALUES(1250, 1610, 3);
```

```
INSERT INTO CUMPARA(cod_rezervare, cod_pasager, numar_locuri)  
VALUES(1260, 1620, 1);
```

```
INSERT INTO CUMPARA(cod_rezervare, cod_pasager, numar_locuri)  
VALUES(1270, 1630, 1);
```

```
INSERT INTO CUMPARA(cod_rezervare, cod_pasager, numar_locuri)  
VALUES(1280, 1640, 1);
```

```
INSERT INTO CUMPARA(cod_rezervare, cod_pasager, numar_locuri)  
VALUES(1290, 1640, 1);
```

```
INSERT INTO CUMPARA(cod_rezervare, cod_pasager, numar_locuri)  
VALUES(1300, 1640, 1);
```

```
--INSERTURI PENTRU OFERA
```

```
INSERT INTO OFERA(cod_clasa, cod_facilitate)  
VALUES(1330,1510);
```

```
INSERT INTO OFERA(cod_clasa, cod_facilitate)
```

```
VALUES(1350,1510);

INSERT INTO OFERA(cod_clasa, cod_facilitate)
VALUES(1310,1520);

INSERT INTO OFERA(cod_clasa, cod_facilitate)
VALUES(1340,1520);
INSERT INTO OFERA(cod_clasa, cod_facilitate)
VALUES(1330,1530);

INSERT INTO OFERA(cod_clasa, cod_facilitate)
VALUES(1330,1540);

INSERT INTO OFERA(cod_clasa, cod_facilitate)
VALUES(1350,1550);

INSERT INTO OFERA(cod_clasa, cod_facilitate)
VALUES(1340,1550);

INSERT INTO OFERA(cod_clasa, cod_facilitate)
VALUES(1330,1550);

INSERT INTO OFERA(cod_clasa, cod_facilitate)
```

```
VALUES(1330,1560);

INSERT INTO OFERA(cod_clasa, cod_facilitate)
VALUES(1320,1520);

INSERT INTO OFERA(cod_clasa, cod_facilitate)
VALUES(1320,1510);

--INSERTURI PENTRU LUCREAZA

INSERT INTO LUCREAZA(cod_angajat, cod_zbor, ore, spor)
VALUES(1670, 1110, 5, 230);

INSERT INTO LUCREAZA(cod_angajat, cod_zbor, ore, spor)
VALUES(1700, 1120, 9, 370);

INSERT INTO LUCREAZA(cod_angajat, cod_zbor, ore, spor)
VALUES(1700, 1130, 4, 200);

INSERT INTO LUCREAZA(cod_angajat, cod_zbor, ore, spor)
VALUES(1700, 1140, 8, 325);

INSERT INTO LUCREAZA(cod_angajat, cod_zbor, ore, spor)
VALUES(1670, 1150, 12, 430);
```

```
INSERT INTO LUCREAZA(cod_angajat, cod_zbor, ore, spor)  
VALUES(1710, 1160, 3, 100);
```

```
INSERT INTO LUCREAZA(cod_angajat, cod_zbor, ore, spor)  
VALUES(1710, 1170, 9, 350);
```

```
INSERT INTO LUCREAZA(cod_angajat, cod_zbor, ore, spor)  
VALUES(1690, 1180, 8, 300);
```

```
INSERT INTO LUCREAZA(cod_angajat, cod_zbor, ore, spor)  
VALUES(1690, 1190, 7, 200);
```

```
INSERT INTO LUCREAZA(cod_angajat, cod_zbor, ore, spor)  
VALUES(1690, 1200, 5, 100);
```

```
INSERT INTO LUCREAZA(cod_angajat, cod_zbor, ore, spor)  
VALUES(1740, 1110, 5, 170);
```

```
INSERT INTO LUCREAZA(cod_angajat, cod_zbor, ore, spor)  
VALUES(1740, 1120, 9, 240);
```

```
INSERT INTO LUCREAZA(cod_angajat, cod_zbor, ore, spor)  
VALUES(1750, 1130, 4, 120);
```

```
INSERT INTO LUCREAZA(cod_angajat, cod_zbor, ore, spor)
VALUES(1750, 1140, 8, 160);
```

```
INSERT INTO LUCREAZA(cod_angajat, cod_zbor, ore, spor)
VALUES(1750, 1150, 12, 400);
```

```
INSERT INTO LUCREAZA(cod_angajat, cod_zbor, ore, spor)
VALUES(1720, 1160, 3, 70);
```

```
INSERT INTO LUCREAZA(cod_angajat, cod_zbor, ore, spor)
VALUES(1720, 1170, 9, 220);
```

```
INSERT INTO LUCREAZA(cod_angajat, cod_zbor, ore, spor)
VALUES(1740, 1180, 8, 100);
```

```
INSERT INTO LUCREAZA(cod_angajat, cod_zbor, ore, spor)
VALUES(1730, 1190, 7, 150);
```

```
INSERT INTO LUCREAZA(cod_angajat, cod_zbor, ore, spor)
VALUES(1730, 1200, 5, 200);
```

```
INSERT INTO LUCREAZA(cod_angajat, cod_zbor, ore, spor)
VALUES(1680, 1110, 1, 15);
```

```
INSERT INTO LUCREAZA(cod_angajat, cod_zbor, ore, spor)
VALUES(1680, 1120, 2, null);
```

```
INSERT INTO LUCREAZA(cod_angajat, cod_zbor, ore, spor)
VALUES(1680, 1130, 1, 14);
```

```
INSERT INTO LUCREAZA(cod_angajat, cod_zbor, ore, spor)
VALUES(1760, 1140, 2, null);
```

```
INSERT INTO LUCREAZA(cod_angajat, cod_zbor, ore, spor)
VALUES(1760, 1150, 2, null);
```

```
INSERT INTO LUCREAZA(cod_angajat, cod_zbor, ore, spor)
VALUES(1680, 1160, 1, null);
```

```
INSERT INTO LUCREAZA(cod_angajat, cod_zbor, ore, spor)
VALUES(1760, 1170, 2, 35);
```

```
INSERT INTO LUCREAZA(cod_angajat, cod_zbor, ore, spor)
VALUES(1760, 1180, 2, null);
```

```
INSERT INTO LUCREAZA(cod_angajat, cod_zbor, ore, spor)
```

```
VALUES(1760, 1190, 2, 40);

INSERT INTO LUCREAZA(cod_angajat, cod_zbor, ore, spor)
VALUES(1760, 1200, 1, null);

--INSERTURI PENTRU CONECTATA

INSERT INTO CONECTATA(cod_poarta, cod_pista)
VALUES(470,720);

INSERT INTO CONECTATA(cod_poarta, cod_pista)
VALUES(480,720);

INSERT INTO CONECTATA(cod_poarta, cod_pista)
VALUES(490,730);

INSERT INTO CONECTATA(cod_poarta, cod_pista)
VALUES(500,740);

INSERT INTO CONECTATA(cod_poarta, cod_pista)
VALUES(530,750);

INSERT INTO CONECTATA(cod_poarta, cod_pista)
VALUES(540,760);
```

```
INSERT INTO CONECTATA(cod_poarta, cod_pista)  
VALUES(550,770);
```

```
INSERT INTO CONECTATA(cod_poarta, cod_pista)  
VALUES(570,780);
```

```
INSERT INTO CONECTATA(cod_poarta, cod_pista)  
VALUES(580,790);
```

```
INSERT INTO CONECTATA(cod_poarta, cod_pista)  
VALUES(590,800);
```

```
INSERT INTO CONECTATA(cod_poarta, cod_pista)  
VALUES(600,810);
```

```
INSERT INTO CONECTATA(cod_poarta, cod_pista)  
VALUES(610,820);
```

```
INSERT INTO CONECTATA(cod_poarta, cod_pista)  
VALUES(620,830);
```

```
INSERT INTO CONECTATA(cod_poarta, cod_pista)  
VALUES(630,840);
```

```
INSERT INTO CONECTATA(cod_poarta, cod_pista)  
VALUES(640,850);
```

```
INSERT INTO CONECTATA(cod_poarta, cod_pista)  
VALUES(650,860);
```

```
INSERT INTO CONECTATA(cod_poarta, cod_pista)  
VALUES(670,870);
```

```
INSERT INTO CONECTATA(cod_poarta, cod_pista)  
VALUES(680,880);
```

```
INSERT INTO CONECTATA(cod_poarta, cod_pista)  
VALUES(690,890);
```

```
INSERT INTO CONECTATA(cod_poarta, cod_pista)  
VALUES(710,900);
```

```
INSERT INTO CONECTATA(cod_poarta, cod_pista)  
VALUES(710,910);
```

```
INSERT INTO CONECTATA(cod_poarta, cod_pista)
```

```
VALUES(710,920);

--INSERTURI PENTRU REZERVA

INSERT INTO REZERVA(cod_rezervare, cod_loc, cod_zbor,
data_rezervare)

VALUES (1290, 1360, 1110, TO_DATE('2023-05-17', 'YYYY-MM-DD'));

INSERT INTO REZERVA(cod_rezervare, cod_loc, cod_zbor,
data_rezervare)

VALUES (1300, 1360, 1120, TO_DATE('2023-05-18', 'YYYY-MM-DD'));

INSERT INTO REZERVA(cod_rezervare, cod_loc, cod_zbor,
data_rezervare)

VALUES (1210, 1390, 1140, TO_DATE('2023-04-15', 'YYYY-MM-DD'));

INSERT INTO REZERVA(cod_rezervare, cod_loc, cod_zbor,
data_rezervare)

VALUES (1220, 1400, 1150, TO_DATE('2023-03-20', 'YYYY-MM-DD'));

INSERT INTO REZERVA(cod_rezervare, cod_loc, cod_zbor,
data_rezervare)

VALUES (1220, 1410, 1150, TO_DATE('2023-03-20', 'YYYY-MM-DD'));

INSERT INTO REZERVA(cod_rezervare, cod_loc, cod_zbor,
data_rezervare)
```

```
VALUES (1230, 1450, 1160, TO_DATE('2022-10-10', 'YYYY-MM-DD'));

INSERT INTO REZERVA(cod_rezervare, cod_loc, cod_zbor,
data_rezervare)

VALUES (1240, 1460, 1170, TO_DATE('2022-09-15', 'YYYY-MM-DD'));

INSERT INTO REZERVA(cod_rezervare, cod_loc, cod_zbor,
data_rezervare)

VALUES (1240, 1450, 1180, TO_DATE('2022-09-15', 'YYYY-MM-DD'));

INSERT INTO REZERVA(cod_rezervare, cod_loc, cod_zbor,
data_rezervare)

VALUES (1250, 1490, 1190, TO_DATE('2021-11-15', 'YYYY-MM-DD'));

INSERT INTO REZERVA(cod_rezervare, cod_loc, cod_zbor,
data_rezervare)

VALUES (1250, 1500, 1190, TO_DATE('2021-11-15', 'YYYY-MM-DD'));

INSERT INTO REZERVA(cod_rezervare, cod_loc, cod_zbor,
data_rezervare)

VALUES (1250, 1480, 1190, TO_DATE('2021-11-15', 'YYYY-MM-DD'));

INSERT INTO REZERVA(cod_rezervare, cod_loc, cod_zbor,
data_rezervare)

VALUES (1260, 1440, 1200, TO_DATE('2021-10-20', 'YYYY-MM-DD'));
```

```
INSERT INTO REZERVA(cod_rezervare, cod_loc, cod_zbor,  
data_rezervare)  
  
VALUES (1270, 1430, 1200, TO_DATE('2023-02-20', 'YYYY-MM-DD'));  
  
INSERT INTO REZERVA(cod_rezervare, cod_loc, cod_zbor,  
data_rezervare)  
  
VALUES (1280, 1420, 1120, TO_DATE('2023-02-20', 'YYYY-MM-DD'));
```

Acum voi insera screenshot-urile pentru crearea tabelelor:

Oracle SQL Developer : C:\Users\40773\project.sql

File Edit View Navigate Run Source Tools Window Help

SQL History project.sql Welcome Page

SQL Worksheet History

c_nutri(p_tara);

Worksheet: Query Builder

```
Lara VARCHAR2(25),
CONSTRAINT pk_angajat PRIMARY KEY(cod_angajat)
);

CREATE TABLE PILOT(
cod_angajat NUMBER(5),
educatie VARCHAR2(50),
experienta NUMBER(2),
CONSTRAINT pk_pilot FOREIGN KEY (cod_angajat) REFERENCES ANGAJAT(cod_angajat)
);

CREATE TABLE INSOTITOR_DE_BORD(
cod_angajat NUMBER(5),
experienta NUMBER(2),
limba_spirala VARCHAR2(25),
CONSTRAINT pk_insotitor FOREIGN KEY (cod_angajat) REFERENCES ANGAJAT(cod_angajat)
);

CREATE TABLE INGINER(
cod_angajat NUMBER(5),
educatie VARCHAR2(50),
CONSTRAINT fk_inginer FOREIGN KEY (cod_angajat) REFERENCES ANGAJAT(cod_angajat)
);

Script Output | Query Result |
Task completed in 0.166 seconds
```

Table PILOT created.

Table INSOTITOR_DE_BORD created.

Table INGINER created.

Line 156 Column 12 | Insert | Modified: Windows CR

- 0 X

Cloudy 8:10 PM 1/12/2024

Oracle SQL Developer : C:\Users\40773\project.sql

File Edit View Navigate Run Source Tools Window Help

SQL History project.sql Welcome Page

SQL Worksheet History

c_nutri(p_tara);

Worksheet: Query Builder

```
CONSTRAINT pk_insotitor FOREIGN KEY (cod_angajat) REFERENCES ANGAJAT(cod_angajat)
);

CREATE TABLE INGINER(
cod_angajat NUMBER(5),
educatie VARCHAR2(50),
CONSTRAINT fk_inginer FOREIGN KEY (cod_angajat) REFERENCES ANGAJAT(cod_angajat)
);

CREATE TABLE ZBOR(
cod_zbor NUMBER(5),
data DATE,
inistrator NUMBER(5),
cod_avion NUMBER(5),
cod_pista NUMBER(5),
cod_pista2 NUMBER(5),
cod_ruta NUMBER(5),
CONSTRAINT pk_zbor PRIMARY KEY(cod_zbor),
CONSTRAINT fk_zbor4 FOREIGN KEY(cod_avion) REFERENCES AVION(cod_avion),
CONSTRAINT fk_zbor1 FOREIGN KEY(cod_pista) REFERENCES PISTA(cod_pista),
CONSTRAINT fk_zbor2 FOREIGN KEY(cod_pista2) REFERENCES PISTA(cod_pista),
CONSTRAINT fk_zbor3 FOREIGN KEY(cod_ruta) REFERENCES RUTA(cod_ruta)
);

Script Output | Query Result |
Task completed in 0.225 seconds
```

Table INGINER created.

Table ZBOR created.

Line 169 Column 16 | Insert | Modified: Windows CR

- 0 X

Cloudy 8:10 PM 1/12/2024

Oracle SQL Developer : C:\Users\40773\project.sql

File Edit View Navigate Run Source Team Tools Window Help

SQL History project.sql Welcome Page

SQL Worksheet History

c_nutri(p_tara); 1 of 1

Worksheet: Query Builder

```
cod_pista1 NUMBER(5),
cod_pista2 NUMBER(5),
cod_ruta NUMBER(5),
CONSTRAINT pk_zbor PRIMARY KEY(cod_zbor),
CONSTRAINT fk_zbor4 FOREIGN KEY(cod_avion) REFERENCES AVION(cod_avion),
CONSTRAINT fk_zbor1 FOREIGN KEY(cod_pista1) REFERENCES PISTA(cod_pista),
CONSTRAINT fk_zbor2 FOREIGN KEY(cod_pista2) REFERENCES PISTA(cod_pista),
CONSTRAINT fk_zbor3 FOREIGN KEY(cod_ruta) REFERENCES RUTA(cod_ruta)
);

CREATE TABLE ARE(
cod_aeroport1 NUMBER(5),
cod_aeroport2 NUMBER(5),
cod_ruta NUMBER(5),
CONSTRAINT pk_are PRIMARY KEY(cod_aeroport1, cod_aeroport2, cod_ruta),
CONSTRAINT fk_are1 FOREIGN KEY(cod_aeroport1) REFERENCES AEROPORT(cod_aeroport),
CONSTRAINT fk_are2 FOREIGN KEY(cod_aeroport2) REFERENCES AEROPORT(cod_aeroport),
CONSTRAINT fk_are3 FOREIGN KEY(cod_ruta) REFERENCES RUTA(cod_ruta)
);
CREATE TABLE OFERA(
cod_clasa NUMBER(5),
cod_facilitate NUMBER(5),
CONSTRAINT pk_ofera PRIMARY KEY(cod_clasa, cod_facilitate),
CONSTRAINT fk_ofera1 FOREIGN KEY(cod_clasa) REFERENCES CLASA(cod_clasa),

```

Script Output * Query Result * Task completed in 0.197 seconds

Table ZBOR created.

Table ARE created.

Line 182 Column 14 | Insert | Modified: Windows CR

Cloudy 8:10 PM 1/12/2024

Oracle SQL Developer : C:\Users\40773\project.sql

File Edit View Navigate Run Source Team Tools Window Help

SQL History project.sql Welcome Page

SQL Worksheet History

c_nutri(p_tara); 1 of 1

Worksheet: Query Builder

```
cod_pista1 NUMBER(5),
cod_pista2 NUMBER(5),
cod_ruta NUMBER(5),
CONSTRAINT fk_zbor2 FOREIGN KEY(cod_pista2) REFERENCES PISTA(cod_pista),
CONSTRAINT fk_zbor3 FOREIGN KEY(cod_ruta) REFERENCES RUTA(cod_ruta)
);

CREATE TABLE ARE(
cod_aeroport1 NUMBER(5),
cod_aeroport2 NUMBER(5),
cod_ruta NUMBER(5),
CONSTRAINT pk_are PRIMARY KEY(cod_aeroport1, cod_aeroport2, cod_ruta),
CONSTRAINT fk_are1 FOREIGN KEY(cod_aeroport1) REFERENCES AEROPORT(cod_aeroport),
CONSTRAINT fk_are2 FOREIGN KEY(cod_aeroport2) REFERENCES AEROPORT(cod_aeroport),
CONSTRAINT fk_are3 FOREIGN KEY(cod_ruta) REFERENCES RUTA(cod_ruta)
);
CREATE TABLE OFERA(
cod_clasa NUMBER(5),
cod_facilitate NUMBER(5),
CONSTRAINT pk_ofera PRIMARY KEY(cod_clasa, cod_facilitate),
CONSTRAINT fk_ofera1 FOREIGN KEY(cod_clasa) REFERENCES CLASA(cod_clasa),
CONSTRAINT fk_ofera2 FOREIGN KEY(cod_facilitate) REFERENCES FACILITATE(cod_facilitate)
);
CREATE TABLE CUMPARA(
```

Script Output * Query Result * Task completed in 0.179 seconds

Table ARE created.

Table OFERA created.

Line 188 Column 16 | Insert | Modified: Windows CR

Cloudy 8:10 PM 1/12/2024

Oracle SQL Developer : C:\Users\40773\project.sql

File Edit View Navigate Run Source Team Tools Window Help

SQL History project.sql Welcome Page

SQL Worksheet History

c_rute(p_tara); ofr; cumpara;

Worksheet: Query Builder

```
CONSTRAINT pk_aeroport PRIMARY KEY(cod_aeroport1, cod_aeroport2, cod_ruta),
CONSTRAINT fk_aer FOREIGN KEY(cod_aeroport1) REFERENCES AEROPORT(cod_aeroport),
CONSTRAINT fk_aer2 FOREIGN KEY(cod_aeroport2) REFERENCES AEROPORT(cod_aeroport),
CONSTRAINT fk_ar3 FOREIGN KEY(cod_ruta) REFERENCES RUTA(cod_ruta)
);

CREATE TABLE OFERA(
cod_clasa NUMBER(5),
cod_facilitate NUMBER(5),
CONSTRAINT pk_ofera PRIMARY KEY(cod_clasa, cod_facilitate),
CONSTRAINT fk_oferal FOREIGN KEY(cod_clasa) REFERENCES CLASA(cod_clasa),
CONSTRAINT fk_ofera2 FOREIGN KEY(cod_facilitate) REFERENCES FACILITATE(cod_facilitate)
);

CREATE TABLE CUMPARA(
cod_rezervare NUMBER(5),
cod_pasager NUMBER(5),
numar_locuri NUMBER(2),
CONSTRAINT pk_cumpara PRIMARY KEY(cod_rezervare, cod_pasager),
CONSTRAINT fk_cumparal FOREIGN KEY(cod_rezervare) REFERENCES REZERVARE(cod_rezervare),
CONSTRAINT fk_cumpar2 FOREIGN KEY(cod_pasager) REFERENCES PASAGER(cod_pasager)
);

Script Output * Query Result *
Task completed in 0.177 seconds
```

Table OFERA created.

Table CUMPARA created.

Line 201 Column 16 | Insert | Modified: Windows CR

Cloudy 8:10 PM 1/12/2024

Oracle SQL Developer : C:\Users\40773\project.sql

File Edit View Navigate Run Source Team Tools Window Help

SQL History project.sql Welcome Page

SQL Worksheet History

c_rute(p_tara); ofr; cumpara;

Worksheet: Query Builder

```
CREATE TABLE CUMPARA(
cod_rezervare NUMBER(5),
cod_pasager NUMBER(5),
numar_locuri NUMBER(2),
CONSTRAINT pk_cumpara PRIMARY KEY(cod_rezervare, cod_pasager),
CONSTRAINT fk_cumparal FOREIGN KEY(cod_rezervare) REFERENCES REZERVARE(cod_rezervare),
CONSTRAINT fk_cumpar2 FOREIGN KEY(cod_pasager) REFERENCES PASAGER(cod_pasager)
);

CREATE TABLE REZERVA(
cod_rezervare NUMBER(5),
cod_loc NUMBER(5),
cod_zbor NUMBER(5),
data_rezervare DATE,
CONSTRAINT pk_rezerva PRIMARY KEY(cod_rezervare, cod_loc, cod_zbor),
CONSTRAINT fk_rezerval FOREIGN KEY(cod_rezervare) REFERENCES REZERVARE(cod_rezervare),
CONSTRAINT fk_rezerval FOREIGN KEY(cod_loc) REFERENCES LOC(cod_loc),
CONSTRAINT fk_rezerva3 FOREIGN KEY(cod_zbor) REFERENCES ZBOR(cod_zbor)
);

CREATE TABLE LUCREAZA;
```

Script Output * Query Result *
Task completed in 0.168 seconds

Table CUMPARA created.

Table REZERVA created.

Line 213 Column 16 | Insert | Modified: Windows CR

Cloudy 8:11 PM 1/12/2024

Oracle SQL Developer : C:\Users\40773\project.sql

```

CREATE TABLE REZERVA(
    cod_rezervare NUMBER(5),
    cod_loc NUMBER(5),
    cod_zbor NUMBER(5),
    data_rezervare DATE,
    CONSTRAINT pk_rezervare PRIMARY KEY(cod_rezervare, cod_loc, cod_zbor),
    CONSTRAINT fk_rezervare FOREIGN KEY(cod_rezervare) REFERENCES REZERVARE(cod_rezervare),
    CONSTRAINT fk_rezervare2 FOREIGN KEY(cod_loc) REFERENCES LOC(cod_loc),
    CONSTRAINT fk_rezervare3 FOREIGN KEY(cod_zbor) REFERENCES ZBOR(cod_zbor)
);

CREATE TABLE LUCREAZA(
    cod_angajat NUMBER(5),
    cod_zbor NUMBER(5),
    ore NUMBER(2),
    spor NUMBER(4),
    CONSTRAINT pk_lucreaza PRIMARY KEY(cod_angajat, cod_zbor),
    CONSTRAINT fk_lucreaza FOREIGN KEY(cod_angajat) REFERENCES ANGAJAT(cod_angajat),
    CONSTRAINT fk_lucreaza2 FOREIGN KEY(cod_zbor) REFERENCES ZBOR(cod_zbor)
);

```

Script Output : Task completed in 0.168 seconds

Table REZERVA created.

Table LUCREAZA created.

Line 222 Column 12 | Insert | Modified: Windows CR

Oracle SQL Developer : C:\Users\40773\project.sql

```

CREATE TABLE LUCREAZA(
    cod_angajat NUMBER(5),
    cod_zbor NUMBER(5),
    ore NUMBER(2),
    spor NUMBER(4),
    CONSTRAINT pk_lucreaza PRIMARY KEY(cod_angajat, cod_zbor),
    CONSTRAINT fk_lucreaza FOREIGN KEY(cod_angajat) REFERENCES ANGAJAT(cod_angajat),
    CONSTRAINT fk_lucreaza2 FOREIGN KEY(cod_zbor) REFERENCES ZBOR(cod_zbor)
);

CREATE TABLE CONECTATA(
    cod_poarta NUMBER(5),
    cod_pista NUMBER(5),
    CONSTRAINT pk_CONECTATA PRIMARY KEY(cod_poarta, cod_pista),
    CONSTRAINT fk_CONECTATA1 FOREIGN KEY(cod_poarta) REFERENCES POARTA(cod_poarta),
    CONSTRAINT fk_CONECTATA2 FOREIGN KEY(cod_pista) REFERENCES PISTA(cod_pista)
);

```

Script Output : Task completed in 0.179 seconds

Table LUCREAZA created.

Table CONECTATA created.

Line 231 Column 13 | Insert | Modified: Windows CR

Acum voi insera screenshot-urile pentru inserarile in tabele. Pentru a insera mai putine astfel de capturi de ecran, le voi face dupa adaugarea datelor in tabele, cu selecturile din fiecare:

Oracle SQL Developer : C:\Users\40773\project.sql

File Edit View Navigate Run Source Tools Window Help

SQL History project.sql Welcome Page

SQL Worksheet History

Script Output All Rows Fetched: 13 in 0.003 seconds

```
-- ultimul insert pentru angajat
INSERT INTO ANGAJAT(cod_angajat, nume, prenume, salariu, data_nasterii, data_angajarii, tara)
VALUES(10000, 'Smith', 'Anna', 7000, TO_DATE('17-10-1994', 'dd-mm-yyyy'), TO_DATE('20-07-2019', 'dd-mm-yyyy'), 'Franta');

select * from locatie;
select * from aeroport;
select * from pistă;
select * from roata;
```

All Rows Fetched: 13 in 0.003 seconds

COD LOCATE	ADRESA	TARA	ORAS	LIMBA OFICIALA	MONEDA OFICIALA
1	10 JFK Access Rd. NY 11430	Statele Unite ale Americii	New York	engleză	USD
2	20 10th Street, Hong Kong	Pescaria	Hong Kong	engleză	HKD
3	301 Sky Plaza Rd	Hong Kone	Hong Kone	[null]	HKD
4	45 Rue de la Paix, Paris, France	Franta	Paris	engleză	EUR
5	50 69547 Frankfurt, Hessen, Germany	Germania	Frankfurt	germană	EUR
6	60 272 Gondhwa-ro, Jungr-su, Incheon	Coreea de Sud	Seoul	coreană	KRW
7	70 Delhi, India	India	Delhi	hinduă	INR
8	80 Mascot NSW 2020	Australia	Sydney	engleză	AUD
9	90 1000 Dr. Mihai Br. Mississauga, L5P 1B2, Canada	Canada	Toronto	engleză	CAD
10	100 Calea Bucureștilor	România	Bucuresti	română	RON
11	110 Strada Aeroplanelui 1	România	Iasi	română	RON
12	120 Strada Aeroplanelui 1	România	Craiova	română	RON
13	130 Via dell'Aeroporto di Fiumicino	Italia	Roma	italiană	EUR

Line 2134 Column 6 | Insert | Modified: Windows CR

Cloudy 8:39 PM 1/12/2024

Oracle SQL Developer : C:\Users\40773\project.sql

File Edit View Navigate Run Source Tools Window Help

SQL History project.sql Welcome Page

SQL Worksheet History

Script Output All Rows Fetched: 13 in 0.007 seconds

```
-- ultimul insert pentru angajat
INSERT INTO ANGAJAT(cod_angajat, nume, prenume, salariu, data_nasterii, data_angajarii, tara)
VALUES(10000, 'Smith', 'Anna', 7000, TO_DATE('17-10-1994', 'dd-mm-yyyy'), TO_DATE('20-07-2019', 'dd-mm-yyyy'), 'Franta');

select * from locatie;
select * from aeroport;
select * from pistă;
select * from roata;
```

All Rows Fetched: 13 in 0.007 seconds

COD AEROPORT	COD LOCATE	NUME	ZBOR INTERNATIONAL
1	140	10 John F. Kennedy International Airport	DA
2	150	110 Aeroportul Internațional din Hong Kong	DA
3	160	30 Aeroportul Internațional de la Hong Kone	DA
4	170	40 Aeroportul Internațional Charles de Gaulle	DA
5	180	50 Aeroportul Internațional din Frankfurt	DA
6	190	60 Aeroportul Internațional Incheon	DA
7	200	70 Aeroportul Internațional din Sanchi	DA
8	210	80 Aeroportul Internațional din Sydney	DA
9	220	90 Aeroportul Internațional Pearson, Toronto	DA
10	230	100 Aeroportul Internațional din București	DA
11	240	110 Aeroportul Internațional Iași	DA
12	250	120 Aeroportul Internațional Craiova	DA
13	260	130 Aeroportul Internațional Leonardo da Vinci	DA

Line 2136 Column 21 | Insert | Modified: Windows CR

Cloudy 8:39 PM 1/12/2024

Oracle SQL Developer : C:\Users\40773\project.sql

File Edit View Navigate Run Source Tools Window Help

SQL History project.sql Welcome Page

SQL Worksheet History

Script Output All Rows Fetched: 21 in 0.007 seconds

```

-- ultimul insert pentru angajat
INSERT INTO ANGAJAT(cod_angajat, nume, prenume, salariu, data_nasterii, data_angajarii, tara)
VALUES(10000, 'Smith', 'Anna', 7000, TO_DATE('17-10-1994', 'dd-mm-yyyy'), TO_DATE('20-07-2019', 'dd-mm-yyyy'), 'Franta');

select * from locatie;
select * from aeroport;
select * from pistă;
select * from poartă;
```

Line 213B Column 18 | Insert | Modified: Windows CR

Cloudy -4°C

Search ENG US 8:39 PM 1/12/2024

Oracle SQL Developer : C:\Users\40773\project.sql

File Edit View Navigate Run Source Tools Window Help

SQL History project.sql Welcome Page

SQL Worksheet History

Script Output All Rows Fetched: 25 in 0.01 seconds

```

-- ultimul insert pentru angajat
INSERT INTO ANGAJAT(cod_angajat, nume, prenume, salariu, data_nasterii, data_angajarii, tara)
VALUES(10000, 'Smith', 'Anna', 7000, TO_DATE('17-10-1994', 'dd-mm-yyyy'), TO_DATE('20-07-2019', 'dd-mm-yyyy'), 'Franta');

select * from locatie;
select * from aeroport;
select * from pistă;
select * from poartă;
select * from terminal;
```

Line 214D Column 17 | Insert | Modified: Windows CR

Cloudy -4°C

Search ENG US 8:39 PM 1/12/2024

Oracle SQL Developer : C:\Users\40773\project.sql

File Edit View Navigate Run Source Team Tools Window Help

SQL History project.sql Welcome Page

SQL Worksheet History

SQL Script Editor

Query Result 1 of 15 All Rows Fetched: 20 in 0.012 seconds

```
select * from locatie;
select * from aeroport;
select * from pista;
select * from poarta;
select * from terminal;
select * from conectata;
```

All Rows Fetched: 20 in 0.012 seconds

COD_TERMINAL	COD_AEROPORT	CAPACITATE	POTZIE
1	270	140	2500 SUD
2	280	140	3000 NORD-EST
3	290	140	3000 NORD-CENTRAL
4	300	160	3500 CENTRAL
5	310	160	3500 WEST
6	320	160	1500 WEST
7	330	160	1500 WEST
8	340	170	2100 SUD
9	350	180	3500 SUD-VEST
10	360	180	3500 SUD-VEST
11	370	190	2400 NORD
12	380	190	2400 NORD-EST
13	390	210	3500 CENTRAL
14	400	220	4800 NORD-VEST
15	410	230	4800 NORD-VEST
16	420	230	1500 SUD-EST
17	430	240	2100 SUD
18	440	250	3500 NORD
19	450	260	3000 SUD
20	460	260	2500 CENTRAL

Line 2142 Column 17 | Insert | Modified: Windows CR

Cloudy 8:39 PM 1/12/2024

Oracle SQL Developer : C:\Users\40773\project.sql

File Edit View Navigate Run Source Team Tools Window Help

SQL History project.sql Welcome Page

SQL Worksheet History

SQL Script Editor

Query Result 1 of 15 All Rows Fetched: 22 in 0.009 seconds

```
select * from locatie;
select * from aeroport;
select * from pista;
select * from poarta;
select * from terminal;
select * from conectata;
select * from aree;
```

All Rows Fetched: 22 in 0.009 seconds

COD_POARTA	COD_PISTA
1	470
2	480
3	490
4	500
5	510
6	540
7	550
8	570
9	580
10	590
11	600
12	610
13	620
14	630
15	640
16	650
17	670
18	680
19	690
20	710
21	710
22	920

Line 2144 Column 17 | Insert | Modified: Windows CR

Cloudy 8:39 PM 1/12/2024

Oracle SQL Developer : C:\Users\40773\project.sql

File Edit View Navigate Run Source Tools Window Help

SQL History project.sql Welcome Page

SQL Worksheet History

SQL Worksheet: 1 of 15

Script Output: All Rows Fetched: 10 in 0.007 seconds

```
select * from aeropost;
```

```
select * from pista;
```

```
select * from poarta;
```

```
select * from terminal;
```

```
select * from conectata;
```

```
select * from are;
```

```
select * from ruta;
```

Line 2148 Column 17 | Insert | Modified: Windows: CR

Cloudy 8:39 PM 1/12/2024

Oracle SQL Developer : C:\Users\40773\project.sql

File Edit View Navigate Run Source Tools Window Help

SQL History project.sql Welcome Page

SQL Worksheet History

SQL Worksheet: 1 of 15

Script Output: All Rows Fetched: 10 in 0.009 seconds

```
select * from pista;
```

```
select * from poarta;
```

```
select * from terminal;
```

```
select * from conectata;
```

```
select * from are;
```

```
select * from ruta;
```

```
select * from zbor;
```

Line 2148 Column 17 | Insert | Modified: Windows: CR

Cloudy 8:39 PM 1/12/2024

Oracle SQL Developer : C:\Users\40773\project.sql

File Edit View Navigate Run Source Team Tools Window Help

SQL History project.sql Welcome Page

SQL Worksheet History

Script Output All Rows Fetched: 10 in 0.002 seconds

Worksheet: Query Builder

```
select * from poarta;
select * from terminal;
select * from conectata;
select * from are;
select * from ruta;
select * from zbor;
select * from lucreaza;
```

All Rows Fetched: 10 in 0.002 seconds

	COD_ZBOR	DATA	INTARZIRE	COD_AVION	COD_PISTA1	COD_PISTA2	COD_RUTA
1	111017	MAY-23	0	1080	820	780	930
2	111018	MAY-23	0	1080	720	840	940
3	111019	MAY-23	0	1030	790	870	950
4	111020	MAY-23	45	1040	750	820	970
5	111021	MAY-23	10	1040	750	820	970
6	111022	MAY-23	0	1050	750	910	980
7	111023	MAY-23	0	1050	930	750	990
8	111024	MAY-23	0	1050	910	770	1000
9	111025	MAY-23	0	1070	750	810	1010
10	111026	MAY-23	0	1060	890	800	1020

Line 2150 Column 13 | Insert | Modified: Windows CR

Cloudy 8:40 PM 1/12/2024

Oracle SQL Developer : C:\Users\40773\project.sql

File Edit View Navigate Run Source Team Tools Window Help

SQL History project.sql Welcome Page

SQL Worksheet History

Script Output All Rows Fetched: 10 in 0.002 seconds

Worksheet: Query Builder

```
select * from poarta;
select * from terminal;
select * from conectata;
select * from are;
select * from ruta;
select * from zbor;
select * from lucreaza;
```

All Rows Fetched: 10 in 0.002 seconds

	COD_ZBOR	DATA	INTARZIRE	COD_AVION	COD_PISTA1	COD_PISTA2	COD_RUTA
1	111017	MAY-23	0	1080	820	780	930
2	111018	MAY-23	0	1080	720	840	940
3	111019	MAY-23	0	1030	790	870	950
4	111020	MAY-23	45	1040	720	850	960
5	111021	MAY-23	10	1040	750	820	970
6	111022	MAY-23	0	1050	750	910	980
7	111023	MAY-23	0	1050	930	750	990
8	111024	MAY-23	0	1050	910	770	1000
9	111025	MAY-23	0	1070	750	810	1010
10	111026	MAY-23	0	1060	890	810	1020

Line 2150 Column 13 | Insert | Modified: Windows CR

Cloudy 8:40 PM 1/12/2024

Oracle SQL Developer : C:\Users\40773\project.sql

File Edit View Navigate Run Source Team Tools Window Help

SQL History project.sql Welcome Page

SQL Worksheet History

Script Output All Rows Fetched: 30 in 0.007 seconds

SQL All Rows Fetched: 30 in 0.007 seconds

Worksheet: Query Builder

```
select * from terminal;
select * from conectata;
select * from are;
select * from ruta;
select * from zbor;
select * from lucrareaza;
select * from angajat;
```

All Rows Fetched: 30 in 0.007 seconds

	COD_ANGAJAT	COD_ZBOR	ORE	SPOR
1	1670	1110	5	230
2	1700	1110	7	200
3	1700	1130	4	200
4	1670	1150	125	430
5	1710	1160	9	100
6	1710	1160	10	250
7	1650	1180	8	300
8	1650	1180	7	100
9	1650	1200	9	100
10	1740	1110	9	170
11	1740	1110	9	170
12	1750	1130	4	140
13	1750	1130	4	120
14	1750	1150	125	400
15	1750	1150	125	400
16	1720	1160	9	70
17	1720	1160	10	220
18	1740	1180	8	100
19	1740	1180	7	250
20	1730	1200	9	200
21	1680	1110	12	(null)
22	1680	1130	1	14
23	1680	1130	1	(null)
24	1760	1150	12	(null)
25	1760	1150	12	(null)
26	1680	1180	12	(null)
27	1760	1180	12	35
28	1760	1180	12	(null)
29	1760	1200	12	40
30	1760	1200	1	(null)

Line 2152 Column 14 | Insert | Modified: Windows CR

Cloudy 8:40 PM 1/12/2024

Oracle SQL Developer : C:\Users\40773\project.sql

File Edit View Navigate Run Source Team Tools Window Help

SQL History project.sql Welcome Page

SQL Worksheet History

Script Output All Rows Fetched: 11 in 0.007 seconds

SQL All Rows Fetched: 11 in 0.007 seconds

Worksheet: Query Builder

```
select * from conectata;
select * from are;
select * from ruta;
select * from zbor;
select * from lucrareaza;
select * from angajat;
select * from inginer;
```

All Rows Fetched: 11 in 0.007 seconds

	COD_ANGAJAT	NUME	SALARIU	PRENUME	DATA_NASTERII	DATA_ANGAJARII	TARA
1	1670	Poescu	8000	Ion	01-JAN-50	01-JAN-20	Romania
2	1680	Mihai	10000	Andrei	15-MAY-65	15-MAY-15	Romania
3	1690	Poica	15000	Marina	10-MAY-82	15-MAR-11	Italia
4	1700	Muller	10000	Hans	20-JUN-88	10-SEP-19	Germania
5	1710	Popescu	12000	Paul	05-JUL-75	15-JUL-19	Romania
6	1720	Lobea	12000	Maria	08-SEP-51	18-NOV-18	Espania
7	1730	Popescu	10000	Andrei	15-JUL-81	15-JUL-19	Romania
8	1740	Klim	15000	Mihai	19-JUL-53	12-AUG-20	Corea de Sud
9	1750	Mohamed	16000	Ahmed	12-NOV-54	30-JUL-15	Pakistan
10	1760	Popescu	10000	Adriana	15-JUL-81	15-JUL-19	Romania
11	10000	Smith	7000	Anna	17-OCT-54	30-JUL-19	Franta

Line 2154 Column 14 | Insert | Modified: Windows CR

Cloudy 8:40 PM 1/12/2024

Oracle SQL Developer : C:\Users\40773\project.sql

File Edit View Navigate Run Source Tools Window Help

SQL History project.sql Welcome Page

SQL Worksheet History

Script Output: project.sql 1 of 15 rows fetched in 0.006 seconds

All Rows Fetched: 2 in 0.006 seconds

Worksheet: Query Builder

```
select * from are;
select * from ruta;
select * from zbor;
select * from lucreaza;
select * from angajat;
select * from inginer;
select * from insotitor_de_bord;
```

Script Output: Query Result 1 · Query Result 2 · Query Result 3 · Query Result 4 · Query Result 5 · Query Result 6

COD ANGAJAT | EDUCATIE

1	1680	Universitatea Politehnica Bucuresti
2	1760	Universitatea Politehnica Bucuresti

Line 215B Column 14 | Insert | Modified: Windows CR

Cloudy -4°C

Search ENG US 8:40 PM 1/12/2024

Oracle SQL Developer : C:\Users\40773\project.sql

File Edit View Navigate Run Source Tools Window Help

SQL History project.sql Welcome Page

SQL Worksheet History

Script Output: project.sql 1 of 15 rows fetched in 0.006 seconds

All Rows Fetched: 4 in 0.006 seconds

Worksheet: Query Builder

```
select * from ruta;
select * from zbor;
select * from lucreaza;
select * from angajat;
select * from inginer;
select * from insotitor_de_bord;
select * from pilot;
```

Script Output: Query Result 1 · Query Result 2 · Query Result 3 · Query Result 4 · Query Result 5 · Query Result 6

COD ANGAJAT | EXPERIENTA | LIMBA STRANA

1	1740	3	Engleza
2	1720	5	Germana
3	1730	4	Spaniola

Line 215B Column 14 | Insert | Modified: Windows CR

Cloudy -4°C

Search ENG US 8:40 PM 1/12/2024

Oracle SQL Developer : C:\Users\40773\project.sql

File Edit View Navigate Run Source Team Tools Window Help

SQL History project.sql Welcome Page

SQL Worksheet History

SQL Script Output

SQL All Rows Fetched: 4 in 0.006 seconds

Worksheet: Query Builder

```
select * from zbor;
```

```
select * from lucreaza;
```

```
select * from angajat;
```

```
select * from inginer;
```

```
select * from insotitor_de_bord;
```

```
select * from pilot;
```

```
select * from avion;
```

Script Output: [Query Result 1] [Query Result 2] [Query Result 3] [Query Result 4] [Query Result 5] [Query Result 6]

1 COD_ANGAJAT | EDUCATIE | EXPERIENTA

1 1670 Universitatea Politehnica Bucuresti 5

2 1710 California Institute of Technology 8

3 1690 Massachusetts Institute of Technology 12

4

Line 2160 Column 14 | Insert | Modified: Windows: CR

Cloudy 8:40 PM 1/12/2024

Oracle SQL Developer : C:\Users\40773\project.sql

File Edit View Navigate Run Source Team Tools Window Help

SQL History project.sql Welcome Page

SQL Worksheet History

SQL Script Output

SQL All Rows Fetched: 8 in 0.007 seconds

Worksheet: Query Builder

```
select * from lucreaza;
```

```
select * from angajat;
```

```
select * from inginer;
```

```
select * from insotitor_de_bord;
```

```
select * from pilot;
```

```
select * from avion;
```

```
select * from loc;
```

Script Output: [Query Result 1] [Query Result 2] [Query Result 3] [Query Result 4] [Query Result 5] [Query Result 6]

1 COD_AVION | NUMAR_LOCURI | INALTIME | LUNGIME | ZBOR_INTERCONTINENTAL | MODEL

1 1030 150 10 15DA Boeing 747

2 1040 140 12 16DA Airbus A320

3 1050 200 12 16DA Boeing 787

4 1060 80 6 10MD Embraer E170

5 1070 250 12 16DA Airbus A330

6 1080 120 10 14NU Boeing 737

7 1090 180 11 16DA Airbus A350

8 1100 90 7 11NU Embraer E195

Line 2163 Column 14 | Insert | Modified: Windows: CR

Cloudy 8:40 PM 1/12/2024

Oracle SQL Developer : C:\Users\40773\project.sql

File Edit View Navigate Run Source Team Tools Window Help

SQL History project.sql Welcome Page

SQL Worksheet History

Script Output All Rows Fetched: 8 in 0.007 seconds

All Rows Fetched: 8 in 0.007 seconds

Worksheet: Query Builder

```
select * from lucreaza;
select * from angajat;
select * from inginer;
select * from insotitor_de_bord;
select * from pilot;
select * from avion;
select * from loc;
```

Script Output | Query Result 1 | Query Result 2 | Query Result 3 | Query Result 4 | Query Result 5 | Query Result 6

	COD_AVION	NUMAR_LOCURI	FINALTIME	LUNGIME	ZBOR_INTERCONTINENTAL	MODEL
1	1030	150	10	15DA	Boeing 747	
2	1040	130	12	13DA	Airbus A320	
3	1050	200	12	18DA	Boeing 787	
4	1060	180	9	14DA	Boeing 7380	
5	1070	250	18	19DA	Airbus A380	
6	1080	120	9	14NU	Boeing 737	
7	1090	180	11	18NU	Airbus A350	
8	1100	90	7	11NU	Boeing 6195	

Line 2162 Column 14 | Insert | Modified: Windows CR

Cloudy 8:41 PM 1/12/2024

Oracle SQL Developer : C:\Users\40773\project.sql

File Edit View Navigate Run Source Team Tools Window Help

SQL History project.sql Welcome Page

SQL Worksheet History

Script Output All Rows Fetched: 15 in 0.007 seconds

All Rows Fetched: 15 in 0.007 seconds

Worksheet: Query Builder

```
select * from angajat;
select * from inginer;
select * from insotitor_de_bord;
select * from pilot;
select * from avion;
select * from loc;
select * from clasa;
```

Script Output | Query Result 1 | Query Result 2 | Query Result 3 | Query Result 4 | Query Result 5 | Query Result 6

All Rows Fetched: 15 in 0.007 seconds

	COD_LOC	COD_AVION	POZITIE	NUMAR_LOC	COD_CLASA
1	1360	1050	Centru	14	1310
2	1410	1050	Maritime	14	1310
3	1380	1040	Maritime	14	1330
4	1390	1050	Central	20	1330
5	1400	1050	Maritime	1	1340
6	1410	1050	Central	67	1350
7	1420	1050	Central	114	1350
8	1430	1050	Central	2	1350
9	1440	1050	Maritime	22	1350
10	1450	1050	Central	104	1350
11	1460	1050	Central	35	1340
12	1470	1050	Central	1	1340
13	1480	1050	Central	98	1340
14	1490	1070	Central	2	1330
15	1500	1050	Maritime	26	1330

Line 2164 Column 14 | Insert | Modified: Windows CR

Cloudy 8:41 PM 1/12/2024

Oracle SQL Developer : C:\Users\40773\project.sql

File Edit View Navigate Run Source Team Tools Window Help

SQL History project.sql Welcome Page

SQL Worksheet History

Script Output Query Result 1 Query Result 2 Query Result 3 Query Result 4 Query Result 5 Query Result 6

All Rows Fetched: 5 in 0.008 seconds

SQL All Rows Fetched: 5 in 0.008 seconds

Line 2168 Column 14 | Insert | Modified: Windows: CR

Cloudy 8:41 PM 1/12/2024

```
select * from inginer;
select * from insctitor_de_bord;
select * from pilot;
select * from avion;
select * from loc;
select * from clasa;
select * from ofera;
```

	COD_CLASA	PRET
1	1310	500
2	1320	600
3	1330	1200
4	1340	1100
5	1350	1000

Oracle SQL Developer : C:\Users\40773\project.sql

File Edit View Navigate Run Source Team Tools Window Help

SQL History project.sql Welcome Page

SQL Worksheet History

Script Output Query Result 1 Query Result 2 Query Result 3 Query Result 4 Query Result 5 Query Result 6

All Rows Fetched: 10 in 0.007 seconds

SQL All Rows Fetched: 10 in 0.007 seconds

Line 2168 Column 14 | Insert | Modified: Windows: CR

Cloudy 8:41 PM 1/12/2024

```
select * from insctitor_de_bord;
select * from pilot;
select * from avion;
select * from loc;
select * from clasa;
select * from ofera;
select * from facilitate;
```

	COD_CLASA	COD_FACILITATE
1	1310	1520
2	1320	1530
3	1330	1530
4	1330	1540
5	1340	1550
6	1350	1560
7	1350	1570
8	1340	1580
9	1350	1590
10	1360	1600

Oracle SQL Developer : C:\Users\40773\project.sql

File Edit View Navigate Run Source Tools Window Help

SQL History project.sql Welcome Page

SQL Worksheet History

SQL Worksheet 1 of 15

Script Output All Rows Fetched: 6 in 0.008 seconds

```
select * from pilot;
select * from avion;
select * from loc;
select * from clasa;
select * from ofera;
select * from facilitate;
select * from pasager;
```

Line 2170 Column 14 | Insert | Modified: Windows CR

CCD	FACILITATE	NUME	PRIET_SUPLIMENTAR	STANDARD
1	1510 WiFi		10 DA	
2	1520 Cabina calda		10 DA	
3	1530 Divertiment la bord		5 DA	
4	1540 Servicii de imbarcare		10 DA	
5	1550 Scaun de calda		12 NU	
6	1560 Acces la salonul VIP		20 DA	

Cloudy -4°C 8:41 PM 1/12/2024

Oracle SQL Developer : C:\Users\40773\project.sql

File Edit View Navigate Run Source Tools Window Help

SQL History project.sql Welcome Page

SQL Worksheet History

SQL Worksheet 1 of 15

Script Output All Rows Fetched: 10 in 0.007 seconds

```
select * from avion;
select * from loc;
select * from clasa;
select * from ofera;
select * from facilitate;
select * from pasager;
select * from cumpara;
```

Line 2170 Column 14 | Insert | Modified: Windows CR

CCD	PASAGER	NUME	PRENUME	NATIONALITATE	NUMAR PASAPORT
1	1570 Popescu	Ion	Roman?	Roman?	1345678
2	1580 Popescu	Maria	Rebeca?	Roman?	87654321
3	1590 Smith	John	Englez?	Englez?	98765432
4	1600 Müller	Hans	German?	German?	54321678
5	1610 Ivanov	Ivan	Vladimir?	Rus?	87654321
6	1620 Löwen	(null)	Spaniol?	Spaniol?	43210765
7	1630 Kovalcs	János	Maghiar?	Maghiar?	6543210987
8	1640 Kovács	János	Maghiar?	Maghiar?	87654321
9	1650 Sato	Takeshi	Japoniez?	Japoniez?	3543765
10	1660 Schreiber	Marta	Roman?	Roman?	87654321

Cloudy -4°C 8:41 PM 1/12/2024

Oracle SQL Developer : C:\Users\40773\project.sql

File Edit View Navigate Run Source Tools Window Help

SQL History project.sql Welcome Page

SQL Worksheet History

Worksheet: Query Builder

```
select * from loc;
select * from clasa;
select * from ofera;
select * from facilitate;
select * from pasager;
select * from cumpara;
select * from rezervare;
```

Script Output | Query Result 1 | Query Result 2 | Query Result 3 | Query Result 4 | Query Result 5 | Query Result 6 |

All Rows Fetched: 10 in 0.01 seconds

COD, REZERVARE, COD_PASAGER, NUMAR_LOCURI

	1210	1570	1
1	1570	1580	1
2	1530	1570	1
3	1520	1570	1
4	1540	1600	1
5	1520	1610	1
6	1560	1620	1
7	1570	1630	1
8	1580	1640	1
9	1590	1640	1
10	1590	1640	1

Line 2174 Column 14 | Insert | Modified: Windows | 8:41 PM
1/12/2024 ENG US WiFi ☰

The screenshot shows the Oracle SQL Developer interface. The top menu bar includes File, Edit, View, Navigate, Run, Source, Team, Tools, Window, Help. The title bar indicates the file is C:\Users\40773\project.sql. The left sidebar has tabs for SQL History, project.sql, Welcome Page, and SQL Worksheet: History. Below the sidebar is a toolbar with various icons. The main area is divided into two panes: a Query Builder pane on the left containing a series of SELECT statements, and a Results pane on the right displaying the output of the last query. The results show a table with columns COD_RESERVAIRE, PRET_TOTAL, and TOTAL. The bottom status bar shows Line 2176 Column 14, Insert, Modified: Windows: CR, and system icons for battery, network, and time (8:41 PM, 1/12/2024). A weather icon in the bottom left corner shows 4°C Cloudy.

```
File Edit View Navigate Run Source Team Tools Window Help
SQL History project.sql Welcome Page
SQL Worksheet: History
@ mariap19
Query Builder
select * from clasa;
select * from ofera;
select * from facilitate;
select * from pasager;
select * from cumpara;
select * from rezerva;
select * from rezerva;
Script Output: [Query Result 1] [Query Result 2] [Query Result 3] [Query Result 4] [Query Result 5] [Query Result 6]
All Rows Fetched: 10 in 0.006 seconds
COD_RESERVAIRE | PRET_TOTAL
1          1210          515
2          1220         1650
3          1230          125
4          1240          2420
5          1250          3100
6          1260          1115
7          1270          1010
8          1280          1010
9          1290         1100
10         1300          1200
```

The screenshot shows the Oracle SQL Developer interface. In the top-left pane, there is a script editor with several SELECT statements:

```

select * from ofera;
select * from facilitate;
select * from pasager;
select * from cumpara;
select * from rezervare;
select * from rezerva;

```

In the bottom-right pane, there is a results grid titled "DATA REZERVARE" with columns "COD REZERVARE", "COD LOC", and "ZBOR". The data is as follows:

COD REZERVARE	COD LOC	ZBOR
1	1250	1360 111017-MAV-23
2	1250	1370 111017-MAV-23
3	1210	1390 114015-APR-23
4	1230	1410 116012-MAR-23
5	1250	1410 116020-MAR-23
6	1230	1450 116012-OCT-22
7	1230	1450 116012-OCT-22
8	1240	1450 118015-SEP-22
9	1250	1460 118015-SEP-22
10	1250	1500 118015-NOV-21
11	1250	1480 118015-NOV-21
12	1250	1490 120020-FEB-23
13	1270	1430 120020-FEB-23
14	1260	1420 112020-FEB-23

6. Formulati în limbaj natural o problema pe care să o rezolvati folosind un subprogram stocat independent care să utilizeze toate cele 3 tipuri de colectii studiate. Apelati subprogramul.

Aceasta procedura afiseaza informatii despre rezervarile facute pentru zboruri pentru fiecare data a fiecaruizbor. Pentru fiecare data, se vor afisa pasagerii care au rezervari si locurile rezervate.

```

CREATE OR REPLACE PROCEDURE PasageriSiLocuriOcupateZbor IS
    TYPE ListaPasageri IS TABLE OF VARCHAR2(50);
    TYPE LocuriRezervate IS VARRAY(20) OF NUMBER;

    TYPE InregistrareZbor IS RECORD (
        pasageri ListaPasageri,
        locuri_rezervate LocuriRezervate
    );

```

```

TYPE OrarZboruri IS TABLE OF InregistrareZbor INDEX BY
VARCHAR2(20);

rezervari_zboruri OrarZboruri;

BEGIN
    FOR zbor_rec IN (SELECT DISTINCT TO_CHAR(data, 'YYYY-MM-DD')
AS data_zbor
                     FROM ZBOR)
LOOP
    BEGIN
        SELECT DISTINCT P.NUME || ' ' || P.PRENUME
        BULK COLLECT INTO
rezervari_zboruri(zbor_rec.data_zbor).pasageri
        FROM zbor z
        JOIN rezerva r ON z.cod_zbor = r.cod_zbor
        JOIN rezervare rez ON rez.cod_rezervare =
r.cod_rezervare
        JOIN cumpara c ON c.cod_rezervare = rez.cod_rezervare
        JOIN pasager p ON p.cod_pasager = c.cod_pasager
        WHERE TO_CHAR(Z.DATA, 'YYYY-MM-DD') =
zbor_rec.data_zbor;

        SELECT DISTINCT l.numar_loc
        BULK COLLECT INTO
rezervari_zboruri(zbor_rec.data_zbor).locuri_rezervate
        FROM zbor z
        JOIN rezerva r ON z.cod_zbor = r.cod_zbor
        JOIN loc l ON l.cod_loc = r.cod_loc
        WHERE TO_CHAR(Z.DATA, 'YYYY-MM-DD') =
zbor_rec.data_zbor;

        DBMS_OUTPUT.PUT_LINE('Data Zbor: ' ||
zbor_rec.data_zbor);

        IF rezervari_zboruri.EXISTS(zbor_rec.data_zbor) THEN
            DBMS_OUTPUT.PUT('Pasageri: ');
    END;
END;

```

```

        FOR i IN
1..rezervari_zboruri(zbor_rec.data_zbor).pasageri.COUNT LOOP

DBMS_OUTPUT.PUT(rezervari_zboruri(zbor_rec.data_zbor).pasageri(i))
);

        IF i <
rezervari_zboruri(zbor_rec.data_zbor).pasageri.COUNT THEN
            DBMS_OUTPUT.PUT(', ');
        END IF;
    END LOOP;

    DBMS_OUTPUT.NEW_LINE;

    DBMS_OUTPUT.PUT('Locuri Rezervate: ');

        FOR i IN
1..rezervari_zboruri(zbor_rec.data_zbor).locuri_rezervate.COUNT
LOOP

DBMS_OUTPUT.PUT(rezervari_zboruri(zbor_rec.data_zbor).locuri_reze
rvate(i));

        IF i <
rezervari_zboruri(zbor_rec.data_zbor).locuri_rezervate.COUNT THEN
            DBMS_OUTPUT.PUT(', ');
        END IF;
    END LOOP;

    DBMS_OUTPUT.NEW_LINE;
ELSE
    DBMS_OUTPUT.PUT_LINE('Niciun pasager pentru acest
zbor.');
END IF;
EXCEPTION
    WHEN NO_DATA_FOUND THEN
        DBMS_OUTPUT.PUT_LINE('No data found for the date:
' || zbor_rec.data_zbor);
    WHEN OTHERS THEN

```

```

        DBMS_OUTPUT.PUT_LINE('An error occurred for the
date: ' || zbor_rec.data_zbor);
      END;

    END LOOP;

EXCEPTION
  WHEN OTHERS THEN
    DBMS_OUTPUT.PUT_LINE('An unexpected error occurred.');
END PasageriSiLocuriOcupateZbor;
/

```

set serveroutput on;

```

BEGIN
  PasageriSiLocuriOcupateZbor;
END;
/

```

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

- Script Output:** Shows the execution of the PL/SQL block. The output includes messages like "Niciun pasager pentru acest zbor.", "No data found for the date: 2023-05-18", and "An error occurred for the date: 2023-05-19".
- Procedure PASAGERISILOCURIOCUPATEZBOR compiled**: Shows the compilation status of the procedure.
- Data Zbor:** A table listing passengers and their reservation counts. The data is as follows:

Pasageri	Locuri Rezervate
Kovacs János	14
Pasageri: Kovacs János	14
Data Zbor: 2023-05-18	
Pasageri: Popescu Ion	20
Data Zbor: 2023-05-19	
Data Zbor: 2023-05-21	

7. Formulati in limbaj natural o problema pe care sa o rezolvati folosind un subprogram stocat independent care sa utilizeze 2 tipuri diferite de cursoare studiate, unul dintre acestea fiind cursor parametrizat, dependent de celalalt cursor. Apelati subprogramul.

Aceasta functie efectueaza o actualizare a salariilor angajatilor in functie de distanta parcursa pe ruta care porneste dintr-o tara specificata. Angajatii actualizati si detaliile lor sunt apoi returnati sub forma de cursor pentru utilizarea ulterioara. Sunt gestionate exceptii: daca nu exista date pentru o ruta potrivita sau pentru angajati, erori la actualizarea salariului sau daca tara nu exista.

```
CREATE OR REPLACE FUNCTION marireSalarii(p Tara
locatie.tara%TYPE) RETURN SYS_REFCURSOR IS
    v_cod_ruta ruta.cod_ruta%TYPE;
    v_dist ruta.distanta%TYPE;
    v_rows_updated NUMBER := 0;

    no_rute_data_found EXCEPTION;
    no_angajati_data_found EXCEPTION;
    invalid_tara_input EXCEPTION;
    update_failed EXCEPTION;

    c_rute SYS_REFCURSOR;

    -- Cursor care selecteaza angajatii implicați în rutele
    -- specificate
    CURSOR c_angajati (parametru NUMBER) IS
        SELECT a.cod_angajat, a.nume, a.prenume, a.salariu
        FROM angajat a
            JOIN lucreaza l ON a.cod_angajat = l.cod_angajat
            JOIN zbor z ON z.cod_zbor = l.cod_zbor
        WHERE z.cod_ruta = parametru
        FOR UPDATE OF a.salariu NOWAIT;
```

```

-- Cursor pentru angajatii actualizati
v_cursor SYS_REFCURSOR;
v_cod_angajat angajat.cod_angajat%TYPE;
v_nume angajat.nume%TYPE;
v_prenume angajat.prenume%TYPE;
v_salariu angajat.salariu%TYPE;

BEGIN
    -- Verificam daca p_tara contine cifre
    IF REGEXP_LIKE(p_tara, '\d') THEN
        RAISE invalid_tara_input;
    END IF;

    -- Deschidem cursorul pentru rute
    OPEN c_rute FOR
        SELECT r.cod_ruta, r.distanta
        FROM ruta r
            JOIN are ar ON r.cod_ruta = ar.cod_ruta
            JOIN aeroport a ON (a.cod_aeroport =
ar.cod_aeroport1
                                OR a.cod_aeroport =
ar.cod_aeroport2)
            JOIN locatie l ON l.cod_locatie = a.cod_locatie
        WHERE upper(l.tara) = upper(p_tara)
        AND ar.cod_aeroport1 != ar.cod_aeroport2;

    -- Deschidem cursorul de returnare în afara buclei
    OPEN v_cursor FOR
        SELECT a.cod_angajat, a.nume, a.prenume, a.salariu
        FROM angajat a
        WHERE 1 = 0; -- Conditie falsa pentru a initializa
cursorul, deoarece nu avem date inca

LOOP
    BEGIN
        -- Ia datele despre ruta curenta
        FETCH c_rute INTO v_cod_ruta, v_dist;

```

```

        EXIT WHEN c_rute%NOTFOUND;
        -- Verific?m dac? mai sunt rute
        IF c_rute%NOTFOUND THEN
            RAISE no_rute_data_found;
        END IF;

        -- iteram prin angajatii din ruta curenta
        FOR i IN c_angajati(v_cod_ruta) LOOP
            BEGIN
                -- Actualizam salariul angajatului
                UPDATE angajat
                SET salariu = salariu + v_dist
                WHERE cod_angajat = i.cod_angajat;

                -- Verificam daca s-au actualizat randuri
                IF SQL%ROWCOUNT > 0 THEN
                    v_rows_updated := v_rows_updated +
SQL%ROWCOUNT;

                    -- Adaugam angajatul in cursorul de
returnare
                    OPEN v_cursor FOR
                        SELECT a.cod_angajat, a.nume,
a.prenume, a.salariu
                        FROM angajat a
                        WHERE a.cod_angajat = i.cod_angajat;

                    FETCH v_cursor INTO v_cod_angajat,
v_nume, v_prenume, v_salariu;
                    DBMS_OUTPUT.PUT_LINE('Cod Angajat: ' ||
v_cod_angajat || ', Nume: ' || v_nume || ', Prenume: ' ||
v_prenume || ', Salariu: ' || v_salariu);
                END IF;

            EXCEPTION
                WHEN no_angajati_data_found THEN
                    DBMS_OUTPUT.PUT_LINE('Nu s-au g?sit date
pentru angajat.');
            END;
    
```

```
        END LOOP;

        EXCEPTION
            WHEN no_rute_data_found THEN
                DBMS_OUTPUT.PUT_LINE('Nu mai sunt date pentru
rute. Ie?ire din bucl?.');
                EXIT;
            END;

        END LOOP;

        -- Închidem cursorul pentru rute
        CLOSE c_rute;

        RETURN v_cursor;

    EXCEPTION
        WHEN invalid_tara_input THEN
            DBMS_OUTPUT.PUT_LINE('Intrare invalida pentru p_tara. Nu
ar trebui sa contine cifre.');
            RETURN NULL;

        WHEN update_failed THEN
            DBMS_OUTPUT.PUT_LINE('Eroare la actualizarea angajatului:
Niciun rand actualizat.');
            RETURN NULL;

        WHEN OTHERS THEN
            DBMS_OUTPUT.PUT_LINE('A aparut o eroare neasteptata: ' ||
SQLERRM);
            RETURN NULL;

    END marireSalarii;
/
```

```

ACCEPT tara CHAR PROMPT 'Introduceti tara: ';

DECLARE
    v_cursor SYS_REFCURSOR;
    v_cod_angajat angajat.cod_angajat%TYPE;
    v_nume angajat.nume%TYPE;
    v_prenume angajat.prenume%TYPE;
    v_salariu angajat.salariu%TYPE;

BEGIN
    v_cursor := marireSalarii('&tara');

    LOOP
        FETCH v_cursor INTO v_cod_angajat, v_nume, v_prenume,
v_salariu;
        EXIT WHEN v_cursor%NOTFOUND;

        DBMS_OUTPUT.PUT_LINE('Cod Angajat: ' || v_cod_angajat ||
', Nume: ' || v_nume || ', Prenume: ' || v_prenume || ', Salariu:
' || v_salariu);
    END LOOP;

    CLOSE v_cursor;

EXCEPTION
    WHEN OTHERS THEN
        DBMS_OUTPUT.PUT_LINE('A aparut o eroare: ' || SQLERRM);

END;
/

```

rollback;

```

  Oracle SQL Developer : C:\Users\40773\project.sql
  File Edit View Navigate Run Source Tools Window Help
  SQL History project.sql Welcome Page
  SQL Worksheet History
  SQL Worksheet 1 of 1 mariap19
  SELECT INTO angajat 1 of 1
  Worksheets Query Builder
  BEGIN
    v_cursor := marireSalarii('stara');

    LOOP
      FETCH v_cursor INTO v_cod_angajat, v_nume, v_prenume, v_salariu;
      EXIT WHEN v_cursor%NOTFOUND;

      DBMS_OUTPUT.PUT_LINE('Cod Angajat: ' || v_cod_angajat || ', Nume: ' || v_nume || ', Prenume: ' || v_prenume || ', Salariu: ' || v_salariu);
    END LOOP;

    CLOSE v_cursor;

  EXCEPTION
    WHEN OTHERS THEN
      DBMS_OUTPUT.PUT_LINE('A aparut o eroare: ' || SQLERRM);

  END;
/
ROLLBACK;

```

Script Output: Task completed in 0.363 seconds

```

  Cod Angajat: 1680, Nume: Ionescu, Prenume: Andrei, Salariu: 13000
  Cod Angajat: 1700, Nume: Muller, Prenume: Hans, Salariu: 11000
  Cod Angajat: 1750, Nume: Muhammad, Prenume: Ahmed, Salariu: 17000
  Cod Angajat: 1650, Nume: Popa, Prenume: Maria, Salariu: 15700
  Cod Angajat: 1730, Nume: Dupont, Prenume: Jean, Salariu: 9700
  Cod Angajat: 1760, Nume: James, Prenume: Anna, Salariu: 7700

  PL/SQL procedure successfully completed.

```

Line 2449 Column 16 | Insert | Modified: Windows_CN | 8:56 PM | 1/12/2024

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

Aceasta functie returneaza o colectie de locuri disponibile pentru o rezervare pentru zborurile care trec prin orasul specificat. Functia verifica daca orasul primit este in baza de date. Daca este se cauta zborurile asociate orasului si se identifica locurile disponibile pentru aceste zboruri.

```
CREATE OR REPLACE TYPE Locuri_Object AS Object (
    cod_loc NUMBER(5),
    cod_avion NUMBER(5),
    pozitie VARCHAR2(9),
    numar_loc NUMBER(3),
    cod_clasa NUMBER(5)
);
```

```
CREATE OR REPLACE TYPE Locuri_Avion AS TABLE OF Locuri_Object;
```

```
CREATE OR REPLACE FUNCTION VerificaLocuriLibere(
    p_oras_cautat VARCHAR2
) RETURN Locuri_Avion IS
    locuri_disponibile Locuri_Avion := Locuri_Avion();
    v_city_count NUMBER;
    v_un_loc loc%ROWTYPE;

    v_ruta_nr NUMBER;

    CURSOR c_zboruri IS
        SELECT z.cod_zbor, z.cod_avion
        FROM zbor z
        JOIN ruta r ON r.cod_ruta = z.cod_ruta
        JOIN are ar ON ar.cod_ruta = r.cod_ruta
        JOIN aeroport a ON a.cod_aeroport = ar.cod_aeroport
        JOIN locatie l ON l.cod_locatie = a.cod_locatie
        WHERE l.oras = p_oras_cautat;

    InvalidCityException EXCEPTION;
    NoFlightException EXCEPTION;
    NoAvailableSeatException EXCEPTION;

BEGIN
    SELECT COUNT(*)
    INTO v_city_count
```

```

FROM locatie l
WHERE l.oras = p_oras_cautat;

SELECT COUNT(*)
INTO v_ruta_nr
FROM zbor z
JOIN ruta r ON r.cod_ruta = z.cod_ruta
JOIN are ar ON ar.cod_ruta = r.cod_ruta
JOIN aeroport a ON a.cod_aeroport = ar.cod_aeroport1
JOIN locatie l ON l.cod_locatie = a.cod_locatie
WHERE l.oras = p_oras_cautat;

IF v_ruta_nr = 0 THEN
    RAISE NoFlightException;
END IF;

IF v_city_count = 0 THEN
    RAISE InvalidCityException;
END IF;

FOR zbor_rec IN c_zboruri LOOP
    FOR v_un_loc IN (SELECT *
                      FROM loc l
                     WHERE l.cod_avion = zbor_rec.cod_avion
                       AND l.cod_loc NOT IN (SELECT cod_loc
                                              FROM rezerva r
                                             WHERE r.cod_zbor =
zbor_rec.cod_zbor))
        LOOP
            locuri_disponibile.extend;
            locuri_disponibile(locuri_disponibile.last) :=
Locuri_Object(
                v_un_loc.cod_loc,
                v_un_loc.cod_avion,
                v_un_loc.pozitie,
                v_un_loc.numar_loc,
                v_un_loc.cod_clasa
);

```

```

        END LOOP;

    END LOOP;

    IF locuri_disponibile.count = 0 THEN
        RAISE NoAvailableSeatException;
    END IF;

    RETURN locuri_disponibile;

EXCEPTION
    WHEN InvalidCityException THEN
        DBMS_OUTPUT.PUT_LINE('Nu exista orasul');
        RETURN NULL;

    WHEN NoFlightException THEN
        DBMS_OUTPUT.PUT_LINE('Nu exista un astfel de zbor');
        RETURN NULL;

    WHEN NoAvailableSeatException THEN
        DBMS_OUTPUT.PUT_LINE('Nu sunt locuri disponibile');
        RETURN NULL;

    WHEN OTHERS THEN
        DBMS_OUTPUT.PUT_LINE('An error occurred: ' || SQLERRM);
        RETURN NULL;
END VerificaLocuriLibere;
/

```

SET SERVEROUTPUT ON;

```

DECLARE
    v_oras_cautat VARCHAR2(50);
    v_locuri_disponibile Locuri_Avion;
    v_numar_test NUMBER := &p_numar_test;
BEGIN
    IF v_numar_test NOT IN (1, 2, 3, 4) THEN

```

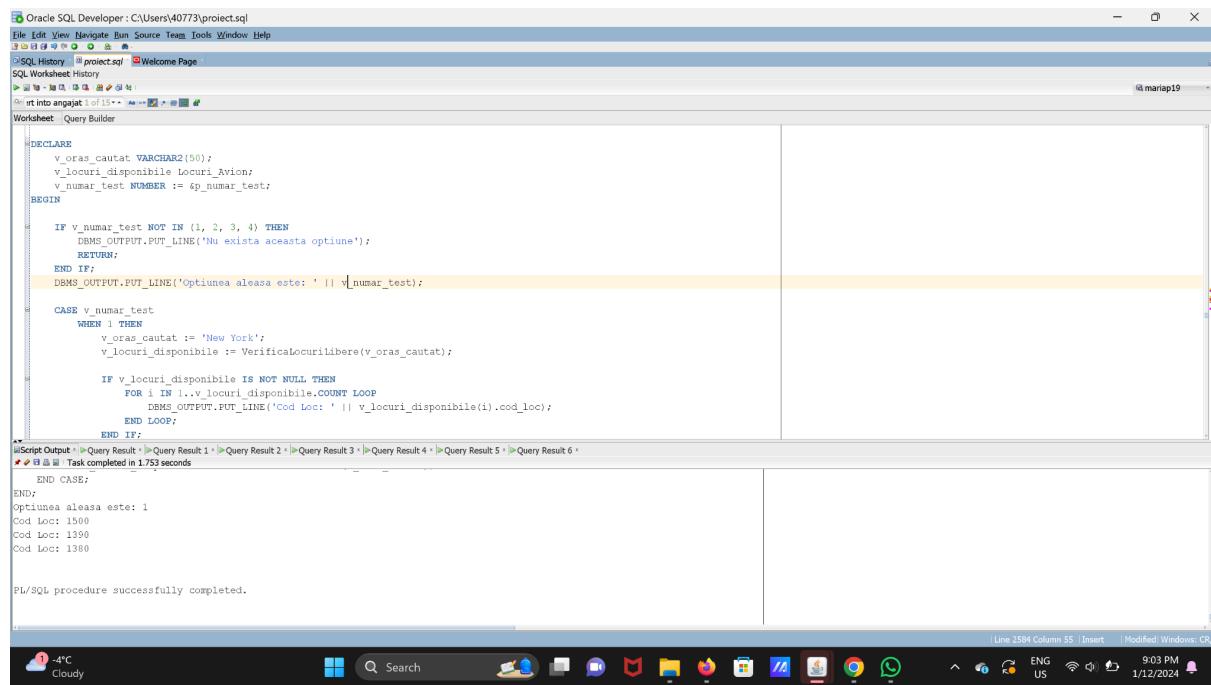
```

        DBMS_OUTPUT.PUT_LINE('Nu exista aceasta optiune');
        RETURN;
    END IF;
    DBMS_OUTPUT.PUT_LINE('Optiunea aleasa este: ' ||
v_numar_test);

CASE v_numar_test
    WHEN 1 THEN
        v_oras_cautat := 'New York';
        v_locuri_disponibile :=
VerificaLocuriLibere(v_oras_cautat);

        IF v_locuri_disponibile IS NOT NULL THEN
            FOR i IN 1..v_locuri_disponibile.COUNT LOOP
                DBMS_OUTPUT.PUT_LINE('Cod Loc: ' ||
v_locuri_disponibile(i).cod_loc);
            END LOOP;
        END IF;
    WHEN 2 THEN
        v_oras_cautat := 'Berlin';
        v_locuri_disponibile :=
VerificaLocuriLibere(v_oras_cautat);
    WHEN 3 THEN
        v_oras_cautat := 'havana'; --aici ceva nu e bine
        v_locuri_disponibile :=
VerificaLocuriLibere(v_oras_cautat);
    WHEN 4 THEN
        v_oras_cautat := 'Frankfurt';
        v_locuri_disponibile :=
VerificaLocuriLibere(v_oras_cautat);
    END CASE;
END;
/

```

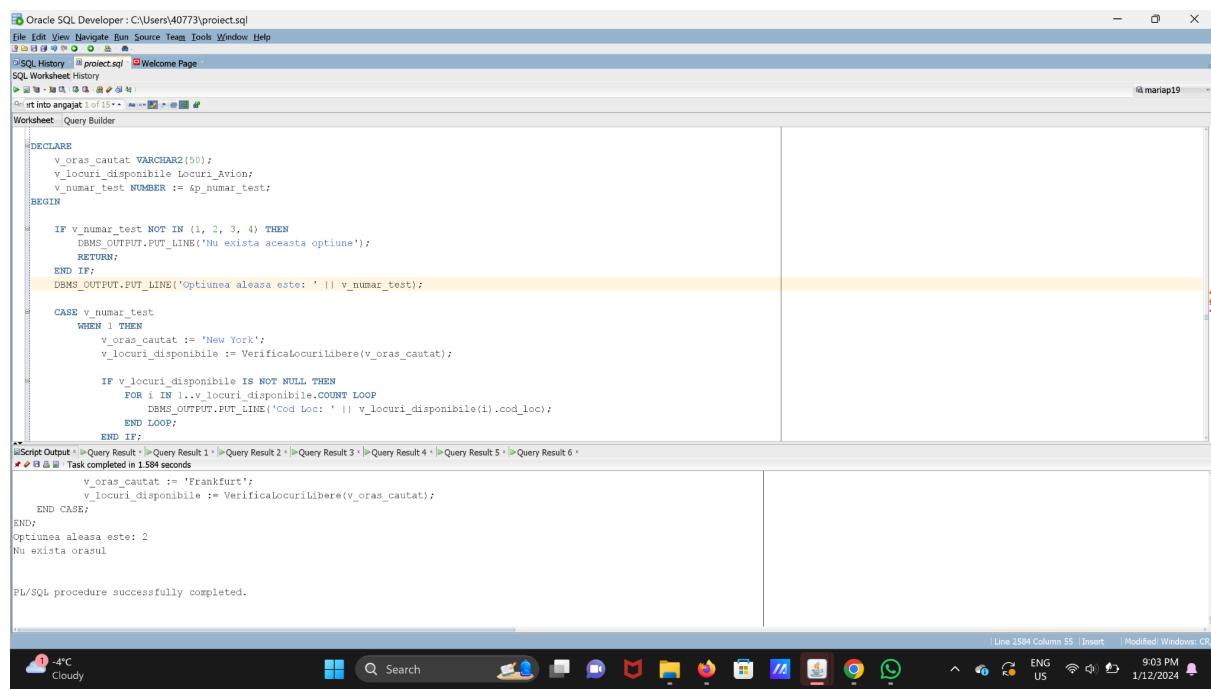


```
DECLARE
    v_oras_cautat VARCHAR2(50);
    v_locuri_disponibile Locuri_Avion;
    v_numar_test NUMBER := 4p_numar_test;
BEGIN
    IF v_numar_test NOT IN (1, 2, 3, 4) THEN
        DBMS_OUTPUT.PUT_LINE('Nu exista aceasta optiune');
        RETURN;
    END IF;
    DBMS_OUTPUT.PUT_LINE('Optiunea aleasa este: ' || v_numar_test);

    CASE v_numar_test
    WHEN 1 THEN
        v_oras_cautat := 'New York';
        v_locuri_disponibile := VerificaLocuriLibere(v_oras_cautat);

        IF v_locuri_disponibile IS NOT NULL THEN
            FOR i IN 1..v_locuri_disponibile.COUNT LOOP
                DBMS_OUTPUT.PUT_LINE('Cod Loc: ' || v_locuri_disponibile(i).cod_loc);
            END LOOP;
        END IF;
    END CASE;
END;
Optiunea aleasa este: 1
Cod Loc: 1500
Cod Loc: 1390
Cod Loc: 1380

PL/SQL procedure successfully completed.
```

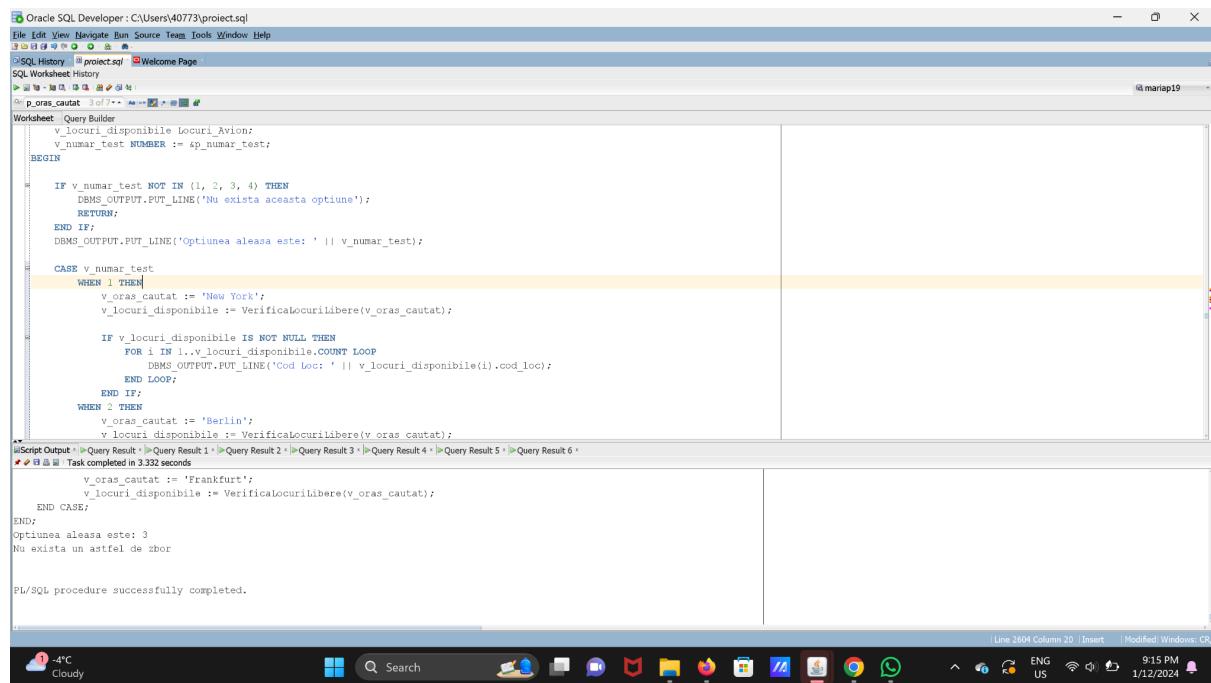


```
DECLARE
    v_oras_cautat VARCHAR2(50);
    v_locuri_disponibile Locuri_Avion;
    v_numar_test NUMBER := 4p_numar_test;
BEGIN
    IF v_numar_test NOT IN (1, 2, 3, 4) THEN
        DBMS_OUTPUT.PUT_LINE('Nu exista aceasta optiune');
        RETURN;
    END IF;
    DBMS_OUTPUT.PUT_LINE('Optiunea aleasa este: ' || v_numar_test);

    CASE v_numar_test
    WHEN 1 THEN
        v_oras_cautat := 'Frankfurt';
        v_locuri_disponibile := VerificaLocuriLibere(v_oras_cautat);

        IF v_locuri_disponibile IS NOT NULL THEN
            FOR i IN 1..v_locuri_disponibile.COUNT LOOP
                DBMS_OUTPUT.PUT_LINE('Cod Loc: ' || v_locuri_disponibile(i).cod_loc);
            END LOOP;
        END IF;
    END CASE;
END;
Optiunea aleasa este: 2
Nu exista orasul

PL/SQL procedure successfully completed.
```

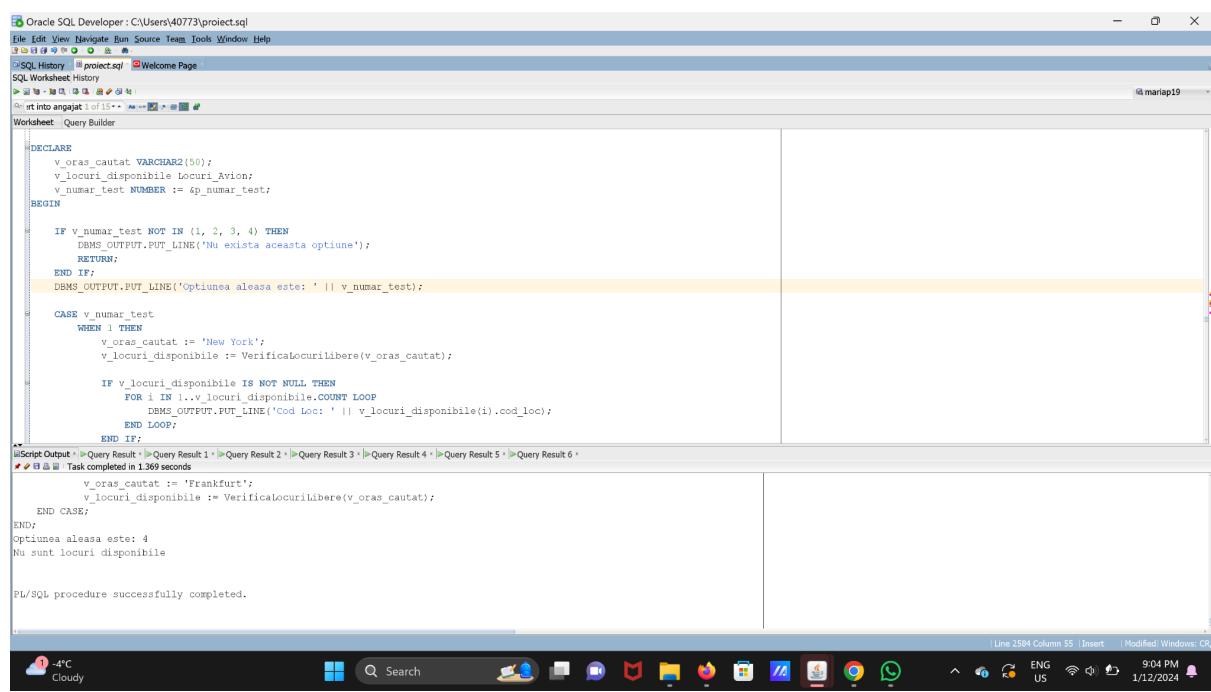


```
Oracle SQL Developer : C:\Users\40773\project.sql
File Edit View Navigate Run Source Tools Window Help
SQL History project.sql Welcome Page
SQL Worksheet History
p_oras_cautat 3 of 4
Worksheet - Query Builder
v_locuri_disponibile Locuri Avion;
v_numar_test NUMBER := 4;
BEGIN
    IF v_numar_test NOT IN (1, 2, 3, 4) THEN
        DBMS_OUTPUT.PUT_LINE('Nu exista aceasta optiune');
        RETURN;
    END IF;
    DBMS_OUTPUT.PUT_LINE('Optiunea aleasa este: ' || v_numar_test);

    CASE v_numar_test
        WHEN 1 THEN
            v_oras_cautat := 'New York';
            v_locuri_disponibile := VerificaLocuriLibere(v_oras_cautat);

            IF v_locuri_disponibile IS NOT NULL THEN
                FOR i IN 1..v_locuri_disponibile.COUNT LOOP
                    DBMS_OUTPUT.PUT_LINE('Cod Loc: ' || v_locuri_disponibile(i).cod_loc);
                END LOOP;
            END IF;
        WHEN 2 THEN
            v_oras_cautat := 'Berlin';
            v_locuri_disponibile := VerificaLocuriLibere(v_oras_cautat);
    END CASE;
END;
Optiunea aleasa este: 3
Nu exista un astfel de zbor

PL/SQL procedure successfully completed.
```



```
Oracle SQL Developer : C:\Users\40773\project.sql
File Edit View Navigate Run Source Tools Window Help
SQL History project.sql Welcome Page
SQL Worksheet History
p_oras_cautat 1 of 1
Worksheet - Query Builder
DECLARE
    v_oras_cautat VARCHAR2(50);
    v_locuri_disponibile Locuri Avion;
    v_numar_test NUMBER := 4;
BEGIN
    IF v_numar_test NOT IN (1, 2, 3, 4) THEN
        DBMS_OUTPUT.PUT_LINE('Nu exista aceasta optiune');
        RETURN;
    END IF;
    DBMS_OUTPUT.PUT_LINE('Optiunea aleasa este: ' || v_numar_test);

    CASE v_numar_test
        WHEN 1 THEN
            v_oras_cautat := 'New York';
            v_locuri_disponibile := VerificaLocuriLibere(v_oras_cautat);

            IF v_locuri_disponibile IS NOT NULL THEN
                FOR i IN 1..v_locuri_disponibile.COUNT LOOP
                    DBMS_OUTPUT.PUT_LINE('Cod Loc: ' || v_locuri_disponibile(i).cod_loc);
                END LOOP;
            END IF;
        WHEN 2 THEN
            v_oras_cautat := 'Frankfurt';
            v_locuri_disponibile := VerificaLocuriLibere(v_oras_cautat);
    END CASE;
END;
Optiunea aleasa este: 4
Nu sunt locuri disponibile

PL/SQL procedure successfully completed.
```

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

Procedura primește ca parametru o data de zbor și returnează informațiile despre facilitatea pe care a cumpărat-o pasagerul care are cel mai mic număr al locului (toată linia) din cadrul unui zbor dat. Aceste informații sunt apoi returnate prin intermediul parametrului de tip OUT, care conține numele pasagerului, numele facilității, prețul suplimentar și dacă este o facilitate standard sau nu.

```
CREATE OR REPLACE TYPE Facilitate_Nume_Object AS OBJECT (
    nume_pasager VARCHAR2(25),
    nume_facilitate VARCHAR2(25),
    pret_suplimentar NUMBER(3),
    standard CHAR(2)
);
```

```
CREATE OR REPLACE PROCEDURE obtineFacilitate(p_data_zbor IN DATE,
                                             p_facilitate OUT Facilitate_Nume_Object) IS

    TYPE ReservationCursor IS REF CURSOR;

    v_cursor ReservationCursor;

    v_nume_pasager.nume%TYPE;

    v_cod_loc loc.cod_loc%TYPE;

    v_nr_min_loc NUMBER := 2000;
```

```

v_nr_loc NUMBER;

v_verificare NUMBER;

v_nume_facilitate facilitate.nume_facilitate%TYPE;
v_pret_suplimentar facilitate.pret_suplimentar%TYPE;
v_standard facilitate.standard%TYPE;

cod_loc_cursor loc.cod_loc%TYPE;
nume_pasager_cursor pasager.nume%TYPE;

DATA_NU_EXISTA EXCEPTION;
PRAGMA EXCEPTION_INIT(DATA_NU_EXISTA, -20000);

BEGIN

SELECT COUNT(*)
INTO v_verificare
FROM zbor
WHERE data = p_data_zbor;

IF v_verificare = 0 THEN
    RAISE DATA_NU_EXISTA;
END IF;

--cursorul care codul, locul, codul pasagerului pt
rezervarile
--din cadrul zborului de pe acea data
OPEN v_cursor FOR
    SELECT r.cod_loc cod_loc, p.nume nume_pasager
    FROM zbor z JOIN rezerva r ON z.cod_zbor = r.cod_zbor
                  JOIN rezervare rez ON r.cod_rezervare =
rez.cod_rezervare

```

```

        JOIN cumpara c ON rez.cod_rezervare =
c.cod_rezervare
                JOIN pasager p ON c.cod_pasager =
p.cod_pasager
                WHERE z.data = p_data_zbor;

-- verificam care dintre linii corespunde locului cu cel mai
mic numar
LOOP
    FETCH v_cursor INTO cod_loc_cursor, nume_pasager_cursor;
    EXIT WHEN v_cursor%NOTFOUND;
    SELECT numar_loc
    INTO v_nr_loc
    FROM loc
    WHERE cod_loc = cod_loc_cursor;

    IF v_nr_loc < v_nr_min_loc THEN
        v_nr_min_loc := v_nr_loc;
        v_nume := nume_pasager_cursor;
        v_cod_loc := cod_loc_cursor;
    END IF;
END LOOP;

CLOSE v_cursor;

-- Obtine informatiile despre facilitate pentru locul cu cel
mai mic numar
SELECT f.nume_facilitate, f.pret_suplimentar, f.standard
INTO v_nume_facilitate, v_pret_suplimentar, v_standard
FROM loc l JOIN clasa c ON l.cod_clasa = c.cod_clasa
            JOIN ofera o ON c.cod_clasa = o.cod_clasa
            JOIN facilitate f ON o.cod_facilitate =
f.cod_facilitate
WHERE l.cod_loc = v_cod_loc;

p_facilitate := Facilitate_Nume_Object(v_nume,
v_nume_facilitate, v_pret_suplimentar, v_standard);

```

```

EXCEPTION

    WHEN DATA_NU_EXISTA THEN
        DBMS_OUTPUT.PUT_LINE('Nu există date specificate în
tabelul zbor');

    WHEN NO_DATA_FOUND THEN
        DBMS_OUTPUT.PUT_LINE('Nu sunt astfel de facilități');

    WHEN TOO_MANY_ROWS THEN
        DBMS_OUTPUT.PUT_LINE('Sunt prea multe facilități');

    WHEN OTHERS THEN
        DBMS_OUTPUT.PUT_LINE('An error occurred: ' || SQLERRM);

END;
/


--MERGE CUM AR TREBUI
set serveroutput on;
DECLARE
    v_data_zbor DATE := TO_DATE('2023-05-17', 'YYYY-MM-DD');
    v_facilitate Facilitate_Nume_Object;
BEGIN
    obtineFacilitate(p_data_zbor => v_data_zbor, p_facilitate =>
v_facilitate);

    DBMS_OUTPUT.PUT_LINE('Nume pasager: ' ||
v_facilitate.nume_pasager);
    DBMS_OUTPUT.PUT_LINE('Nume facilitate: ' ||
v_facilitate.nume_facilitate);
    DBMS_OUTPUT.PUT_LINE('Pret suplimentar: ' ||
v_facilitate.pret_suplimentar);
    DBMS_OUTPUT.PUT_LINE('Standard: ' || v_facilitate.standard);
END;

```

```
/



--TOO_MANY_ROWS
DECLARE
    v_data_zbor DATE := TO_DATE('2023-05-21', 'YYYY-MM-DD');
    v_facilitate Facilitate_Nume_Object;
BEGIN
    obtineFacilitate(p_data_zbor => v_data_zbor, p_facilitate =>
v_facilitate);

        DBMS_OUTPUT.PUT_LINE('Nume pasager: ' ||
v_facilitate.nume_pasager);
        DBMS_OUTPUT.PUT_LINE('Nume facilitate: ' ||
v_facilitate.nume_facilitate);
        DBMS_OUTPUT.PUT_LINE('Pret suplimentar: ' ||
v_facilitate.pret_suplimentar);
        DBMS_OUTPUT.PUT_LINE('Standard: ' || v_facilitate.standard);
END;
/



--NU_EXISTA_DATA
DECLARE
    v_data_zbor DATE := TO_DATE('2023-07-21', 'YYYY-MM-DD');
    v_facilitate Facilitate_Nume_Object;
BEGIN
    obtineFacilitate(p_data_zbor => v_data_zbor, p_facilitate =>
v_facilitate);

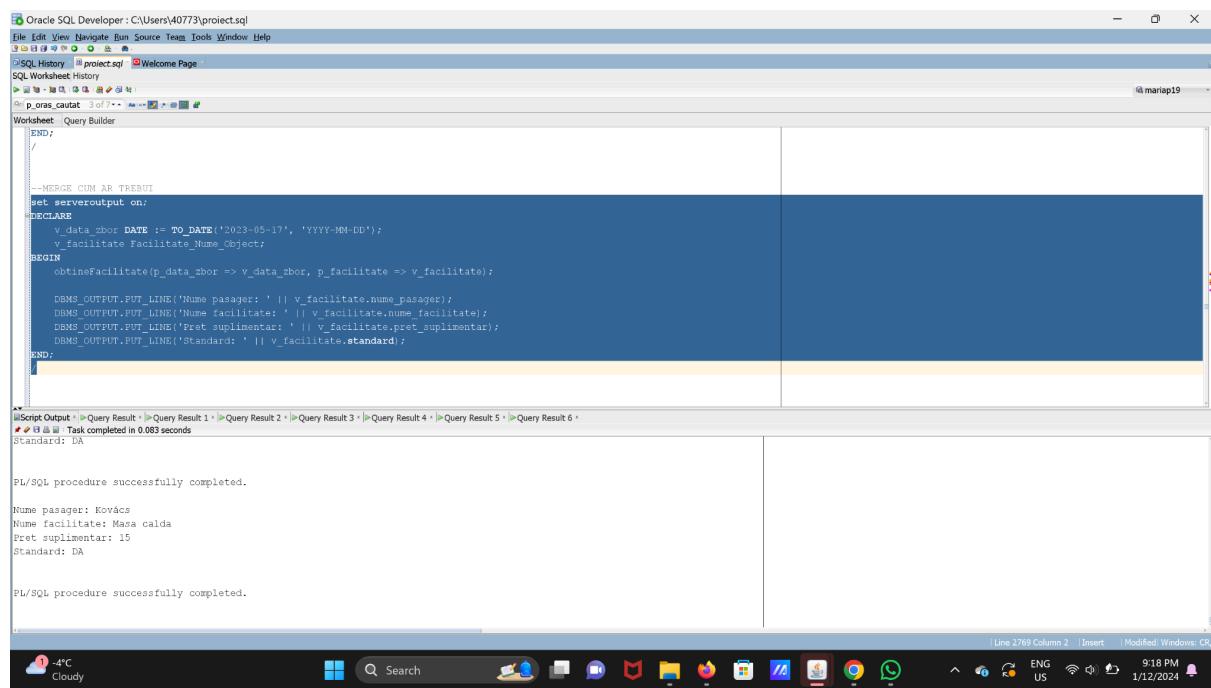
        DBMS_OUTPUT.PUT_LINE('Nume pasager: ' ||
v_facilitate.nume_pasager);
        DBMS_OUTPUT.PUT_LINE('Nume facilitate: ' ||
v_facilitate.nume_facilitate);
        DBMS_OUTPUT.PUT_LINE('Pret suplimentar: ' ||
v_facilitate.pret_suplimentar);
        DBMS_OUTPUT.PUT_LINE('Standard: ' || v_facilitate.standard);
END;
/
```

```

--NO_DATA_FOUND
DECLARE
    v_data_zbor DATE := TO_DATE('2023-05-22', 'YYYY-MM-DD');
    v_facilitate Facilitate_Nume_Object;
BEGIN
    obtineFacilitate(p_data_zbor => v_data_zbor, p_facilitate =>
v_facilitate);

    DBMS_OUTPUT.PUT_LINE('Nume pasager: ' ||
v_facilitate.nume_pasager);
    DBMS_OUTPUT.PUT_LINE('Nume facilitate: ' ||
v_facilitate.nume_facilitate);
    DBMS_OUTPUT.PUT_LINE('Pret suplimentar: ' ||
v_facilitate.pret_suplimentar);
    DBMS_OUTPUT.PUT_LINE('Standard: ' || v_facilitate.standard);
END;
/

```



```

--MERGE CUM AR TREBUT
set serveroutput on;
DECLARE
    v_data_zbor DATE := TO_DATE('2023-05-17', 'YYYY-MM-DD');
    v_facilitate Facilitate_Nume_Object;
BEGIN
    obtineFacilitate(p_data_zbor => v_data_zbor, p_facilitate => v_facilitate);

    DBMS_OUTPUT.PUT_LINE('Nume pasager: ' || v_facilitate.nume_pasager);
    DBMS_OUTPUT.PUT_LINE('Nume facilitate: ' || v_facilitate.nume_facilitate);
    DBMS_OUTPUT.PUT_LINE('Pret suplimentar: ' || v_facilitate.pret_suplimentar);
    DBMS_OUTPUT.PUT_LINE('Standard: ' || v_facilitate.standard);
END;
/

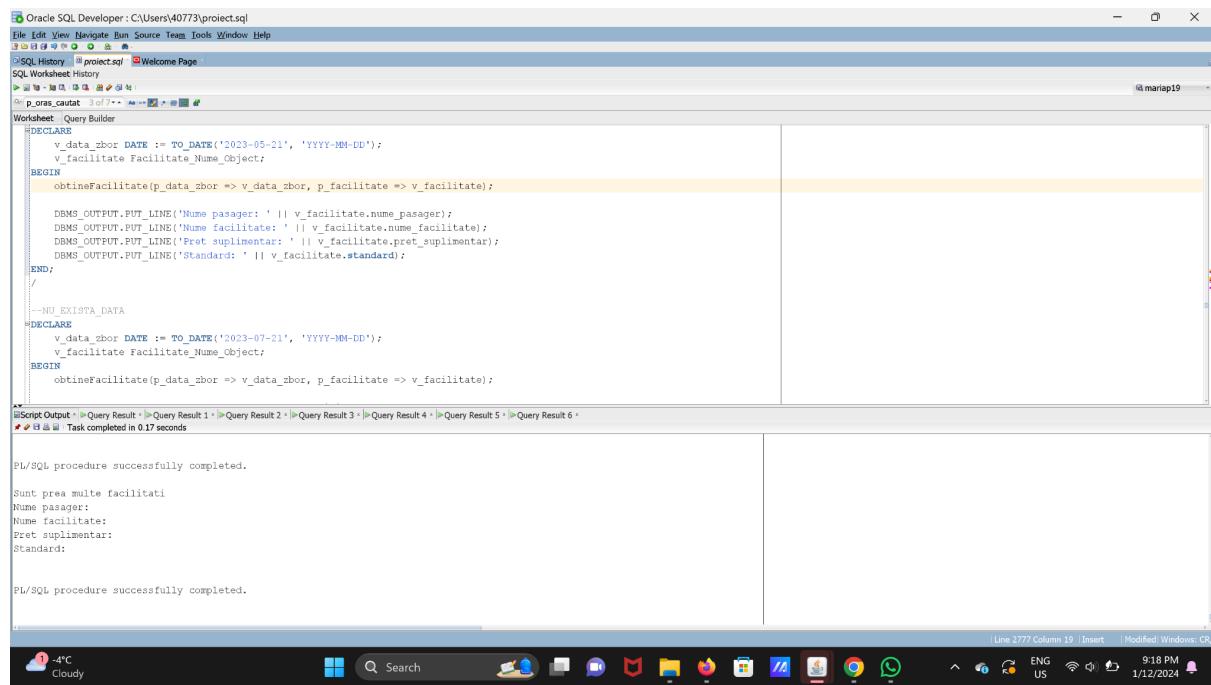
```

Script Output: Task completed in 0.083 seconds

PL/SQL procedure successfully completed.

Nume pasager: Kovács
Nume facilitate: Masa calda
Pret suplimentar: 15
Standard: DA

PL/SQL procedure successfully completed.



```
DECLARE
    v_data_zbor DATE := TO_DATE('2023-05-21', 'YYYY-MM-DD');
    v_facilitate Facilitate_Nume_Object;
    v_facilitate_Nume_Object;
BEGIN
    obtineFacilitate(p.data_zbor => v_data_zbor, p.facilitate => v_facilitate);

    DBMS_OUTPUT.PUT_LINE('Nume pasager: ' || v_facilitate.num_pasager);
    DBMS_OUTPUT.PUT_LINE('Nume facilitate: ' || v_facilitate.nume_facilitate);
    DBMS_OUTPUT.PUT_LINE('Pret suplimentar: ' || v_facilitate.pret_suplimentar);
    DBMS_OUTPUT.PUT_LINE('Standard: ' || v_facilitate.standard);
END;
/

--NU EXISTA DATA
DECLARE
    v_data_zbor DATE := TO_DATE('2023-07-21', 'YYYY-MM-DD');
    v_facilitate Facilitate_Nume_Object;
BEGIN
    obtineFacilitate(p.data_zbor => v_data_zbor, p.facilitate => v_facilitate);
END;
```

Script Output :> Query Result 1 :> Query Result 2 :> Query Result 3 :> Query Result 4 :> Query Result 5 :> Query Result 6 :>

Task completed in 0.17 seconds

PL/SQL procedure successfully completed.

Sunt prea multe facilitati

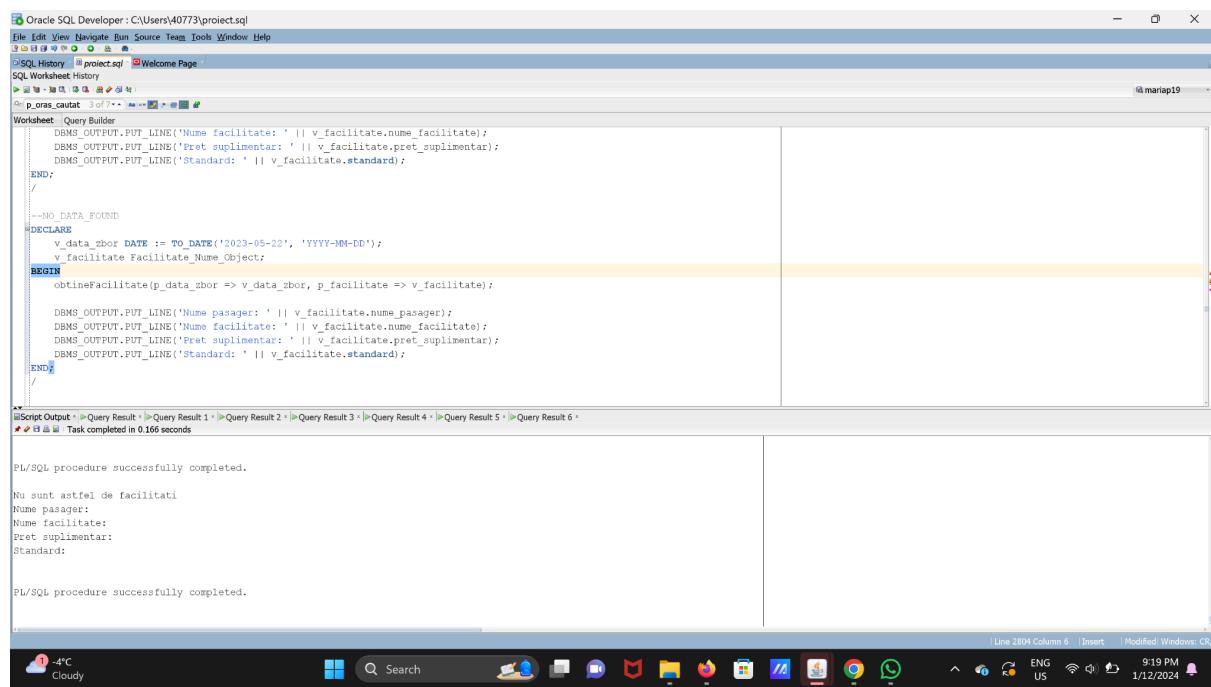
Nume pasager:

Nume facilitate:

Pret suplimentar:

Standard:

PL/SQL procedure successfully completed.



```
DECLARE
    v_data_zbor DATE := TO_DATE('2023-05-22', 'YYYY-MM-DD');
    v_facilitate Facilitate_Nume_Object;
    v_facilitate_Nume_Object;
BEGIN
    obtineFacilitate(p.data_zbor => v_data_zbor, p.facilitate => v_facilitate);

    DBMS_OUTPUT.PUT_LINE('Nume pasager: ' || v_facilitate.num_pasager);
    DBMS_OUTPUT.PUT_LINE('Nume facilitate: ' || v_facilitate.nume_facilitate);
    DBMS_OUTPUT.PUT_LINE('Pret suplimentar: ' || v_facilitate.pret_suplimentar);
    DBMS_OUTPUT.PUT_LINE('Standard: ' || v_facilitate.standard);
END;
/

--NO DATA_FOUND
DECLARE
    v_data_zbor DATE := TO_DATE('2023-05-22', 'YYYY-MM-DD');
    v_facilitate Facilitate_Nume_Object;
BEGIN
    obtineFacilitate(p.data_zbor => v_data_zbor, p.facilitate => v_facilitate);

    DBMS_OUTPUT.PUT_LINE('Nume pasager: ' || v_facilitate.num_pasager);
    DBMS_OUTPUT.PUT_LINE('Nume facilitate: ' || v_facilitate.nume_facilitate);
    DBMS_OUTPUT.PUT_LINE('Pret suplimentar: ' || v_facilitate.pret_suplimentar);
    DBMS_OUTPUT.PUT_LINE('Standard: ' || v_facilitate.standard);
END;
/
```

Script Output :> Query Result 1 :> Query Result 2 :> Query Result 3 :> Query Result 4 :> Query Result 5 :> Query Result 6 :>

Task completed in 0.66 seconds

PL/SQL procedure successfully completed.

Nu sunt astfel de facilitati

Nume pasager:

Nume facilitate:

Pret suplimentar:

Standard:

PL/SQL procedure successfully completed.

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

Avem un trigger care se activeaza inainte de operatiile LMD. Daca se incearca insert, se verifica daca ziua saptamanii este una de weekend. Pentru update, se verifica daca acesta se incearca intre orele 2am si 4 am. Pentru delete, se verifica daca utilizatorul care incearca sa le faca este "c##maria". Daca una dintre aceste conditii nu este indeplinita, atunci acel caz va fi oprit.

```
CREATE OR REPLACE TRIGGER lmdtriggercomanda
    BEFORE INSERT OR UPDATE OR DELETE ON poarta
DECLARE
    v_zi_sapt VARCHAR2(20);
    v_ora NUMBER;
BEGIN
    IF INSERTING THEN
        SELECT TO_CHAR(SYSDATE, 'DY') INTO v_zi_sapt FROM DUAL;

        IF v_zi_sapt NOT IN ('SAT', 'SUN') THEN
            RAISE_APPLICATION_ERROR(-20001, 'Inserarea poate fi
efectuata doar in weekenduri.');
        END IF;

    ELSIF UPDATING THEN
        SELECT TO_NUMBER(TO_CHAR(SYSDATE, 'HH24')) INTO v_ora
        FROM DUAL;

        IF v_ora < 2 OR v_ora >= 4 THEN
            RAISE_APPLICATION_ERROR(-20002, 'Actualizarea poate
fi efectuata doar intre orele 2AM si 4AM.');
        END IF;

    ELSE
        -- Deleting
        IF SYS_CONTEXT('USERENV', 'SESSION_USER') <> 'C##MARIA'
    THEN
        RAISE_APPLICATION_ERROR(-20003, 'Nu aveti permisiunea
de a sterge un angajat.');
    END IF;
    END IF;
END;
/
```



```
-- aceasta functie este facuta sa realizeze diferite comenzi LMD.  
Se va da numarul estimat de pasageri pentru  
-- poarta ca parametru. Daca numarul este mai mare de 200. se va  
introduce o noua inregistrare in tabel. Daca  
-- Numarul este intre 100 si 200, atunci se actualizeaza  
capacitatea unei porti cu 10%, poarta asociata  
-- terminalului cu cel mai mare numar de porti. Daca numarul este  
sub 100, atunci se sterg poartile care au  
-- in tabelul conectata asociata o pista care nu e folosita in  
cadrul zborurilor
```

```
CREATE OR REPLACE FUNCTION inupdelpoarta(p_nr_potential NUMBER)  
RETURN NUMBER IS
```

```
BEGIN
```

```
    -- inseamna ca pentru fiecare poarta se estimeaza ca vor fi  
    -- mult de 200 de pasageri
```

```
    IF p_nr_potential > 200 THEN
```

```
        INSERT INTO poarta  
        VALUES(8000, 460, 300, 'VEST');
```

```
    ELSIF p_nr_potential > 100 THEN
```

```
        UPDATE poarta  
        SET capacitate = capacitate + capacitate * 0.1  
        WHERE cod_poarta IN (  
            SELECT cod_poarta  
            FROM poarta  
            WHERE cod_terminal = (  
                SELECT cod_terminal  
                FROM (  
                    SELECT cod_terminal, COUNT(*) AS  
terminal_count
```

```

        FROM poarta
        GROUP BY cod_terminal
        ORDER BY COUNT(*) DESC
    )
    WHERE ROWNUM = 1
)
);

ELSE
    DELETE FROM poarta
    WHERE cod_poarta IN (
        SELECT cod_poarta
        FROM conectata C
        WHERE cod_pista = (
            SELECT MAX(cod_pista)
            FROM pista
            WHERE cod_pista NOT IN (

```

```

SET SERVEROUTPUT ON;
DECLARE
    v_result NUMBER;
BEGIN
    v_result := inupdelpoarta(250);
    DBMS_OUTPUT.PUT_LINE('Rezultat pentru p_nr_potential = 250: ' || TO_CHAR(v_result));
END;
/

```

Script Output • Query Result 1 • Query Result 2 • Query Result 3 • Query Result 4 • Query Result 5 • Query Result 6 • Task completed in 0.105 seconds

PL/SQL procedure successfully completed.

Function INUPDELPOARTA compiled

Rezultat pentru p_nr_potential = 250: 0

PL/SQL procedure successfully completed.

```

SET SERVEROUTPUT ON;
DECLARE
    v_result NUMBER;
BEGIN
    v_result := inupdelpoarta(250);
    DBMS_OUTPUT.PUT_LINE('Rezultat pentru p_nr_potential = 250: '
|| TO_CHAR(v_result));

```

The screenshot shows the Oracle SQL Developer interface. In the top-left pane, there is a code editor window titled 'p_pras_cautat' with the following PL/SQL code:

```

DECLARE
    v_result NUMBER;
BEGIN
    v_result := inupdelpoarta(250);
    DBMS_OUTPUT.PUT_LINE('Rezultat pentru p_nr_potential = 250: ' || TO_CHAR(v_result));
END;
/

```

Below the code editor, the 'Script Output' section displays the results of the execution:

```

Function INUPDELPOARTA compiled
Rezultat pentru p_nr_potential = 250: 0
PL/SQL procedure successfully completed.
Rezultat pentru p_nr_potential = 250: 0
PL/SQL procedure successfully completed.

```

The bottom right corner of the screen shows the Windows taskbar with various icons and system status information.

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

Acest trigger este activat înainte de operațiile lmd în cazul insertului, se verifică dacă există angajați din țara din care este angajatul care este inserat în cazul delete-ului, se verifică să nu mai existe niciun angajat angajat pe aceeași data primul trigger nu va merge deoarece va avea problema mutating, iar al doilea rezolvă problema.

```

create or replace trigger lmdTriggerAngajat
    before insert or delete on angajat
    for each row
declare
    v_aeroporturi number;
    v_ang NUMBER;
begin
    if inserting then
        -- Verifica dacă există aeroporturi în țara de origine a
        noului angajat sau dacă

```

```

-- exista angajati din aceeasi tara
select count(1)
into v_aeroporturi
from aeroport a JOIN locatie l ON l.cod_locatie =
a.cod_locatie
where tara = :new.tara;

select count(1)
into v_ang
from angajat
where tara = :new.tara;

if v_aeroporturi = 0 and v_ang = 0 then
    raise_application_error(-20001, 'Nu se poate adauga
angajatul. Nu exista aeroporturi in tara de origine sau angajati
din aceeasi tara');
end if;

ELSE -- DELETING
-- Verifica daca mai exista minim un angajat cu aceeasi
data_angajare
SELECT COUNT(*)
INTO v_ang
FROM angajat
WHERE data_angajarii = :old.data_angajarii;

IF v_ang > 1 THEN
    RAISE_APPLICATION_ERROR(-20003, 'Nu se poate sterge
angajatul. Mai exista cel putin un alt angajat cu aceeasi
data_angajare.');
END IF;
END IF;

end lmdTriggerAngajat;
/
CREATE OR REPLACE TRIGGER lmdTriggerAngajat

```

```

FOR INSERT OR UPDATE OR DELETE ON angajat
COMPOUND TRIGGER

    -- Utilizam o colectie de tip tabela asociativa pentru a
    stoca informatiile necesare
    TYPE t_tara_angajati IS TABLE OF VARCHAR2(30);
    v_tara_angajati t_tara_angajati := t_tara_angajati();

    TYPE t_date_angajare IS TABLE OF angajat.data_angajarii%TYPE;
    v_date_angajare t_date_angajare := t_date_angajare();

    BEFORE EACH ROW IS
    BEGIN

        v_tara_angajati.EXTEND;
        v_tara_angajati(v_tara_angajati.LAST) := :NEW.tara;

        v_date_angajare.EXTEND;
        v_date_angajare(v_date_angajare.LAST) :=
        :OLD.data_angajarii;

    END BEFORE EACH ROW;

    AFTER STATEMENT IS
        v_aeroporturi NUMBER;
        v_ang NUMBER;
        v_exista BOOLEAN := FALSE;
        v_data NUMBER;
    BEGIN
        -- Verificam conditiile pentru inserare
        IF INSERTING THEN
            FOR i IN 1 .. v_tara_angajati.COUNT LOOP
                SELECT COUNT(1)
                INTO v_aeroporturi
                FROM locatie
                WHERE tara = v_tara_angajati(i);

                IF v_aeroporturi = 1 THEN
                    v_exista := TRUE;
                END IF;
            END LOOP;
        END IF;
    END AFTER STATEMENT;

```

```

        END IF;

        SELECT COUNT(1)
        INTO v_ang
        FROM angajat
        WHERE tara = v_tara_angajati(i);

        IF v_ang = 1 THEN
            v_exista := TRUE;
        END IF;

        IF v_exista = FALSE THEN
            RAISE_APPLICATION_ERROR(-20003, 'Nu se poate
adauga angajatul. Nu exista aeroporturi in tara sa sau alti
angajati de acolo');
        END IF;

        END LOOP;

    END IF;

    -- Verificam conditiile pentru stergere
    IF DELETING THEN
        FOR i IN 1 .. v_date_angajare.COUNT LOOP
            SELECT COUNT(*)
            INTO v_data
            FROM angajat
            WHERE data_angajarii = v_date_angajare(i);

            IF v_data > 0 THEN
                RAISE_APPLICATION_ERROR(-20003, 'Nu se poate
sterge angajatul. Mai exist? cel pu?in un alt angajat cu aceea?i
data_angajare.');
            END IF;
        END LOOP;
    END IF;
END AFTER STATEMENT;

```

```

END lmdTriggerAngajat;
/


-- aceasta procedura are doi parametri: primul reprezinta un
string care contine tipul comenzii lmd, iar
-- al doilea o variabila boolean care verifica daca ar trebui sa
mearga sau sa fie oprita de triggerul definit
-- insert, true: pentru angajatii care au lucarat in aceleasi
zboruri ca JAMES, se fac inserari deriveate de la
-- datele lor.
-- insert, false:incearca inserarea derivata din informatiile
angajatilor care lucreaza in zborurile care
-- sunt legate de aeroporturi din tara de care sunt legate cele
mai multe zboruri. Pentru a calcula aceasta tara, am
-- verificat pentru fiecare in parte daca numarul de zboruri de
care este legata este egal cu maximul
-- dintre numerele zborurilor de care este legata fiecare tara.
Am folosit o cerere sincronizata in care
-- intervin mai mult de trei tabele, grupari de date cu subcereri
nesincronizate in care intervin cel putin 3
-- tabele, functii grup, filtrare la nivel de grupuri.
-- delete, false: se incercă stergerea angajatului cu cel mai
mare cod, dar mai există altul cu aceeași
-- data a angajării.

CREATE OR REPLACE PROCEDURE InUpDelAng(p_tip_lmd VARCHAR2,
p_acceptat BOOLEAN) IS
    TYPE CursorLMDType IS REF CURSOR;
    CursorLMD CursorLMDType;

    v_cod_ang angajat.cod_angajat%TYPE;
    v_tara angajat.tara%TYPE;
    v_salariu angajat.salariu%TYPE;

    v_linii_actualizate NUMBER;

BEGIN
    IF UPPER(p_tip_lmd) = 'INSERT' THEN
        IF p_acceptat = TRUE THEN

```

```

OPEN CursorLMD FOR
    SELECT ang.cod_angajat, ang.tara
    FROM angajat ang JOIN lucreaza lu ON
ang.cod_angajat = lu.cod_angajat
        WHERE lu.cod_zbor IN (SELECT l.cod_zbor
                                FROM angajat a JOIN
lucreaza l ON a.cod_angajat = l.cod_angajat
                                WHERE a.numere = 'James')
        AND ang.numere <> 'James'

    MINUS

    SELECT ang.cod_angajat, ang.tara
    FROM angajat ang JOIN lucreaza lu ON
ang.cod_angajat = lu.cod_angajat
        WHERE lu.cod_zbor NOT IN (SELECT l.cod_zbor
                                FROM angajat a JOIN
lucreaza l ON a.cod_angajat = l.cod_angajat
                                WHERE a.numere =
'James');

LOOP
    FETCH CursorLMD INTO v_cod_ang, v_tara;
    EXIT WHEN CursorLMD%NOTFOUND;

    INSERT INTO angajat
        VALUES (v_cod_ang + 3000, null, null, null, null,
null, v_tara);

END LOOP;

CLOSE CursorLMD;
ELSIF p_acceptat = FALSE THEN
    OPEN CursorLMD FOR
        SELECT ang.cod_angajat, ang.salariu
        FROM zbor zb JOIN lucreaza lu ON (lu.cod_zbor =
zb.cod_zbor)

```

```
        JOIN angajat ang ON(ang.cod_angajat
= lu.cod_angajat)
        WHERE lu.cod_zbor IN (SELECT Z.cod_zbor
        FROM LOCATIE L JOIN AEROPORT
A ON L.cod_locatie = A.cod_locatie
        JOIN ARE ar
ON ar.cod_aeroport1 = A.cod_aeroport OR
ar.cod_aeroport2 = A.cod_aeroport
        JOIN RUTA R
ON R.cod_ruta = ar.cod_ruta
        JOIN ZBOR Z
ON ar.cod_ruta = Z.cod_ruta
        WHERE L.tara IN (
        SELECT
L.tara
        FROM
LOCATIE L JOIN AEROPORT A ON L.cod_locatie = A.cod_locatie
        JOIN ARE ar ON A.cod_aeroport = ar.cod_aeroport1 OR
A.cod_aeroport = ar.cod_aeroport2
        JOIN RUTA R ON ar.cod_ruta = R.cod_ruta
        GROUP BY
L.tara
        HAVING
COUNT(*) = (
        SELECT MAX(numar_zboruri)
        FROM (
        SELECT L.tara, COUNT(*) AS numar_zboruri
        FROM LOCATIE L
        JOIN AEROPORT A ON L.cod_locatie = A.cod_locatie
```

```

JOIN ARE ar ON A.cod_aeroport = ar.cod_aeroport1 OR
A.cod_aeroport = ar.cod_aeroport2

JOIN RUTA R ON ar.cod_ruta = R.cod_ruta

GROUP BY L.tara
)

Subcerere

))));

      LOOP
        FETCH CursorLMD INTO v_cod_ang, v_salariu;
        EXIT WHEN CursorLMD%NOTFOUND;

        INSERT INTO angajat
          VALUES (v_cod_ang + 3000, null, v_salariu + 1000,
null, null, null, 'Rusia');

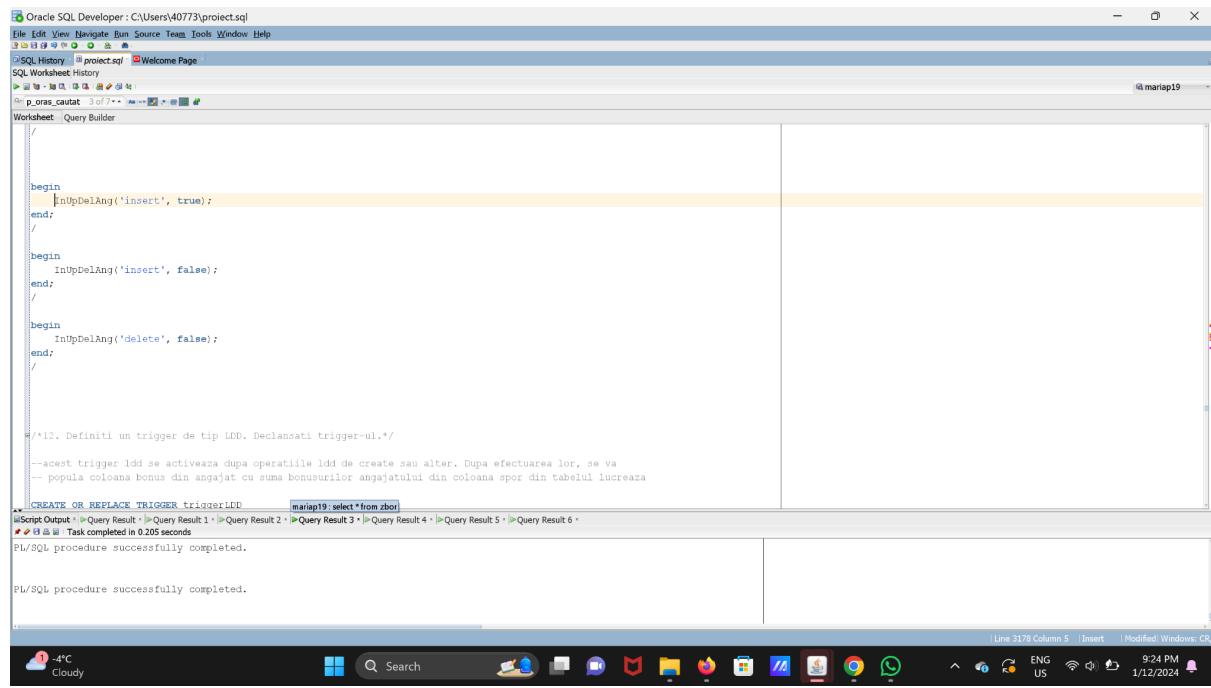
      END LOOP;

      CLOSE CursorLMD;
    END IF;

ELSIF UPPER(p_tip_lmd) = 'DELETE' THEN
  IF p_acceptat = FALSE THEN
    DELETE FROM angajat
    WHERE cod_angajat = (SELECT MAX(cod_angajat)
                          FROM angajat);
  END IF;
END IF;
END InUpDelAng;
/

```

```
begin
    InUpDelAng('insert', true);
end;
/
begin
    InUpDelAng('insert', false);
end;
/
begin
    InUpDelAng('delete', false);
end;
/
```



Oracle SQL Developer : C:\Users\40773\project.sql

File Edit View Navigate Run Source Team Tools Window Help

SQL History project.sql Welcome Page

SQL Worksheet History

p_ora_cautat 3 of 1 * - +

Worksheet Query Builder

```
        FROM angajat;
```

```
END IF;
END IF;
END InUpDelAng;
/
```

```
begin
    InUpDelAng('insert', true);
end;
/
rollback;

set serveroutput on;
begin
    InUpDelAng('insert', false);
end;
/
begin
    InUpDelAng('delete', false);
end;
```

Script Output > Query Result > Query Result 1 > Query Result 2 > Query Result 3 > Query Result 4 > Query Result 5 > Query Result 6

Task completed in 0.257 seconds

Line 3185 Column 17 | Insert | Modified: Windows CR

Cloudy 9:40 PM 1/12/2024 ENG US

Oracle SQL Developer : C:\Users\40773\project.sql

File Edit View Navigate Run Source Team Tools Window Help

SQL History project.sql Welcome Page

SQL Worksheet History

p_ora_cautat 3 of 1 * - +

Worksheet Query Builder

```
begin
    InUpDelAng('insert', true);
end;
/
rollback;

set serveroutput on;
begin
    InUpDelAng('insert', false);
end;
/
begin
    InUpDelAng('delete', false);
end;
/
```

```
/*10. Definire un trigger select*from zbo*cu clansati trigger-ul.*/
Script Output > Query Result > Query Result 1 > Query Result 2 > Query Result 3 > Query Result 4 > Query Result 5 > Query Result 6
```

Task completed in 0.123 seconds

```
InUpDelAng('delete', false);
end;
Error report -
ORA-02290: integrity constraint (C##MARIA.FK_LUCREAZAI) violated - child record found
ORA-06512: at "C##MARIA.INUODELANG", line 90
ORA-06512: at line 2
02292. 00000 - "integrity constraint (%s.%s) violated - child record found"
Cause: attempted to delete a parent key value that had a foreign
dependency.
Action: delete dependencies first then parent or disable constraint.
```

Line 3189 Column 6 | Insert | Modified: Windows CR

Cloudy 9:44 PM 1/12/2024 ENG US

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

Acet trigger ldd se activeaza dupa operatiile ldd de create sau alter. Dupa efectuarea lor, se va popula coloana bonus din angajat cu suma bonusurilor angajatului din coloana spor din tabelul lucreaza.

```
alter table angajat
add bonus number;

CREATE OR REPLACE TRIGGER triggerLDD
AFTER CREATE OR ALTER ON SCHEMA
DECLARE
BEGIN
    FOR ang_rec IN (SELECT a.cod_angajat, SUM(l.spor) AS
total_spor
        FROM angajat a
        JOIN lucreaza l ON a.cod_angajat =
l.cod_angajat
        GROUP BY a.cod_angajat)
    LOOP
        UPDATE angajat
        SET bonus = ang_rec.total_spor
        WHERE cod_angajat = ang_rec.cod_angajat;
    END LOOP;

    DBMS_OUTPUT.PUT_LINE('Triggerul LDD a fost declansat');

END triggerLDD;
/
```



```
CREATE OR REPLACE TYPE EXEMPLU_PENTRU_TRIGGER IS TABLE OF
VARCHAR2(10);
```

```

File Edit View Navigate Run Source Tools Window Help
SQL History project.sql Welcome Page
SQL Worksheet History
p_oras_cautat 3 of 1 maria19
Worksheet: Query Builder
    WHERE cod_angajat = ang_rec.cod_angajat;
END LOOP;

DBMS_OUTPUT.PUT_LINE('Triggerul LDD a fost declarat');

END triggerLDD;
/


CREATE OR REPLACE TYPE EXEMPLU_PENTRU_TRIGGER IS TABLE OF VARCHAR2(10);

Select * from angajat;

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

CREATE OR REPLACE PACKAGE pachetCompanieAeriana AS
  TYPE ListaPasageri IS TABLE OF VARCHAR2(50);
  TYPE LocuriReservate IS VARRAY(10) OF NUMBER;

```

Script Output • Query Result 1 • Query Result 2 • Query Result 3 • Query Result 4 • Query Result 5 • Query Result 6

All Rows Fetched: 10 in 0.00 seconds

cod_angajat	nume	salariu	prenume	data_nasterii	data_angajarii	tara	bonus
1	Ivanescu	12000	Andrei	15-FEB-85	01-JUN-15	Romania	600
2	Ionescu	12000	Andrei	01-JUN-15	01-JUN-15	Romania	250
3	Ponta	15700	Maria	10-MAY-88	15-MAR-11	Italia	600
4	Popovici	14000	Ionel	01-JAN-90	01-JAN-13	Romania	500
5	Smith	18000	John	03-DEC-95	25-MAY-19	Marea Britanie	450
6	Popescu	13000	Andreea	10-JUL-92	01-JUN-18	Romania	300
7	Dubcont	19700	Paul	12-AFR-87	05-FEB-16	Franța	350
8	Filip	15000	Mihai	29-IUN-93	12-AUG-20	Corea de Sud	500
9	Mohammed	13500	Ali	01-JAN-90	01-JAN-13	Romania	600
10	James	17600	Anna	17-OCT-94	05-JUL-19	Marea Britanie	750

Line 3228 Column 23 | Insert | Modified: Windows-1252

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

```

CREATE OR REPLACE PACKAGE pachetCompanieAeriana AS
  PROCEDURE PasageriSiLocuriOcupateZbor;
  FUNCTION marireSalarii(p_tara locatie.tara%TYPE) RETURN
  SYS_REFCURSOR;

```

```

TYPE Locuri_Object IS RECORD (
    cod_loc NUMBER(5),
    cod_avion NUMBER(5),
    pozitie VARCHAR2(9),
    numar_loc NUMBER(3),
    cod_clasa NUMBER(5)
);

TYPE Locuri_Avion IS TABLE OF Locuri_Object;

FUNCTION VerificaLocuriLibere(
    p_oras_cautat VARCHAR2
) RETURN Locuri_Avion;

TYPE Facilitate_Nume_Object IS RECORD (
    nume_pasager VARCHAR2(25),
    nume_facilitate VARCHAR2(25),
    pret_suplimentar NUMBER(3),
    standard CHAR(2)
);

PROCEDURE obtineFacilitate(p_data_zbor IN DATE, p_facilitate
OUT Facilitate_Nume_Object);

FUNCTION inupdelpoarta(p_nr_potential NUMBER) RETURN NUMBER;

END pachetCompanieAeriana;
/

```

CREATE OR REPLACE PACKAGE BODY pachetCompanieAeriana AS

PROCEDURE PasageriSiLocuriOcupateZbor IS

TYPE ListaPasageri IS TABLE OF VARCHAR2(50);

TYPE LocuriRezervate IS VARRAY(20) OF NUMBER;

```

        TYPE InregistrareZbor IS RECORD (
            pasageri ListaPasageri,
            locuri_rezervate LocuriRezervate
        );

        TYPE OrarZboruri IS TABLE OF InregistrareZbor INDEX BY
VARCHAR2(20);

        rezervari_zboruri OrarZboruri;

        BEGIN
            FOR zbor_rec IN (SELECT DISTINCT TO_CHAR(data,
'YYYY-MM-DD') AS data_zbor
                            FROM ZBOR)
            LOOP
                BEGIN
                    SELECT DISTINCT P.NUME || ' ' || P.PRENUME
                    BULK COLLECT INTO
rezervari_zboruri(zbor_rec.data_zbor).pasageri
                    FROM zbor z
                    JOIN rezerva r ON z.cod_zbor = r.cod_zbor
                    JOIN rezervare rez ON rez.cod_rezervare =
r.cod_rezervare
                    JOIN cumpara c ON c.cod_rezervare =
rez.cod_rezervare
                    JOIN pasager p ON p.cod_pasager = c.cod_pasager
                    WHERE TO_CHAR(Z.DATA, 'YYYY-MM-DD') =
zbor_rec.data_zbor;

                    SELECT DISTINCT l.numar_loc
                    BULK COLLECT INTO
rezervari_zboruri(zbor_rec.data_zbor).locuri_rezervate
                    FROM zbor z
                    JOIN rezerva r ON z.cod_zbor = r.cod_zbor
                    JOIN loc l ON l.cod_loc = r.cod_loc
                    WHERE TO_CHAR(Z.DATA, 'YYYY-MM-DD') =
zbor_rec.data_zbor;
    
```

```

        DBMS_OUTPUT.PUT_LINE('Data Zbor: ' ||
zbor_rec.data_zbor);

        IF rezervari_zboruri.EXISTS(zbor_rec.data_zbor)
THEN
        DBMS_OUTPUT.PUT('Pasageri: ');

        FOR i IN
1..rezervari_zboruri(zbor_rec.data_zbor).pasageri.COUNT LOOP

DBMS_OUTPUT.PUT(rezervari_zboruri(zbor_rec.data_zbor).pasageri(i));
;

        IF i <
rezervari_zboruri(zbor_rec.data_zbor).pasageri.COUNT THEN
            DBMS_OUTPUT.PUT(' , ');
        END IF;
        END LOOP;

        DBMS_OUTPUT.NEW_LINE;

        DBMS_OUTPUT.PUT('Locuri Rezervate: ');

        FOR i IN
1..rezervari_zboruri(zbor_rec.data_zbor).locuri_rezervate.COUNT
LOOP

DBMS_OUTPUT.PUT(rezervari_zboruri(zbor_rec.data_zbor).locuri_reze
rvate(i));

        IF i <
rezervari_zboruri(zbor_rec.data_zbor).locuri_rezervate.COUNT THEN
            DBMS_OUTPUT.PUT(' , ');
        END IF;
        END LOOP;

        DBMS_OUTPUT.NEW_LINE;
ELSE

```

```

        DBMS_OUTPUT.PUT_LINE('Niciun pasager pentru
acest zbor.');
    END IF;
EXCEPTION
    WHEN NO_DATA_FOUND THEN
        DBMS_OUTPUT.PUT_LINE('No data found for the
date: ' || zbor_rec.data_zbor);
    WHEN OTHERS THEN
        DBMS_OUTPUT.PUT_LINE('An error occurred for
the date: ' || zbor_rec.data_zbor);
    END;

END LOOP;

EXCEPTION
    WHEN OTHERS THEN
        DBMS_OUTPUT.PUT_LINE('An unexpected error
occurred.');
END PasageriSiLocuriOcupateZbor;

```

```

FUNCTION marireSalarii(p_tara_locatie.tara%TYPE) RETURN
SYS_REFCURSOR IS
    v_cod_ruta ruta.cod_ruta%TYPE;
    v_dist ruta.distanta%TYPE;
    v_rows_updated NUMBER := 0;

    no_rute_data_found EXCEPTION;
    no_angajati_data_found EXCEPTION;
    invalid_tara_input EXCEPTION;
    update_failed EXCEPTION;

    c_rute SYS_REFCURSOR;

    -- Cursor care selecteaza angajatii implicate in rutele
specificate
    CURSOR c_angajati (parametru NUMBER) IS
        SELECT a.cod_angajat, a.nume, a.prenume, a.salariu
        FROM angajat a

```

```

        JOIN lucreaza l ON a.cod_angajat = l.cod_angajat
        JOIN zbor z ON z.cod_zbor = l.cod_zbor
        WHERE z.cod_ruta = parametru
        FOR UPDATE OF a.salariu NOWAIT;

-- Cursor pentru angajatii actualizati
v_cursor SYS_REFCURSOR;
v_cod_angajat angajat.cod_angajat%TYPE;
v_nume angajat.nume%TYPE;
v_prenume angajat.prenume%TYPE;
v_salariu angajat.salariu%TYPE;

BEGIN
-- Verificam daca p_tara contine cifre
IF REGEXP_LIKE(p_tara, '\d') THEN
    RAISE invalid_tara_input;
END IF;

-- Deschidem cursorul pentru rute
OPEN c_rute FOR
    SELECT r.cod_ruta, r.distanta
    FROM ruta r
        JOIN are ar ON r.cod_ruta = ar.cod_ruta
        JOIN aeroport a ON (a.cod_aeroport =
ar.cod_aeroport1
                                OR a.cod_aeroport =
ar.cod_aeroport2)
        JOIN locatie l ON l.cod_locatie = a.cod_locatie
        WHERE upper(l.tara) = upper(p_tara)
        AND ar.cod_aeroport1 != ar.cod_aeroport2;

-- Deschidem cursorul de returnare in afara buclei
OPEN v_cursor FOR
    SELECT a.cod_angajat, a.nume, a.prenume, a.salariu
    FROM angajat a
    WHERE 1 = 0; -- Conditie falsa pentru a initializa
cursorul, deoarece nu avem date inca

LOOP
```

```

BEGIN
    -- Ia datele despre ruta curenta
    FETCH c_rute INTO v_cod_ruta, v_dist;

    EXIT WHEN c_rute%NOTFOUND;
    -- Verific?m dac? mai sunt rute
    IF c_rute%NOTFOUND THEN
        RAISE no_rute_data_found;
    END IF;

    -- iteram prin angajatii din ruta curenta
    FOR i IN c_angajati(v_cod_ruta) LOOP
        BEGIN
            -- Actualizam salariul angajatului
            UPDATE angajat
            SET salariu = salariu + v_dist
            WHERE cod_angajat = i.cod_angajat;

            -- Verificam daca s-au actualizat randuri
            IF SQL%ROWCOUNT > 0 THEN
                v_rows_updated := v_rows_updated +
SQL%ROWCOUNT;

                -- Adaugam angajatul in cursorul de
returnare
                OPEN v_cursor FOR
                    SELECT a.cod_angajat, a.nume,
a.prenume, a.salariu
                    FROM angajat a
                    WHERE a.cod_angajat =
i.cod_angajat;

                FETCH v_cursor INTO v_cod_angajat,
v_nume, v_prenume, v_salariu;
                DBMS_OUTPUT.PUT_LINE('Cod Angajat: ' ||
v_cod_angajat || ', Nume: ' || v_nume || ', Prenume: ' ||
v_prenume || ', Salariu: ' || v_salariu);
            END IF;
        END;
    END;

```

```

        EXCEPTION
            WHEN no_angajati_data_found THEN
                DBMS_OUTPUT.PUT_LINE('Nu s-au gasit
date pentru angajat.');
            END;
        END LOOP;

        EXCEPTION
            WHEN no_rute_data_found THEN
                DBMS_OUTPUT.PUT_LINE('Nu mai sunt date pentru
rute. Ie?ire din bucl?.');
            EXIT;
        END;

    END LOOP;

    -- Închidem cursorul pentru rute
    CLOSE c_rute;

    RETURN v_cursor;

    EXCEPTION
        WHEN invalid_tara_input THEN
            DBMS_OUTPUT.PUT_LINE('Intrare invalida pentru p_tara.
Nu ar trebui sa contine cifre.');
            RETURN NULL;

        WHEN update_failed THEN
            DBMS_OUTPUT.PUT_LINE('Eroare la actualizarea
angajatului: Niciun rand actualizat.');
            RETURN NULL;

        WHEN OTHERS THEN
            DBMS_OUTPUT.PUT_LINE('A aparut o eroare neasteptata:
'|| SQLERRM);
            RETURN NULL;

```

```

END marireSalarii;

FUNCTION VerificaLocuriLibere(
    p_oras_cautat VARCHAR2
) RETURN Locuri_Avion IS
    locuri_disponibile Locuri_Avion := Locuri_Avion();
    v_city_count NUMBER;
    v_un_loc loc%ROWTYPE;

    v_ruta_nr NUMBER;

    CURSOR c_zboruri IS
        SELECT z.cod_zbor, z.cod_avion
        FROM zbor z
        JOIN ruta r ON r.cod_ruta = z.cod_ruta
        JOIN are ar ON ar.cod_ruta = r.cod_ruta
        JOIN aeroport a ON a.cod_aeroport = ar.cod_aeroport1
        JOIN locatie l ON l.cod_locatie = a.cod_locatie
        WHERE l.oras = p_oras_cautat;

    InvalidCityException EXCEPTION;
    NoFlightException EXCEPTION;
    NoAvailableSeatException EXCEPTION;

BEGIN
    SELECT COUNT(*)
    INTO v_city_count
    FROM locatie l
    WHERE l.oras = p_oras_cautat;

    SELECT COUNT(*)
    INTO v_ruta_nr
    FROM zbor z
    JOIN ruta r ON r.cod_ruta = z.cod_ruta
    JOIN are ar ON ar.cod_ruta = r.cod_ruta
    JOIN aeroport a ON a.cod_aeroport = ar.cod_aeroport1

```

```

JOIN locatie l ON l.cod_locatie = a.cod_locatie
WHERE l.oras = p_oras_cautat;

IF v_ruta_nr = 0 THEN
    RAISE NoFlightException;
END IF;

IF v_city_count = 0 THEN
    RAISE InvalidCityException;
END IF;

FOR zbor_rec IN c_zboruri LOOP
    FOR v_un_loc IN (SELECT *
                      FROM loc l
                     WHERE l.cod_avion =
zbor_rec.cod_avion
                      AND l.cod_loc NOT IN (SELECT cod_loc
                                             FROM rezerva r
                                             WHERE
r.cod_zbor = zbor_rec.cod_zbor))
        LOOP
            locuri_disponibile.extend;
            locuri_disponibile(locuri_disponibile.last) :=

Locuri_Object(
                v_un_loc.cod_loc,
                v_un_loc.cod_avion,
                v_un_loc.pozitie,
                v_un_loc.numar_loc,
                v_un_loc.cod_clasa
            );
    END LOOP;
END LOOP;

IF locuri_disponibile.count = 0 THEN
    RAISE NoAvailableSeatException;
END IF;

```

```

        RETURN locuri_disponibile;

EXCEPTION
    WHEN InvalidCityException THEN
        DBMS_OUTPUT.PUT_LINE('Nu exista orasul');
        RETURN NULL;

    WHEN NoFlightException THEN
        DBMS_OUTPUT.PUT_LINE('Nu exista un astfel de zbor');
        RETURN NULL;

    WHEN NoAvailableSeatException THEN
        DBMS_OUTPUT.PUT_LINE('Nu sunt locuri disponibile');
        RETURN NULL;

    WHEN OTHERS THEN
        DBMS_OUTPUT.PUT_LINE('An error occurred: ' ||
SQLERRM);
        RETURN NULL;
END VerificaLocuriLibere;

```

```

PROCEDURE obtineFacilitate(p_data_zbor IN DATE, p_facilitate
OUT Facilitate_Nume_Object) IS

    TYPE ReservationCursor IS REF CURSOR;

    v_cursor ReservationCursor;

    v_nume pasager.nume%TYPE;

    v_cod_loc loc.cod_loc%TYPE;

    v_nr_min_loc NUMBER := 2000;

```

```

v_nr_loc NUMBER;

v_verificare NUMBER;

v_nume_facilitate facilitate.nume_facilitate%TYPE;
v_pret_suplimentar facilitate.pret_suplimentar%TYPE;
v_standard facilitate.standard%TYPE;

cod_loc_cursor loc.cod_loc%TYPE;
nume_pasager_cursor pasager.nume%TYPE;

DATA_NU_EXISTA EXCEPTION;
PRAGMA EXCEPTION_INIT(DATA_NU_EXISTA, -20000);

BEGIN

    SELECT COUNT(*)
    INTO v_verificare
    FROM zbor
    WHERE data = p_data_zbor;

    IF v_verificare = 0 THEN
        RAISE DATA_NU_EXISTA;
    END IF;

    --cursorul care codul, locul, codul pasagerului pt
rezervarile
    --din cadrul zborului de pe acea data
    OPEN v_cursor FOR
        SELECT r.cod_loc cod_loc, p.nume nume_pasager
        FROM zbor z JOIN rezerva r ON z.cod_zbor = r.cod_zbor
                      JOIN rezervare rez ON r.cod_rezervare =
rez.cod_rezervare
                      JOIN cumpara c ON rez.cod_rezervare =
c.cod_rezervare

```

```

        JOIN pasager p ON c.cod_pasager =
p.cod_pasager
        WHERE z.data = p_data_zbor;

        -- verificam care dintre linii corespunde locului cu cel
mai mic numar
        LOOP
            FETCH v_cursor INTO cod_loc_cursor,
nume_pasager_cursor;
            EXIT WHEN v_cursor%NOTFOUND;
            SELECT numar_loc
            INTO v_nr_loc
            FROM loc
            WHERE cod_loc = cod_loc_cursor;

            IF v_nr_loc < v_nr_min_loc THEN
                v_nr_min_loc := v_nr_loc;
                v_nume := nume_pasager_cursor;
                v_cod_loc := cod_loc_cursor;
            END IF;
        END LOOP;

        CLOSE v_cursor;

        -- Obtine informatiile despre facilitate pentru locul cu
cel mai mic numar
        SELECT f.nume_facilitate, f.pret_suplimentar, f.standard
        INTO v_nume_facilitate, v_pret_suplimentar, v_standard
        FROM loc l JOIN clasa c ON l.cod_clasa = c.cod_clasa
                    JOIN ofera o ON c.cod_clasa = o.cod_clasa
                    JOIN facilitate f ON o.cod_facilitate =
f.cod_facilitate
        WHERE l.cod_loc = v_cod_loc;

        p_facilitate := Facilitate_Nume_Object(v_nume,
v_nume_facilitate, v_pret_suplimentar, v_standard);

```

```

EXCEPTION

    WHEN DATA_NU_EXISTA THEN
        DBMS_OUTPUT.PUT_LINE('Nu există date specificate în
tabelul zbor');

    WHEN NO_DATA_FOUND THEN
        DBMS_OUTPUT.PUT_LINE('Nu sunt astfel de facilități');

    WHEN TOO_MANY_ROWS THEN
        DBMS_OUTPUT.PUT_LINE('Sunt prea multe facilități');

    WHEN OTHERS THEN
        DBMS_OUTPUT.PUT_LINE('An error occurred: ' ||
SQLERRM);

END;

FUNCTION inupdelpoarta(p_nr_potential NUMBER) RETURN NUMBER
IS

BEGIN

    -- inseamna ca pentru fiecare poarta se estimeaza ca vor
    -- fi mai mult de 200 de pasageri

    IF p_nr_potential > 200 THEN

        INSERT INTO poarta
        VALUES(8000, 460, 300, 'VEST');

    ELSIF p_nr_potential > 100 THEN

        UPDATE poarta
        SET capacitate = capacitate + capacitate * 0.1
        WHERE cod_poarta IN (

```

```

        SELECT cod_poarta
        FROM poarta
        WHERE cod_terminal = (
            SELECT cod_terminal
            FROM (
                SELECT cod_terminal, COUNT(*) AS
terminal_count
                    FROM poarta
                    GROUP BY cod_terminal
                    ORDER BY COUNT(*) DESC
            )
            WHERE ROWNUM = 1
        )
    );

ELSE
    DELETE FROM poarta
    WHERE cod_poarta IN (
        SELECT cod_poarta
        FROM conectata C
        WHERE cod_pista = (
            SELECT MAX(cod_pista)
            FROM pista
            WHERE cod_pista NOT IN (
                SELECT cod_pista
                FROM pista P
                JOIN zbor z ON P.cod_pista = z.cod_pista1
OR P.cod_pista = z.cod_pista2
            )
        )
    );
END IF;

RETURN 0;
END inupdelpoarta;

```

```
END pachetCompanieAeriana;
/

-- APELURI PENTRU PACHET

-- pentru 6

set serveroutput on;
BEGIN
    PachetCompanieAeriana.PasageriSiLocuriOcupateZbor;
END;
/

-- pentru 7

ACCEPT tara CHAR PROMPT 'Introduceti tara: ';

DECLARE
```

```

v_cursor SYS_REFCURSOR;
v_cod_angajat angajat.cod_angajat%TYPE;
v_nume angajat.nume%TYPE;
v_prenume angajat.prenume%TYPE;
v_salariu angajat.salariu%TYPE;

BEGIN
    v_cursor := pachetCompanieAeriana.marireSalarii('&tara');

    LOOP
        FETCH v_cursor INTO v_cod_angajat, v_nume, v_prenume,
v_salariu;
        EXIT WHEN v_cursor%NOTFOUND;

        DBMS_OUTPUT.PUT_LINE('Cod Angajat: ' || v_cod_angajat ||
', Nume: ' || v_nume || ', Prenume: ' || v_prenume || ', Salariu:
' || v_salariu);
    END LOOP;

    CLOSE v_cursor;

EXCEPTION
    WHEN OTHERS THEN
        DBMS_OUTPUT.PUT_LINE('Ap?rut o eroare: ' || SQLERRM);

END;
/
rollback;

-- pentru 8

SET SERVEROUTPUT ON;

DECLARE
    v_oras_cautat VARCHAR2(50);
    v_locuri_disponibile pachetCompanieAeriana.Locuri_Avion;
    v_numar_test NUMBER := '&p_numar_test';
BEGIN

```

```

IF v_numar_test NOT IN (1, 2, 3, 4) THEN
    DBMS_OUTPUT.PUT_LINE('Nu exista aceasta optiune');
    RETURN;
END IF;

CASE v_numar_test
    WHEN 1 THEN
        v_oras_cautat := 'New York';
        v_locuri_disponibile :=
pachetCompanieAeriana.VerificaLocuriLibere(v_oras_cautat);

        IF v_locuri_disponibile IS NOT NULL THEN
            FOR i IN 1..v_locuri_disponibile.COUNT LOOP
                DBMS_OUTPUT.PUT_LINE('Cod Loc: ' ||
v_locuri_disponibile(i).cod_loc);
            END LOOP;
        END IF;
    WHEN 2 THEN
        v_oras_cautat := 'Berlin';
        v_locuri_disponibile :=
pachetCompanieAeriana.VerificaLocuriLibere(v_oras_cautat);
    WHEN 3 THEN
        v_oras_cautat := 'havana';
        v_locuri_disponibile :=
pachetCompanieAeriana.VerificaLocuriLibere(v_oras_cautat);
    WHEN 4 THEN
        v_oras_cautat := 'Frankfurt';
        v_locuri_disponibile :=
pachetCompanieAeriana.VerificaLocuriLibere(v_oras_cautat);
END CASE;
END;
/
-- pentru 9

```

```

--MERGE CUM AR TREBUI
set serveroutput on;
DECLARE
    v_data_zbor DATE := TO_DATE('2023-05-17', 'YYYY-MM-DD');
    v_facilitate pachetCompanieAeriana.Facilitate_Nume_Object;
BEGIN
    pachetCompanieAeriana.obtineFacilitate(p_data_zbor =>
v_data_zbor, p_facilitate => v_facilitate);

    DBMS_OUTPUT.PUT_LINE('Nume pasager: ' ||
v_facilitate.nume_pasager);
    DBMS_OUTPUT.PUT_LINE('Nume facilitate: ' ||
v_facilitate.nume_facilitate);
    DBMS_OUTPUT.PUT_LINE('Pret suplimentar: ' ||
v_facilitate.pret_suplimentar);
    DBMS_OUTPUT.PUT_LINE('Standard: ' || v_facilitate.standard);
END;
/


--TOO_MANY_ROWS
DECLARE
    v_data_zbor DATE := TO_DATE('2023-05-21', 'YYYY-MM-DD');
    v_facilitate pachetCompanieAeriana.Facilitate_Nume_Object;
BEGIN
    pachetCompanieAeriana.obtineFacilitate(p_data_zbor =>
v_data_zbor, p_facilitate => v_facilitate);

END;
/


--NU_EXISTA_DATA
DECLARE
    v_data_zbor DATE := TO_DATE('2023-07-21', 'YYYY-MM-DD');
    v_facilitate pachetCompanieAeriana.Facilitate_Nume_Object;
BEGIN
    pachetCompanieAeriana.obtineFacilitate(p_data_zbor =>
v_data_zbor, p_facilitate => v_facilitate);

```

```

END;
/

--NO_DATA_FOUND
DECLARE
    v_data_zbor DATE := TO_DATE('2023-05-22', 'YYYY-MM-DD');
    v_facilitate pachetCompanieAeriana.Facilitate_Nume_Object;
BEGIN
    pachetCompanieAeriana.obtineFacilitate(p_data_zbor =>
v_data_zbor, p_facilitate => v_facilitate);

END;
/


-- pentru 10

SET SERVEROUTPUT ON;
DECLARE
    v_result NUMBER;
BEGIN
    v_result := pachetCompanieAeriana.inupdelpoarta(250);
    DBMS_OUTPUT.PUT_LINE('Rezultat pentru p_nr_potential = 250: '
|| TO_CHAR(v_result));

END;
/


SET SERVEROUTPUT ON;
DECLARE
    v_result NUMBER;
BEGIN
    v_result := pachetCompanieAeriana.inupdelpoarta(150);
    DBMS_OUTPUT.PUT_LINE('Rezultat pentru p_nr_potential = 250: '
|| TO_CHAR(v_result));

END;
/

```



Screenshot of two integrated development environments (IDEs) showing database management and PL/SQL development.

Top Window (Database Explorer):

- Database Explorer:** Shows a project named "project" with three databases: "RUTA", "TERMINAL", and "BILET".
- Console:** Displays the following SQL code and its execution results:

```
PachetCompanieAeriana.PasageriSiLocuriOcupateZbor;
END;
[2024-01-12 22:24:42] completed in 5 ms
C##PROJECT> BEGIN
PachetCompanieAeriana.PasageriSiLocuriOcupateZbor;
END;
[2024-01-12 22:24:57] completed in 9 ms
Data Zbor: 2023-05-17
Pasageri: Kovács János
Locuri Rezervate: 14
Data Zbor: 2023-05-18
Pasageri: Kovács János
Locuri Rezervate: 114, 14
Data Zbor: 2023-05-19
Pasageri: Popescu Ion
Locuri Rezervate: 20
Data Zbor: 2023-05-21
Pasageri: Ionescu Maria
Locuri Rezervate: 21, 67
Data Zbor: 2023-05-22
Pasageri: Smith John
Locuri Rezervate: 12
Data Zbor: 2023-05-23
```

Bottom Window (Oracle SQL Developer):

- File Edit View Navigate Run Source Team Tools Window Help**
- SQL History:** Shows a query named "marire" with 5 rows.
- Worksheet:** Displays a PL/SQL script:

```
END;
/
-- pentru 7

ACCEPT tara CHAR PROMPT 'Introduceti tara: ';

DECLARE
    v_cursor SYS_REFCURSOR;
    v_cod_angajat angajat.cod_angajat%TYPE;
    v_nume_angajat angajat.nume%TYPE;
    v_prenume_angajat angajat.prenume%TYPE;
    v_salariu angajat.salariu%TYPE;

BEGIN
    v_cursor := pachetCompanieAeriana.marireSalarii('stara');

    LOOP
        FETCH v_cursor INTO v_cod_angajat, v_nume, v_prenume, v_salariu;
        EXIT WHEN v_cursor%NOTFOUND;
    END LOOP;
END;
```
- Script Output:** Shows the output of the PL/SQL procedure:

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

Oracle SQL Developer : C:\Users\40773\project.sql

File Edit View Navigate Run Source Tools Window Help

SQL History project.sql Welcome Page

SQL Worksheet History

Worksheet: Query Builder

```
v_locuri_disponibile := pachetCompanieAeriana.Locuri_Avion;
v_numar_test NUMBER := '4p_numar_test';
BEGIN
  IF v_numar_test NOT IN (1, 2, 3, 4) THEN
    DBMS_OUTPUT.PUT_LINE('Nu exista aceasta optiune');
    RETURN;
  END IF;

  CASE v_numar_test
  WHEN 1 THEN
    v_oras_cautat := 'New York';
    v_locuri_disponibile := pachetCompanieAeriana.VerificaLocuriLibere(v_oras_cautat);
    IF v_locuri_disponibile IS NOT NULL THEN
      FOR i IN 1..v_locuri_disponibile.COUNT LOOP
        dbms_output.put_line(i);
      END LOOP;
    ELSE
      dbms_output.put_line('Nu exista locuri disponibile pentru acest destinatie');
    END IF;
  WHEN 2 THEN
    v_oras_cautat := 'Berlin';
    v_locuri_disponibile := pachetCompanieAeriana.VerificaLocuriLibere(v_oras_cautat);
  WHEN 3 THEN
    v_oras_cautat := 'Hong Kong'; --aici ceva nu e bine
    v_locuri_disponibile := pachetCompanieAeriana.VerificaLocuriLibere(v_oras_cautat);
  WHEN 4 THEN
    v_oras_cautat := 'Frankfurt';
    v_locuri_disponibile := pachetCompanieAeriana.VerificaLocuriLibere(v_oras_cautat);
  END CASE;
END;
```

Cod Loc: 1500
Cod Loc: 1390
Cod Loc: 1380

PL/SQL procedure successfully completed.

Script Output:

Task completed in 1.566 seconds

Line 3846 Column 1 Insert Modified: Windows CR

Cloudy -4°C Search ENG US 10:50 PM 1/12/2024

Oracle SQL Developer : C:\Users\40773\project.sql

File Edit View Navigate Run Source Tools Window Help

SQL History project.sql Welcome Page

SQL Worksheet History

Worksheet: Query Builder

```
v_locuri_disponibile := pachetCompanieAeriana.Locuri_Avion;
v_numar_test NUMBER := '4p_numar_test';
BEGIN
  IF v_numar_test NOT IN (1, 2, 3, 4) THEN
    DBMS_OUTPUT.PUT_LINE('Nu exista aceasta optiune');
    RETURN;
  END IF;

  CASE v_numar_test
  WHEN 1 THEN
    v_oras_cautat := 'New York';
    v_locuri_disponibile := pachetCompanieAeriana.VerificaLocuriLibere(v_oras_cautat);
    IF v_locuri_disponibile IS NOT NULL THEN
      FOR i IN 1..v_locuri_disponibile.COUNT LOOP
        dbms_output.put_line(i);
      END LOOP;
    ELSE
      dbms_output.put_line('Nu exista locuri disponibile pentru acest destinatie');
    END IF;
  WHEN 2 THEN
    v_oras_cautat := 'Berlin';
    v_locuri_disponibile := pachetCompanieAeriana.VerificaLocuriLibere(v_oras_cautat);
  WHEN 3 THEN
    v_oras_cautat := 'Hong Kong'; --aici ceva nu e bine
    v_locuri_disponibile := pachetCompanieAeriana.VerificaLocuriLibere(v_oras_cautat);
  WHEN 4 THEN
    v_oras_cautat := 'Frankfurt';
    v_locuri_disponibile := pachetCompanieAeriana.VerificaLocuriLibere(v_oras_cautat);
  END CASE;
END;
```

Nu exista orasul

PL/SQL procedure successfully completed.

Script Output:

Task completed in 1.683 seconds

Line 3946 Column 1 Insert Modified: Windows CR

Cloudy -4°C Search ENG US 10:50 PM 1/12/2024

Oracle SQL Developer : C:\Users\40773\project.sql

File Edit View Navigate Run Source Tools Window Help

SQL History project.sql Welcome Page

SQL Worksheet History

Worksheet: Query Builder

```
v_locuri_disponibile := pacchetCompanieAeriana.Locuri_Avion;
v_numar_test NUMBER := '4p_numar_test';
BEGIN

    IF v_numar_test NOT IN (1, 2, 3, 4) THEN
        DBMS_OUTPUT.PUT_LINE('Nu exista aceasta optiune');
        RETURN;
    END IF;

    CASE v_numar_test
        WHEN 1 THEN
            v_oras_cautat := 'New York';
            v_locuri_disponibile := pacchetCompanieAeriana.VerificaLocuriLibere(v_oras_cautat);

            IF v_locuri_disponibile IS NOT NULL THEN
                FOR i IN 1..v_locuri_disponibile.COUNT LOOP
                    DBMS_OUTPUT.PUT_LINE('Locuri disponibile pentru ' || i);
                END LOOP;
            ELSE
                DBMS_OUTPUT.PUT_LINE('Nu sunt locuri disponibile');
            END IF;
        WHEN 2 THEN
            v_oras_cautat := 'Berlin';
            v_locuri_disponibile := pacchetCompanieAeriana.VerificaLocuriLibere(v_oras_cautat);
        WHEN 3 THEN
            v_oras_cautat := 'Hong Kong'; --saii ceva nu e bine
            v_locuri_disponibile := pacchetCompanieAeriana.VerificaLocuriLibere(v_oras_cautat);
        WHEN 4 THEN
            v_oras_cautat := 'Frankfurt';
            v_locuri_disponibile := pacchetCompanieAeriana.VerificaLocuriLibere(v_oras_cautat);
    END CASE;
END;
```

Nu sunt locuri disponibile

PL/SQL procedure successfully completed.

Line 3846 Column 1 | Insert | Modified: Windows CR

Cloudy 1 -4°C Search ENG US 10:51 PM 1/12/2024

Oracle SQL Developer : C:\Users\40773\project.sql

File Edit View Navigate Run Source Tools Window Help

SQL History project.sql Welcome Page

SQL Worksheet History

Worksheet: Query Builder

```
v_locuri_disponibile := pacchetCompanieAeriana.VerificaLocuriLibere(v_oras_cautat);
END CASE;
/
-- pentru 9

--MERGE CUN A TREBUI
set serveroutput on;
DECLARE
    v_data_zbor DATE := TO_DATE('2023-05-17', 'YYYY-MM-DD');
    v_facilitate pacchetCompanieAeriana.Facilitate_Nume_Record;
BEGIN
    pacchetCompanieAeriana.obtineFacilitate(p_data_zbor => v_data_zbor, p_facilitate => v_facilitate);

    DBMS_OUTPUT.PUT_LINE('Nume pasager: ' || v_facilitate.nume_pasager);
    DBMS_OUTPUT.PUT_LINE('Nume facilitate: ' || v_facilitate.nume_facilitate);
    DBMS_OUTPUT.PUT_LINE('Pret suplimentar: ' || v_facilitate.pret_suplimentar);
    DBMS_OUTPUT.PUT_LINE('Standard: ' || v_facilitate.standard);
END;
```

PL/SQL procedure successfully completed.

Name: pasager: Kovács
Name: facilitate: Masa calda
Pret suplimentar: 15
Standard: DA

Line 3905 Column 2 | Insert | Modified: Windows CR

Cloudy 1 -4°C Search ENG US 10:58 PM 1/12/2024

Oracle SQL Developer : C:\Users\40773\project.sql

File Edit View Navigate Run Source Tools Window Help

SQL History project.sql Welcome Page

SQL Worksheet History

maria 5 of 74

Worksheet - Query Builder

```
procEDURE
  pachetCompanieAeriana.obtineFacilitate(p_data_zbor => v_data_zbor, p_facilitate => v_facilitate);

  DBMS_OUTPUT.PUT_LINE('Num pasager: ' || v_facilitate.num_pasager);
  DBMS_OUTPUT.PUT_LINE('Num facilitate: ' || v_facilitate.num_facilitate);
  DBMS_OUTPUT.PUT_LINE('Pret suplimentar: ' || v_facilitate.pret_suplimentar);
  DBMS_OUTPUT.PUT_LINE('Standard: ' || v_facilitate.standard);

END;
/


--TOO_MANY_ROWS
DECLARE
  v_data_zbor DATE := TO_DATE('2023-05-21', 'YYYY-MM-DD');
  v_facilitate pachetCompanieAeriana.Facilitate_Nume_Record;
BEGIN
  pachetCompanieAeriana.obtineFacilitate(p_data_zbor => v_data_zbor, p_facilitate => v_facilitate);

END;
/


--NO_EXISTA_DATE
DECLARE
  v_data_zbor DATE := TO_DATE('2023-07-21', 'YYYY-MM-DD');
  
```

Script Output | Query Result | Task completed in 0.202 seconds

PL/SQL procedure successfully completed.

Sunt prea multe facilitati

PL/SQL procedure successfully completed.

Oracle SQL Developer : C:\Users\40773\project.sql

File Edit View Navigate Run Source Team Tools Window Help

SQL History project.sql Welcome Page

SQL Worksheet: History

mari@ mari 5:07 AM

Worksheet: Query Builder

```
begin
    pachetCompanieAeriana.obtineFacilitate(p_data_zbor => v_data_zbor, p_facilitate => v_facilitate);
END;
/

--NU_EXISTA_DATE
DECLARE
    v_data_zbor DATE := TO_DATE('2023-07-21', 'YYYY-MM-DD');
    v_facilitate pachetCompanieAeriana.Facilitate_Nume_Record;
BEGIN
    pachetCompanieAeriana.obtineFacilitate(p_data_zbor => v_data_zbor, p_facilitate => v_facilitate);

END;
/

--NO_DATA_FOUND
DECLARE
    v_data_zbor DATE := TO_DATE('2023-05-22', 'YYYY-MM-DD');
    v_facilitate pachetCompanieAeriana.Facilitate_Nume_Record;
BEGIN
    pachetCompanieAeriana.obtineFacilitate(p_data_zbor => v_data_zbor, p_facilitate => v_facilitate);

END;
/

```

Script Output: 1 query result
Task completed in 0.104 seconds

source: D:\FII\proiect\project.sql

PL/SQL procedure successfully completed.

Nu există date specificată în tabelul zbor

PL/SQL procedure successfully completed.

4°C Cloudy

Search

Line 3927 Column 1 Insert Modified: Windows: CR

ENG US 10:59 PM 1/12/2024

```
Oracle SQL Developer : C:\Users\40773\project.sql
File Edit View Navigate Run Source Tools Window Help
SQL History project.sql Welcome Page
SQL Worksheet History
mariap19 5 of 7 44
Worksheet: Query Builder
--> pachetCompanieAeriana
DECLARE
    v_data_zbor DATE := TO_DATE('2023-07-21', 'YYYY-MM-DD');
    v_facilitate pachetCompanieAeriana.Facilitate_Nume_Record;
BEGIN
    pachetCompanieAeriana.obtineFacilitate(p_data_zbor => v_data_zbor, p_facilitate => v_facilitate);
END;
/
-- NO DATA FOUND
DECLARE
    v_data_zbor DATE := TO_DATE('2023-06-22', 'YYYY-MM-DD');
    v_facilitate pachetCompanieAeriana.Facilitate_Nume_Record;
BEGIN
    pachetCompanieAeriana.obtineFacilitate(p_data_zbor => v_data_zbor, p_facilitate => v_facilitate);
END;
/
-- pentru 10
SET SERVEROUTPUT ON;
Script Output > Query Result
Task completed in 0.082 seconds
No object data specified in cursor ZBOR

PL/SQL procedure successfully completed.

Nu sunt astfel de facilitati

PL/SQL procedure successfully completed.

Line 3936 Column 2 | Insert | Modified: Windows CR
Cloudy 4°C Search ENG US 10:59 PM 1/12/2024
```

```
Oracle SQL Developer : C:\Users\40773\project.sql
File Edit View Navigate Run Source Tools Window Help
SQL History project.sql Welcome Page
SQL Worksheet History
mariap19 5 of 7 44
Worksheet: Query Builder
pachetCompanieAeriana.obtineFacilitate(p_data_zbor => v_data_zbor, p_facilitate => v_facilitate),
END;
/
-- pentru 10
SET SERVEROUTPUT ON;
DECLARE
    v_result NUMBER;
BEGIN
    v_result := pachetCompanieAeriana.inupdelpoarta(250);
    DBMS_OUTPUT.PUT_LINE('Rezultat pentru p_nr_potential = 250: ' || TO_CHAR(v_result));
END;
/
SET SERVEROUTPUT ON;
DECLARE
    v_result NUMBER;
BEGIN
    v_result := pachetCompanieAeriana.inupdelpoarta(150);
    DBMS_OUTPUT.PUT_LINE('Rezultat pentru p_nr_potential = 250: ' || TO_CHAR(v_result));
Script Output > Query Result
Task completed in 0.22 seconds
Nu sunt astfel de facilitati

PL/SQL procedure successfully completed.

Rezultat pentru p_nr_potential = 250: 0

PL/SQL procedure successfully completed.

Line 3949 Column 2 | Overwrite | Modified: Windows CR
Cloudy 4°C Search ENG US 10:59 PM 1/12/2024
```

The screenshot shows the Oracle SQL Developer interface. In the top-left pane, there is a large redacted area covering several lines of code. Below this, the 'Script Output' tab is visible, showing the following text:

```

Script Output * Query Result *
Task completed in 0.203 seconds
Resultat pentru p_nr_potential = 250: 0

PL/SQL procedure successfully completed.

Resultat pentru p_nr_potential = 250: 0

PL/SQL procedure successfully completed.

```

The bottom right corner of the screen shows the Windows taskbar with various icons and system status information.

14. 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)

Vom face un pachet în cadrul căruia vom avea diferite proceduri care, apelate între ele, vor citi dintr-un fisier dat ca parametru pasageri(nume, prenume, naționalitate, nrPasaport, oraș de plecare, oraș de ajungere). Se va căuta cea mai scurta distanță (în sensul schimbărilor de aeroporturi) cu ajutorul funcției bfs. Dacă există o astfel de rută, se va adăuga pasagerul în baza de date.

```

CREATE OR REPLACE PACKAGE pachetIntegrareFlux AS

    PROCEDURE citire_fisier_interactiuni(pth VARCHAR2);

```

```
FUNCTION proceseaza_linie(line VARCHAR2) RETURN VARCHAR2;

PROCEDURE push(nradaugat NUMBER);

FUNCTION front RETURN NUMBER;

PROCEDURE pop;

FUNCTION queueempty RETURN BOOLEAN;

FUNCTION bfs(orasStart NUMBER, orasStop NUMBER) RETURN
NUMBER;

FUNCTION retCod(p_oras VARCHAR2) RETURN NUMBER;

st NUMBER := 1;
dr NUMBER := 0;

TYPE queueType IS TABLE OF NUMBER INDEX BY BINARY_INTEGER;
coada queueType;

TYPE verifViz IS TABLE OF NUMBER INDEX BY PLS_INTEGER;

END pachetIntegrareFlux;
/

CREATE OR REPLACE PACKAGE BODY pachetIntegrareFlux AS

PROCEDURE citire_fisier_interactiuni(pth VARCHAR2) IS
v_insertCommand VARCHAR2(1024);
file_handle UTL_FILE.FILE_TYPE;
line VARCHAR2(1024);

fisier_lipsa EXCEPTION;
```

```

extensie_invalida EXCEPTION;
output_chunk VARCHAR2(32767);
commit_flag BOOLEAN := TRUE;
BEGIN

    file_handle := UTL_FILE.FOPEN('ORACLE_HOME', pth, 'R');

    LOOP
        BEGIN
            UTL_FILE.GET_LINE(file_handle, line);
            v_insertCommand := proceseaza_linie(line);
            output_chunk := v_insertCommand;
            DBMS_OUTPUT.PUT_LINE(output_chunk);

        EXCEPTION
            WHEN UTL_FILE.READ_ERROR THEN
                EXIT;
            WHEN NO_DATA_FOUND THEN
                EXIT;
            WHEN OTHERS THEN
                DBMS_OUTPUT.PUT_LINE('Error processing line:
' || SQLERRM);
        END;
    END LOOP;

    IF commit_flag THEN
        COMMIT;
    END IF;

    IF UTL_FILE.IS_OPEN(file_handle) THEN
        UTL_FILE.FCLOSE(file_handle);
    END IF;

    EXCEPTION
        WHEN fisier_lipsa THEN
            RAISE_APPLICATION_ERROR(-20002, 'Nu a fost gasit
fisierul la adresa: ' || pth);
        WHEN extensie_invalida THEN
            RAISE_APPLICATION_ERROR(-20003, 'Era asteptat un
fisier .txt');
    END;

```

```

WHEN OTHERS THEN
    IF UTL_FILE.IS_OPEN(file_handle) THEN
        UTL_FILE.FCLOSE(file_handle);
    END IF;
    DBMS_OUTPUT.PUT_LINE('A aparut eroarea: ' ||
SQLERRM);
END;

FUNCTION proceseaza_linie(line VARCHAR2) RETURN VARCHAR2 IS
    v_nume VARCHAR2(50);
    v_prenume VARCHAR2(50);
    v_nationalitate VARCHAR2(50);
    v_ident NUMBER;
    v_oras_start VARCHAR2(50);
    v_oras_stop VARCHAR2(50);
    v_separator VARCHAR2(1) := ' ';
    v_start_pos NUMBER := 1;
    v_end_pos NUMBER;

    v_cod_oras_start NUMBER;
    v_cod_oras_stop NUMBER;
    v_distanță NUMBER;

BEGIN
    FOR i IN 1..6 LOOP
        v_end_pos := INSTR(line, v_separator, v_start_pos);

        IF i = 6 AND v_end_pos = 0 THEN
            v_end_pos := LENGTH(line) + 1;
        END IF;

        CASE i

```

```

        WHEN 1 THEN v_nume := SUBSTR(line, v_start_pos,
v_end_pos - v_start_pos);
        WHEN 2 THEN v_prenume := SUBSTR(line,
v_start_pos, v_end_pos - v_start_pos);
        WHEN 3 THEN v_nationalitate := SUBSTR(line,
v_start_pos, v_end_pos - v_start_pos);
        WHEN 4 THEN v_ident := TO_NUMBER(SUBSTR(line,
v_start_pos, v_end_pos - v_start_pos)) DEFAULT NULL ON CONVERSION
ERROR);
        WHEN 5 THEN v_oras_start := SUBSTR(line,
v_start_pos, v_end_pos - v_start_pos);
        WHEN 6 THEN v_oras_stop := SUBSTR(line,
v_start_pos, v_end_pos - v_start_pos);
    END CASE;

    v_start_pos := v_end_pos + 1;
END LOOP;

-- Obtine codul aeroportului pentru orasul de start
v_cod_oras_start := retCod(v_oras_start);

-- Obtine codul aeroportului pentru orasul de stop
v_cod_oras_stop := retCod(v_oras_stop);

-- Verifica daca exista ruta intre orasele date, primind
distanta
v_distanta := bfs(v_cod_oras_start, v_cod_oras_stop);

IF v_distanta != -1 THEN
    -- Adaugam pasagerul în baza de date
    INSERT INTO pasager (cod_pasager, nume, prenume,
nationalitate, numar_pasaport)
        VALUES (cheietest2.NEXTVAL, v_nume, v_prenume,
v_nationalitate, v_ident);

    COMMIT;
    RETURN 'Pasager adaugat cu succes. Distanta va fi: '
|| v_distanta;
ELSE

```

```
        RETURN 'Nu exista ruta valida intre orasele
specificate pentru pasagerul ' || v_nume;
    END IF;

EXCEPTION
    WHEN NO_DATA_FOUND THEN
        RETURN 'La procesarea linie, nu s-a gasit data pentru
ORA-01403';
    WHEN OTHERS THEN
        RETURN 'La procesarea linie, a aparut eroarea: ' ||

SQLERRM;
END proceseaza_linie;
```

```
PROCEDURE push(nradaugat NUMBER) IS
BEGIN
    dr := dr + 1;
    coada(dr) := nradaugat;
END push;
```

```
FUNCTION front RETURN NUMBER IS
BEGIN
    RETURN coada(st);
END front;
```

```
PROCEDURE pop IS
BEGIN
    st := st + 1;
END pop;
```

```
FUNCTION queueempty RETURN BOOLEAN IS
BEGIN
    RETURN st > dr;
END queueempty;
```

```

FUNCTION bfs(orasStart NUMBER, orasStop NUMBER) RETURN NUMBER
IS
    viz verifViz;
    nodCurent NUMBER;
BEGIN
    st := 1;
    dr := 0;
    push(orasStart);
    viz(orasStart) := 1;

    WHILE NOT queueEmpty() LOOP
        nodCurent := front();
        pop();

        IF nodCurent = orasStop THEN
            RETURN viz(nodCurent) -1 ;
        END IF;

        FOR vecin IN (SELECT cod_aeroport2 AS codVecin
                      FROM are
                      WHERE cod_aeroport1 = nodCurent
                      UNION
                      SELECT cod_aeroport1 AS codVecin
                      FROM are
                      WHERE cod_aeroport2 = nodCurent)
        LOOP
            IF NOT viz.EXISTS(vecin.codVecin) THEN
                viz(vecin.codVecin) := viz(nodCurent) + 1;
                push(vecin.codVecin);
            END IF;

            END LOOP;

        END LOOP;

        RETURN -1;
    END bfs;

```

```
FUNCTION retCod(p_oras VARCHAR2) RETURN NUMBER IS
    v_cod_aeroport NUMBER := 0;
BEGIN
    --verificam care este codul aeroportului din orasul dat
    SELECT cod_aeroport
    INTO v_cod_aeroport
    FROM locatie l JOIN aeroport a ON l.cod_locatie =
a.cod_locatie
        WHERE l.oras = p_oras;

    RETURN v_cod_aeroport;
EXCEPTION
    WHEN NO_DATA_FOUND THEN
        RETURN 0;
END retCod;

END pachetIntegrareFlux;
/

SET SERVEROUTPUT ON;

DECLARE
BEGIN

pachetIntegrareFlux.citire_fisier_interactiuni('C:\app\40773\prod
uct\21c\dbhomeXE\intrare.txt');

END;
/
```

Oracle SQL Developer : C:\Users\40773\project.sql

File Edit View Navigate Run Source Tools Window Help

SQL History project.sql Welcome Page

SQL Worksheet History

mariap19

marina 5 of 5 mariap19

Worksheet Query Builder

```
SET SERVEROUTPUT ON;
DECLARE
BEGIN
    pachetIntegraleFlux.citire_fisier_interactiuni('C:\app\40773\product\21c\dbhomeXE\intrare.txt');
END;
```

```
select l.oras
from locatie l join aeroport a on l.cod_locatie = a.cod_locatie
where cod_aeroport = 240;
select * from are;
select * from aeroport;
```

Script Output Task completed in 0.08 seconds

456/33 PL/SQL: ORA-00947: not enough values
Errors: check compile log

Package PACHETINTEGRAREFLUX compiled

Package Body PACHETINTEGRAREFLUX compiled

Pasager adaugat cu succes. Distanța va fi: 2
Pasager adaugat cu succes. Distanța va fi: 2

PL/SQL procedure successfully completed.

Line 42/61 Column 2 Insert Modified: Windows: CR

Cloudy -4°C

Search

ENGLISH US 11:08 PM 1/12/2024