**Proiect SGBD**

**Administrare Hotel**

STAN C.G. ANA-MARIA

GRUPA 241

UNIVERSITATEA DIN BUCUREȘTI

FACULTATEA DE MATEMATICĂ ȘI INFORMATICĂ

SPECIALIZAREA INFORMATICĂ

PROFESOR COORDONATOR

**NATALIA GABRIELA MOANGA**

**07.01.2022**

**CUPRINS**

|  |  |  |
| --- | --- | --- |
| Capitolul 1 | **Cerința 1 - Prezentați pe scurt baza de date (utilitatea ei).** | Pag. 3 |
| Capitolul 2 | **Cerința 2 - Realizați diagrama entitate-relație (ERD).** | Pag. 9 |
| Capitolul 3 | **Cerința 3 - Pornind de la diagrama entitate-relație realizați diagrama conceptuală a modelului propus, integrând toate atributele necesare.** | Pag. 11 |
| Capitolul 4 | **Cerința 4 - Implementați în Oracle diagrama conceptuală realizată: definiți toate tabelele, implementând toate constrângerile de integritate necesare (chei primare, cheile externe etc).**  **Cerința 5 - Adăugați informații coerente în tabelele create (minim 5 înregistrări pentru fiecare entitate independentă; minim 10 înregistrări pentru tabela asociativă).** | Pag. 14  Pag. 17 |
| Capitolul 5 | **Cerința 6 - Formulați în limbaj natural o problemă pe care să o rezolvați folosind un subprogram stocat care să utilizeze două tipuri de colecție studiate. Apelați subprogramul.**  **Cerința 7 - Formulați în limbaj natural o problemă pe care să o rezolvați folosind un subprogram stocat care să utilizeze un tip de cursor studiat. Apelați subprogramul.**  **Cerința 8 – Formulați în limbaj natural o problemă pe care să o rezolvați folosind un subprogram stocat de tip funcție care să utilizeze într-o singură comandă SQL 3 dintre tabelele definite. Tratați toate excepțiile care pot apărea. Apelați subprogramul astfel încât să evidențiați toate cazurile tratate.**  **Cerința 9 - Formulați în limbaj natural o problemă pe care să o rezolvați folosind un subprogram stocat de tip procedură care să utilizeze într-o singură comandă SQL 5 dintre tabelele definite. Tratați toate excepțiile care pot apărea, incluzând excepțiile NO\_DATA\_FOUND și TOO\_MANY\_ROWS. Apelați subprogramul astfel încât să evidențiați toate cazurile tratate.** | Pag. 36  Pag. 38  Pag. 40  Pag. 43 |
| Capitolul 6 | **Cerința 10 - Definiți un *trigger* de tip LMD la nivel de comandă. Declanșați *trigger*-ul.**  **Cerința 11 - Definiți un *trigger* de tip LMD la nivel de linie. Declanșați *trigger*-ul.**  **Cerința 12 - Definiți un *trigger* de tip LDD. Declanșați *trigger*-ul.** | Pag. 46  Pag. 47  Pag. 48 |
| Capitolul 7 | **Cerința 13 - Definiți un pachet care să conțină toate obiectele definite în cadrul proiectului.** | Pag. 50 |

**Capitolul 1**

**Prezentați pe scurt baza de date (utilitatea ei)**

**(Cerința 1)**

**Informații generale:**

Proiectul meu are ca obiectiv ușurarea stocării de informații legate de gestionarea unui hotel. Accentul este pus pe rezervarea unei camere, informații despre aceasta și două categorii principale de angajați (menajere și recepționiști). Această idee este utilă, deoarece este foarte greu să ții evidența rezervărilor într-un hotel cu un număr semnificativ de mare de camere și personal. Proiectarea unui sistem care să nu prezinte probleme de gestionare, cu materialele didactice predate, este aproape imposibil pe tema hoteliera. Așadar, o să prezint mai jos restricțiile care trebuie luate în considerare pentru proiectarea unei diagrame cât mai bune.

În proiect se consideră mai multe hoteluri, fiecare având proprii săi angajați și camere. Mai multe hoteluri pot să aibă același client, dar fiecare hotel va avea codul camerelor, al angajaților și al rezervării diferit.

O recepționeră poate prelua mai multe rezervări deodată. Un client pentru a obține o cameră de hotel trebuie sa facă neapărat o rezervare. Clientul va provenii dintr-o țară. Țările introduse în tabel vor fi în special cele din care provine cel puțin un client, dar vor exista și unele țări fără nici un client. Tabelul asociativ numit APARE\_IN\_ISTORICUL va conține informații despre prețul camerei, clientul și camera rezervata.

**Prezentarea constrângerilor:**

În proiectul meu exista mai multe **restricții de funcționare**, care vor fi prezentate mai jos. Acestea sunt obligatoriu de respectat pentru buna funcționare a bazei de date:

* Există cel puțin 2 hoteluri în modelul de date (2 hoteluri pot avea același nume<<EXEMPLU: HILTON, HOLIDAY INN>>, adresa trebuie să fie unică și ambele câmpuri să nu fie null)
* Un angajat poate să lucreze pentru un singur hotel, iar acesta poate să fie doar de tip CAMERISTA sau RECEPTIONISTA
* Camerele diferă de la hotel la hotel, o camera dintr-un hotel nu poate exista și în altele, dar numărul camerei poate să existe și în alt hotel
* Hotelul trebuie să dețină cel puțin o cameră și cel puțin un angajat
* Un client provine dintr-o singură țară, iar acesta poate să facă mai multe rezervări, în perioade diferite, la mai multe hoteluri
* Nu pot să existe 2 rezervări care se suprapun pe o cameră
* Unele camere pot să nu aibă rezervări făcute pe ele
* O rezervare este făcută pe o singura cameră
* O recepționistă poate să preia mai multe rezervări, dar în același timp poate să nu preia niciuna
* Codul angajatului din entitatea rezervare trebuie neapărat să îi aparțină unei RECEPTIONISTE
* Codul angajatului din entitatea cameriste trebuie neapărat să îi aparțină unei CAMERISTE
* Mai multe cameriste pot îngriji mai multe camere, dar în același timp o cameristă poate sa îngrijească singură o cameră sau să nu îngrijească nici una
* În tabelul asociativ la atributul numardeoreingrijire se vor introduce numere mici de forma 1,2,3 pentru a avea sens
* În baza de date vor fi luați în considerare doar clienții care fac cel puțin o rezervare, angajații care aparțin unui hotel și camerele care sunt deținute de cel puțin un hotel
* Un hotel trebuie să aibă neapărat o cameră
* În entitatea istoric camera pot exista camere în care nu s-a cazat nimeni

**Descrierea entităților:**

Entitățile prezentate în modelul de date sunt HOTEL, ANGAJATI, CAMERA, CLIENTUL, TARA, REZERVARE și sub entitățile care aparțin de entitatea ANGAJATI: RECEPTIONISTA, MENAJERA. În plus, există și tabelele asociative ESTE\_ATASATA și APARE\_IN\_ISTORICUL.

HOTEL = reprezintă clădirea cu camere/apartamente date spre închiriere pe o perioadă de timp, de obicei scurtă, destinate călătoriilor sau pentru afaceri. Aceasta este entitatea principală de care sunt legate entitățile importante ANGAJATI și CAMERE. Cheia primară este ***cod\_hotel***.

CAMERE = reprezintă spațiul pe care clienții îl închiriază prin rezervare. În plus, camerele au și informații despre numărul de locuri, prețul camerei etc. Cheia primară este ***cod\_camera***, iar cod\_hotel este FK.

ANGAJAT = persoana fizică, aparținând de hotel, care are două sub entități. Menajerele sunt persoanele fizice care se ocupă cu îngrijirea camerelor de hotel, iar recepționistele se ocupă cu preluarea rezervărilor. Cheia primara este ***cod\_angajat***, iar cod\_hotel este FK.

APARE\_IN\_ISTORICUL= reprezintă informații despre camerele de hotel, clientul care a stat în cameră și cât a plătit. Cheia primară este compusă și este formată din cele doua FK cod\_camera și cod\_client.

REZERVARE = este făcută de client și poate să conțină una sau mai multe camere. Cheia primară este ***cod\_rezervare***, iar cele trei FK sunt cod\_client, cod\_camera și cod\_angajat (specific unei receptioniste).

CLIENTUL = persoana fizică care apelează la serviciile hotelului, adică la închirierea de camere prin intermediul a una sau mai multor rezervări. Cheia primară este ***cod\_client***, iar FK este cod\_tara.

TARA = reprezintă locul din care clienții provin. Cheia primară este ***cod\_tara***.

**Descrierea relațiilor si prezentarea cardinalității:**

HOTELUL deține CAMERE = reflectă relația dinte entitatea principală Hotel și Camere, hotelul se bazează pe acestea, deci cardinalitatea minimă este de 1:1 (ca un hotel să funcționeze are nevoie de cel puțin o cameră) și cardinalitatea maximă este de 1:n (un hotel poate avea mai multe camere.

HOTELUL are ANGAJATI = Prezintă relația dinte entitatea principală Hotel și Angajați, aceștia sunt unici pentru fiecare hotel în parte și pot fi de 2 tipuri. Cardinalitatea minimă este 1:1 pentru că hotelul are nevoie de minim un angajat pentru a funcționa și cardinalitatea maximă 1:n pentru că avem nevoie de mai mulți angajați pentru a funcționa la capacitatea maximă.

REZERVAREA este pe CAMERA = Reprezintă faptul că o rezervare este legată în mod direct de CAMERA. Astfel, fiecare rezervare este unică și poate să fie pe o singură cameră, deci cardinalitatea minimă este egală cu cea maximă adică 1:1, pentru că o rezervare nu poate să conțină mai multe camere (nu ar merge legarea cu cheia externă).

CLIENTUL face REZERVARE = reflectă relația dintre entitatea CLIENT și REZERVARE. Clientul plasează o rezervare, acest lucru conducând la cardinalul minim de 1:1, un client ca să existe trebuie să fi făcut o rezervare, iar cardinalitatea maximă este de 1:n, un client poate sa facă mai multe rezervări.

CLIENTUL provine din o TARA = reflectă faptul că fiecare client aparține de o țară, asta făcând cardinalitatea maximă de 0:1, există țări din care nu provin clienți, iar cardinalitatea maximă este de n:1, o țară poate avea mai mulți clienți care provin din ea.

RECEPTIONISTELE preiau REZERVAREA = arată relația dintre sub-entitatea recepționiste, care aparține angajaților și entitatea rezervare. Astfel, o recepționistă poate să preia mai multe rezervări, cardinalitatea maximă fiind de 1:n, iar cardinalitatea minimă este 1:0, o recepționista poate să nu preia nici o rezervare.

MENAJERA este atașată de CAMERA = arată relația dintre sub-entitatea MENAJERA și entitatea cameră. Astfel, relația maximă este de n:n, mai multe menajere îngrijesc mai multe camere, iar cardinalitatea minimă este de 1:1, o menajeră poate îngriji doar o cameră.

**Descrierea atributelor, incluzând tipul de date și eventualele constrângeri, valori implicite, valori posibile ale atributelor:**

Entitatea independentă HOTEL are atributele:

* cod\_hotel = variabilă de tip întreg, de lungime maximă 4, care reprezintă codul unui hotel
* nume\_hotel = variabilă de tip caracter, de lungime maximă 40, care reprezintă numele hotelului, nu trebuie să fie null
* adresa = variabilă de tip caracter, de lungime maximă 60, care reprezintă adresa hotelului, nu trebuie să fie null și trebuie să fie unică

Entitatea independentă CAMERA are atributele:

* cod\_camera = variabilă de tip întreg de lungime maximă 4, care reprezintă codul/idul camerei
* numar\_camera = variabilă de tip întreg, de lungime maximă 4, este unică în fiecare hotel și reprezintă numărul camerei
* cod\_hotel = variabilă de tip întreg de lungime maximă 4, care reprezintă cheia externă din entitatea HOTEL. Atributul trebuie să corespundă la o valoare din entitatea HOTEL

Entitatea independentă REZERVARE are atributele:

* cod\_rezervare = variabilă de tip întreg de lungime maximă 5, care reprezintă codul rezervării
* data\_start = variabilă de tip dată calendaristică, care reprezintă data la care începe sejurul la hotel
* data\_final = variabilă de tip dată calendaristica, care reprezintă data la care se termină sejurul la hotel
* data\_rezervare = variabilă de tip dată calendaristica care reprezintă data când s-a făcut rezervarea
* cod\_client = variabilă de tip întreg de lungime maximă 4 care reprezintă codul clientului care se afla și în entitatea CLIENTUL. Atributul trebuie să corespundă la o valoare din entitatea CLIENTUL.
* cod\_camera = variabilă de tip întreg de lungime maximă 4 care reprezintă codul camerei care se afla și în entitatea CAMERA. Atributul trebuie să corespundă la o valoare din entitatea CAMERA.

Entitatea independentă CLIENTUL are atributele:

* cod\_client = variabilă de tip întreg de lungime maximă 4, care reprezintă codul clientului
* nume = variabilă de tip caracter de lungime maximă 20, care reprezintă numele clientului
* prenume = variabilă de tip caracter de lungime maximă 30, care reprezintă prenumele clientului
* email = variabilă de tip caracter de lungime maximă 40, care poate să fie nulă și reprezintă emailul clientului
* telefon = variabilă de tip caracter care are lungimea maximă de 10 și nu poate să fie nulă, reprezentând telefonul clientului
* cod\_tara = variabilă de tip întreg de lungime maximă 3, care reprezintă codul țării din care provine clintul. Atributul trebuie să corespundă la o valoare din entitatea TARA

Entitatea independentă TARA are atributele:

* cod\_tara = variabilă de tip întreg de lungime maximă 3, care reprezintă codul țării
* nume\_tara = variabilă de tip caracter de lungime maximă 20, care reprezintă numele țării
* regiune\_tara = variabilă de tip caracter de lungime maximă 20, care reprezintă continentul de unde este țara (Europa, America de Nord, Africa ș.a.)

Entitatea asociativă APARE\_IN\_ISTORICUL are atributele:

* cod\_camera= variabilă de tip întreg de lungime maximă 4 care reprezintă codul camerei care se află și în entitatea CAMERA. Atributul trebuie să corespundă la o valoare din entitatea CAMERA.
* cod\_client = variabilă de tip întreg de lungime maximă 4 care reprezintă codul clientului care se afla și în entitatea CLIENT. Atributul trebuie să corespundă la o valoare din entitatea CLIENT.
* Pret = variabilă de tip întreg de lungime maximă 6 care reprezintă cât a plătit clientul pentru cameră

Entitatea independentă ANGAJATI are atributele:

* cod\_angajat = variabilă de tip întreg de lungime maximă 4, care reprezintă codul angajatului
* cod\_hotel = variabilă de tip întreg de lungime maximă 4, care reprezintă cheia externă din entitatea HOTEL. Atributul trebuie să corespundă la o valoare din entitatea HOTEL
* numeangajat = variabilă de tip caracter de lungime 20, care reprezintă numele angajatului
* prenumeangajat = variabilă de tip caracter de lungime 20, care reprezintă numele angajatului
* salariu\_lunar = variabilă de tip întreg de lungime 7, care reprezintă salariul lunar al angajatului
* tipul\_angajatului = variabilă de tip caracter de lungime maximă 20, care poate să fie receptionist sau menajeră

Sub-entitatea RECEPTIONISTA care face parte din ANGAJATI:

* comision\_clientreceptionista = variabilă de tip întreg de lungime maximă 2, care reprezintă comisionul pe care recepționista îl primește pentru o rezervare finalizată

Sub-entitatea MENAJERA care face parte din ANGAJATI:

* firmamenaj = variabilă de tip caracter de lungime maximă 20, care reprezintă firma de la care provine menajera

Entitatea asociativa ESTE\_ATASATA care are atributele:

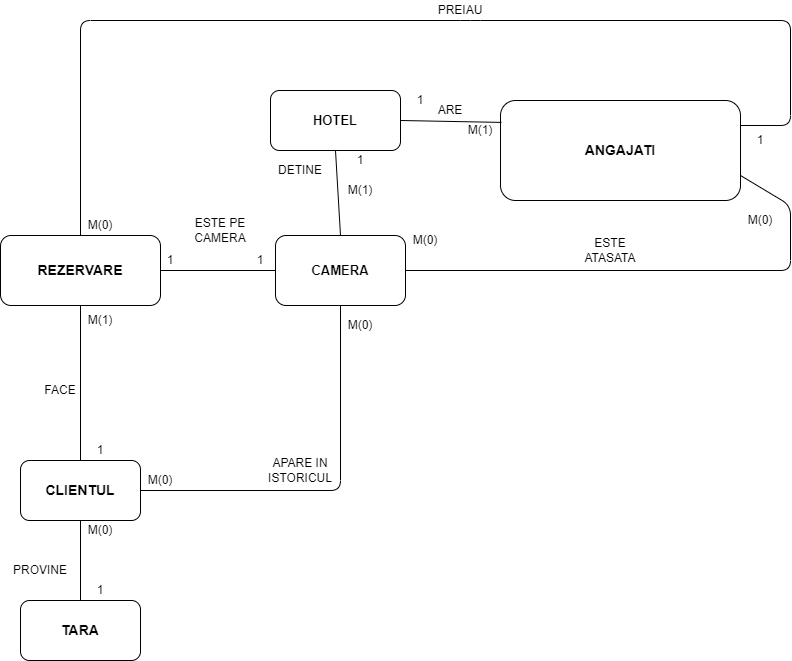
* cod\_angajat = codul exclusiv al unei MENAJERE, variabilă de tip întreg de lungime maximă 4, care reprezintă codul angajatului și este cheie externă din entitatea ANGAJATI
* cod\_camera = variabilă de tip întreg de lungime maximă 4 care reprezintă codul camerei care se afla și în entitatea CAMERA. Atributul trebuie să corespundă la o valoare din entitatea CAMERA.
* numărdeoreîngrijire = variabilă de tip întreg de lungime maximă 2 care reprezintă numărul de ore în care o menajeră îngrijește o cameră pe zi

**Capitolul 2**

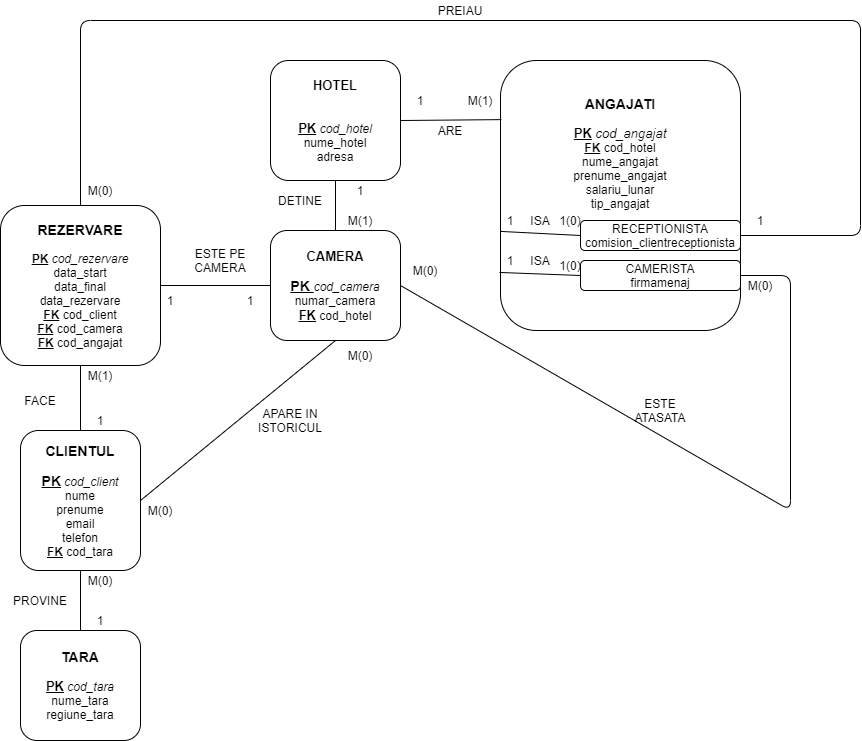
**Realizați diagrama entitate-relație (ERD)**

**(Cerința 2)**

**Diagrama ERD fără atribute:**



**Diagrama ERD cu atribute:**

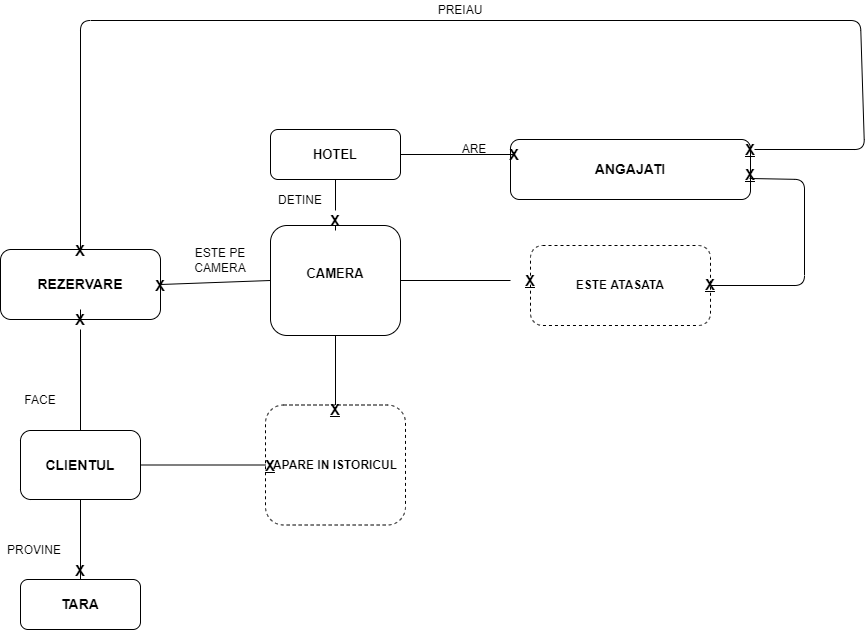
****

**Capitolul 3**

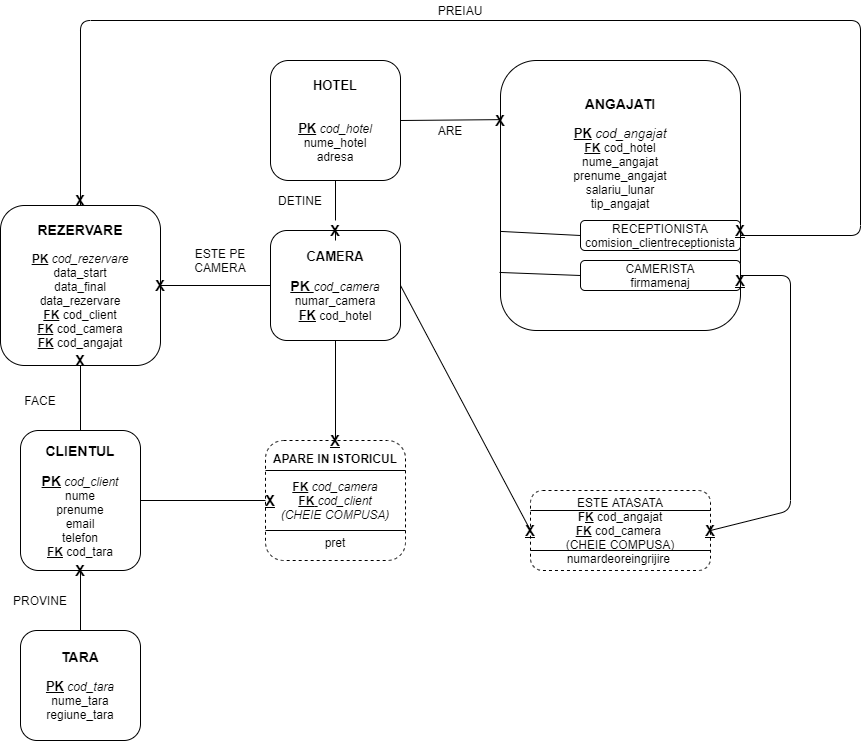
**Pornind de la diagrama entitate-relație realizați diagrama conceptuală a modelului propus, integrând toate atributele necesare**

**(Cerința 3)**

**Diagrama conceptuală fără atribute:**

****

**Diagrama conceptuală cu atribute:**

****

**Enumerarea schemelor relaționale corespunzătoare diagramei conceptuale:**

**HOTEL(**cod\_hotel#, nume\_hotel, adresa**)**

**REZERVARE(**cod\_rezervare#, data\_start, data\_final, data\_rezervare, cod\_client#, cod\_camera#, cod\_angajat#**)**

**CAMERA(**cod\_camera#, numar\_camera, cod\_hotel#**)**

**CLIENTUL(**cod\_client#, nume, prenume, email, telefon, cod\_tara#**)**

**TARA(**cod\_tara#, nume\_tara, regiune\_tara**)**

**APARE\_IN\_ISTORICUL(**cod\_camera#, cod\_client#, pret**)**

**ANGAJATI(**cod\_angajat#, cod\_hotel#, nume\_angajat,prenume\_angajat,salariu\_lunar, tip\_angajat, comicion\_clientreceptionista, firmamenaj**)**

**ESTE\_ATASATA(**cod\_angajat#, cod\_camera#, numardeoreingrijire**)**

**Capitolul 4**

**Crearea tabelelor în SQL și inserarea de date coerente în fiecare dintre acestea**

**(Cerința 4 si 5)**

**Implementați în Oracle diagrama conceptuală realizată:**

**Entitatea Hotel:**

CREATE TABLE HOTEL(

cod\_hotel number(4) CONSTRAINT pkey1\_h primary key,

nume\_hotel varchar2(40) not null,

adresa varchar2(60) not null UNIQUE

);

**Entitatea Camera:**

CREATE TABLE CAMERA(

cod\_camera number(4) constraint pkey2\_C PRIMARY key,

numar\_camera number(4) not null,

cod\_hotel number(4),

CONSTRAINT fk\_h FOREIGN KEY(cod\_hotel) REFERENCES HOTEL(cod\_hotel)

);

**Entitatea Tara:**

CREATE TABLE TARA(

cod\_tara number(3) constraint pkey5\_t primary key,

nume\_tara varchar2(20) not null,

regiune\_tara varchar2(20) not null

);

**Entitatea Clientul:**

CREATE TABLE CLIENTUL(

cod\_client number(4) CONSTRAINT pkey4\_client PRIMARY KEY,

nume varchar2(20) not null,

prenume varchar2(30) not null,

email varchar2(40),

telefon varchar2(10) not null,

cod\_tara number(3) not null,

CONSTRAINT fk\_tara FOREIGN KEY(cod\_tara) REFERENCES TARA(cod\_tara)

);

**Entitatea Angajati:**

CREATE TABLE ANGAJATI(

cod\_angajat number(4) constraint pkey\_ang PRIMARY KEY,

cod\_hotel number(4),

nume\_angajat varchar2(20) not null,

prenume\_angajat varchar2(20) not null,

salariu\_lunar NUMBER(7) not null,

tip\_angajat VARCHAR2(20),

comision\_clientreceptionista number(2),

firmamenaj varchar2(20),

CONSTRAINT fk\_h2 FOREIGN KEY(cod\_hotel) REFERENCES HOTEL(cod\_hotel)

);

**Entitatea Rezervare:**

CREATE TABLE REZERVARE(

cod\_rezervare number(5) constraint pkey3\_R PRIMARY KEY,

data\_start date,

data\_final date,

data\_rezervare date,

cod\_client number(4),

cod\_camera number(4),

cod\_angajat number(4),

CONSTRAINT fk\_client1 FOREIGN KEY(cod\_client) REFERENCES CLIENTUL(cod\_client),

CONSTRAINT fk\_camera1 FOREIGN KEY(cod\_camera) REFERENCES CAMERA(cod\_camera),

CONSTRAINT fk\_receptionista1 FOREIGN KEY(cod\_angajat) REFERENCES ANGAJATI(cod\_angajat)

); --angajat

**Entitatea Apare\_In\_Istoricul:**

CREATE TABLE APARE\_IN\_ISTORICUL(

cod\_camera number(4),

CONSTRAINT fk\_camera2istcam FOREIGN KEY(cod\_camera) REFERENCES CAMERA(cod\_camera),

cod\_client number(4),

CONSTRAINT fk\_client1istcam FOREIGN KEY(cod\_client) REFERENCES CLIENTUL(cod\_client),

pret number(6) not null,

CONSTRAINT pk\_compusistcam PRIMARY KEY(cod\_camera, cod\_client)

);

**Entitatea Este\_Atasata:**

CREATE TABLE ESTE\_ATASATA(

cod\_angajat number(4),

cod\_camera number(4),

numardeoreingrijire number(2),

CONSTRAINT fk\_camera3 FOREIGN KEY(cod\_camera) REFERENCES CAMERA(cod\_camera),

CONSTRAINT fk\_angajat1 FOREIGN KEY(cod\_angajat) REFERENCES ANGAJATI(cod\_angajat),

CONSTRAINT pk\_compus3 PRIMARY KEY(cod\_camera, cod\_angajat)

);

**Inserarea în tabel și rezultatul inserării:**

**Entitatea Hotel:**

Secvența folosită:

CREATE SEQUENCE secvhotel

INCREMENT BY 3

START WITH 5

MAXVALUE 1000

NOCYCLE;

Inserare:

INSERT INTO HOTEL

VALUES(secvhotel.nextval,'Ramada Park Hotel','Romania, Bucuresti, Sec.1, Poligrafiei Avenue 3');

INSERT INTO HOTEL

VALUES(SECVHOTEL.nextval,'Rin Grand Hotel','Romania, Bucuresti, Sec.4, Str. Vitan-Barzesti 7');

INSERT INTO HOTEL

VALUES(SECVHOTEL.nextval,'Club Hotel Falcon','Turcia, Antalya, Sirinyali Mah. 1512');

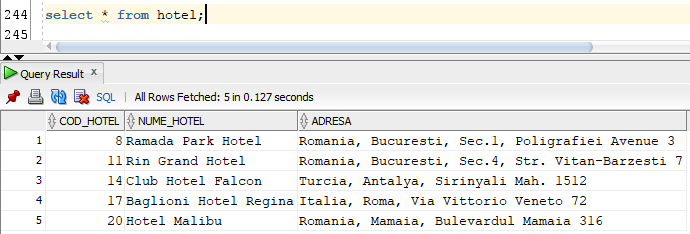
INSERT INTO HOTEL

VALUES(SECVHOTEL.nextval,'Baglioni Hotel Regina','Italia, Roma, Via Vittorio Veneto 72');

INSERT INTO HOTEL

VALUES(SECVHOTEL.nextval,'Hotel Malibu','Romania, Mamaia, Bulevardul Mamaia 316');

Rezultat:



**Entitatea Camera:**

Secvența folosită:

CREATE SEQUENCE secvcamera

INCREMENT BY 5

START WITH 10

MAXVALUE 1000

NOCYCLE;

Inserare:

Insert into camera

Values(secvcamera.nextval,100,8);

Insert into camera

Values(secvcamera.nextval,105,8);

Insert into camera

Values(secvcamera.nextval,110,8);

Insert into camera

Values(secvcamera.nextval,111,8);

Insert into camera

Values(secvcamera.nextval,112,8);

Insert into camera

Values(secvcamera.nextval,100,11);

Insert into camera

Values(secvcamera.nextval,122,11);

Insert into camera

Values(secvcamera.nextval,120,11);

Insert into camera

Values(secvcamera.nextval,124,11);

Insert into camera

Values(secvcamera.nextval,120,14);

Insert into camera

Values(secvcamera.nextval,200,14);

Insert into camera

Values(secvcamera.nextval,201,14);

Insert into camera

Values(secvcamera.nextval,202,14);

Insert into camera

Values(secvcamera.nextval,300,17);

Insert into camera

Values(secvcamera.nextval,200,17);

Insert into camera

Values(secvcamera.nextval,350,17);

Insert into camera

Values(secvcamera.nextval,355,17);

Insert into camera

Values(secvcamera.nextval,300,20);

Insert into camera

Values(secvcamera.nextval,400,20);

Insert into camera

Values(secvcamera.nextval,440,20);

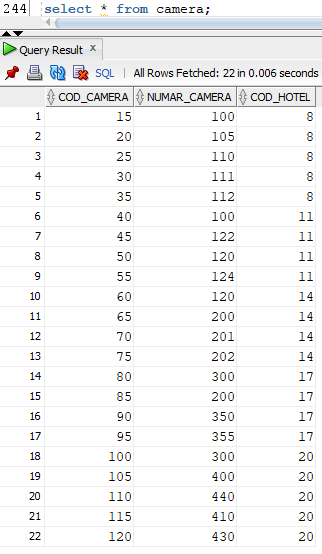
Insert into camera

Values(secvcamera.nextval,410,20);

Insert into camera

Values(secvcamera.nextval,430,20);

Rezultat:



**Entitatea Tara:**

Secvența folosită:

CREATE SEQUENCE secvtara

INCREMENT BY 2

START WITH 1

MAXVALUE 100

NOCYCLE;

Inserare:

Insert into tara

Values(secvtara.nextval,'Romania','Europa');

Insert into tara

Values(secvtara.nextval,'Italia','Europa');

Insert into tara

Values(secvtara.nextval,'Spania','Europa');

Insert into tara

Values(secvtara.nextval,'Germania','Europa');

Insert into tara

Values(secvtara.nextval,'Uganda','Africa');

Insert into tara

Values(secvtara.nextval,'Nigeria','Africa');

Insert into tara

Values(secvtara.nextval,'SUA','America de Nord');

Insert into tara

Values(secvtara.nextval,'Canada','America de Nord');

Insert into tara

Values(secvtara.nextval,'Columbia','America de Sud');

Insert into tara

Values(secvtara.nextval,'Argentina','America de Sud');

Insert into tara

Values(secvtara.nextval,'China','Asia');

Insert into tara

Values(secvtara.nextval,'Japonia','Asia');

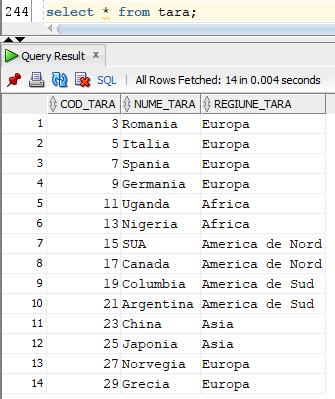
Insert into tara

Values(secvtara.nextval,'Norvegia','Europa');

Insert into tara

Values(secvtara.nextval,'Grecia','Europa');

Rezultat:



**Entitatea Clientul:**

Secvența folosită:

CREATE SEQUENCE secclient

INCREMENT BY 3

START WITH 50

MAXVALUE 1000

NOCYCLE;

Inserare:

Insert into clientul

Values(secclient.nextval,'Cristi','Gigi','cristi@gmail.ro','07717171',3);

Insert into clientul

Values(secclient.nextval,'Popa','Dorian','dorian11@gmail.com','07717333',3);

Insert into clientul

Values(secclient.nextval,'Giorno','Giovanna','GiGiovanna@jojo.it','07734223',5);

Insert into clientul

Values(secclient.nextval,'Ramos','Sergio','s.ramos@RLMAD.sp','044123536',7);

Insert into clientul

Values(secclient.nextval,'Polo','Marco',NULL,'0752223',5);

Insert into clientul

Values(secclient.nextval,'Veratii','Mario','Vertatmari2@hermes.it','07034221',5);

Insert into clientul

Values(secclient.nextval,'Tartini','Giuseppe',NULL,'09934211',5);

Insert into clientul

Values(secclient.nextval,'Olushola','Ajose',NULL,'0778223',13);

Insert into clientul

Values(secclient.nextval,'Drakos','Papadopoulos','Drakos22@gmail.com','07780253',29);

Insert into clientul

Values(secclient.nextval,'Demetriou','Georgiou',NULL,'077802901',29);

Insert into clientul

Values(secclient.nextval,'Alma','Lionel','almlin@yahoo.com','071102901',21);

Insert into clientul

Values(secclient.nextval,'Yuji','Itadori','Yuri123@juju.jp','0041412',25);

Insert into clientul

Values(secclient.nextval,'Hill','Jenna','jennaH2@yahoo.com','0700412',15);

Insert into clientul

Values(secclient.nextval,'Jones','Nick','Jnick@gmail.com','011212',15);

Insert into clientul

Values(secclient.nextval,'Popescu','Marian','Popmar@yahoo.com','0717333',3);

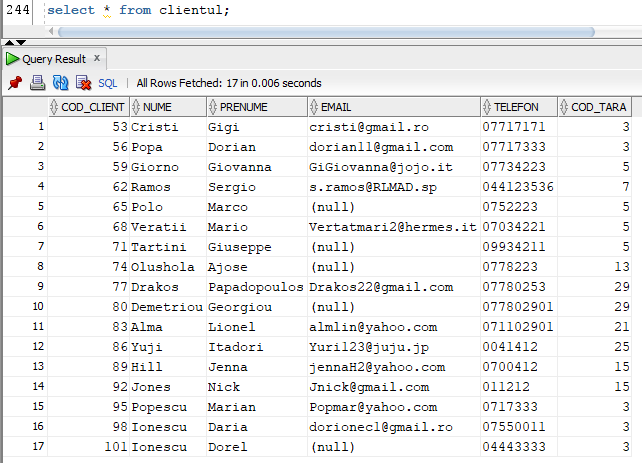
Insert into clientul

Values(secclient.nextval,'Ionescu','Daria','dorionec1@gmail.ro','07550011',3);

Insert into clientul

Values(secclient.nextval,'Ionescu','Dorel',NULL,'04443333',3);

Rezultat:



**Entitatea Angajati:**

Secvența folosită:

CREATE SEQUENCE secang

INCREMENT BY 4

START WITH 20

MAXVALUE 1000

NOCYCLE;

Inserare:

Insert into angajati

values(secang.nextval,8,'Soare','Tilea',2500,'Camerista',NULL,'MenajSRL');

Insert into angajati

values(secang.nextval,8,'Saftoiu','Iulian',3000,'Receptionista',2,NULL);

Insert into angajati

values(secang.nextval,8,'Boroi','Aurelia',2700,'Camerista',NULL,'AuPair');

Insert into angajati

values(secang.nextval,8,'Gogeanu','Lavinia',4000,'Receptionista',4,NULL);

Insert into angajati

values(secang.nextval,11,'Vasilescu','Ana',3300,'Camerista',NULL,'EutochemSRL');

Insert into angajati

values(secang.nextval,11,'Toma','Marcel',3100,'Camerista',NULL,'MenajSRL');

Insert into angajati

values(secang.nextval,11,'Cojoc','Florentin',4500,'Receptionista',3,NULL);

Insert into angajati

values(secang.nextval,11,'Iordanescu','Alexandru',4000,'Receptionista',2,NULL);

Insert into angajati

values(secang.nextval,14,'Mircea','Rodica',2200,'Camerista',NULL,'EutochemSRL');

Insert into angajati

values(secang.nextval,14,'Toma','Ion',5000,'Camerista',NULL,'AuPair');

Insert into angajati

values(secang.nextval,14,'Ciobanu','Sorina',4200,'Receptionista',2,NULL);

Insert into angajati

values(secang.nextval,14,'Lazar','Mihnea',3600,'Receptionista',2,NULL);

Insert into angajati

values(secang.nextval,17,'Pirvu','Felix',2530,'Camerista',NULL,'MenajSRL');

Insert into angajati

values(secang.nextval,17,'Giurgiu','Elena',2000,'Camerista',NULL,'AuPair');

Insert into angajati

values(secang.nextval,17,'Zamfir','Mihai',5000,'Receptionista',5,NULL);

Insert into angajati

values(secang.nextval,17,'Ungureanu','Ilinca',2234,'Receptionista',3,NULL);

Insert into angajati

values(secang.nextval,20,'Ionescu','Florenta',3200,'Camerista',NULL,'AuPair');

Insert into angajati

values(secang.nextval,20,'Apostol','Catalina',4100,'Camerista',NULL,'AuPair');

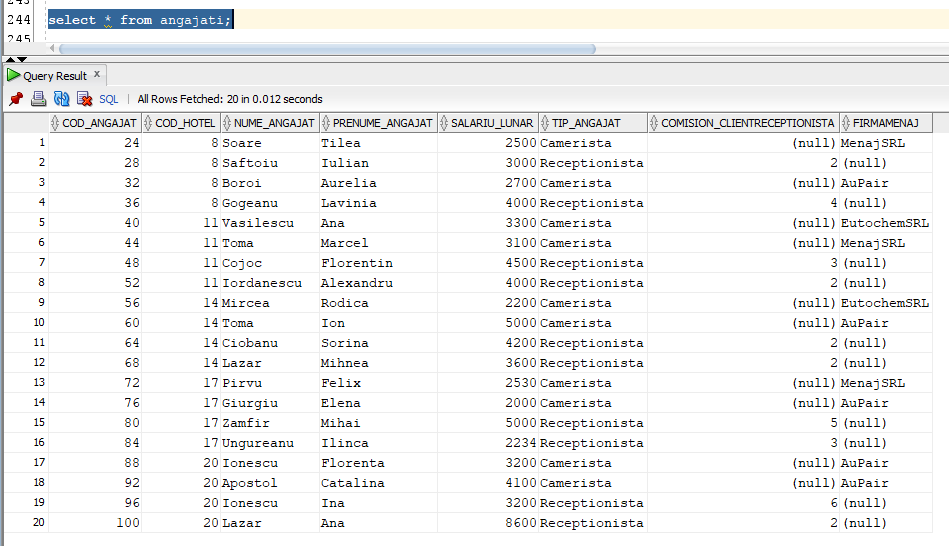
Insert into angajati

values(secang.nextval,20,'Ionescu','Ina',3200,'Receptionista',6,NULL);

Insert into angajati

values(secang.nextval,20,'Lazar','Ana',8600,'Receptionista',2,NULL);

Rezultat:



**Entitatea Rezervare:**

Secvența folosită:

CREATE SEQUENCE secrez

INCREMENT BY 8

START WITH 17

MAXVALUE 10000

NOCYCLE;

Inserare: Insert into rezervare

values(secrez.nextval,to\_date('10-07-2021','dd-mm-yyyy'),to\_date('17-07-2021','dd-mm-yyyy'),sysdate,53, 100,96);

Insert into rezervare

values(secrez.nextval,to\_date('29-08-2021','dd-mm-yyyy'),to\_date('07-09-2021','dd-mm-yyyy'),to\_date('10-04-2021','dd-mm-yyyy'),56, 20,36);

Insert into rezervare

values(secrez.nextval,to\_date('22-08-2021','dd-mm-yyyy'),to\_date('10-09-2021','dd-mm-yyyy'),to\_date('01-04-2021','dd-mm-yyyy'),59, 30,28);

Insert into rezervare

values(secrez.nextval,to\_date('21-06-2021','dd-mm-yyyy'),to\_date('28-06-2021','dd-mm-yyyy'),to\_date('02-05-2021','dd-mm-yyyy'),62, 40,48);

Insert into rezervare

values(secrez.nextval,to\_date('01-09-2021','dd-mm-yyyy'),to\_date('11-09-2021','dd-mm-yyyy'),to\_date('03-04-2021','dd-mm-yyyy'),65, 45,48);

Insert into rezervare

values(secrez.nextval,to\_date('22-09-2021','dd-mm-yyyy'),to\_date('01-10-2021','dd-mm-yyyy'),to\_date('01-03-2021','dd-mm-yyyy'),68, 70,64);

Insert into rezervare

values(secrez.nextval,to\_date('22-09-2021','dd-mm-yyyy'),to\_date('01-10-2021','dd-mm-yyyy'),to\_date('01-03-2021','dd-mm-yyyy'),71, 75,68);

Insert into rezervare

values(secrez.nextval,to\_date('19-08-2021','dd-mm-yyyy'),to\_date('02-09-2021','dd-mm-yyyy'),to\_date('10-04-2021','dd-mm-yyyy'),74, 100,100);

Insert into rezervare

values(secrez.nextval,to\_date('22-08-2021','dd-mm-yyyy'),to\_date('10-08-2021','dd-mm-yyyy'),to\_date('02-04-2021','dd-mm-yyyy'),77, 105,100);

Insert into rezervare

values(secrez.nextval,to\_date('21-10-2021','dd-mm-yyyy'),to\_date('08-11-2021','dd-mm-yyyy'),to\_date('02-02-2021','dd-mm-yyyy'),80, 110,96);

Insert into rezervare

values(secrez.nextval,to\_date('01-10-2021','dd-mm-yyyy'),to\_date('09-10-2021','dd-mm-yyyy'),to\_date('22-04-2021','dd-mm-yyyy'),83, 80,80);

Insert into rezervare

values(secrez.nextval,to\_date('22-09-2021','dd-mm-yyyy'),to\_date('01-10-2021','dd-mm-yyyy'),to\_date('27-03-2021','dd-mm-yyyy'),86, 85,84);

Insert into rezervare

values(secrez.nextval,to\_date('22-11-2021','dd-mm-yyyy'),to\_date('02-12-2021','dd-mm-yyyy'),to\_date('05-03-2021','dd-mm-yyyy'),89, 90,84);

Insert into rezervare

values(secrez.nextval,to\_date('28-12-2021','dd-mm-yyyy'),to\_date('03-01-2022','dd-mm-yyyy'),to\_date('01-02-2021','dd-mm-yyyy'),92, 15,28);

Insert into rezervare

values(secrez.nextval,to\_date('17-07-2021','dd-mm-yyyy'),to\_date('27-07-2021','dd-mm-yyyy'),to\_date('17-05-2021','dd-mm-yyyy'),95, 25,36);

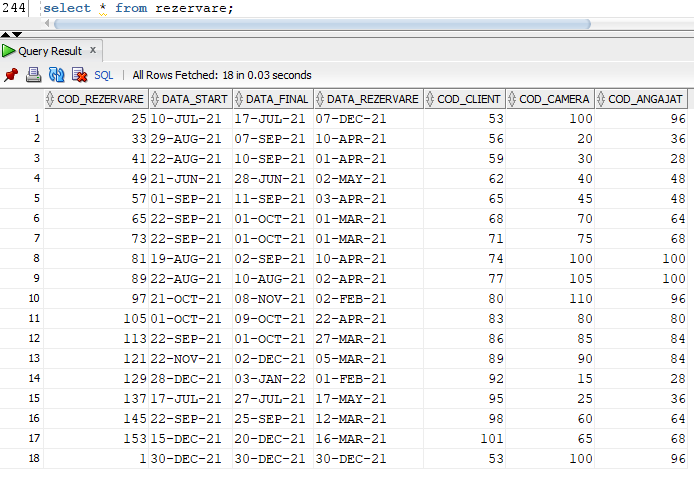
Insert into rezervare

values(secrez.nextval,to\_date('22-09-2021','dd-mm-yyyy'),to\_date('25-09-2021','dd-mm-yyyy'),to\_date('12-03-2021','dd-mm-yyyy'),98, 60,64);

Insert into rezervare

values(secrez.nextval,to\_date('15-12-2021','dd-mm-yyyy'),to\_date('20-12-2021','dd-mm-yyyy'),to\_date('16-03-2021','dd-mm-yyyy'),101, 65,68);

Rezultat:



**Entitatea Apare\_In\_Istoricul:**

Inserare:

insert into APARE\_IN\_ISTORICUL

VALUES(15,53,5500);

insert into APARE\_IN\_ISTORICUL

VALUES(15,101,2500);

insert into APARE\_IN\_ISTORICUL

VALUES(20,56,6700);

insert into APARE\_IN\_ISTORICUL

VALUES(20,98,4400);

insert into APARE\_IN\_ISTORICUL

VALUES(25,59,5700);

insert into APARE\_IN\_ISTORICUL

VALUES(30,65,4000);

insert into APARE\_IN\_ISTORICUL

VALUES(35,77,7700);

insert into APARE\_IN\_ISTORICUL

VALUES(40,83,5500);

insert into APARE\_IN\_ISTORICUL

VALUES(45,86,2200);

insert into APARE\_IN\_ISTORICUL

VALUES(50,86,8800);

insert into APARE\_IN\_ISTORICUL

VALUES(55,89,9100);

insert into APARE\_IN\_ISTORICUL

VALUES(60,92,5200);

insert into APARE\_IN\_ISTORICUL

VALUES(65,95,3200);

insert into APARE\_IN\_ISTORICUL

VALUES(70,53,7600);

insert into APARE\_IN\_ISTORICUL

VALUES(75,86,2200);

insert into APARE\_IN\_ISTORICUL

VALUES(80,56,8800);

insert into APARE\_IN\_ISTORICUL

VALUES(85,89,9100);

insert into APARE\_IN\_ISTORICUL

VALUES(90,71,5200);

insert into APARE\_IN\_ISTORICUL

VALUES(95,68,3200);

insert into APARE\_IN\_ISTORICUL

VALUES(100,86,7200);

insert into APARE\_IN\_ISTORICUL

VALUES(105,89,10000);

insert into APARE\_IN\_ISTORICUL

VALUES(110,56,9000);

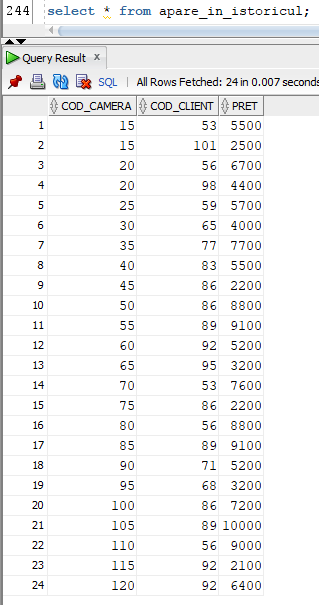
insert into APARE\_IN\_ISTORICUL

VALUES(115,92,2100);

insert into APARE\_IN\_ISTORICUL

VALUES(120,92,6400);

Rezultat:



**Entitatea Este\_Atasata:**

Inserare:

insert into este\_atasata

values(24,15,2);

insert into este\_atasata

values(32,15,1);

insert into este\_atasata

values(24,20,1);

insert into este\_atasata

values(24,25,3);

insert into este\_atasata

values(32,30,1);

insert into este\_atasata

values(24,30,3);

insert into este\_atasata

values(24,35,2);

insert into este\_atasata

values(32,35,1);

insert into este\_atasata

values(44,40,1);

insert into este\_atasata

values(40,40,1);

insert into este\_atasata

values(40,45,2);

insert into este\_atasata

values(44,50,1);

insert into este\_atasata

values(40,50,3);

insert into este\_atasata

values(44,55,1);

insert into este\_atasata

values(40,55,2);

insert into este\_atasata

values(56,60,1);

insert into este\_atasata

values(56,65,1);

insert into este\_atasata

values(60,70,2);

insert into este\_atasata

values(60,75,2);

insert into este\_atasata

values(56,75,1);

insert into este\_atasata

values(76,80,1);

insert into este\_atasata

values(72,80,1);

insert into este\_atasata

values(76,85,2);

insert into este\_atasata

values(76,90,2);

insert into este\_atasata

values(72,95,1);

insert into este\_atasata

values(88,100,1);

insert into este\_atasata

values(88,105,1);

insert into este\_atasata

values(88,110,2);

insert into este\_atasata

values(88,115,2);

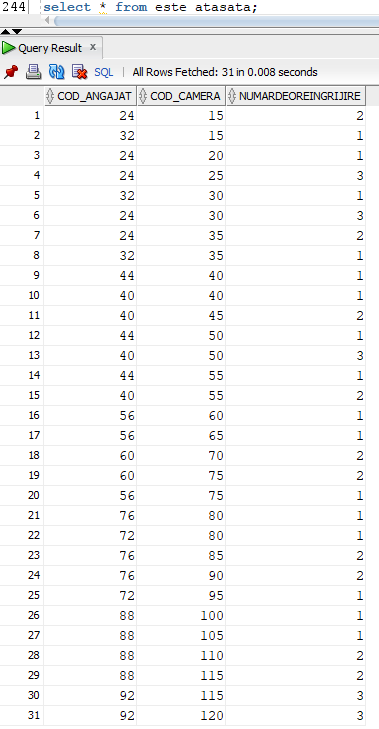
insert into este\_atasata

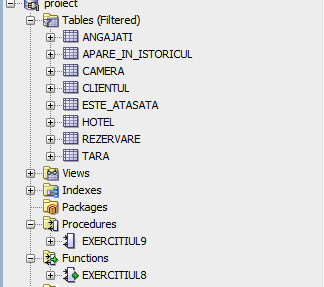
values(92,115,3);

insert into este\_atasata

values(92,120,3);

Rezultat:



****

**Capitolul 5**

**Exercițiile 6, 7, 8 si 9**

**(Cerința 6, 7, 8 si 9)**

* Formulați în limbaj natural o problemă pe care să o rezolvați folosind un **subprogram stocat** care să utilizeze două tipuri de colecție studiate. Apelați subprogramul. (6)

Enunț:

Sa se afișeze pentru fiecare client din entitatea CLIENTUL numele , prenumele, in cate camere a stat pana acum si din ce hotel au fost camerele, daca acesta nu a stat in nici o camera pana acum atunci in loc sa se afișeze hotelurile in care a stat sa apară scris ‚ Clientul X Y nu a stat la nici un hotel’.

Rezolvare:

CREATE OR REPLACE PROCEDURE exercitiul6

is

type t\_imbricat is table of number(3);

type t\_indexat is table of number(3) index by PLS\_INTEGER;

id\_camera t\_imbricat:=t\_imbricat();

id\_client t\_indexat;

nume\_client clientul.nume%type;

prenume\_client clientul.prenume%type;

id\_h hotel.cod\_hotel%type;

nume\_h hotel.nume\_hotel%type;

n\_rez number(3);

tara\_c number(3);

BEGIN

select cod\_client BULK collect into id\_client from clientul;

for i in id\_client.first..id\_client.last loop

SELECT cod\_camera BULK collect into id\_camera

from apare\_in\_istoricul

where id\_client(i) = cod\_client;

select nume,prenume into nume\_client,prenume\_client

from clientul

where id\_client(i) = cod\_client;

dbms\_output.put\_line('Clientul '||nume\_client||' '|| prenume\_client||' a stat pana acum in '||id\_camera.count()||' camere');

if id\_camera.count() !=0 then

for a in id\_camera.first..id\_camera.last loop

select cod\_hotel into id\_h from camera

where id\_camera(a) = cod\_camera;

select nume\_hotel into nume\_h FROM hotel

where cod\_hotel = id\_h;

dbms\_output.put\_line('Camera din hotelul: '||nume\_h);

end loop;

else dbms\_output.put\_line('Clientul '||nume\_client||' '|| prenume\_client||' nu a stat la nici un hotel ');

end if;

id\_camera.delete;

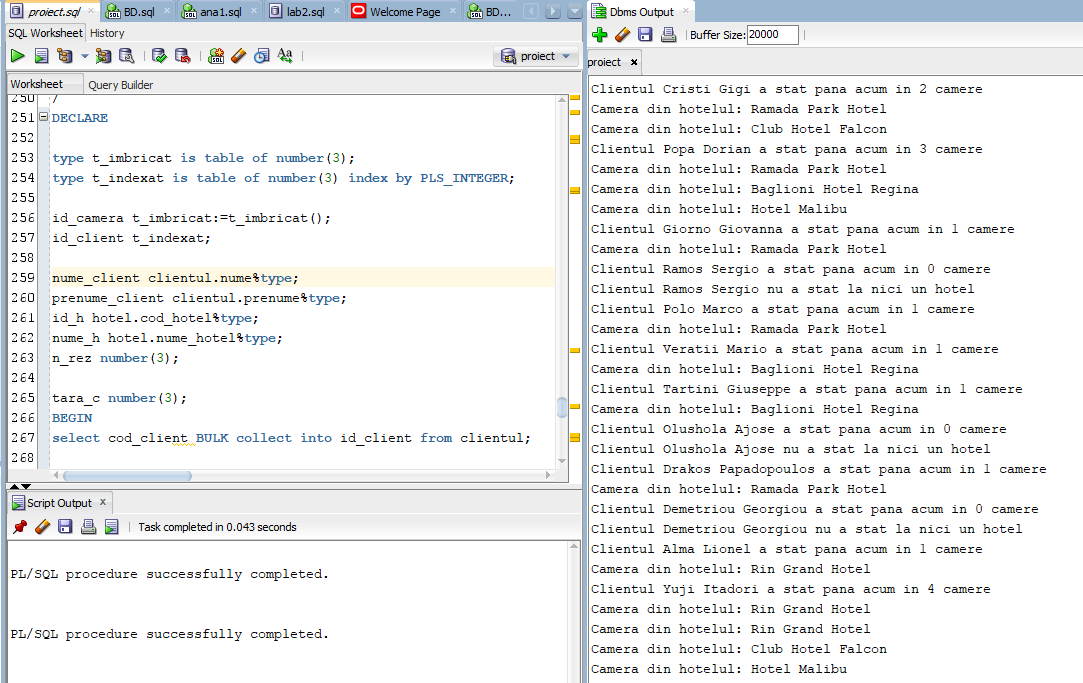
end loop;

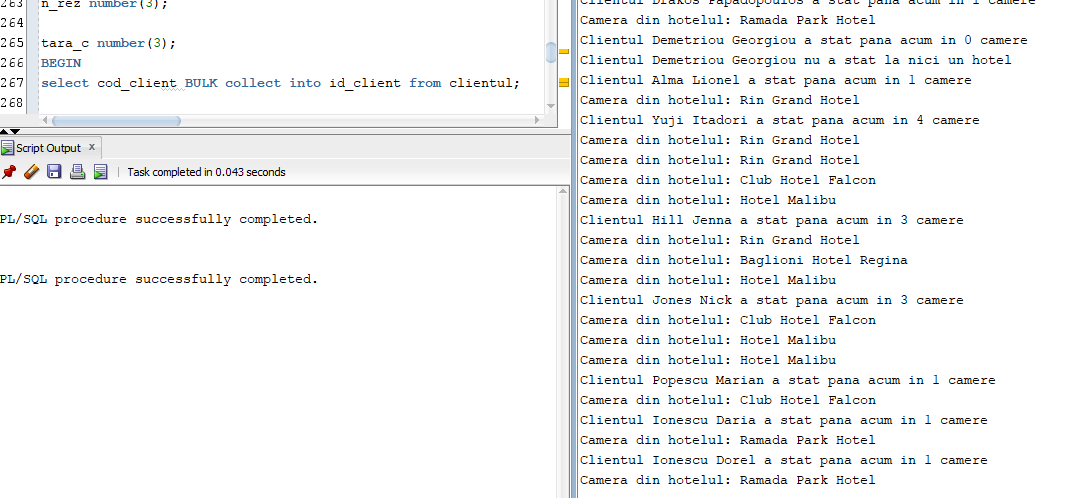
End;

/

Rezultat:

execute exercitiul6;





* Formulați în limbaj natural o problemă pe care să o rezolvați folosind un **subprogram stocat** care să utilizeze un tip de cursor studiat. Apelați subprogramul. (7)

Enunț:

Sa se afișeze id-ul camerei, numărul camerei si numele hotelului de care aparține camera pentru toate camerele care aparțin de hotele cu cod par.

Rezolvare:

CREATE OR REPLACE PROCEDURE exercitiul7

is

TYPE tab\_nr IS TABLE OF NUMBER(4);

n\_hotel hotel.nume\_hotel%type;

t\_id tab\_nr;

t\_nr tab\_nr;

t\_ch tab\_nr;

cursor c\_camera is (select cod\_camera,numar\_camera,cod\_hotel from camera where cod\_hotel in

(select cod\_hotel from hotel

where mod(cod\_hotel,2)=0));

BEGIN

open c\_camera;

fetch c\_camera bulk collect into t\_id,t\_nr,t\_ch;

close c\_camera;

for i in t\_id.first..t\_id.last loop

select nume\_hotel into n\_hotel from hotel where cod\_hotel = t\_ch(i);

dbms\_output.put\_line('Camera cu id-ul '||t\_id(i)||' are numarul '||t\_nr(i)||' si apartine hotelului: '||n\_hotel);

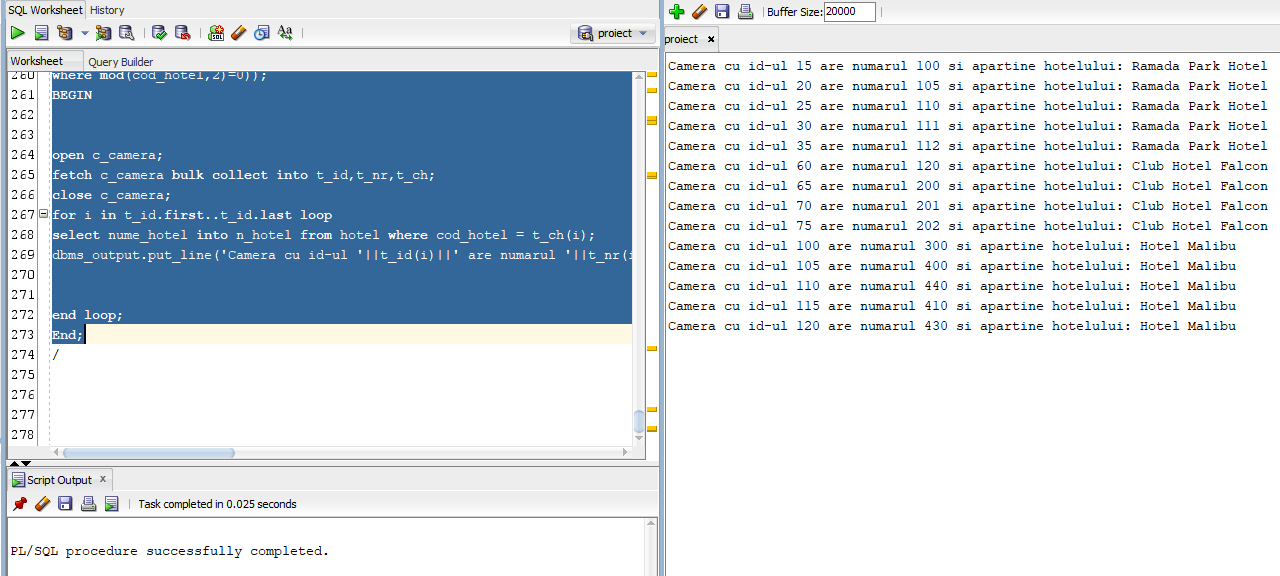
end loop;

End;

/

Rezultat:

execute exercitiul7;



* Formulați în limbaj natural o problemă pe care să o rezolvați folosind un **subprogram stocat de tip funcție** care să utilizeze într-o singură comandă SQL 3 dintre tabelele definite. Tratați toate excepțiile care pot apărea. Apelați subprogramul astfel încât să evidențiați toate cazurile tratate. (8)

Enunț:

Sa se afișeze, pentru o tara data, daca au fost făcute mai multe rezervări după, înainte sau același număr pentru data 02-04-2021. Daca nu exista rezervări acest lucru se va afișa pe ecran.

Rezolvare:

create or replace function exercitiul8

(n\_tara in tara.nume\_tara%type)

return varchar2

is

TYPE tab\_nr IS TABLE OF varchar2(90);

t\_rez tab\_nr;

contor1 number(4);

contor2 number(4);

begin

contor1:=0;

contor2:=0;

select case when r.data\_rezervare >to\_date('05-04-2021', 'DD-MM-YYYY') then 'rezervarea a fost facuta dupa 02-04-2021' else 'rezervarea a fost facuta inainte de 02-04-2021' end bulk collect into t\_rez

from rezervare r

WHERE r.cod\_client in (select c.cod\_client from clientul c

where c.cod\_tara in (select t.cod\_tara from tara t

where t.cod\_tara=c.cod\_tara and upper(nume\_tara)=upper(n\_tara)));

if t\_rez.count= 0 then return 'Nu sunt rezervari din tara data';

else

for i in t\_rez.first..t\_rez.last loop

if t\_rez(i) = 'rezervarea a fost facuta dupa 02-04-2021' then contor1:=contor1+1;

else contor2:=contor2+1;

end if;

end loop;

if contor1>contor2 then return 'Mai multe rezervari au fost facute dupa 02-04-2021';

else if contor2<contor1 then return 'Mai multe rezervari au fost facute inainte de 02-04-2021';

else return 'Au fost facute acelasi numar de rezervari inainte si dupa 02-04-2021';

end if;

end if;

end if;

exception

when NO\_DATA\_FOUND then

DBMS\_OUTPUT.PUT\_LINE('Tara data nu are rezervari');

RAISE\_APPLICATION\_ERROR (-20000,'Tara data nu are nici un client care sa fi facut o rezervare');

when TOO\_MANY\_ROWS then

DBMS\_OUTPUT.PUT\_LINE('Mai multe tari poarta acelasi nume');

RAISE\_APPLICATION\_ERROR (-20001, 'Exista mai multe tari in baza de date cu acelasi nume');

end;

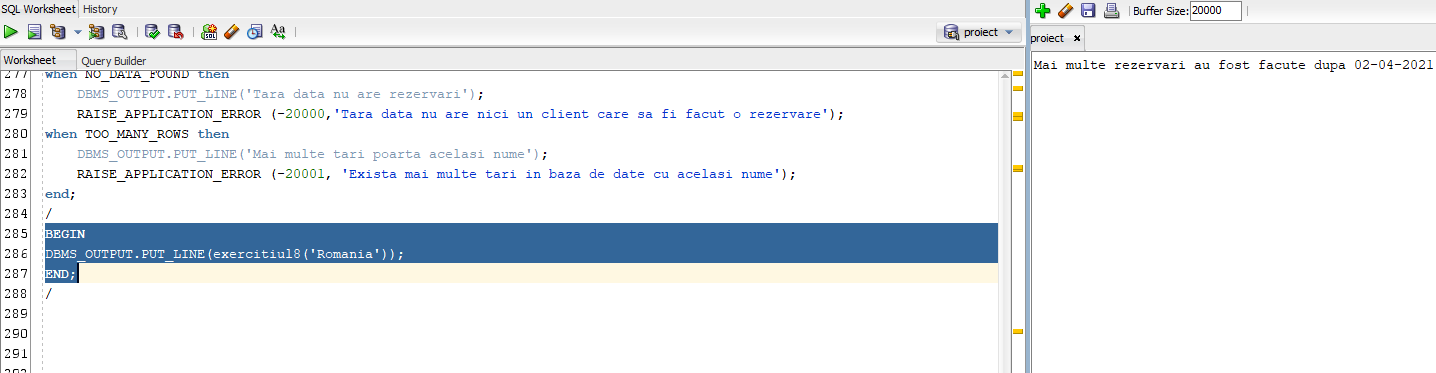
/

Rezultat:

BEGIN

DBMS\_OUTPUT.PUT\_LINE(exercitiul8('Romania'));

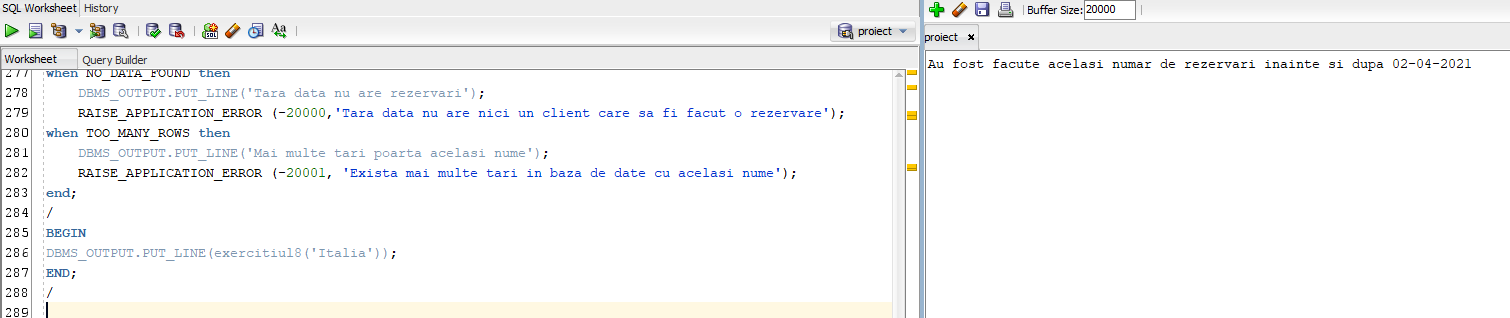
END;



BEGIN

DBMS\_OUTPUT.PUT\_LINE(exercitiul8('Italia'));

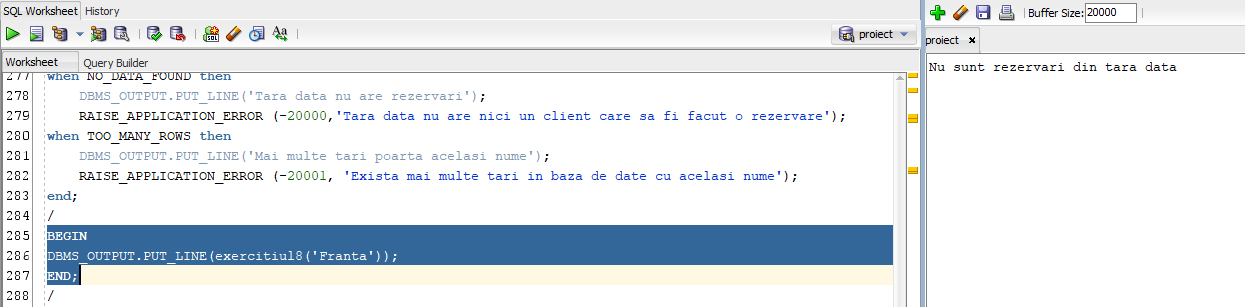
END;



BEGIN

DBMS\_OUTPUT.PUT\_LINE(exercitiul8('Franta'));

END;



* Formulați în limbaj natural o problemă pe care să o rezolvați folosind un **subprogram stocat de tip procedură** care să utilizeze într-o singură comandă SQL 5 dintre tabelele definite. Tratați toate excepțiile care pot apărea, incluzând excepțiile NO\_DATA\_FOUND și TOO\_MANY\_ROWS. Apelați subprogramul astfel încât să evidențiați toate cazurile tratate. (9)

Enunț:

Sa se afișeze, pentru o tara data, hotelurile unde au stat clienții din aceasta.

Rezolvare:

create or replace procedure exercitiul9

(ntara tara.nume\_tara%type)

is

id\_tara tara.cod\_tara%type;

TYPE tab\_nr IS TABLE OF varchar2(60);

n\_hotel tab\_nr;

begin

select h.nume\_hotel bulk collect into n\_hotel from hotel h

where h.cod\_hotel in (select ca.cod\_hotel from camera ca

where ca.cod\_camera in(select r.cod\_camera from rezervare r

WHERE r.cod\_client in (select c.cod\_client from clientul c

where c.cod\_tara in (select t.cod\_tara from tara t

where t.cod\_tara=c.cod\_tara and upper(nume\_tara)=upper(ntara)))));

for i in n\_hotel.first..n\_hotel.last loop

DBMS\_OUTPUT.PUT\_LINE(n\_hotel(i));

end loop;

exception

when NO\_DATA\_FOUND then

DBMS\_OUTPUT.PUT\_LINE('Tara data nu are rezervari');

RAISE\_APPLICATION\_ERROR (-20000,'Tara data nu are nici un client care sa fi facut o rezervare');

when TOO\_MANY\_ROWS then

DBMS\_OUTPUT.PUT\_LINE('Mai multe tari poarta acelasi nume');

RAISE\_APPLICATION\_ERROR (-20001, 'Exista mai multe tari in baza de date cu acelasi nume');

when VALUE\_ERROR then

DBMS\_OUTPUT.PUT\_LINE('Tara data nu are rezervari');

RAISE\_APPLICATION\_ERROR (-20002, 'Tara data nu are nici un client care sa fi facut o rezervare, deci NULL');

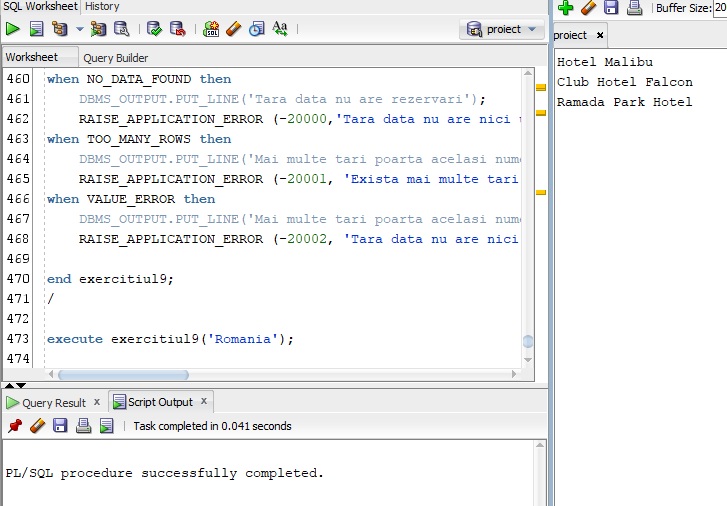
end exercitiul9;

/

Rezultat:

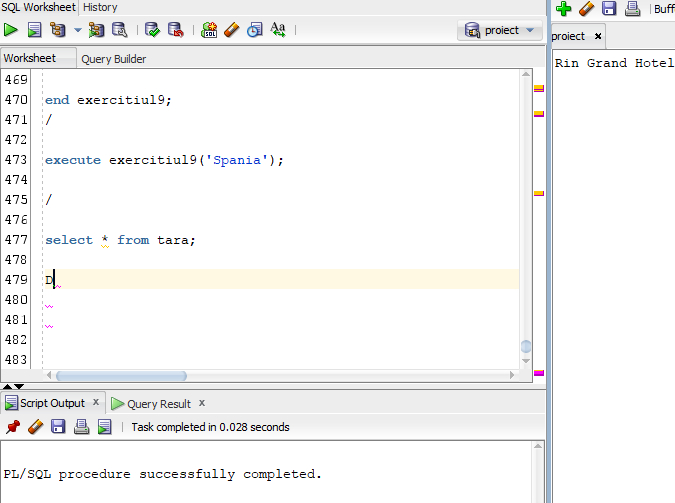
execute exercitiul9('Romania');

/



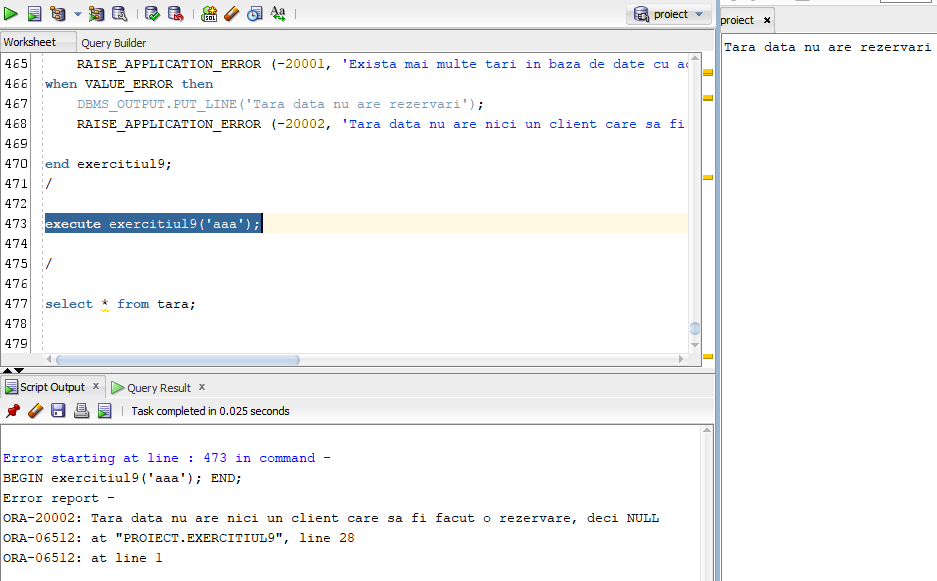
execute exercitiul9('Spania');

/



execute exercitiul9('aaa');

/



**Capitolul 6**

**Triggeri**

**(Cerința 10, 11 si 12)**

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

Enunț:

Rezervările nu se pot modifica pe întâi Decembrie sau Ianuarie. In plus, modificările se pot face intre orele 10-22 doar in zilele lucrătoare ale săptămânii.

Rezolvare:

CREATE OR REPLACE TRIGGER ext10

BEFORE INSERT OR UPDATE OR DELETE ON rezervare

BEGIN

IF ((to\_char(sysdate,'DD/MM')='01/12')or(to\_char(sysdate,'DD/MM')='01/01') )

OR (TO\_CHAR(SYSDATE,'HH24') NOT BETWEEN 10 AND 22)

OR ((to\_char(sysdate,'D')=6) or (to\_char(sysdate,'D')=7))

THEN

RAISE\_APPLICATION\_ERROR(-20001,'tabelul nu poate fi actualizat');

END IF;

END;

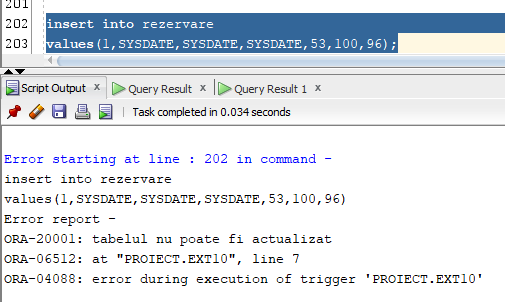
/

Rezultat:

Exemplu făcut in o zi de sâmbătă:

insert into rezervare

values(1,SYSDATE,SYSDATE,SYSDATE,53,100,96);



Ștergere:

drop trigger ext10;

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

Enunț:

Înainte de a da update la rezervare (mai exact câmpurile cod\_rezervare, data\_start, data\_final, data\_rezervare, cod\_client) triggerul se va activa daca încercam sa modificam cod\_rezervare, cod\_client sau daca data\_rezervare este diferita de sysdate. In plus, modificarea nu se poate aplica decât daca diferența dintre vechile dăți(data\_start, respectiv data\_final) este mai mica de o luna.

Rezolvare:

CREATE OR REPLACE TRIGGER ext11

BEFORE UPDATE of cod\_rezervare,data\_start,data\_final,data\_rezervare,cod\_client ON rezervare

for each row

when((old.cod\_rezervare!= new.cod\_rezervare)

or(old.cod\_client!=new.cod\_client)

or(new.data\_rezervare!= sysdate)

or(MONTHS\_BETWEEN( old.data\_start, new.data\_start)>1)

or(MONTHS\_BETWEEN( old.data\_final, new.data\_final)>1))

begin

RAISE\_APPLICATION\_ERROR(-20000,'Valori invalide');

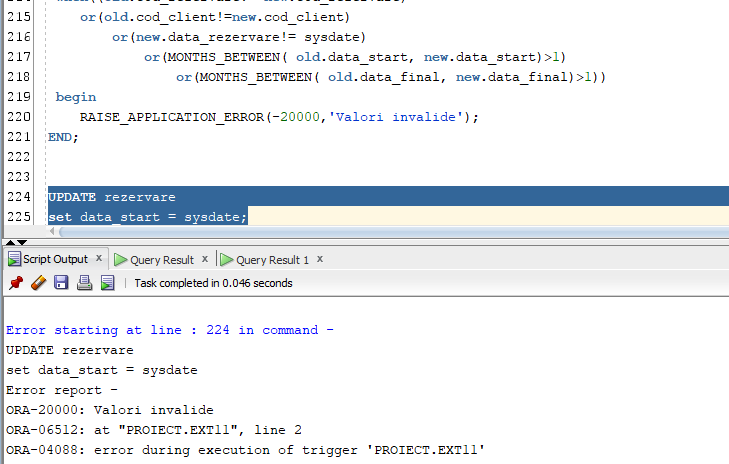
END;

/

Rezultat:

UPDATE rezervare

set data\_start = sysdate;



Ștergere:

drop trigger ext11;

* Definiți un *trigger* de tip LDD. Declanșați *trigger*-ul. (12)

Enunț:

Triggerul sa nu lase pe nimeni sa dea drop la orice tabel creat.

Rezolvare:

CREATE OR REPLACE TRIGGER ext12

AFTER DROP ON SCHEMA

BEGIN

RAISE\_APPLICATION\_ERROR (-20002, 'Tabelele nu pot fi sterse');

ROLLBACK;

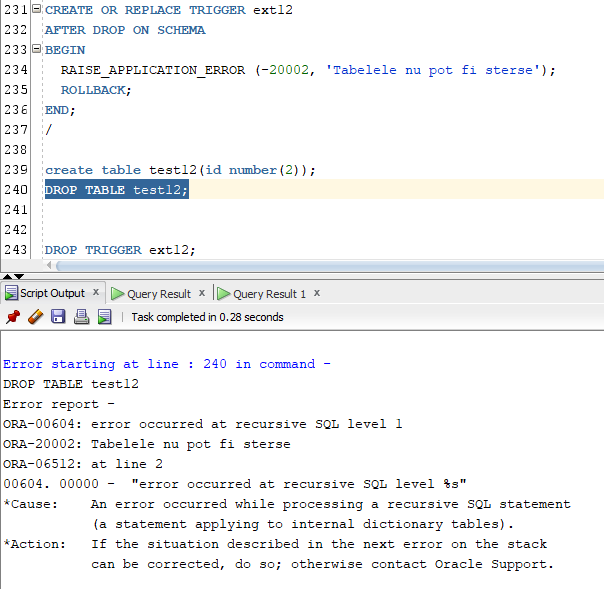
END;

/

Rezultat:

create table test12(id number(2));

DROP TABLE test12;



Ștergere:

drop trigger ext12;

**Capitolul 7**

**Cerințe opționale pentru nota finală**

**(Cerința 13)**

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

Create or Replace PACKAGE ex13 is

procedure exercitiul6;

procedure exercitiul7;

function exercitiul8

(n\_tara in tara.nume\_tara%type)

return varchar2;

procedure exercitiul9

(ntara tara.nume\_tara%type);

end ex13;

/

Create or Replace PACKAGE BODY ex13

is

procedure exercitiul6

IS

type t\_imbricat is table of number(3);

type t\_indexat is table of number(3) index by PLS\_INTEGER;

id\_camera t\_imbricat:=t\_imbricat();

id\_client t\_indexat;

nume\_client clientul.nume%type;

prenume\_client clientul.prenume%type;

id\_h hotel.cod\_hotel%type;

nume\_h hotel.nume\_hotel%type;

n\_rez number(3);

tara\_c number(3);

BEGIN

select cod\_client BULK collect into id\_client from clientul;

for i in id\_client.first..id\_client.last loop

SELECT cod\_camera BULK collect into id\_camera

from apare\_in\_istoricul

where id\_client(i) = cod\_client;

select nume,prenume into nume\_client,prenume\_client

from clientul

where id\_client(i) = cod\_client;

dbms\_output.put\_line('Clientul '||nume\_client||' '|| prenume\_client||' a stat pana acum in '||id\_camera.count()||' camere');

if id\_camera.count() !=0 then

for a in id\_camera.first..id\_camera.last loop

select cod\_hotel into id\_h from camera

where id\_camera(a) = cod\_camera;

select nume\_hotel into nume\_h FROM hotel

where cod\_hotel = id\_h;

dbms\_output.put\_line('Camera din hotelul: '||nume\_h);

end loop;

else dbms\_output.put\_line('Clientul '||nume\_client||' '|| prenume\_client||' nu a stat la nici un hotel ');

end if;

id\_camera.delete;

end loop;

End;

procedure exercitiul7

IS

TYPE tab\_nr IS TABLE OF NUMBER(4);

n\_hotel hotel.nume\_hotel%type;

t\_id tab\_nr;

t\_nr tab\_nr;

t\_ch tab\_nr;

cursor c\_camera is (select cod\_camera,numar\_camera,cod\_hotel from camera where cod\_hotel in

(select cod\_hotel from hotel

where mod(cod\_hotel,2)=0));

BEGIN

open c\_camera;

fetch c\_camera bulk collect into t\_id,t\_nr,t\_ch;

close c\_camera;

for i in t\_id.first..t\_id.last loop

select nume\_hotel into n\_hotel from hotel where cod\_hotel = t\_ch(i);

dbms\_output.put\_line('Camera cu id-ul '||t\_id(i)||' are numarul '||t\_nr(i)||' si apartine hotelului: '||n\_hotel);

end loop;

End;

function exercitiul8

(n\_tara in tara.nume\_tara%type)

return varchar2

is

TYPE tab\_nr IS TABLE OF varchar2(90);

t\_rez tab\_nr;

contor1 number(4);

contor2 number(4);

begin

contor1:=0;

contor2:=0;

select case when r.data\_rezervare >to\_date('05-04-2021', 'DD-MM-YYYY') then 'rezervarea a fost facuta dupa 02-04-2021' else 'rezervarea a fost facuta inainte de 02-04-2021' end bulk collect into t\_rez

from rezervare r

WHERE r.cod\_client in (select c.cod\_client from clientul c

where c.cod\_tara in (select t.cod\_tara from tara t

where t.cod\_tara=c.cod\_tara and upper(nume\_tara)=upper(n\_tara)));

if t\_rez.count= 0 then return 'Nu sunt rezervari din tara data';

else

for i in t\_rez.first..t\_rez.last loop

if t\_rez(i) = 'rezervarea a fost facuta dupa 02-04-2021' then contor1:=contor1+1;

else contor2:=contor2+1;

end if;

end loop;

if contor1>contor2 then return 'Mai multe rezervari au fost facute dupa 02-04-2021';

else if contor2<contor1 then return 'Mai multe rezervari au fost facute inainte de 02-04-2021';

else return 'Au fost facute acelasi numar de rezervari inainte si dupa 02-04-2021';

end if;

end if;

end if;

exception

when NO\_DATA\_FOUND then

DBMS\_OUTPUT.PUT\_LINE('Tara data nu are rezervari');

RAISE\_APPLICATION\_ERROR (-20000,'Tara data nu are nici un client care sa fi facut o rezervare');

when TOO\_MANY\_ROWS then

DBMS\_OUTPUT.PUT\_LINE('Mai multe tari poarta acelasi nume');

RAISE\_APPLICATION\_ERROR (-20001, 'Exista mai multe tari in baza de date cu acelasi nume');

end;

procedure exercitiul9

(ntara tara.nume\_tara%type)

is

id\_tara tara.cod\_tara%type;

TYPE tab\_nr IS TABLE OF number(8);

id\_client tab\_nr;

id\_rezervare tab\_nr;

id\_camera camera.cod\_camera%type;

id\_hotel hotel.cod\_hotel%type;

n\_hotel hotel.nume\_hotel%type;

n\_camera camera.numar\_camera%type;

n\_nume clientul.nume%type;

n\_prenume clientul.prenume%type;

d\_rezervare rezervare.data\_rezervare%type;

begin

select cod\_tara into id\_tara from tara where upper(ntara) = upper(nume\_tara);

select cod\_client bulk collect into id\_client from clientul where id\_tara=cod\_tara;

for i in id\_client.first..id\_client.last loop

select nume,prenume into n\_nume,n\_prenume from clientul where cod\_client = id\_client(i);

select cod\_rezervare bulk collect into id\_rezervare from rezervare where cod\_client=id\_client(i);

for j in id\_rezervare.first..id\_rezervare.last loop

select data\_rezervare into d\_rezervare from rezervare where cod\_rezervare = id\_rezervare(j);

select cod\_camera into id\_camera from rezervare where cod\_rezervare = id\_rezervare(j);

select numar\_camera,cod\_hotel into n\_camera,id\_hotel from camera where cod\_camera=id\_camera;

select nume\_hotel into n\_hotel from hotel where cod\_hotel=id\_hotel;

DBMS\_OUTPUT.PUT\_LINE(n\_nume||' '||n\_prenume||' din '||ntara||' a facut o rezervare pe camera cu numarul '||n\_camera||' in data de '||d\_rezervare||' la hotelul '||n\_hotel);

end loop;

end loop;

exception

when NO\_DATA\_FOUND then

DBMS\_OUTPUT.PUT\_LINE('Tara data nu are rezervari');

RAISE\_APPLICATION\_ERROR (-20000,'Tara data nu are nici un client care sa fi facut o rezervare');

when TOO\_MANY\_ROWS then

DBMS\_OUTPUT.PUT\_LINE('Mai multe tari poarta acelasi nume');

RAISE\_APPLICATION\_ERROR (-20001, 'Exista mai multe tari in baza de date cu acelasi nume');

end exercitiul9;

end;

Rezultat:

