**Proiect**

**Baze de Date**

STAN C.G. ANA-MARIA

GRUPA 141

UNIVERSITATEA DIN BUCUREȘTI

FACULTATEA DE MATEMATICĂ ȘI INFORMATICĂ

SPECIALIZAREA INFORMATICĂ

PROFESOR COORDONATOR

**BENEGUI CEZARA**

**28.05.2021**

**CUPRINS**

|  |  |  |
| --- | --- | --- |
| Capitolul 1 | **Cerința 1 - Descrierea modelului real, a utilității acestuia și a regulilor de funcționare** | Pag. 3 |
| Capitolul 2 | **Cerința 2 - Prezentarea constrângerilor (restricții, reguli) impuse asupra modelului** | Pag. 3 |
| Capitolul 3 | **Cerința 3 - Descrierea entităților, incluzând precizarea cheii primare** | Pag. 4 |
| Capitolul 4 | **Cerința 4 - Descrierea relațiilor, incluzând precizarea cardinalității acestora** | Pag. 5 |
| Capitolul 5 | **Cerința 5 - Descrierea atributelor, incluzând tipul de date și eventualele constrângeri, valori implicite, valori posibile ale atributelor** | Pag. 6 |
| Capitolul 6 | **Cerința 6 - Realizarea diagramei entitate-relație corespunzătoare descrierii de la cerințele 3-5** | Pag. 9 |
| Capitolul 7 | **Cerința 7 - Realizarea diagramei conceptuale corespunzătoare diagramei entitate-relație proiectată la cerința 6.** | Pag. 10 |
| Capitolul 8 | **Cerința 8 - Enumerarea schemelor relaționale corespunzătoare diagramei conceptuale proiectată la cerința 7** | Pag. 11 |
| Capitolul 9 | **Cerința 9 - Realizarea normalizării până la forma normală 3 (FN1-FN3)** | Pag. 11 |
| Capitolul 10 | **Cerința 10 - Crearea tabelelor în SQL și inserarea de date coerente în fiecare dintre acestea (minimum 5 înregistrări în fiecare tabel neasociativ; minimum 10 înregistrări în tabelele asociative)**  **Cerința 13 - Crearea unei secvențe ce va fi utilizată în inserarea înregistrărilor în tabele (cerința 10)** | Pag. 14  Pag. 17 |
| Capitolul 11 | **Cerința 11 - Formulați în limbaj natural și implementați 5 cereri SQL complexe** | Pag. 21 |
| Capitolul 12 | **Cerința 12 - Implementarea a 3 operații de actualizare sau suprimare a datelor utilizând subcereri** | Pag. 24 |
| Capitolul 13 | **Cerința 16 - Formulați în limbaj natural și implementați în SQL: o cerere ce utilizează operația outerjoin pe minimum 4 tabele și două cereri ce utilizează operația division** | Pag. 27 |
| Capitolul 14 | **Bibliografie** | Pag. 30 |

**Capitolul 1**

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

**(Cerința 1)**

Proiectul meu are ca obiectiv 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, număr de persoane și numărul camerei.

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

**Capitolul 2**

**Prezentarea constrângerilor (restricții, reguli) impuse asupra modelului**

**(Cerința 2)**

* 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

**Capitolul 3**

**Descrierea entităților, incluzând precizarea cheii primare**

**(Cerința 3)**

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.

ISTORIC\_CAMERA = 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***.

**Capitolul 4**

**Descrierea relațiilor, incluzând precizarea cardinalității acestora**

**(Cerința 4)**

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.

**Capitolul 5**

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

**(Cerința 5)**

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:

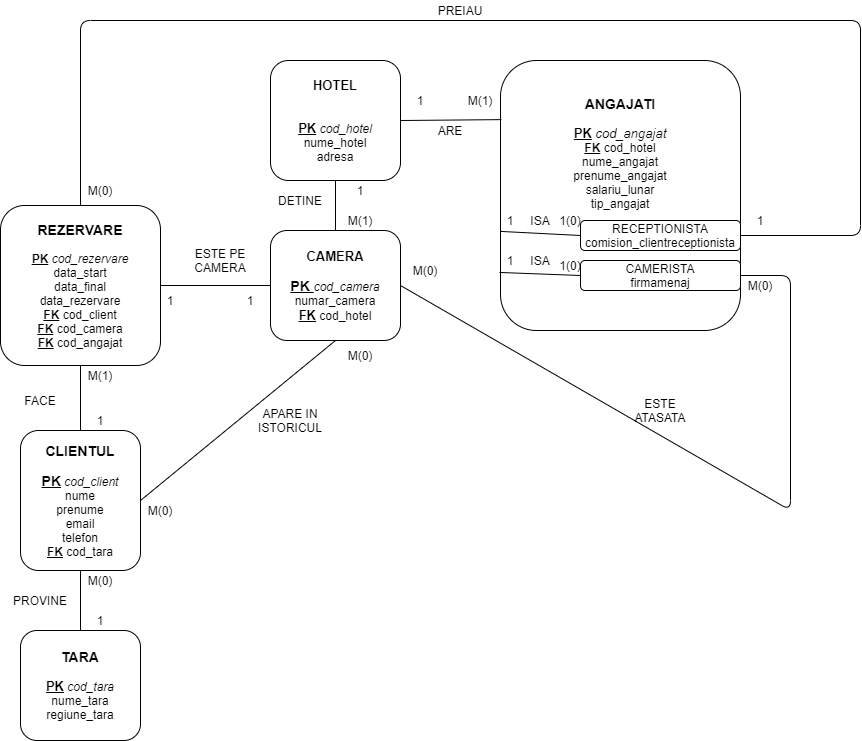
* cod\_angajat = codul exclusiv al unei MENAJERE, variabilă de tip întreg de lungime maximă 4, care reprezintă codul angajatului și este cheie externă din entitatea ANGAJATI
* cod\_camera = variabilă de tip întreg de lungime maximă 4 care reprezintă codul camerei care se afla și în entitatea CAMERA. 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 6**

**Realizarea diagramei entitate-relație corespunzătoare**

**descrierii de la cerințele 3-5**

**(Cerința 6)**

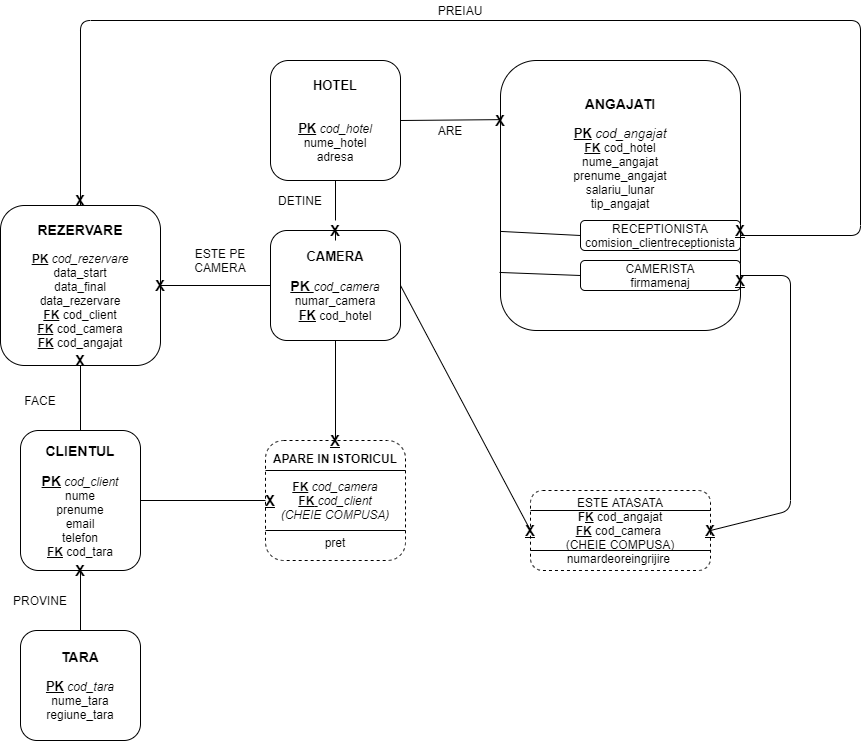


**Capitolul 7**

**Realizarea diagramei conceptuale corespunzătoare diagramei entitate-relație proiectată la cerința 6**

**(Cerința 7)**

Diagrama conceptuală obținută trebuie să conțină minimum 6 tabele (fără considerarea subentităților), dintre care cel puțin un tabel asociativ



**Capitolul 8**

**Enumerarea schemelor relaționale corespunzătoare diagramei conceptuale proiectată la cerința 7**

**(Cerința 8)**

**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**)**

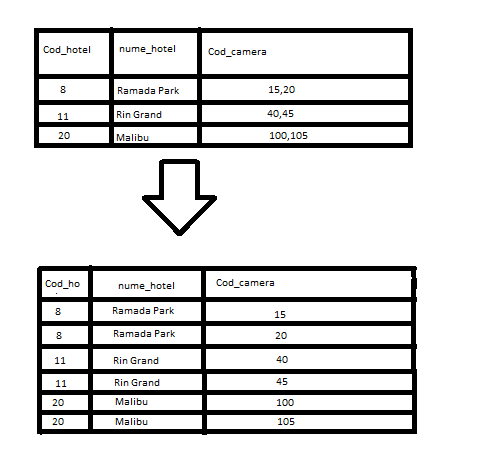
**Capitolul 9**

**Realizarea normalizării până la forma normală 3 (FN1-FN3)**

**(Cerința 9)**

Transformare non-FN1 -> FN1:

O să luam ca exemplu tabelul hotel combinat cu tabelul cameră. Primul tabel este non-FN1, deoarece câmpul cod\_camera stochează toate camerele fiecărui hotel. Astfel, regula FN1 nu este respectata. Transformam din primul tabel non-FN1 in FN1 adaugand o linie noua pentru fiecare camera.



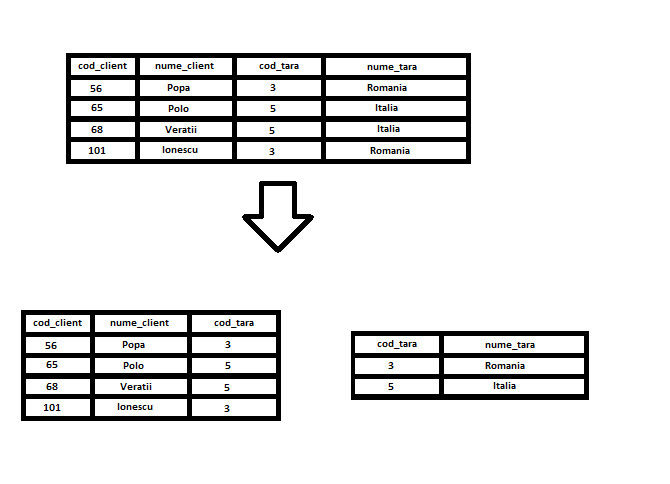
Pentru transformarea FN1 avem regula: O relație este în prima formă normală dacă fiecărui atribut care o compune îi corespunde o valoare indivizibilă (atomică). Astfel, pentru transformarea în FN1 o să facem un rând special pentru fiecare cameră de hotel.

Transformare FN1->FN2:

Primul tabel stochează codul clienților, codul și numele țării de proveniență. Primul tabel prezentat este FN1, așadar transformăm în FN2 pentru că aparține primei forme de normalizare. Ca să devină FN2 fiecare atribut care nu este cheie (nu participă la cheia primară) va deveni dependent de întreaga cheie primară. Separăm tabelul inițial în doua tabele care aparțin lui FN1 și FN2, pentru că o relație este în primă formă normală dacă fiecărui atribut care o compune îi corespunde o valoare indivizibilă și fiecare atribut care nu este cheie depinde de întreaga cheie primară. În plus, tabelul de mai jos aparține și FN3.

#Cod\_client={nume\_client, #cod\_tara }

#Cod\_tara={#cod\_tara, nume\_tara}

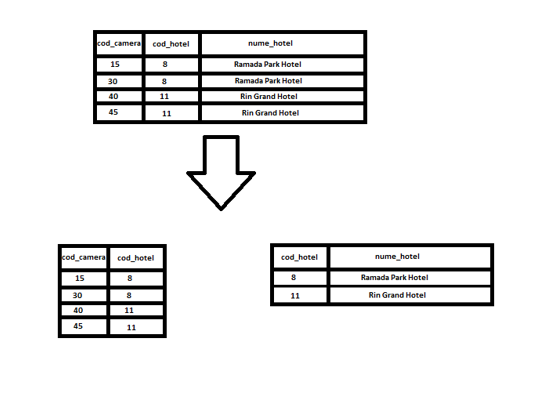
****

Exemplu FN2-FN3:

O relație non-FN3 este primul tabel în care se regăsesc detalii despre codul camerei, codul hotelului și numele hotelului. Pentru a deveni FN3 aceasta trebuie să fie FN2 și fiecare atribut care nu este cheie (nu participă la o cheie) depinde direct de cheia primară. Tabelele sunt în FN2 pentru că atributele sunt dependente de cheia primară adică cod\_camera. Astfel, pentru a deveni FN3 separăm în 2 tabele diferite în care atributele depind direct de cheia primară.

#Cod\_camera={cod\_hotel}

#Cod\_hotel={nume\_hotel}



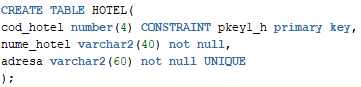
**Capitolul 10**

**Cerința 10 - Crearea tabelelor în SQL și inserarea de date coerente în fiecare dintre acestea (minimum 5 înregistrări în fiecare tabel neasociativ; minimum 10 înregistrări în tabelele asociative)**

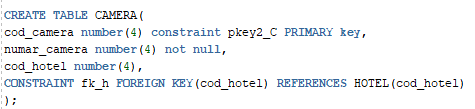
**Cerința 13 - Crearea unei secvențe ce va fi utilizată în inserarea înregistrărilor în tabele (cerința 10).**

* **Crearea tabelelor – cerința 10**

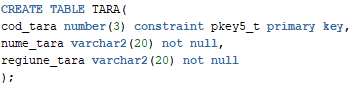
Entitatea Hotel:

****

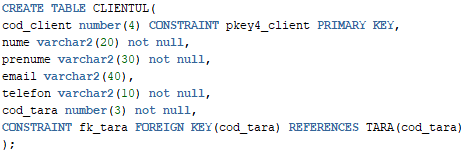
Entitatea Camera:



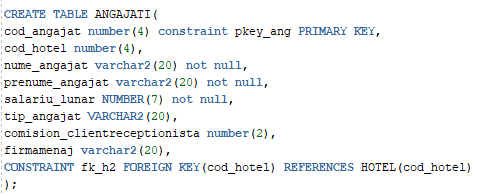
Entitatea Tara:



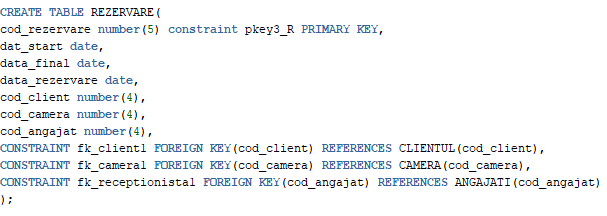
Entitatea Clientul:



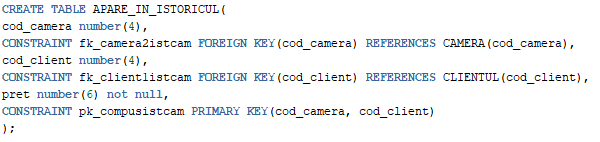
Entitatea Angajati:



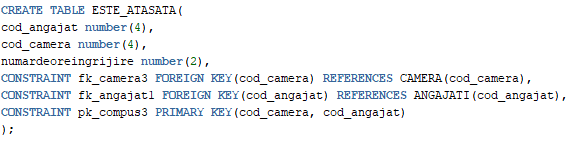
Entitatea Rezervare:



Entitatea Apare\_In\_Istoricul:

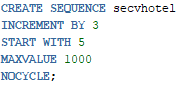


Entitatea Este\_Atasata:

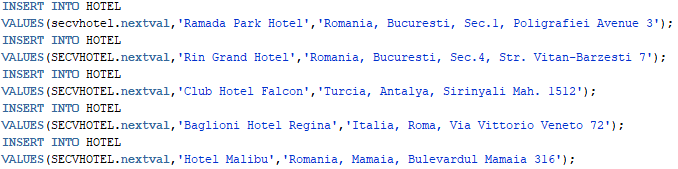


* **Inserarea în tabel și rezultatul inserării – cerința 13**

Secvența folosită



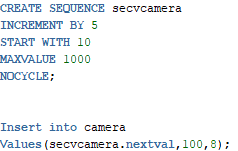
Inserare:



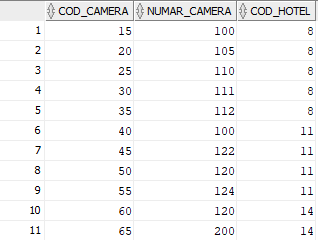
Rezultat:

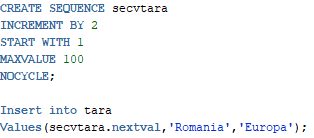


La următoarele entități voi prezenta doar un exemplu de inserare, secvența și rezultatul:

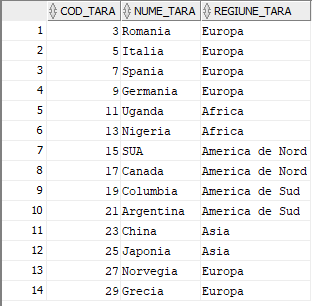


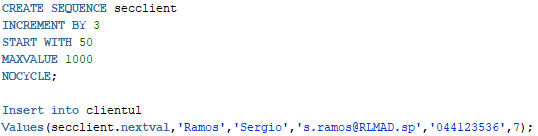
Rezultat (câteva exemple):





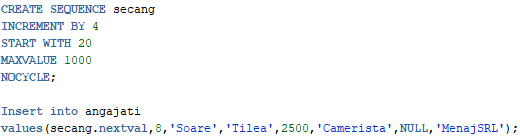
Rezultat (câteva exemple):





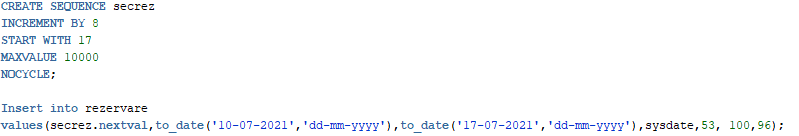
Rezultat (câteva exemple):



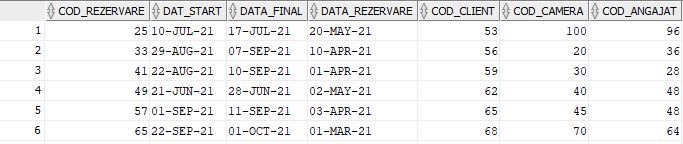


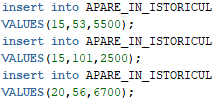
Rezultat (câteva exemple):



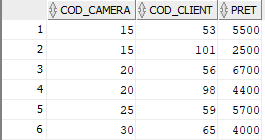


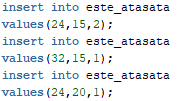
Rezultat (câteva exemple):



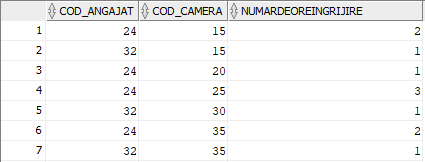


Rezultat (câteva exemple):





Rezultat (câteva exemple):



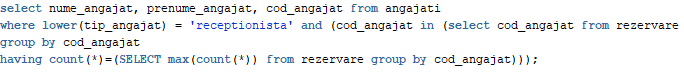
**Capitolul 11**

**Formulați în limbaj natural și implementați 5 cereri SQL complexe**

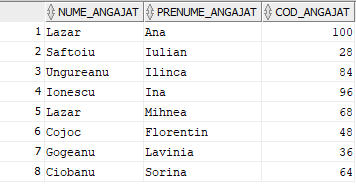
**(Cerința 11)**

* 1. **Să se afișeze numele, prenumele și codul recepționistelor care au făcut cele mai multe rezervări**

Cererea:



Rezultatul:

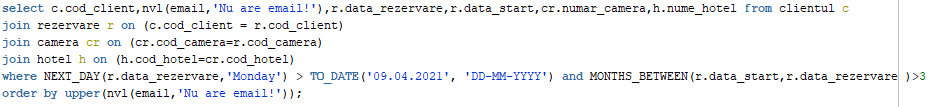


Explicație:

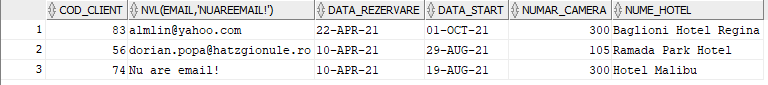
Folosim funcția lower pentru a fi siguri că alegem toate receptionistele iar apoi în al doilea select facem cererea (folosim și group by, having) și alegem angajații care au făcut celel mai multe rezervări.

* 1. **Să se aleagă codul clientului, emailul, data rezervării, data când începe rezervarea, numărul camerei și numele hotelului clienților care au următoarea zi de luni după efectuarea rezervării după data de 9 aprilie 2021 și distanța dintre data rezervării și data de pe cât începe sejurul să fie mai mare de 3 luni. Să se ordoneze alfabetic după email ignorand faptul că încep sau nu cu literă mare. În plus, dacă emailul este null să se înlocuiască cu 'Nu are email!'.**

Cererea:



Rezultatul:

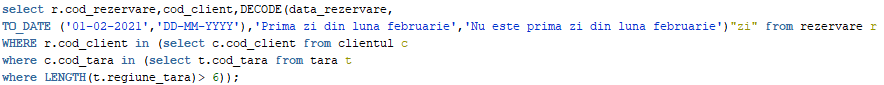


Explicație:

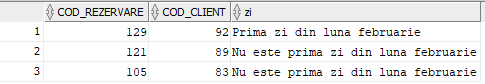
Aplicăm un join pe 4 tabele pentru a obține date despre rezervare, client, cameră și hotel. În where aplicăm funcția next day pentru a alege prima zi de luni după dată rezervării . La final folosim upper pentru a ordona după email, neținând cont dacă caracterele sunt mari sau mici.

* 1. **Să se afișeze codul clientului, codul rezervării și dacă data rezervării este prima zi din luna februarie a clienților care provin dintr-un continent care are mai mult de 6 litere.**

Cererea:



Rezultatul:

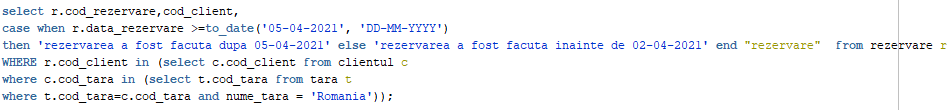


Explicație:

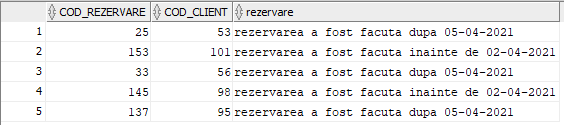
Folosim o cerere nesincronizată pe 3 tabele, apoi aplicăm decode pentru a verifica dacă este prima zi din luna februarie sau nu. În final, verificăm dacă lungimea numelui continentului din care provine țara clientului este mai mare de 6 caractere.

* 1. **Să se afișeze codul rezervării, codul clientului și dacă rezervarea a fost făcută după sau înainte de 5 aprilie 2021, a rezervărilor făcute de clienți care provin din România.**

Cererea:



Rezultatul:

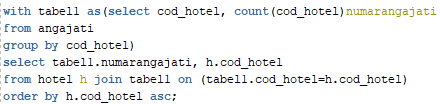


Explicație:

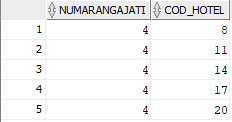
Folosim o cerere sincronizată pe 3 tabele prentru a verifica dacă rezervarea este făcută de un client din România. Apoi folosim o funcție case pentru a verifica dacă data rezervării este înainte dau după 5 aprilie 2021.

* 1. **Să se afișeze cu WITH câți angajați are fiecare hotel, să se ordoneze ascendent după codul hotelului.**

Cererea:



Rezultatul:



Explicație:

Facem un nou tabel în care introducem coloana numarangajati (numărăm cu count de câte ori apare fiecare cod\_hotel în tabelul angajați).

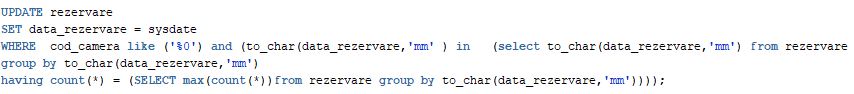
**Capitolul 12**

**Implementarea a 3 operații de actualizare sau suprimare a datelor utilizând subcereri**

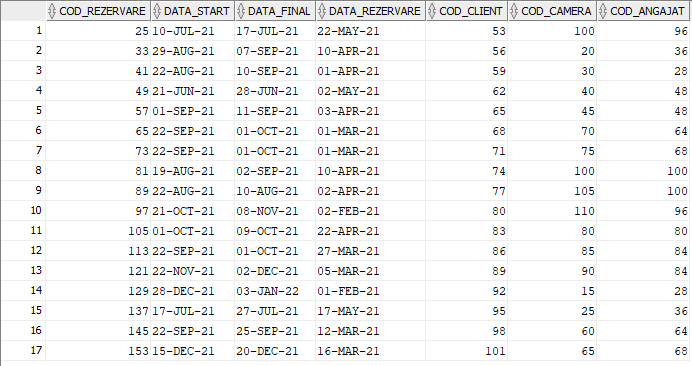
**(Cerința 12)**

* 1. **Să se seteze data rezervării în data curentă atunci când codul camerei se termină în 0 și luna în care s-a făcut rezervarea este luna cu cele mai multe rezervări.**

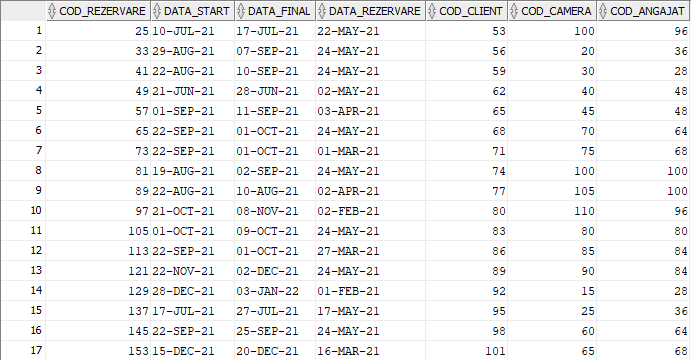
Cererea:



Înainte de update:

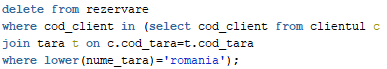


După update:

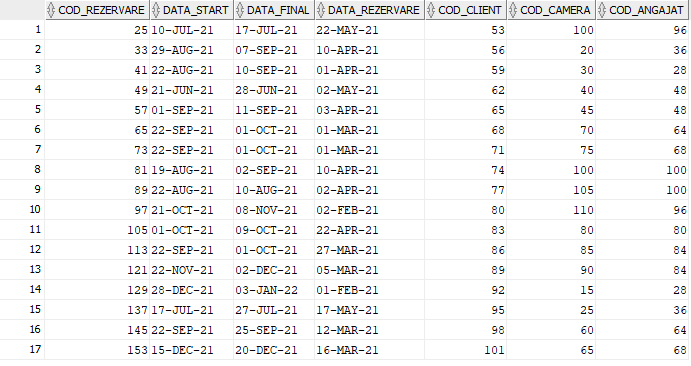


* 1. **Să se șteargă rezervările clienților care provin din România**

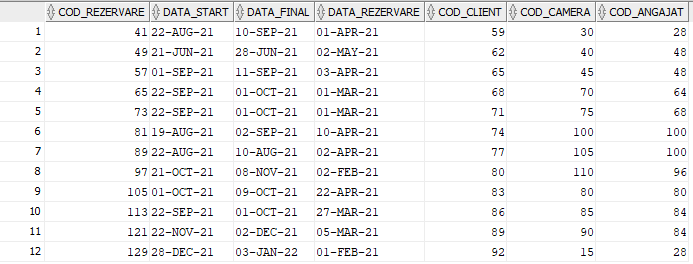
Cererea:



Înainte de delete:

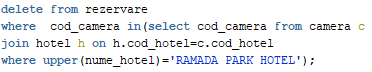


După delete:

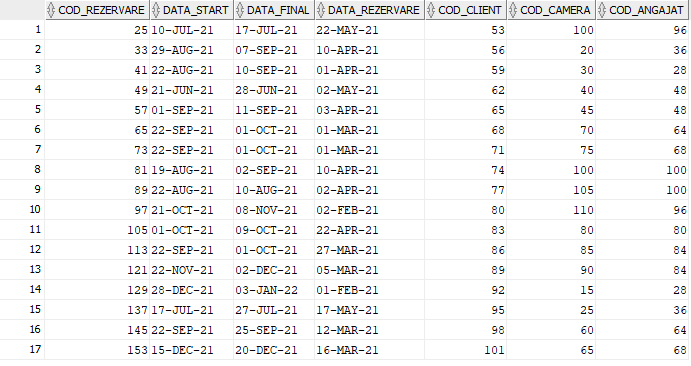


* 1. **Să se șteargă rezervările făcute la RAMDA PARK HOTEL**

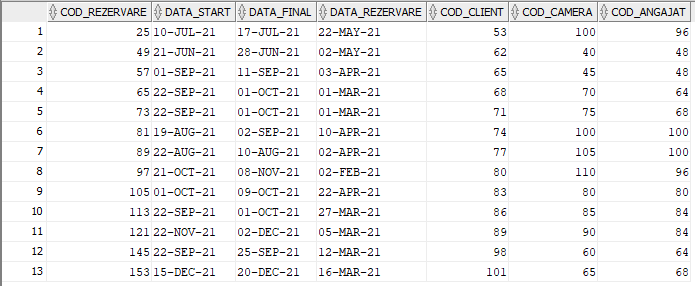
Cererea:



Înainte de delete:



După delete:



**Capitolul 13**

**Formulați în limbaj natural și implementați în SQL:**

**o cerere ce utilizează operația outerjoin pe minimum 4 tabele**

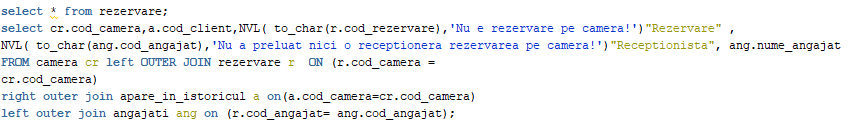
**și două cereri ce utilizează operația division**

**(Cerința 16)**

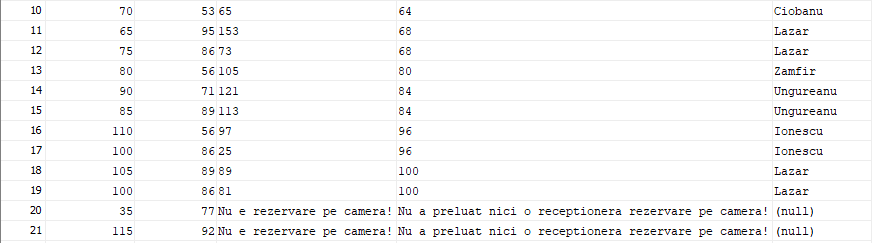
**13.1. Cerință outerjoin**:

Să se afișeze codul camerei, codul ultimului client care a stat în cameră, codul rezervării curente pe cameră (dacă nu există, să se completeze cu „Nu e rezervare pe camera!”), numele și codul angajatului care a făcut rezervarea (daca nu există, să se completeze cu „Nu a preluat nici o recepționera rezervarea pe camera!”), care apar în tabelele cameră și rezervare, apoi al celor care apar și în istoricul camerei.

Rezolvare:



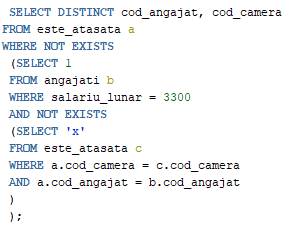
Exemplu de rezultat:



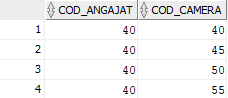
**13.2. Cerință division:**

Să se obțină camerele pe care le îngrijesc și codurile angajaților distincți care au salariul egal cu 3300.

Rezolvare:



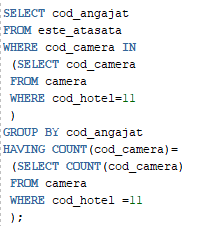
Rezultat:



**13.3. Cerință division**:

Să se obțină codul agnaților distincți care lucrează în hotelul cu codul 11.

Rezolvare:



Rezultat:



**BIBLIOGRAFIE**

Cezarabenegui.com/courses/ (laboratoare si cursuri)

http://www.elth.pub.ro/~preda/teaching/BDE/BDE\_5.pdf