

Proiect Baze de Date
Gestionarea sejururilor din cadrul unui lanț de hoteluri

Capatina Razvan-Nicolae
Grupa 152

Cuprins

Gestiunea sejururilor din cadrul unui lant de hoteluri

1. Descrierea modelului real, a utilitatii acestuia si a regulilor de functionare	3
2. Prezentarea constrangerilor impuse asupra modelului	3
3. Descrierea entitatilor si precizarea cheii primare	3
4. Descrierea relatiilor si precizarea cardinalitatii acestora	4
5. Descrierea atributelor, a tipurilor de date si a eventualelor strangeri	5
6. Diagrama Entitate-Relatie	8
7. Diagrama Conceptuala	9
8. Reprezentarea schemelor relationale corespunzatoare diagramei conceptuale	10
9. Realizarea normalizarii pana la forma normala 3	10
9.1 Forma normala 1 (FN1)	10
9.2 Forma normala 2 (FN2)	11
9.3 Forma normala 3 (FN3)	12
10. Realizarea normalizarii BCNF, FN4 si FN5	13
10.1 Forma normala Boyce-Codd (BCNF)	13
10.2 Forma normala 4 (FN4)	14
10.3 Forma normala 5 (FN5)	15
11. Aplicarea denormalizarii si justificarea necesitatii acesteaia	16
12. Optimizarea unei cereri	17
13. Crearea unei sevante ce va fi utilizata in inserarea inregistrarilor in tabele	20
14. Crearea tabelelor in SQL si inserarea de date coerente in fiecare dintre acestea	37
15. Formularea in limbaj natural si implementarea a 5 cereri SQL complexe	68
16. O cerere ce utilizeaza operatia Outer-Join pe minimum 4 tabele, o cerere ce utilizeaza operatia Division si o cerere care implementeaza analiza Top-n	72
17. Implementarea a 3 operatii de actualizare si de suprimare a datelor utilizand subcereri	75
17.1 Actualizare	75
17.2 Suprimare	75

Gestiunea sejururilor din cadrul unui lant de hoteluri

Descrierea modelului real, a utilitatii acestuia si a regulilor de functionare

Modelul de date va gestiona informatii legate de un lant de hoteluri. Vor fi luate in calcul ce hoteluri exista, ce camere apartin de fiecare hotel si daca au sau nu anumite facilitati. Fiecare hotel se va afla intr-o zona si va avea sau nu un restaurant si o zona de parcare formata din mai multe locuri. In cazul in care exista, restaurantul va servi mai multe feluri de mancare. De asemenea, vor exista mai multi angajati in cadrul lantului de hoteluri, acestia ocupand diverse functii. In cadrul restaurantului vor lucra bucaturi si chelneri, in timp ce in cadrul hotelului vor lucra receptionisti si administratori. Zona de parcare, in cazul in care exista, va fi supravegheta de un paznic. O camera va putea fi rezervata de mai multe persoane. O persoana poate rezerva mai multe camere. Fiecare hotel, restaurant si zona de parcare va fi situata in exact o locatie. Furnizorii, in cazul existentei lor, vor putea incheia mai multe contracte cu mai multe restaurante din cadrul lantului de hoteluri, contribuind la aprovizionarea acestora. Un tip de contract va putea fi incheiat intre mai multi furnizori si mai multe restaurante, iar un restaurant poate realiza mai multe intelegeri cu diversi furnizori.

Prezentarea constrangerilor impuse asupra modelului

Fiecare hotel, restaurant si zona de parcare trebuie sa apartina de exact o locatie, iar o locatie poate contine 0 sau mai multe hoteluri, restaurante si zone de parcare.

O locatie trebuie sa aiba o adresa nevida.

Un hotel trebuie sa aiba o denumire si va contine camere. O camera trebuie sa apartina de exact un hotel si trebuie sa se afle la un etaj in cadrul acestuia si sa aiba, de asemenea, asociat un numar. Numarul de stele si anul inflantarii unui hotel sunt optionale.

O camera poate contine 0 sau mai multe facilitati, iar o facilitate poate fi regasita la 0 sau mai multe camere. O facilitate trebuie sa aiba o anumita denumire.

Un client poate rezerva 0 sau mai multe camere, iar o camera poate fi rezervata de 0 sau mai multi clienti. Rezervarea va memoriza obligatoriu data de inceput si data de sfarsit.

Un client trebuie sa aiba un nume, un prenume si un numar de telefon. Introducerea CNP-ului clientului si a unei adrese de mail este optionala.

O zona de parcare apartine de exact un hotel si contine locuri de parcare. Un loc de parcare apartine de exact o zona de parcare si trebuie sa aiba un numar asociat.

In cadrul lantului de hoteluri exista mai multi angajati. Un angajat poate ocupa functia de bucatar, chelner, paznic, receptionist sau administrator. Intr-un hotel trebuie sa lucreze receptionisti si administratori. De asemenea, intr-un restaurant trebuie sa lucreze chelneri si bucaturi. O zona de parcare va fi supravegheta de zero sau mai multi paznici.

Fiecare angajat trebuie sa aiba un nume, un prenume, un numar de telefon, un salar mentionat si un CNP. Adresa de mail a unui angajat poate sa fie omisa.

Un restaurant trebuie sa aiba un nume, dar nu este obligatoriu sa aiba precizat numarul de stele sau anul inflantarii sale. Un restaurant poate servi mai multe feluri de mancare, iar un fel de mancare poate fi servit la 0 sau mai multe restaurante. Un fel de mancare trebuie sa aiba precizat denumirea sa, cat si pretul.

Un furnizor trebuie sa aiba un nume, iar un contract trebuie sa aiba precizate data de inceput si data de sfarsit a acestuia.

Un furnizor poate incheia 0 sau mai multe tipuri de contracte cu 0 sau mai multe restaurante. Un restaurant poate avea incheiate 0 sau mai multe contracte cu 0 sau mai multi furnizori, iar un tip de contract poate fi incheiat intre 0 sau mai multe restaurante si 0 sau mai multi furnizori.

Descrierea entitatilor si precizarea cheii primare

Pentru modelul de date referitor la gestiunea sejururilor din cadrul unui lant de hoteluri, structurile LOCATIE, HOTEL, CAMERA, CLIENT, FACILITATE, ZONA_DE_PARCARE, LOC_DE_PARCARE, CONTRACT, FURNIZOR, RESTAURANT, FEL_DE_MANCARE, ANGAJAT, PAZNIC, RECEPTIONIST, ADMINISTRATOR, CHELNER, BUCATAR reprezinta entitati.

Toate entitatile care vor fi prezентate sunt independente, cu exceptia entitatilor dependente CAMERA si LOC_DE_PARCARE si a subentitatilor PAZNIC, RECEPTIONIST, ADMINISTRATOR, CHELNER si BUCATAR.

HOTEL = cladire ce va contine mai multe camere si in care pot fi cazati clientii. Entitatea va avea o denumire, un numar de stele, ceea ce va estima calitatea serviciilor din cadrul hotelului, cat si un an al inflantarii acestuia. Cheia primara este **cod_hotel**.

CAMERA = o incadare din cadrul unui hotel unde vor putea fi cazati clienti. Camera se va afla la un anumit etaj al hotelului si va avea asociat un numar. Entitatea este dependenta de HOTEL, iar cheia primara este compusa din **cod_camera** si **cod_hotel**.

CLIENT = persoana fizica ce poate rezerva camere in cadrul lantului de hoteluri. Entitatea va avea un nume, un prenume, un CNP, un numar de telefon si o adresa de mail. Cheia primara este **cod_client**.

FACILITATE = o trasatura suplimentara a camerei care poate fi regasita la mai multe dintre acestea sau la niciuna. Are o anumita denumire si indica daca o camera este single, dubla. Cheia primara este **cod_facilitate**.

LOCATIE = zona in care se pot afla hotelurile, restaurantele si parcarile. Aceasta entitate contine adresa zonei si are cheia primara **cod_locatie**.

ZONA_DE_PARCARE = o suprafață formata din mai multe locuri de parcare unde clientii pot parca autovehiculul.

Cheia primara este **cod_zona_de_parcare**.

LOC_DE_PARCARE = un sector din zona de parcare predestinat unui singur autovehicul. Aceast loc are un numar asociat si este o entitate dependenta de ZONA_DE_PARCARE, avand cheia primara compusa din **cod_loc_de_parcare** si **cod_zona_de_parcare**.

RESTAURANT = cladire unde clientii pot servi masa si in care lucreaza angajati din cadrul lantului de hoteluri: chelneri si bucaturi. Restaurantul are un an in care a fost inflantat, o denumire, cat si un numar de stele ce aproximeaza calitatea serviciilor sale. Cheia primara este **cod_restaurant**.

FEL_DE_MANCARE = un produs din cadrul meniului unui restaurant. Aceasta are o denumire si un pret. Cheia primara este **cod_fel_de_mancare**.

CONTRACT = o intelegera intre un furnizor si un restaurant prin care primul va asigura aprovizionarea ultimului. Acest contract va contine data de inceput, cat si cea de sfarsit. Cheia primara este **cod_contract**.

FURNIZOR = persoana fizica sau juridica care poate aproviziona un restaurant, avand o denumire. Cheia primara este **cod_furnizor**.

ANGAJAT = persoana fizica, angajata in cadrul lantului de hoteluri, care se ocupa cu diverse sarcini si are atribuite anumite responsabilitati.

Angajatul are un nume, un prenume, un CNP, un numar de telefon, un salar si o adresa de mail. Cheia primara este **cod_angajat**.

PAZNIC = persoana fizica, un angajat al lantului de hoteluri care se ocupa cu supravegherea zonei de parcare. PAZNIC este o subentitate a entitatii ANGAJAT, avand cheia primara **cod_angajat**.

RECEPTIONIST = persoana fizica, un angajat al hotelului care se ocupa cu preluarea clientilor ce tocmai au sosit si cu atribuirea unor camere acestora.

RECEPTIONIST este o subentitate a entitatii ANGAJAT, avand cheia primara **cod_angajat**.

ADMINISTRATOR = persoana fizica, un angajat al hotelului care se ocupa cu asigurarea bunei desfasurari a activitatilor in cadrul acestuia. ADMINISTRATOR este o subentitate a entitatii ANGAJAT, avand cheia primara **cod_angajat**.

CHELNER = persoana fizica, un angajat al unui restaurant din cadrul lantului de hoteluri. Aceasta are sarcina de a prelua si transmite catre bucaturi comenzile clientilor. CHELNER este o subentitate a entitatii ANGAJAT, avand cheia primara **cod_angajat**.

BUCATAR = persoana fizica, un angajat al restaurantului care se ocupa cu prepararea comenzilor oferite din partea clientilor.

BUCATAR este o subentitate a entitatii ANGAJAT, avand cheia primara **cod_angajat**.

Descrierea relatiilor si precizarea cardinalitatii acestora

CAMERA_presinta_FACILITATE = relatie de tip many-to-many intre CAMERA si FACILITATE, reprezentand ce beneficii sau trasaturi suplimentare are camera respectiva. Relatia are cardinalitatea minima 0:0 (o camera poate sa nu aiba nicio facilitate, iar o facilitate poate sa nu fie regasita la nicio camera) si cardinalitatea maxima n:n (o camera poate avea mai multe facilitati, iar o facilitate poate fi regasita la mai multe camere).

CLIENT_rezerva_CAMERA = relatie de tip many-to-many intre CLIENT si CAMERA, reprezentand ce rezervari au fost realizate de catre clienti si asupra caror camere. Relatia are cardinalitatea minima 0:0 (un client nu a rezervat nicio camera si o camera nu a fost rezervata de niciun client) si cardinalitatea maxima n:n (un client a rezervat mai multe camere si o camera a fost rezervata de mai multi clienti).

HOTEL_contine_CAMERA = relatie intre HOTEL si CAMERA, indicand ce camere aparțin de un hotel. Relatia are cardinalitatea minima 1:0 (un hotel poate avea 0 sau mai multe camere si o camera trebuie sa apartina de exact un hotel) si cardinalitatea maxima 1:n (un hotel poate contine 0 sau mai multe camere si o camera apartine de exact un hotel).

HOTEL_se_afla_in_LOCATIE = relatie intre HOTEL si LOCATIE, indicand zona in care este amplasat hotelul. Relatia are cardinalitatea minima 0:1 (un hotel trebuie sa se afle in exact o locatie, dar o locatie poate sa nu contina niciun hotel) si cardinalitatea maxima n:1 (un hotel trebuie sa se afle in exact o locatie, dar o locatie poate sa contina mai multe hoteluri).

ZONA_DE_PARCARE_se_afla_in_LOCATIE = relatie intre ZONA_DE_PARCARE si LOCATIE, indicand zona in care este situata parcarea. Relatia are cardinalitatea minima 0:1 (o zona de parcare trebuie sa se afle in exact o locatie, dar o locatie poate sa nu contina nicio zona de parcare) si cardinalitatea maxima n:1 (o zona de parcare trebuie sa se afle in exact o locatie, dar o locatie poate sa contina mai multe zone de parcare).

RESTAURANT_se_afla_in_LOCATIE = relatie intre RESTAURANT si LOCATIE, indicand zona in care se afla restaurantul. Relatia are cardinalitatea minima 0:1 (un restaurant trebuie sa se afle in exact o locatie, dar o locatie poate sa nu contina niciun restaurant) si cardinalitatea maxima n:1 (un restaurant trebuie sa se afle in exact o locatie, dar o locatie poate sa contina mai multe restaurante).

ZONA_DE_PARCARE_apartine_de_HOTEL = relatie intre ZONA_DE_PARCARE si HOTEL ce indica faptul ca o parcare apartine de un anumit hotel. Relatia are cardinalitatea minima 0:1 (o parcare apartine de exact un hotel, dar un hotel poate sa nu dispuna de nicio parcare) si cardinalitatea maxima n:1 (o parcare apartine de exact un hotel, dar un hotel poate dispune de mai multe parcari).

ZONA_DE_PARCARE_detine_LOC_DE_PARCARE = relatie intre ZONA_DE_PARCARE si LOC_DE_PARCARE, reprezentand ce locuri de parcare se afla in cadrul unei zone de parcare. Relatia are cardinalitatea minima 1:0 (o zona de parcare poate avea 0 sau mai multe locuri de parcare, dar un loc apartine de exact o zona) si cardinalitatea maxima 1:n (o zona poate contina mai multe locuri de parcare, dar un loc apartine de exact o zona).

HOTEL_dispunе_de_RESTAURANT = relatie intre HOTEL si RESTAURANT, indicand faptul ca un hotel prezinta si restaurante unde clientii pot lua masa. Relatia are cardinalitatea minima 1:0 (un hotel poate sa nu aiba un restaurant, dar un restaurant apartine de exact un hotel) si cardinalitatea maxima 1:n (un hotel poate sa aiba mai multe restaurante, dar un restaurant apartine de exact un hotel).

RESTAURANT_serveste_FEL_DE_MANCARE = relatie de tip many-to-many intre RESTAURANT si FEL_DE_MANCARE, indicand ce feluri de mancare se afla in meniu unui restaurant. Relatia are cardinalitatea minima 0:0 (un restaurant poate servi 0 sau mai multe feluri de mancare, dar un fel de mancare poate sa nu fie servit la niciun restaurant) si cardinalitatea maxima n:n (un restaurant poate sa ofere mai multe feluri de mancare si un fel de mancare poate fi servit la mai multe restaurante).

PAZNIC_supravegheaza_ZONA_DE_PARCARE = relatie intre PAZNIC si ZONA_DE_PARCARE ce ilustreaza paznicii care se occupa de supravegherea parcarilor. Relatia are cardinalitatea minima 0:0 (un paznic poate sa nu supravegheze nicio zona de parcare si o parcare poate sa nu fie supravegheata de niciun paznic) si cardinalitatea maxima n:1 (un paznic poate sa supravegheze maxim o parcare, dar o parcare poate fi supravegheata de mai multi paznici).

RECEPTIONIST_lucreaza_HOTEL = relatie intre RECEPTIONIST si HOTEL, reprezentand receptionistii care lucreaza in cadrul hotelului. Relatia are cardinalitatea minima 0:0 (un receptionist poate sa nu lucreze in cadrul unui hotel si un hotel poate sa nu aiba receptionist) si cardinalitatea maxima n:1 (un receptionist poate lucra in maxim un hotel, dar intr-un hotel pot lucra mai multi receptionisti).

ADMINISTRATOR_lucreaza_HOTEL = relatie intre ADMINISTRATOR si HOTEL, reprezentand administratorii care lucreaza in cadrul hotelului. Relatia are cardinalitatea minima 0:0 (un administrator poate sa nu lucreze in cadrul unui hotel si un hotel poate sa nu aiba administrator) si cardinalitatea maxima n:1 (un administrator poate lucra in maxim un hotel, dar intr-un hotel pot lucra mai multi administratori).

CHELNER_lucreaza_RESTAURANT = relatie intre CHELNER si RESTAURANT, reprezentand chelnerii care lucreaza in cadrul unui restaurant. Relatia are cardinalitatea minima 0:0 (un chelner poate sa nu lucreze in cadrul unui restaurant si un restaurant poate sa nu aiba chelneri) si cardinalitatea maxima n:1 (un chelner poate lucra in maxim un restaurant, dar intr-un restaurant pot lucra mai multi chelneri).

BUCATAR_lucreaza_RESTAURANT = relatie intre BUCATAR si RESTAURANT, reprezentand bucatarii care lucreaza in cadrul unui restaurant. Relatia are cardinalitatea minima 0:0 (un bucatar poate sa nu lucreze in cadrul unui restaurant si un restaurant poate sa nu aiba bucatar) si cardinalitatea maxima n:1 (un bucatar poate lucra in maxim un restaurant, dar intr-un restaurant pot lucra mai multi bucatari).

FURNIZOR_semneaza_cu_RESTAURANT_mai_multe_CONTRACTE = relatie de tip 3, ce leaga entitatile FURNIZOR, RESTAURANT si CONTRACT. Relatia ilustreaza ce intelegeri au fost realizate intre restaurante si furnizori in vederea aprovizionarilor. Aceste intelegeri au fost realizate prin diverse contracte.

Cardinalitatea minima este 0:0:0 si cardinalitatea maxima este n:n:n.

ANGAJAT_ISA_PAZNIC = relatie speciala intre supraentitatea ANGAJAT si subentitatea PAZNIC, reprezentand daca un angajat ocupa sau nu functia de paznic. Relatia are cardinalitatea minima 1:0 (un angajat poate sa nu fie paznic, dar un paznic este sigur un angajat) si cardinalitate maxima 1:1 (un angajat este paznic, iar un paznic sigur este angajat).

ANGAJAT_ISA_RECEPTIONIST = relatie speciala intre supraentitatea ANGAJAT si subentitatea RECEPTIONIST, reprezentand daca un angajat ocupa sau nu functia de receptionist. Relatia are cardinalitatea minima 1:0 (un angajat poate sa nu fie receptionist, dar un receptionist este sigur un angajat) si cardinalitate maxima 1:1 (un angajat este receptionist, iar un receptionist sigur este angajat).

ANGAJAT_ISA_ADMINISTRATOR = relatie speciala intre supraentitatea ANGAJAT si subentitatea ADMINISTRATOR, reprezentand daca un angajat ocupa sau nu functia de administrator. Relatia are cardinalitatea minima 1:0 (un angajat poate sa nu fie administrator, dar un administrator este sigur un angajat) si cardinalitate maxima 1:1 (un angajat este administrator, iar un administrator sigur este angajat).

ANGAJAT_ISA_CHELNER = relatie speciala intre supraentitatea ANGAJAT si subentitatea CHELNER, reprezentand daca un angajat ocupa sau nu functia de chelner. Relatia are cardinalitatea minima 1:0 (un angajat poate sa nu fie chelner, dar un chelner este sigur un angajat) si cardinalitate maxima 1:1 (un angajat este chelner, iar un chelner sigur este angajat).

ANGAJAT_ISA_BUCATAR = relatie speciala intre supraentitatea ANGAJAT si subentitatea BUCATAR, reprezentand daca un angajat ocupa sau nu functia de bucatar. Relatia are cardinalitatea minima 1:0 (un angajat poate sa nu fie bucatar, dar un bucatar este sigur un angajat) si cardinalitate maxima 1:1 (un angajat este bucatar, iar un bucatar sigur este angajat).

Descrierea atributelor, a tipurilor de date si a eventualelor constrangeri

Entitatea independenta **HOTEL** are drept atribute:

cod_hotel = variabila de tip intreg, de lungime maxima 10, care reprezinta codul unui hotel **NUMBER(10)**. Este cheie primara.

denumire = variabila de tip caracter, de lungime maxima 50, care reprezinta denumirea hotelului **VARCHAR2(50)**. Nu poate fi null.

numar_stele = variabila de tip intreg, de lungime maxima 1, care reprezinta numarul de stele al hotelului **NUMBER(1)**. Poate fi null.

an_infiintare = variabila de tip intreg, de lungime maxima 4, care reprezinta anul in care a fost infiintat hotelul **NUMBER(4)**. Poate fi null.

cod_locatie = variabila de tip intreg, de lungime maxima 10, care reprezinta codul locatiei in care se afla hotelul **NUMBER(10)**. Atributul trebuie sa corespunda la o valoare a cheii primare din tabelul LOCATIE. Nu poate fi null.

Entitatea dependenta **CAMERA** are drept atribute:

cod_camera = variabila de tip intreg, de lungime maxima 10, care reprezinta codul unei camere **NUMBER(10)**. Nu poate fi null.

cod_hotel = variabila de tip intreg, de lungime maxima 10, care reprezinta codul hotelului in cadrul caruia se afla camera **NUMBER(10)**. Atributul trebuie sa corespunda la o valoare a cheii primare din tabelul HOTEL, nu poate fi null.

Atributele cod_camera si cod_hotel formeaza cheia primara compusa a entitatii CAMERA.

numar = variabila de tip intreg, de lungime maxima 10, care reprezinta numarul asociat camerei **NUMBER(10)**. Nu poate fi null si trebuie sa fie strict mai mare decat 0.

etaj = variabila de tip intreg, de lungime maxim 3, care reprezinta etajul la care se afla camera **NUMBER(3)**. Valoarea implicita este 0 (parter) si nu poate lua valori negative. Nu poate fi null.

Entitatea independenta **LOCATIE** are atributele:

cod_locatie = variabila de tip intreg, de lungime maxima 10, care reprezinta codul unei locatii **NUMBER(10)**. Este cheie primara.

adresa = variabila de tip caracter, de lungime maxima 50, care reprezinta adresa la care se afla locatia **VARCHAR2(50)**. Nu poate fi null.

Entitatea independenta **FACILITATE** are atributele:

cod_facilitate = variabila de tip intreg, de lungime maxima 10, care reprezinta codul unei facilitati **NUMBER(10)**. Este cheie primara.

denumire = variabila de tip caracter, de lungime maxima 50, care reprezinta denumirea facilitatii **VARCHAR2(50)**. Nu poate fi null.

Entitatea independenta **CLIENT** are atributele:

cod_client = variabila de tip intreg, de lungime maxima 10, care reprezinta codul unui client **NUMBER(10)**. Este cheie primara.
nume = variabila de tip caracter, de lungime maxima 50, care reprezinta numele de familie al clientului **VARCHAR2(50)**. Nu poate fi null.
prenume = variabila de tip caracter, de lungime maxima 50, care reprezinta prenumele clientului **VARCHAR2(50)**. Nu poate fi null.
CNP = variabila de tip caracter, de lungime maxima 13, care reprezinta CNP-ul clientului **VARCHAR2(13)**. Poate fi null.
numar_telefon = variabila de tip caracter, de lungime maxima 50, care reprezinta numarul de telefon al clientului **VARCHAR2(50)**. Nu poate fi null.
adresa_mail = variabila de tip caracter, de lungime maxima 50, care reprezinta adresa de mail a clientului **VARCHAR2(50)**. Poate fi null.

Relatia **CAMERA_presinta_FACILITATE** are atributele:

cod_camera = variabila de tip intreg, de lungime maxima 10, care reprezinta codul camerei unde se aplica facilitatea **NUMBER(10)**. Nu poate fi null.
cod_hotel = variabila de tip intreg, de lungime maxima 10, care reprezinta codul hotelului in care se afla camera cu facilitatea **NUMBER(10)**. Nu poate fi null.
Cele doua atribute (cod_camera, cod_hotel) trebuie sa corespunda la o valoare a cheii primare compuse din tabelul **CAMERA**.
cod_facilitate = variabila de tip intreg, de lungime maxima 10, care reprezinta codul facilitatii care se aplica pentru camera **NUMBER(10)**. Atributul trebuie sa corespunda la o valoare a cheii primare din tabelul **FACILITATE**, nu poate fi null.
Atributele cod_camera, cod_hotel si cod_facilitate formeaza cheia primara compusa a relatiei **CAMERA_presinta_FACILITATE**.

Relatia **CLIENT_rezerva_CAMERA** are ca atribute:

cod_client = variabila de tip intreg, de lungime maxima 10, care reprezinta codul clientului care rezerva camera **NUMBER(10)**. Atributul trebuie sa corespunda la o valoare a cheii primare din tabelul **CLIENT**, nu poate fi null.
cod_camera = variabila de tip intreg, de lungime maxima 10, care reprezinta codul camerei ce este rezervata **NUMBER(10)**. Nu poate fi null.
cod_hotel = variabila de tip intreg, de lungime maxima 10, care reprezinta codul hotelului unde are loc rezervarea **NUMBER(10)**. Nu poate fi null.
Cele doua atribute (cod_camera, cod_hotel) trebuie sa corespunda la o valoare a cheii primare compuse din tabelul **CAMERA**.
data_inceput = variabila de tip data calendaristica, reprezentand data la care a fost facuta rezervarea **DATE**. Valoarea implicita este data cand intrarea este adaugata in tabel. Nu poate fi null.
data_sfarsit = variabila de tip data calendaristica, reprezentand data la care expira rezervarea facuta **DATE**. Nu e null si trebuie sa fie mai mare sau egala cu data_inceput.
Atributele cod_client, cod_camera, cod_hotel, data_inceput, data_sfarsit formeaza cheia primara compusa a relatiei **CLIENT_rezerva_CAMERA**.

Entitatea independenta **ZONA_DE_PARCARE** are ca atribute:

cod_zona_de_parcare = variabila de tip intreg, de lungime maxima 10, care reprezinta codul zonei de parcare **NUMBER(10)**. Este cheie primara.
cod_hotel = variabila de tip intreg, de lungime maxima 10, care reprezinta codul hotelului de care apartine zona de parcare **NUMBER(10)**. Atributul trebuie sa corespunda la o valoare a cheii primare din tabelul **HOTEL**, nu poate fi null.
cod_locatie = variabila de tip intreg, de lungime maxima 10, care reprezinta codul locatiei in care se afla zona de parcare **NUMBER(10)**. Atributul trebuie sa corespunda la o valoare a cheii primare din tabelul **LOCATIE**, nu poate fi null.

Entitatea dependenta **LOC_DE_PARCARE** are atributele:

cod_loc_de_parcare = variabila de tip intreg, de lungime maxima 10, care reprezinta codul locului de parcare **NUMBER(10)**. Nu poate fi null.
cod_zona_de_parcare = variabila de tip intreg, de lungime maxima 10, care reprezinta codul zonei de parcare de care apartine locul **NUMBER(10)**. Atributul trebuie sa corespunda la o valoare a cheii primare din tabelul **ZONA_DE_PARCARE**, nu poate fi null.
Atributele cod_loc_deparcare si cod_zona_de_parcare formeaza cheia primara compusa a entitatii **LOC_DE_PARCARE**.
numar = variabila de tip intreg, de lungime maxima 10, care reprezinta numarul asociat locului de parcare **NUMBER(10)**. Nu poate fi null si trebuie sa fie strict pozitiv.

Entitatea independenta **FURNIZOR** are atributele:

cod_furnizor = variabila de tip intreg, de lungime maxima 10, care reprezinta codul furnizorului **NUMBER(10)**. Este cheie primara.
nume = variabila de tip caracter, de lungime maxima 50, care reprezinta numele furnizorului **VARCHAR2(50)**. Nu poate fi null.

Entitatea independenta **CONTRACT** are atributele:

cod_contract = variabila de tip intreg, de lungime maxima 10, care reprezinta codul contractului **NUMBER(10)**. Este cheie primara.
data_inceput = variabila de tip data calendaristica, reprezentand data de inceput a contractului **DATE**. Valoarea implicita este data cand intrarea este adaugata in tabel. Nu poate fi null.
data_sfarsit = variabila de tip data calendaristica, reprezentand data de finalizare a contractului **DATE**. Nu poate fi null si trebuie sa fie mai mare sau egala cu data_inceput.

Entitatea independenta **FEL_DE_MANCARE** are atributele:

cod_fel_de_mancare = variabila de tip intreg, de lungime maxima 10, care reprezinta codul felului de mancare **NUMBER(10)**. Este cheie primara.
denumire = variabila de tip caracter, de lungime maxima 50, care reprezinta denumirea felului de mancare **VARCHAR2(50)**. Nu poate fi null.
pret = variabila de tip intreg, de lungime maxima 5, care reprezinta pretul felului de mancare **NUMBER(5)**. Nu poate fi null.

Relatia **RESTAURANT_serveste_FEL_DE_MANCARE** are ca atribute:

cod_restaurant = variabila de tip intreg, de lungime maxima 10, care reprezinta codul restaurantului in care este servit felul de mancare **NUMBER(10)**. Atributul trebuie sa corespunda la o valoare a cheii primare din tabelul **RESTAURANT**, nu poate fi null.
cod_fel_de_mancare = variabila de tip intreg, de lungime maxima 10, care reprezinta codul felului de mancare ce este servit **NUMBER(10)**. Atributul trebuie sa corespunda la o valoare a cheii primare din tabelul **FEL_DE_MANCARE**, nu poate fi null.
Atributele cod_restaurant si cod_fel_de_mancare formeaza cheia primara compusa a relatiei **RESTAURANT_serveste_FEL_DE_MANCARE**.

Relatia **FURNIZOR_semneaza_cu_RESTAURANT_mai_multe_CONTRACTE** are atributele:

cod_restaurant = variabila de tip intreg, de lungime maxima 10, care reprezinta codul restaurantului implicat in contract **NUMBER(10)**. Atributul trebuie sa corespunda la o valoare a cheii primare din tabelul **RESTAURANT**, nu poate fi null.
cod_furnizor = variabila de tip intreg, de lungime maxima 10, care reprezinta codul furnizorului implicat in contract **NUMBER(10)**. Atributul trebuie sa corespunda la o valoare a cheii primare din tabelul **FURNIZOR**, nu poate fi null.
cod_contract = variabila de tip intreg, de lungime maxima 10, care reprezinta codul contractului semnat intre furnizor si restaurant **NUMBER(10)**. Atributul trebuie sa corespunda la o valoare a cheii primare din tabelul **CONTRACT**, nu poate fi null.
Atributele cod_restaurant, cod_furnizor si cod_contract formeaza cheia primara compusa a relatiei **FURNIZOR_semneaza_cu_RESTAURANT_mai_multe_CONTRACTE**.

Entitatea independenta **RESTAURANT** are atributele:

cod_restaurant = variabila de tip intreg, de lungime maxima 10, care reprezinta codul restaurantului **NUMBER(10)**. Este cheie primara.
cod_hotel = variabila de tip intreg, de lungime maxima 10, care reprezinta codul hotelului care dispune de acest restaurant **NUMBER(10)**. Atributul trebuie sa corespunda la o valoare a cheii primare din tabelul HOTEL, nu poate fi null.
cod_locatie = variabila de tip intreg, de lungime maxima 10, care reprezinta codul locatiei unde se afla restaurantul **NUMBER(10)**. Atributul trebuie sa corespunda la o valoare a cheii primare din tabelul LOCATIE, nu poate fi null.
denumire = variabila de tip caracter, de lungime maxima 50, care reprezinta denumirea restaurantului **VARCHAR2(50)**. Nu poate fi null.
numar_stele = variabila de tip intreg, de lungime maxima 1, care reprezinta numarul de stele asociat restaurantului **NUMBER(1)**. Poate fi null.
an_infiintare = variabila de tip intreg, de lungime maxima 4, care reprezinta anul in care a fost infiintat restaurantul **NUMBER(4)**. Poate fi null.

Entitatea independenta **ANGAJAT** are atributele:

cod_angajat = variabila de tip intreg, de lungime maxima 10, care reprezinta codul angajatului **NUMBER(10)**. Este cheie primara.
nume = variabila de tip caracter, de lungime maxima 50, care reprezinta numele de familie al angajatului **VARCHAR2(50)**. Nu poate fi null.
prenume = variabila de tip caracter, de lungime maxima 50, care reprezinta prenumele angajatului **VARCHAR2(50)**. Nu poate fi null.
CNP = variabila de tip caracter, de lungime maxima 13, care reprezinta CNP-ul angajatului **VARCHAR2(13)**. Nu poate fi null.
numar_telefon = variabila de tip caracter, de lungime maxima 50, care reprezinta numarul de telefon al angajatului **VARCHAR2(50)**. Nu poate fi null.
adresa_mail = variabila de tip caracter, de lungime maxima 50, care reprezinta adresa de mail a angajatului **VARCHAR2(50)**. Poate fi null.
salariu = variabila de tip intreg, de lungime maxima 10, care reprezinta salariul angajatului **NUMBER(10)**. Nu poate fi null.

Subentitatea **PAZNIC** are ca atribute:

cod_angajat = variabila de tip intreg, de lungime maxima 10, care reprezinta codul paznicului **NUMBER(10)**. Este cheie primara si trebuie sa corespunda la o valoare a cheii primare din tabelul ANGAJAT.
cod_zona_de_parcare = variabila de tip intreg, de lungime maxima 10, care reprezinta codul zonei de parcare pe care paznicul trebuie sa o supravegheze. **NUMBER(10)**. Atributul trebuie sa corespunda la o valoare a cheii primare din tabelul ZONA_DE_PARCARE, dar poate fi si null.

Subentitatea **RECEPTIONIST** are atributele:

cod_angajat = variabila de tip intreg, de lungime maxima 10, care reprezinta codul receptionistului **NUMBER(10)**. Este cheie primara si trebuie sa corespunda la o valoare a cheii primare din tabelul ANGAJAT.
cod_hotel = variabila de tip intreg, de lungime maxima 10, care reprezinta codul hotelului unde receptionistul lucreaza **NUMBER(10)**.
Atributul trebuie sa corespunda la o valoare a cheii primare din tabelul HOTEL, dar poate fi si null.

Subentitatea **ADMINISTRATOR** are ca atribute:

cod_angajat = variabila de tip intreg, de lungime maxima 10, care reprezinta codul administratorului **NUMBER(10)**. Este cheie primara si trebuie sa corespunda la o valoare a cheii primare din tabelul ANGAJAT.
cod_hotel = variabila de tip intreg, de lungime maxima 10, care reprezinta codul hotelului unde administratorul lucreaza **NUMBER(10)**.
Atributul trebuie sa corespunda la o valoare a cheii primare din tabelul HOTEL, dar poate fi si null.

Subentitatea **CHELNER** are drept atribute:

cod_angajat = variabila de tip intreg, de lungime maxima 10, care reprezinta codul chelnerului **NUMBER(10)**. Este cheie primara si trebuie sa corespunda la o valoare a cheii primare din tabelul ANGAJAT.
cod_restaurant = variabila de tip intreg, de lungime maxima 10, care reprezinta codul restaurantului unde chelnerul lucreaza **NUMBER(10)**.
Atributul trebuie sa corespunda la o valoare a cheii primare din tabelul RESTAURANT, dar poate fi si null.

Subentitatea **BUCATAR** are atributele:

cod_angajat = variabila de tip intreg, de lungime maxima 10, care reprezinta codul bucatarului **NUMBER(10)**. Este cheie primara si trebuie sa corespunda la o valoare a cheii primare din tabelul ANGAJAT.
cod_restaurant = variabila de tip intreg, de lungime maxima 10, care reprezinta codul restaurantului unde bucatarul lucreaza **NUMBER(10)**.
Atributul trebuie sa corespunda la o valoare a cheii primare din tabelul RESTAURANT, dar poate fi si null.

Diagrama Entitate-Relatie

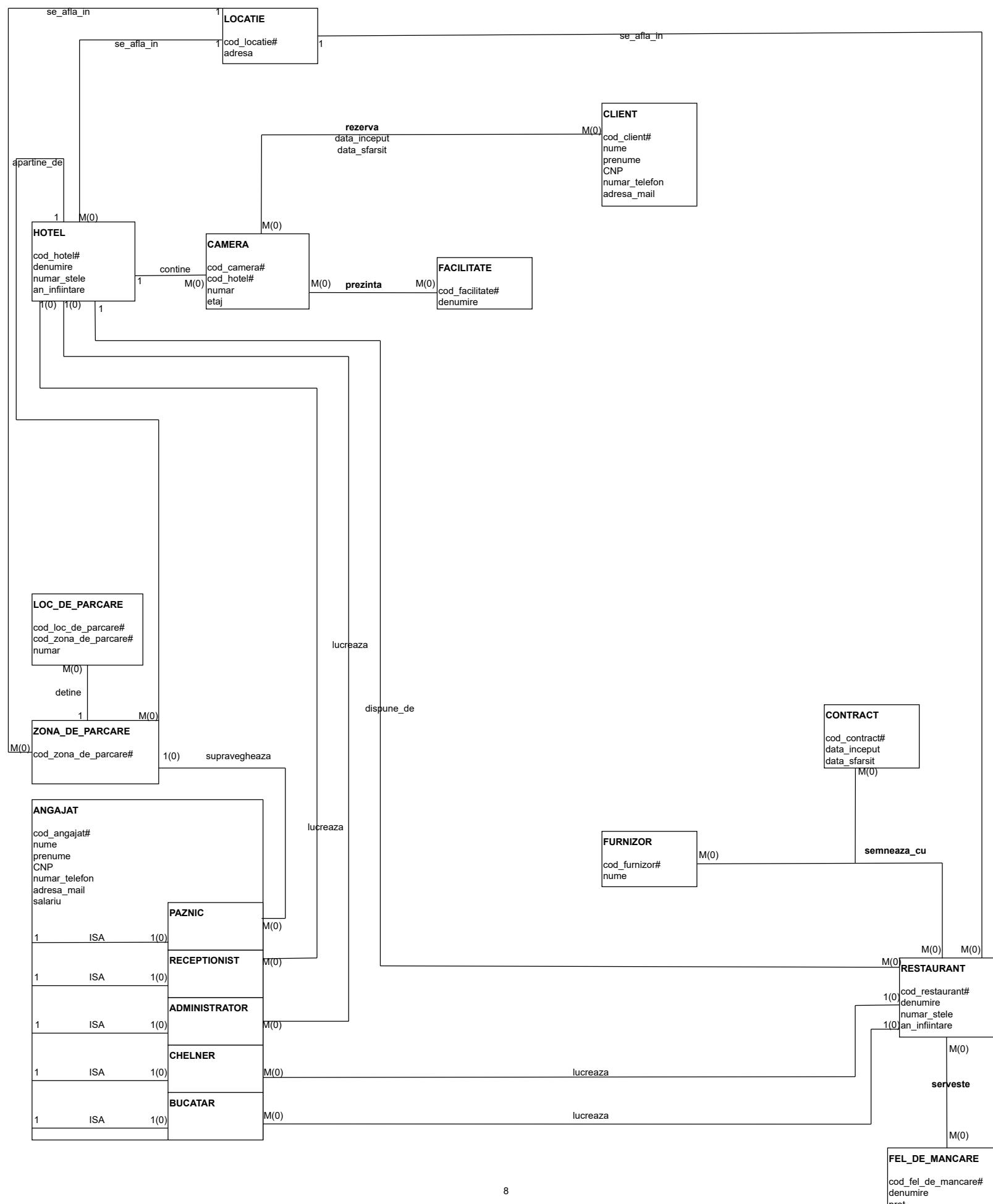
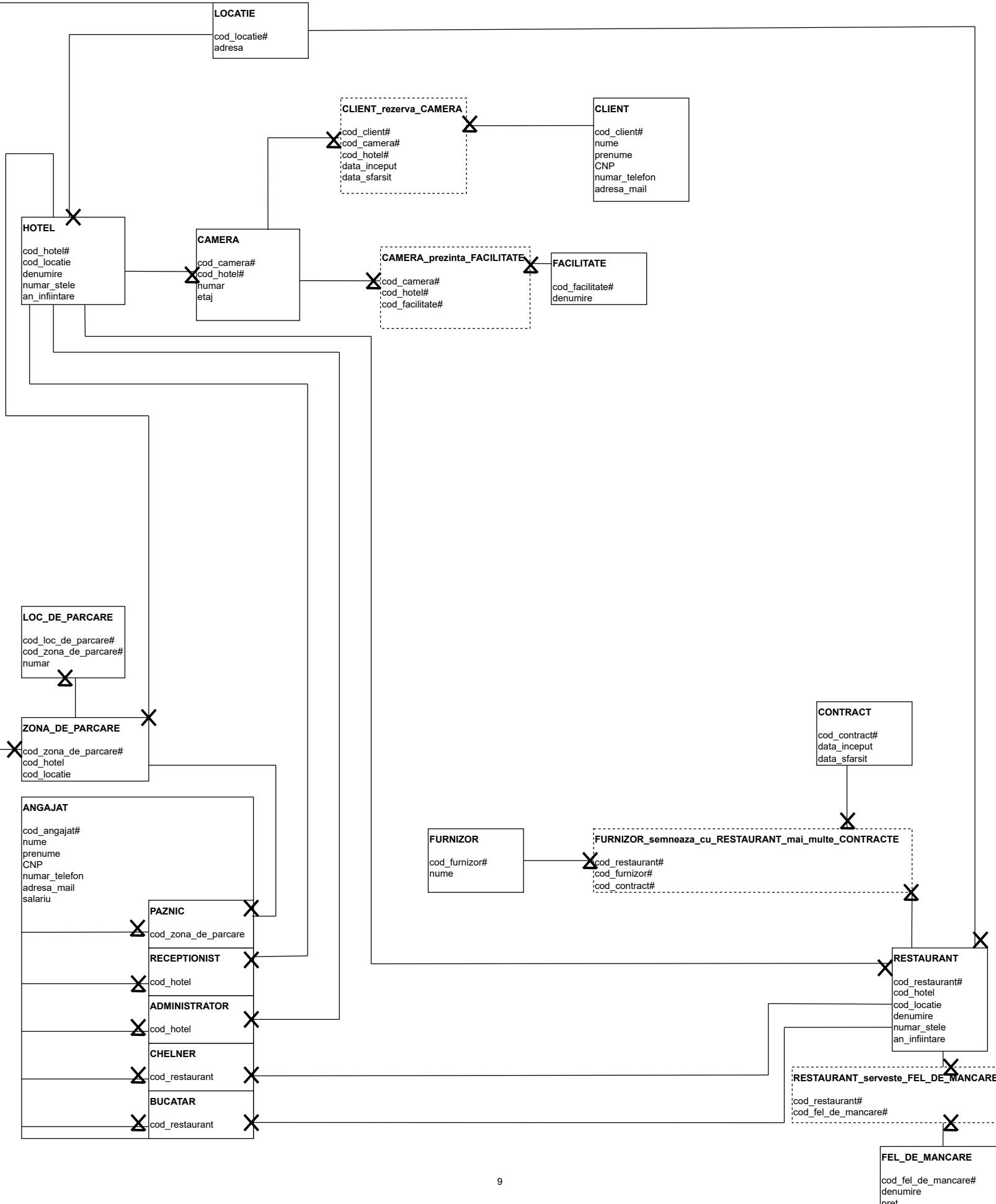


Diagrama Conceptuala



Reprezentarea schemelor relationale corespunzatoare diagramei conceptuale

HOTEL(cod_hotel#, cod_locatie, denumire, numar_stele, an_infiintare)
 CAMERA(cod_camera#, cod_hotel#, numar, etaj)
 LOCATIE(cod_locatie#, adresa)
 CLIENT(cod_client#, nume, prenume, CNP, numar_telefon, adresa_mail)
 FACILITATE(cod_facilitate#, denumire)
 CLIENT_rezerva_CAMERA(cod_client#, cod_camera#, cod_hotel#, data_inceput, data_sfarsit)
 CAMERA_presinta_FACILITATE(cod_camera#, cod_hotel#, cod_facilitate#)
 ZONA_DE_PARCARE(cod_zona_de_parcare#, cod_hotel, cod_locatie)
 LOC_DE_PARCARE(cod_loc_de_parcare#, cod_zona_de_parcare#, numar)
 CONTRACT(cod_contract#, data_inceput, data_sfarsit)
 FURNIZOR(cod_furnizor#, nume)
 FURNIZOR_semneaza_cu_RESTAURANT_mai_multe_CONTRACTE(cod_restaurant#, cod_furnizor#, cod_contract#)
 RESTAURANT(cod_restaurant#, cod_hotel, cod_locatie, denumire, numar_stele, an_infiintare)
 RESTAURANT_serveste_FEL_DE_MANCARE(cod_restaurant#, cod_fel_de_mancare#)
 FEL_DE_MANCARE(cod_fel_de_mancare#, denumire, pret)
 ANGAJAT(cod_angajat#, nume, prenume, CNP, numar_telefon, adresa_mail, salariu)
 PAZNIC(cod_angajat#, cod_zona_de_parcare)
 RECEPTIONIST(cod_angajat#, cod_hotel)
 ADMINISTRATOR(cod_angajat#, cod_hotel)
 CHELNER(cod_angajat#, cod_restaurant)
 BUCATAR(cod_angajat#, cod_restaurant)

Realizarea normalizarii pana la forma normala 3

Forma normala 1 (FN1)

O relatie este in prima forma normala daca fiecarui atribut care o compune ii corespunde o valoare indivizibila (atomica).

Exemplu:

CLIENT (Non FN1)

cod_client#	nume_si_prenume	CNP	numar_telefon	adresa_mail
5	Bogdan Mircea	0000011111222	0777777777	NULL
6	Georgescu Gheorghe	0010011111222	0777377777	georgescu_gheorghe@mail.ro
8	Alexandru Sebastian	1313131313131	0744333432	NULL

Relatia CLIENT(cod_client#, nume_si_prenume, CNP, numar_telefon, adresa_mail) nu este in forma normala 1, deoarece atributul nume_si_prenume nu este atomic, acesta putand fi separat in doua attribute: nume si prenume. Relatia devine CLIENT(cod_client#, nume, prenume, CNP, numar_telefon, adresa_mail).

CLIENT (FN1)

cod_client#	nume	prenume	CNP	numar_telefon	adresa_mail
5	Bogdan	Mircea	0000011111222	0777777777	NULL
6	Georgescu	Gheorghe	0010011111222	0777377777	georgescu_gheorghe@mail.ro
8	Alexandru	Sebastian	1313131313131	0744333432	NULL

Forma normala 2 (FN2)

O relatie R este in a doua forma normala daca si numai daca relatia R este in FN1 si fiecare atribut care nu este cheie (nu participa la cheia primara) este dependent de intreaga cheie primara.

FN2 interzice manifestarea unor dependente functionale pariale in cadrul relatiei R.

Exemplu:

RESTAURANT_serveste_FEL_DE_MANCARE (Non FN2)

cod_restaurant#	cod_fel_de_mancare#	denumire	pret
1	1	Ciorba De Legume	11
1	6	Piure	12
8	6	Piure	12

Un restaurant poate servi mai multe feluri de mancare si un fel de mancare are o denumire si un pret.

Relatia RESTAURANT_serveste_FEL_DE_MANCARE(cod_restaurant#, cod_fel_de_mancare#, denumire, pret) este in FN1, deoarece fiecare atribut este indivizibil. Relatia nu se afla in FN2, deoarece atributele care nu sunt chei primare, denumire si pret, nu sunt dependente de intreaga cheie primara, formata din cod_restaurant# si cod_fel_de_mancare#. Se observa faptul ca denumire si pret depind doar de felul de mancare servit, deci doar de partea cod_fel_de_mancare# din cheia primara.

Astfel avem: {cod_fel_de_mancare#} \rightarrow {denumire, pret} si {cod_restaurant#, cod_fel_de_mancare#} \rightarrow {} (multimea vida)

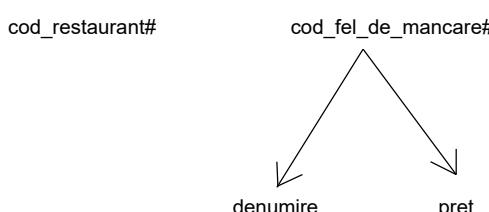
Aplicam regula Casey-Delobel pentru FN2:

Fie relatia R(K1, K2, X, Y), unde K1 si K2 definesc cheia primara, iar X si Y sunt multimi de atribute, astfel incat $K1 \rightarrow X$.

Din cauza dependentei funktionale $K1 \rightarrow X$ care arata ca R nu este in FN2, se inlocuieste R (fara pierdere de informatie) cu doua proiectii: R1(K1, K2, Y) si R2(K1, X).

In acest caz:

R = RESTAURANT_serveste_FEL_DE_MANCARE
 K1 = cod_fel_de_mancare#
 K2 = cod_restaurant#
 X = {denumire, pret}
 Y = multimea vida



Transformare in FN2:

RESTAURANT_serveste_FEL_DE_MANCARE (FN2)	
cod_restaurant#	cod_fel_de_mancare#
1	1
1	6
8	6

FEL_DE_MANCARE (FN2)		
cod_fel_de_mancare#	denumire	pret
1	Ciorba De Legume	11
6	Piure	12

Relatia FEL_DE_MANCARE(cod_fel_de_mancare#, denumire, pret) are toate atributele atomice, aflandu-se astfel in FN1. De asemenea, toate atributele depend de intreaga cheie primara, cod_fel_de_mancare# si relatia se afla astfel in FN2.

Forma normala 3 (FN3)

O relatie R este in a treia forma normala daca si numai daca relatia R este in FN2 si fiecare atribut care nu este cheie (nu participa la o cheie) depinde direct de cheia primara.

Fie R o relatie, X o submultime de atribute ale lui R si A un atribut al relatiei R. A este dependent tranzitiv de X daca exista Y astfel incat $X \rightarrow Y$ si $Y \rightarrow A$.

Exemplu:

ZONA_DE_PARCARE (Non FN3)

cod_zona_de_parcare#	cod_hotel	cod_locatie	denumire
1	1	1	Continental
7	6	5	Mures
8	1	5	Mures

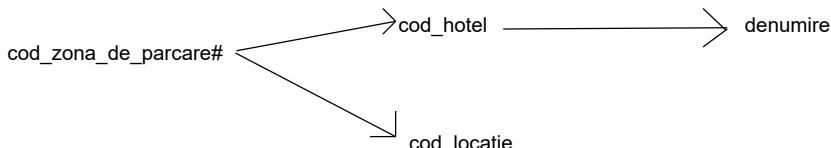
Relatia ZONA_DE_PARCARE(cod_zona_de_parcare#, cod_hotel, cod_locatie, denumire) este in FN2, deoarece este in FN1(fiecare atribut este atomic) si orice atribut care nu este cheie primara este dependent de intreaga cheie primara cod_zona_de_parcare#. Relatia nu este insa in FN3, deoarece denumirea hotelului (atributul denumire) depinde tranzitiv de cheia primara cod_zona_de_parcare# prin intermediul atributului cod_hotel. O relatie este in FN3 daca fiecare atribut care nu este cheie (nu participa la o cheie) nu este dependent tranzitiv de nicio cheie a relatiei ZONA_DE_PARCARE.

Astfel avem: {cod_zona_de_parcare#} \rightarrow {cod_hotel, cod_locatie} si {cod_hotel} \rightarrow {denumire}.
Dependenta tranzitiva este {cod_zona_de_parcare#} \rightarrow {cod_hotel} \rightarrow {denumire}.

Pentru a aduce relatia ZONA_DE_PARCARE in FN3 se aplica regula Casey-Delobel. Relatia se descompune, prin eliminarea dependentelor functionale tranzitive, in proiectiile: ZONA_DE_PARCARE(cod_zona_de_parcare#, cod_hotel, cod_locatie) si HOTEL(cod_hotel#, denumire).

Aplicarea regulii Casey-Delobel pentru FN3: Fie relatia R(K, X1, X2, X3), unde atributul X2 depinde tranzitiv de K, iar K este cheia primara a lui R. Presupunem ca $K \rightarrow X_1 \rightarrow X_2$. Din cauza dependentei funktionale $X_1 \rightarrow X_2$ care arata ca R nu este in FN3, se inlocuieste R (fara pierdere de informatie) prin doua proiectii $R1(K, X_1, X_3)$ si $R2(X_1, X_2)$.

In acest caz: R = ZONA_DE_PARCARE, K = cod_zona_de_parcare#, X2 = denumire, X1 = cod_hotel si X3 = cod_locatie.



Transformare in FN3:

ZONA_DE_PARCARE (FN3)		
cod_zona_de_parcare#	cod_hotel	cod_locatie
1	1	1
7	6	5
8	1	5

HOTEL (FN3)	
cod_hotel#	denumire
1	Continental
5	Mures

Relatia HOTEL(cod_hotel#, denumire) este in FN3, deoarece fiecare atribut este atomic (FN1), fiecare atribut depinde de intreaga cheie primara (FN2) si fiecare atribut depinde doar de cheia primara cod_hotel# (FN3).

Realizarea normalizarii BCNF, FN4 si FN5

Forma normala Boyce-Codd (BCNF)

Determinantul este un atribut sau o multime de atribute neredundante, care constituie un identificator unic pentru alt atribut sau alta multime de atribute ale unei relatiile date.

O relatie R este in forma normala Boyce-Codd daca si numai daca fiecare determinant este o cheie candidat (daca si numai daca pentru orice dependenta functionala totala $X \rightarrow A$, X este o cheie candidat a lui R).

Din regula Casey-Delobel pentru $R(K1\#, K2\#, X)$, presupunand ca exista dependenta: $X \rightarrow K2$ (X determina functional pe $K2$), se va obtine $R1(K1\#, X)$ si $R2(X\#, K2)$, unde X este cheia candidat care in acest caz devine cheie primara.

Exemplu:

Presupunem ca relatia $CLIENT_rezerva_CAMERA(cod_client\#, cod_camera\#, cod_hotel\#, data_inceput, data_sfarsit)$ mai contine atributul $cod_rezervare$, care memoreaza un identificator unic pentru toate rezervarile realizeate in cadrul lantului de hoteluri.
Avem acum relatia $CLIENT_rezerva_CAMERA(cod_client\#, cod_camera\#, cod_hotel\#, data_inceput, data_sfarsit, cod_rezervare)$.

CLIENT_rezerva_CAMERA (Non BCNF)

cod_client#	cod_camera#	cod_hotel#	data_inceput	data_sfarsit	cod_rezervare
1	1	1	02-09-2023	08-09-2023	1
2	2	1	07-05-2023	13-05-2023	2
3	3	1	26-04-2023	02-05-2023	3

Intre atributele relatiei exista dependentele:

$\{cod_client\#, cod_camera\#, cod_hotel\#\} \rightarrow \{data_inceput, data_sfarsit, cod_rezervare\}$,
 $\{cod_rezervare\} \rightarrow \{cod_client\#}$ (se observa ca atributul $cod_rezervare$ este o cheie candidat, cheia $cod_client\#$ depinzand de acest atribut, deoarece o rezervare apartine exact unui client).

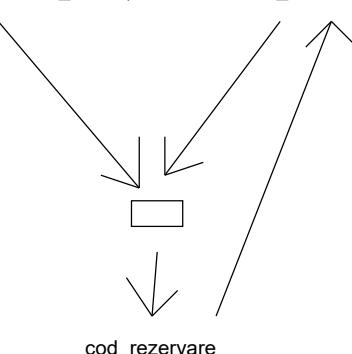
Se aplica regula Casey-Delobel si se aduce relatia in BCNF:

$CLIENT_rezerva_CAMERA(cod_camera\#, cod_hotel\#, data_inceput, data_sfarsit, cod_rezervare)$; ($cod_rezervare$ este cheie externa)
 $REZERVARE(cod_rezervare\#, cod_client)$; ($cod_rezervare\#$ fiind cheie candidat devine cheie primara).

Regula Casey-Delobel s-a aplicat pentru:

$R = CLIENT_rezerva_CAMERA$
 $K1 = (cod_camera\#, cod_hotel\#)$
 $K2 = cod_client\#$
 $X = cod_rezervare$

($cod_camera\#, cod_hotel\#$) $cod_client\#$



Transformare in BCNF:

CLIENT_rezerva_CAMERA (BCNF)

cod_camera#	cod_hotel#	data_inceput	data_sfarsit	cod_rezervare
1	1	02-09-2023	08-09-2023	1
2	1	07-05-2023	13-05-2023	2
3	1	26-04-2023	02-05-2023	3

REZERVARE (BCNF)

cod_rezervare#	cod_client
1	1
2	2
3	3

Forma normala 4 (FN4)

FN4 elimina redundantele datorate relatiilor many-to-many (dependentelor multiple).

O relatie R este in a patra forma normala daca si numai daca relatia este in BCNF si nu contine relatii m:n independente.

Exemplu:

Presupunem ca relatia RESTAURANT_serveste_FEL_DE_MANCARE(cod_restaurant#, cod_fel_de_mancare#) ar contine un al treilea atribut, cod_modalitate_de_servire#, ce ar face parte din cheia primara. Acest atribut ar fi si cheie externa catre alta relatie MODALITATE_DE_SERVIRE(cod_modalitate_de_servire#, descriere), care ar specifica modul in care poate fi servit felul de mancare (exemplu in cadrul restaurantului, la pachet sau alte modalitati) prin intermediul atributului descriere.

RESTAURANT_serveste_FEL_DE_MANCARE (Non FN4)

cod_restaurant#	cod_fel_de_mancare#	cod_modalitate_de_servire#
1	1	1
1	2	2
1	1	2

Intre atributele relatiei exista multidependentele:
 cod_fel_de_mancare# ->-> cod_restaurant#
 cod_fel_de_mancare# ->-> cod_modalitate_de_servire#

Relatia RESTAURANT_serveste_FEL_DE_MANCARE este in BCNF si poate fi adusa in FN4, descompunand prin proiectie in doua relatii:

RESTAURANT_serveste_FEL_DE_MANCARE_1(cod_restaurant#, cod_fel_de_mancare#);
 RESTAURANT_serveste_FEL_DE_MANCARE_2(cod_fel_de_mancare#, cod_modalitate_de_servire#);

RESTAURANT_serveste_FEL_DE_MANCARE = JOIN(RESTAURANT_serveste_FEL_DE_MANCARE_1, RESTAURANT_serveste_FEL_DE_MANCARE_2);

Transformare in FN4:

RESTAURANT_serveste_FEL_DE_MANCARE_1 (FN4)

cod_restaurant#	cod_fel_de_mancare#
1	1
1	2

RESTAURANT_serveste_FEL_DE_MANCARE_2 (FN4)

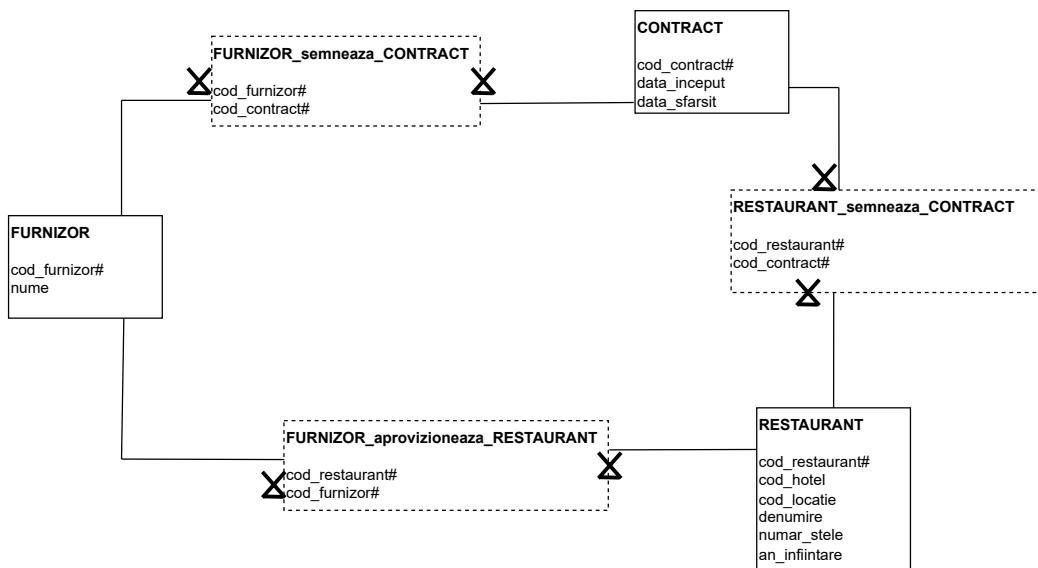
cod_fel_de_mancare#	cod_modalitate_de_servire#
1	1
2	2
1	2

Forma normala 5 (FN5)

FN5 isi propune eliminarea redundantei care apar in relatii many-to-many dependente.
 O relatie R este in forma normala 5 daca si numai daca relatia este in FN4 si nu contine dependente ciclice.

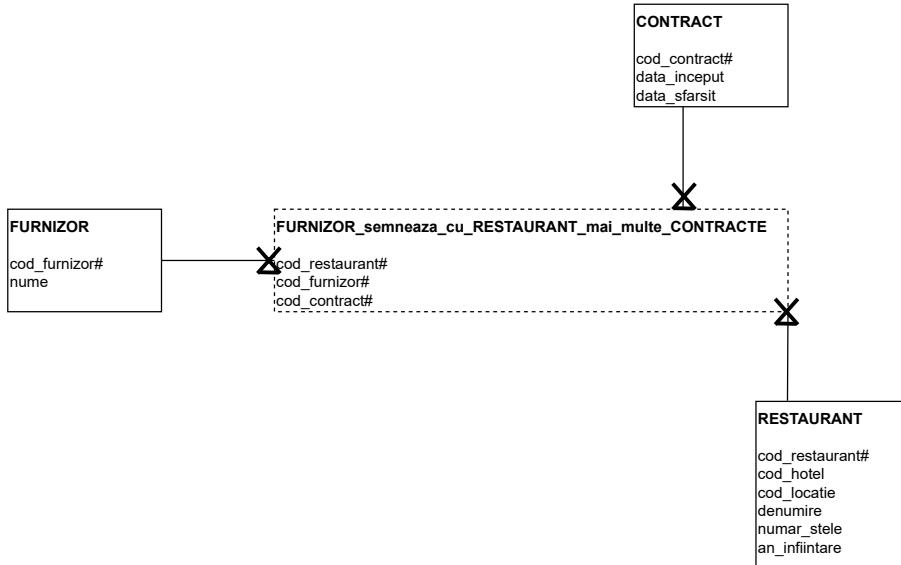
Exemplu:

Non FN5



Transformare in FN5:

FURNIZOR_semneaza_cu_RESTAURANT_mai_multe_CONTRACTE (FN5)



Prima diagrama nu este in FN5, deoarece are 3 relatii de tip 2 ciclice.

Pentru aducerea acesteia in FN5 vom elibera JOIN dependentele.

FURNIZOR_semneaza_cu_RESTAURANT_mai_multe_CONTRACTE =
JOIN (FURNIZOR_aprovizioneaza_RESTAURANT, RESTAURANT_semneaza_CONTRACT, FURNIZOR_semneaza_CONTRACT);

Aplicarea denormalizarii si justificarea necesitatii acestia

Sa presupunem ca relatia FEL_DE_MANCARE(cod_fel_de_mancare#, denumire, pret) ar avea in loc de atributul pret un alt atribut cod_pret, care ar fi cheie externa catre o relatie PRET(cod_pret#, valoare) ce ar memora toate preturile distincte intalnite la toate felurile de mancare.

In aceasta situatie denormalizarea ar inseamna eliminarea relatiei PRET(cod_pret#, valoare) si punerea atributului pret direct in relatia FEL_DE_MANCARE, in locul cheii externe cod_pret.

Denormalizarea ar fi utila in aceasta situatie, deoarece daca am avea nevoie sa accesam pretul unui fel de mancare foarte des am fi mereu nevoiti sa facem JOIN intre FEL_DE_MANCARE si PRET. De asemenea, din punctul de vedere al memoriei, relatia nou-adaugata PRET nu imbunatatesta memoria, chiar daca acum stocam doar valori distincte ale preturilor (in cazul denormalizarii ar putea exista feluri de mancare cu acelasi pret), deoarece acea valoare care s-ar fi repetat in relatia FEL_DE_MANCARE este acum inlocuita de o cheie externa (cod_pret) care se poate repeta pentru feluri de mancare diferite, dar cu acelasi pret.

FEL_DE_MANCARE		
cod_fel_de_mancare#	denumire	cod_pret
6	Piure	1
2	Ciorba De Perisoare	1
5	Cartofi Prajiti	2

PRET	
cod_pret#	valoare
1	12
2	15



FEL_DE_MANCARE (Denormalizat)

Denormalizare:	cod_fel_de_mancare#	denumire	pret
	6	Piure	12
	2	Ciorba De Perisoare	12
	5	Cartofi Prajiti	15

Optimizarea unei cereri

Cerinta:

Sa se afiseze numarul si etajul la care se afla toate camerele din hotelul cu denumirea "Continental".

Metoda optimizata:

Cerere SQL:

```
SELECT c.numar, c.etaj FROM (SELECT cod_hotel, numar, etaj FROM CAMERA) c JOIN (SELECT cod_hotel, denumire FROM HOTEL WHERE INITCAP(denumire) = 'Continental') h ON (c.cod_hotel = h.cod_hotel);
```

Expresie algebraica:

```
R1 = PROJECT(CAMERA c, c.cod_hotel, c.numar, c.etaj);
R2 = SELECT(HOTEL h, INITCAP(h.denumire) = 'Continental');
R3 = PROJECT(R2, h.cod_hotel, h.denumire);
R4 = JOIN(R1, R3, c.cod_hotel = h.cod_hotel);
Rezultat = PROJECT(R4, c.numar, c.etaj);
```

Rezultatul este memorat in Rezultat.

Metoda neoptimizata:

Cerere SQL:

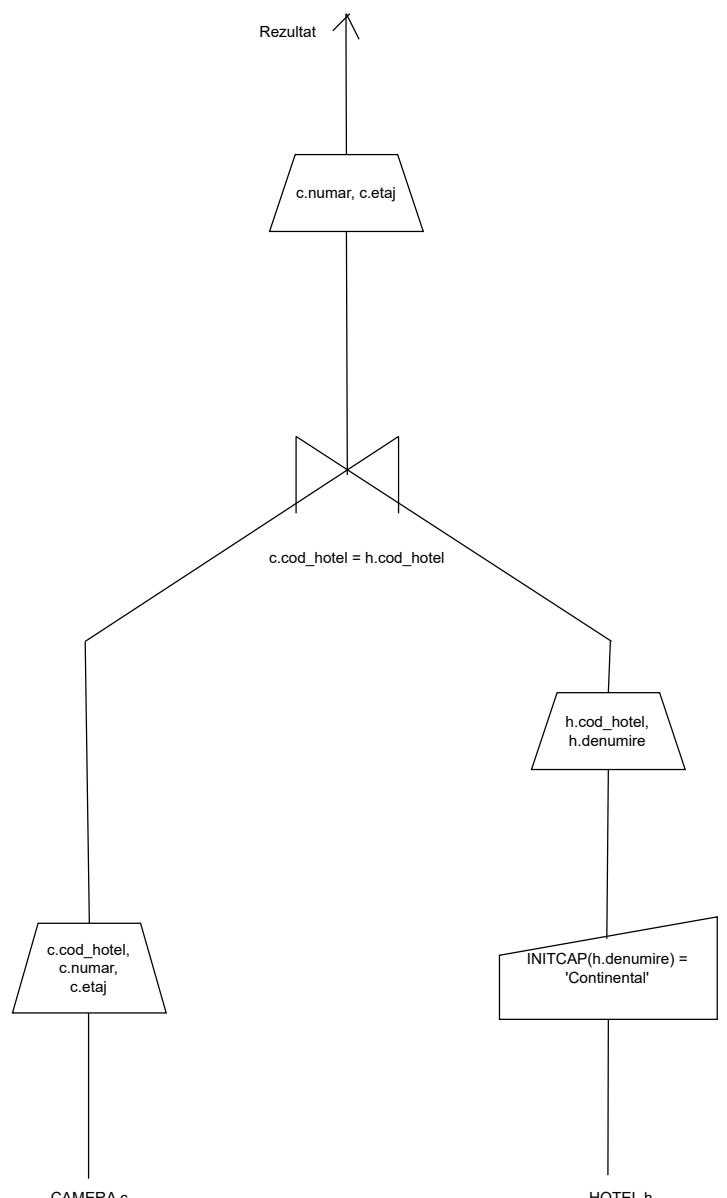
```
SELECT c.numar, c.etaj FROM CAMERA c, HOTEL h WHERE c.cod_hotel = h.cod_hotel AND INITCAP(h.denumire) = 'Continental';
```

Expresie algebraica:

```
R1 = PRODUCT(CAMERA c, HOTEL h);
R2 = SELECT(R1, c.cod_hotel = h.cod_hotel);
R3 = SELECT(R2, INITCAP(h.denumire) = 'Continental');
Rezultat = PROJECT(R3, c.numar, c.etaj);
```

Rezultatul este memorat in Rezultat.

Arbore algebraic metoda optimizata:



Metoda optimizata:

```

SELECT c.numar, c.etalj FROM (SELECT cod_hotel, numar, etaj FROM CAMERA) c JOIN (SELECT cod_hotel, denumire FROM HOTEL
WHERE INITCAP(denumire) = 'Continental') h
ON (c.cod_hotel = h.cod_hotel);
  
```

Script Output x | Query Result x

SQL | All Rows Fetched: 5 in 0.12 seconds

	NUMAR	ETAJ
1	1	0
2	2	1
3	3	0
4	4	2
5	5	0

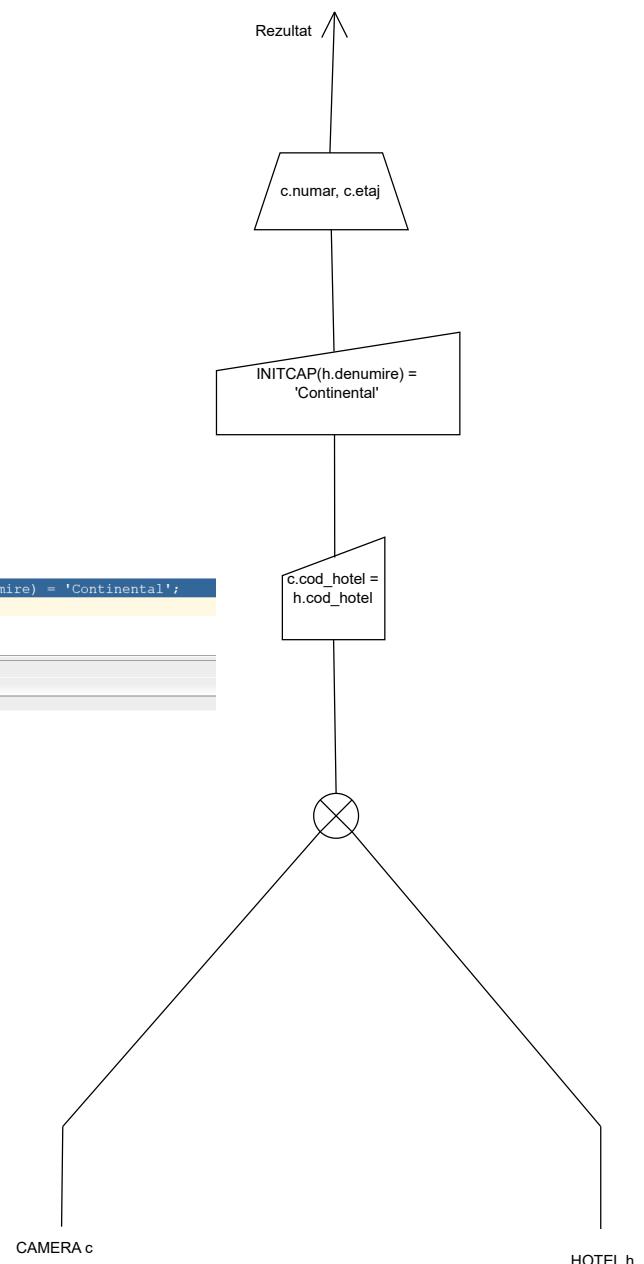
Arbore algebraic metoda neoptimizata:

Metoda neoptimizata:

```
SELECT c.numar, c.etalj FROM CAMERA c, HOTEL h WHERE c.cod_hotel = h.cod_hotel AND INITCAP(h.denumire) = 'Continental';
```

Script Output | Query Result | SQL | All Rows Fetched: 5 in 0.002 seconds

	NUMAR	ETAJ
1	1	0
2	2	1
3	3	0
4	4	2
5	5	0



Principiul folosit este cel de a minimiza cantitatea de date cu care lucrăm cât mai devreme posibil, ca în cazul unor viitoare combinații de date prin JOIN sau produse carteziene să trebuiască să lucrăm cu o cantitate cât mai mică de date. Astfel, proiecțiile se fac cât mai devreme posibil, iar combinarea tabelelor CAMERA și HOTEL s-a realizat printr-un JOIN cu o condiție ($c.cod_hotel = h.cod_hotel$) pentru a filtra datele în timpul primei parcurgeri a produsului cartezian.

Ca și proprietăți ale operatorilor algebrei relationale au fost folosite următoarele:
compunerea proiecțiilor, compunerea selectiilor, comutarea selectiei cu proiecția și comutarea proiecției cu produsul cartezian.

Ca și reguli de optimizare ce deriva din proprietățile operatorilor algebrei relationale au fost folosite următoarele:
Selectiile se execută cât mai devreme posibil, reducând dimensiunea relațiilor.
Produsele carteziene se înlocuiesc cu JOIN-uri, ori de câte ori este posibil.
Proiecțiile se execută la început pentru a îndepărta atributele nefolositoare.

Crearea unei secente ce va fi utilizata in inserarea inregistrarilor in table

```
CREATE SEQUENCE SECVENTA_LOCATIE
INCREMENT BY 1
START WITH 1
MAXVALUE 9999999999
NOCYCLE;

CREATE SEQUENCE SECVENTA_HOTEL
INCREMENT BY 1
START WITH 1
MAXVALUE 9999999999
NOCYCLE;

CREATE SEQUENCE SECVENTA_CAMERA
INCREMENT BY 1
START WITH 1
MAXVALUE 9999999999
NOCYCLE;

CREATE SEQUENCE SECVENTA_FACILITATE
INCREMENT BY 1
START WITH 1
MAXVALUE 9999999999
NOCYCLE;

CREATE SEQUENCE SECVENTA_CLIENT
INCREMENT BY 1
START WITH 1
MAXVALUE 9999999999
NOCYCLE;

CREATE SEQUENCE SECVENTA_ZONA_DE_PARCARE
INCREMENT BY 1
START WITH 1
MAXVALUE 9999999999
NOCYCLE;

CREATE SEQUENCE SECVENTA_LOC_DE_PARCARE
INCREMENT BY 1
START WITH 1
MAXVALUE 9999999999
NOCYCLE;

CREATE SEQUENCE SECVENTA_FURNIZOR
INCREMENT BY 1
START WITH 1
MAXVALUE 9999999999
NOCYCLE;

CREATE SEQUENCE SECVENTA_CONTRACT
INCREMENT BY 1
START WITH 1
MAXVALUE 9999999999
NOCYCLE;

CREATE SEQUENCE SECVENTA_FEL_DE_MANCARE
INCREMENT BY 1
START WITH 1
MAXVALUE 9999999999
NOCYCLE;

CREATE SEQUENCE SECVENTA_RESTAURANT
INCREMENT BY 1
START WITH 1
MAXVALUE 9999999999
NOCYCLE;

CREATE SEQUENCE SECVENTA_ANGAJAT
INCREMENT BY 1
START WITH 1
MAXVALUE 9999999999
NOCYCLE;
```

```

CREATE TABLE LOCATIE
(
cod_locatie NUMBER(10) PRIMARY KEY,
adresa VARCHAR2(50) NOT NULL
);

CREATE TABLE HOTEL
(
cod_hotel NUMBER(10) PRIMARY KEY,
denumire VARCHAR2(50) NOT NULL,
numar_stele NUMBER(1),
an_infiintare NUMBER(4),
cod_locatie NUMBER(10) REFERENCES LOCATIE(cod_locatie) NOT NULL
);

CREATE TABLE CAMERA
(
cod_camera NUMBER(10) NOT NULL,
cod_hotel NUMBER(10) REFERENCES HOTEL(cod_hotel) NOT NULL,
PRIMARY KEY (cod_camera, cod_hotel),
numar NUMBER(10) NOT NULL CHECK(numar > 0),
etaj NUMBER(3) DEFAULT 0 CHECK(etaj >= 0) NOT NULL
);

CREATE TABLE FACILITATE
(
cod_facilitate NUMBER(10) PRIMARY KEY,
denumire VARCHAR2(50) NOT NULL
);

CREATE TABLE CLIENT
(
cod_client NUMBER(10) PRIMARY KEY,
nume VARCHAR2(50) NOT NULL,
prenume VARCHAR2(50) NOT NULL,
CNP VARCHAR2(13),
numar_telefon VARCHAR2(50) NOT NULL,
adresa_mail VARCHAR2(50)
);

CREATE TABLE CAMERA_presinta_FACILITATE
(
cod_camera NUMBER(10) NOT NULL,
cod_hotel NUMBER(10) NOT NULL,
cod_facilitate NUMBER(10) REFERENCES FACILITATE(cod_facilitate) NOT NULL,
FOREIGN KEY (cod_camera, cod_hotel) REFERENCES CAMERA(cod_camera, cod_hotel),
PRIMARY KEY(cod_camera, cod_hotel, cod_facilitate)
);

CREATE TABLE CLIENT_rezerva_CAMERA
(
cod_client NUMBER(10) REFERENCES CLIENT(cod_client) NOT NULL,
cod_camera NUMBER(10) NOT NULL,
cod_hotel NUMBER(10) NOT NULL,
data_inceput DATE DEFAULT SYSDATE NOT NULL,
data_sfarsit DATE NOT NULL,
FOREIGN KEY (cod_camera, cod_hotel) REFERENCES CAMERA(cod_camera, cod_hotel),
CONSTRAINT check_date_client_rezerva_camera CHECK(data_inceput <= data_sfarsit),
PRIMARY KEY(cod_client, cod_camera, cod_hotel, data_inceput, data_sfarsit)
);

CREATE TABLE ZONA_DE_PARCARE
(
cod_zona_de_parcare NUMBER(10) PRIMARY KEY,
cod_hotel NUMBER(10) REFERENCES HOTEL(cod_hotel) NOT NULL,
cod_locatie NUMBER(10) REFERENCES LOCATIE(cod_locatie) NOT NULL
);

CREATE TABLE LOC_DE_PARCARE
(
cod_loc_de_parcare NUMBER(10) NOT NULL,
cod_zona_de_parcare NUMBER(10) NOT NULL,
PRIMARY KEY (cod_loc_de_parcare, cod_zona_de_parcare),
numar NUMBER(10) CHECK(numar > 0) NOT NULL
);

CREATE TABLE FURNIZOR
(
cod_furnizor NUMBER(10) PRIMARY KEY,
nume VARCHAR2(50) NOT NULL
);

```

```

CREATE TABLE CONTRACT
(
    cod_contract NUMBER(10) PRIMARY KEY,
    data_inceput DATE DEFAULT SYSDATE NOT NULL,
    data_sfarsit DATE NOT NULL,
    CONSTRAINT check_date_contract CHECK(data_inceput <= data_sfarsit)
);

CREATE TABLE FEL_DE_MANCARE
(
    cod_fel_de_mancare NUMBER(10) PRIMARY KEY,
    denumire VARCHAR2(50) NOT NULL,
    pret NUMBER(5) NOT NULL
);

CREATE TABLE RESTAURANT
(
    cod_restaurant NUMBER(10) PRIMARY KEY,
    cod_hotel NUMBER(10) REFERENCES HOTEL(cod_hotel) NOT NULL,
    cod_locatie NUMBER(10) REFERENCES LOCATIE(cod_locatie) NOT NULL,
    denumire VARCHAR2(50) NOT NULL,
    numar_stele NUMBER(1),
    an_infiintare NUMBER(4)
);

CREATE TABLE RESTAURANT_serveste_FEL_DE_MANCARE
(
    cod_restaurant NUMBER(10) REFERENCES RESTAURANT(cod_restaurant) NOT NULL,
    cod_fel_de_mancare NUMBER(10) REFERENCES FEL_DE_MANCARE(cod_fel_de_mancare) NOT NULL,
    PRIMARY KEY (cod_restaurant, cod_fel_de_mancare)
);

CREATE TABLE FURNIZOR_semneaza_cu_RESTAURANT_mai_multe_CONTRACTE
(
    cod_restaurant NUMBER(10) REFERENCES RESTAURANT(cod_restaurant) NOT NULL,
    cod_furnizor NUMBER(10) REFERENCES FURNIZOR(cod_furnizor) NOT NULL,
    cod_contract NUMBER(10) REFERENCES CONTRACT(cod_contract) NOT NULL,
    PRIMARY KEY (cod_restaurant, cod_furnizor, cod_contract)
);

CREATE TABLE ANGAJAT
(
    cod_angajat NUMBER(10) PRIMARY KEY,
    nume VARCHAR2(50) NOT NULL,
    prenume VARCHAR2(50) NOT NULL,
    CNP VARCHAR2(13) NOT NULL,
    numar_telefon VARCHAR2(50) NOT NULL,
    adresa_mail VARCHAR2(50),
    salariu NUMBER(10) NOT NULL
);

CREATE TABLE PAZNIC
(
    cod_angajat NUMBER(10) PRIMARY KEY REFERENCES ANGAJAT(cod_angajat),
    cod_zona_de_parcare NUMBER(10) REFERENCES ZONA_DE_PARCARE(cod_zona_de_parcare)
);

CREATE TABLE RECEPTIONIST
(
    cod_angajat NUMBER(10) PRIMARY KEY REFERENCES ANGAJAT(cod_angajat),
    cod_hotel NUMBER(10) REFERENCES HOTEL(cod_hotel)
);

CREATE TABLE ADMINISTRATOR
(
    cod_angajat NUMBER(10) PRIMARY KEY REFERENCES ANGAJAT(cod_angajat),
    cod_hotel NUMBER(10) REFERENCES HOTEL(cod_hotel)
);

CREATE TABLE CHELNER
(
    cod_angajat NUMBER(10) PRIMARY KEY REFERENCES ANGAJAT(cod_angajat),
    cod_restaurant NUMBER(10) REFERENCES RESTAURANT(cod_restaurant)
);

CREATE TABLE BUCATAR
(
    cod_angajat NUMBER(10) PRIMARY KEY REFERENCES ANGAJAT(cod_angajat),
    cod_restaurant NUMBER(10) REFERENCES RESTAURANT(cod_restaurant)
);

```

SQL Worksheet History

Worksheet Query Builder

```
CREATE SEQUENCE SEVENTA_LOCATIE
INCREMENT BY 1
START WITH 1
MAXVALUE 9999999999
NOCYCLE;

CREATE SEQUENCE SEVENTA_HOTEL
INCREMENT BY 1
START WITH 1
MAXVALUE 9999999999
NOCYCLE;
```

Script Output x Query Result x

Task completed in 0.05 seconds

Sequence SEVENTA_LOCATIE created.

Sequence SEVENTA_HOTEL created.

SQL Worksheet History

Worksheet Query Builder

```
CREATE SEQUENCE SEVENTA_CAMERA
INCREMENT BY 1
START WITH 1
MAXVALUE 9999999999
NOCYCLE;

CREATE SEQUENCE SEVENTA_FACILITATE
INCREMENT BY 1
START WITH 1
MAXVALUE 9999999999
NOCYCLE;
```

Script Output x Query Result x

Task completed in 0.059 seconds

Sequence SEVENTA_LOCATIE created.

Sequence SEVENTA_HOTEL created.

Sequence SEVENTA_CAMERA created.

Sequence SEVENTA_FACILITATE created.

SQL Worksheet History

Worksheet Query Builder

```
CREATE SEQUENCE SECVENTA_CLIENT
INCREMENT BY 1
START WITH 1
MAXVALUE 9999999999
NOCYCLE;

CREATE SEQUENCE SECVENTA_ZONA_DE_PARCARE
INCREMENT BY 1
START WITH 1
MAXVALUE 9999999999
NOCYCLE;
```

Script Output x Query Result x

Sequence SECVENTA_HOTEL created.

Sequence SECVENTA_CAMERA created.

Sequence SECVENTA_FACILITATE created.

Sequence SECVENTA_CLIENT created.

Sequence SECVENTA_ZONA_DE_PARCARE created.

SQL Worksheet History

Worksheet Query Builder

```
CREATE SEQUENCE SECVENTA_LOC_DE_PARCARE
INCREMENT BY 1
START WITH 1
MAXVALUE 9999999999
NOCYCLE;

CREATE SEQUENCE SECVENTA_FURNIZOR
INCREMENT BY 1
START WITH 1
MAXVALUE 9999999999
NOCYCLE;
```

Script Output X Query Result X

Sequence SECVENTA_FACILITATE created.

Sequence SECVENTA_CLIENT created.

Sequence SECVENTA_ZONA_DE_PARCARE created.

Sequence SECVENTA_LOC_DE_PARCARE created.

Sequence SECVENTA_FURNIZOR created.

SQL Worksheet | History

Worksheet | Query Builder

```
CREATE SEQUENCE SECVENTA_CONTRACT
INCREMENT BY 1
START WITH 1
MAXVALUE 9999999999
NOCYCLE;

CREATE SEQUENCE SECVENTA_FEL_DE_MANCARE
INCREMENT BY 1
START WITH 1
MAXVALUE 9999999999
NOCYCLE;
```

Script Output X | Query Result X

| Task completed in 0.031 seconds

Sequence SECVENTA_ZONA_DE_PARCARE created.

Sequence SECVENTA_LOC_DE_PARCARE created.

Sequence SECVENTA_FURNIZOR created.

Sequence SECVENTA_CONTRACT created.

Sequence SECVENTA_FEL_DE_MANCARE created.

SQL Worksheet History

Worksheet Query Builder

```
CREATE SEQUENCE SECVENTA_RESTAURANT
INCREMENT BY 1
START WITH 1
MAXVALUE 9999999999
NOCYCLE;

CREATE SEQUENCE SECVENTA_ANGAJAT
INCREMENT BY 1
START WITH 1
MAXVALUE 9999999999
NOCYCLE;
```

Script Output X Query Result X

redo undo save print Task completed in 0.079 seconds

Sequence SECVENTA_FURNIZOR created.

Sequence SECVENTA_CONTRACT created.

Sequence SECVENTA_FEL_DE_MANCARE created.

Sequence SECVENTA_RESTAURANT created.

Sequence SECVENTA_ANGAJAT created.

SQL Worksheet History

Worksheet Query Builder

```
CREATE TABLE LOCATIE
(
    cod_locatie NUMBER(10) PRIMARY KEY,
    adresa VARCHAR2(50) NOT NULL
);

CREATE TABLE HOTEL
(
    cod_hotel NUMBER(10) PRIMARY KEY,
    denumire VARCHAR2(50) NOT NULL,
    numar_stele NUMBER(1),
    an_infiintare NUMBER(4),
    cod_locatie NUMBER(10) REFERENCES LOCATIE(cod_locatie) NOT NULL
);

CREATE TABLE CAMERA
(
    cod_camera NUMBER(10) NOT NULL,
```

Script Output x | Task completed in 0.053 seconds

Sequence SECVENTA_ANGAJAT created.

Table LOCATIE created.

Table HOTEL created.

SQL Worksheet History

Worksheet Query Builder

```
an_infiintare NUMBER(4),
cod_locatie NUMBER(10) REFERENCES LOCATIE(cod_locatie) NOT NULL
);

CREATE TABLE CAMERA
(
    cod_camera NUMBER(10) NOT NULL,
    cod_hotel NUMBER(10) REFERENCES HOTEL(cod_hotel) NOT NULL,
    PRIMARY KEY (cod_camera, cod_hotel),
    numar NUMBER(10) NOT NULL CHECK(numar > 0),
    etaj NUMBER(3) DEFAULT 0 CHECK(etaj >= 0) NOT NULL
);

CREATE TABLE FACILITATE
(
    cod_facilitate NUMBER(10) PRIMARY KEY,
    denumire VARCHAR2(50) NOT NULL
);
```

Script Output x | Query Result x

Task completed in 0.08 seconds

Table HOTEL created.

Table CAMERA created.

Table FACILITATE created.

```
Worksheet Query Builder

CREATE TABLE CLIENT
(
    cod_client NUMBER(10) PRIMARY KEY,
    nume VARCHAR2(50) NOT NULL,
    prenume VARCHAR2(50) NOT NULL,
    CNP VARCHAR2(13),
    numar_telefon VARCHAR2(50) NOT NULL,
    adresa_mail VARCHAR2(50)
);

CREATE TABLE CAMERA_presinta_FACILITATE
(
    cod_camera NUMBER(10) NOT NULL,
    cod_hotel NUMBER(10) NOT NULL,
    cod_facilitate NUMBER(10) REFERENCES FACILITATE(cod_facilitate) NOT NULL,
    FOREIGN KEY (cod_camera, cod_hotel) REFERENCES CAMERA(cod_camera, cod_hotel),
    PRIMARY KEY(cod_camera, cod_hotel, cod_facilitate)
);
```

Script Output x | Query Result x
| Task completed in 0.063 seconds

Table FACILITATE created.

Table CLIENT created.

Table CAMERA_PREZINTA_FACILITATE created.

```
Worksheet | Query Builder

CREATE TABLE CLIENT_rezerva_CAMERA
(
    cod_client NUMBER(10) REFERENCES CLIENT(cod_client) NOT NULL,
    cod_camera NUMBER(10) NOT NULL,
    cod_hotel NUMBER(10) NOT NULL,
    data_inceput DATE DEFAULT SYSDATE NOT NULL,
    data_sfarsit DATE NOT NULL,
    FOREIGN KEY (cod_camera, cod_hotel) REFERENCES CAMERA(cod_camera, cod_hotel),
    CONSTRAINT check_date_client_rezerva_camera CHECK(data_inceput <= data_sfarsit),
    PRIMARY KEY (cod_client, cod_camera, cod_hotel, data_inceput, data_sfarsit)
);

CREATE TABLE ZONA_DE_PARCARE
(
    cod_zona_de_parcare NUMBER(10) PRIMARY KEY,
    cod_hotel NUMBER(10) REFERENCES HOTEL(cod_hotel) NOT NULL,
    cod_locatie NUMBER(10) REFERENCES LOCATIE(cod_locatie) NOT NULL
);
```

Script Output | Query Result | Task completed in 0.076 seconds

Table CLIENT_REZERVA_CAMERA created.

Table ZONA_DE_PARCARE created.

```
SQL Worksheet History
| 0.08 seconds
Worksheet Query Builder
cod_hotel NUMBER(10) REFERENCES HOTEL(cod_hotel) NOT NULL,
cod_locatie NUMBER(10) REFERENCES LOCATIE(cod_locatie) NOT NULL
);

CREATE TABLE LOC_DE_PARCARE
(
    cod_loc_de_parcare NUMBER(10) NOT NULL,
    cod_zona_de_parcare NUMBER(10) NOT NULL,
    PRIMARY KEY (cod_loc_de_parcare, cod_zona_de_parcare),
    numar NUMBER(10) CHECK(numar > 0) NOT NULL
);

CREATE TABLE FURNIZOR
(
    cod_furnizor NUMBER(10) PRIMARY KEY,
    nume VARCHAR2(50) NOT NULL
);

CREATE TABLE CONTRACT
```

Script Output x | Query Result x
Task completed in 0.08 seconds

Table ZONA_DE_PARCARE created.

Table LOC_DE_PARCARE created.

Table FURNIZOR created.

SQL Worksheet History

Worksheet Query Builder

```
    nume VARCHAR2(50) NOT NULL
);

CREATE TABLE CONTRACT
(
    cod_contract NUMBER(10) PRIMARY KEY,
    data_inceput DATE DEFAULT SYSDATE NOT NULL,
    data_sfarsit DATE NOT NULL,
    CONSTRAINT check_date_contract CHECK(data_inceput <= data_sfarsit)
);

CREATE TABLE FEL_DE_MANCARE
(
    cod_fel_de_mancare NUMBER(10) PRIMARY KEY,
    denumire VARCHAR2(50) NOT NULL,
    pret NUMBER(5) NOT NULL
);

CREATE TABLE RESTAURANT
```

Script Output X Query Result X
Task completed in 0.078 seconds

Table FURNIZOR created.

Table CONTRACT created.

Table FEL_DE_MANCARE created.

The screenshot shows the Oracle SQL Developer interface with the 'Worksheet' tab selected. The code area contains the following SQL statements:

```
    pret NUMBER(5) NOT NULL
);

CREATE TABLE RESTAURANT
(
    cod_restaurant NUMBER(10) PRIMARY KEY,
    cod_hotel NUMBER(10) REFERENCES HOTEL(cod_hotel) NOT NULL,
    cod_locatie NUMBER(10) REFERENCES LOCATIE(cod_locatie) NOT NULL,
    denumire VARCHAR2(50) NOT NULL,
    numar_stele NUMBER(1),
    an_infiintare NUMBER(4)
);

CREATE TABLE RESTAURANT_serveste_FEL_DE_MANCARE
(
    cod_restaurant NUMBER(10) REFERENCES RESTAURANT(cod_restaurant) NOT NULL,
    cod_fel_de_mancare NUMBER(10) REFERENCES FEL_DE_MANCARE(cod_fel_de_mancare) NOT NULL,
    PRIMARY KEY (cod_restaurant, cod_fel_de_mancare)
);
```

The 'Script Output' tab is active, showing the message "Task completed in 0.076 seconds".

Table RESTAURANT created.

Table RESTAURANT_SERVESTE_FEL_DE_MANCARE created.

```
Worksheet Query Builder

CREATE TABLE FURNIZOR_semneaza_cu_RESTAURANT_mai_multe_CONTRACTE
(
    cod_restaurant NUMBER(10) REFERENCES RESTAURANT(cod_restaurant) NOT NULL,
    cod_furnizor NUMBER(10) REFERENCES FURNIZOR(cod_furnizor) NOT NULL,
    cod_contract NUMBER(10) REFERENCES CONTRACT(cod_contract) NOT NULL,
    PRIMARY KEY (cod_restaurant, cod_furnizor, cod_contract)
);

CREATE TABLE ANGAJAT
(
    cod_angajat NUMBER(10) PRIMARY KEY,
    nume VARCHAR2(50) NOT NULL,
    prenume VARCHAR2(50) NOT NULL,
    CNP VARCHAR2(13) NOT NULL,
    numar_telefon VARCHAR2(50) NOT NULL,
    adresa_mail VARCHAR2(50),
    salariu NUMBER(10) NOT NULL
);
```

Script Output Query Result

Task completed in 0.061 seconds

Table FURNIZOR_SEMNEAZA_CU_RESTAURANT_MAI_MULTE_CONTRACTE created.

Table ANGAJAT created.

SQL Worksheet History

Worksheet Query Builder

```
CREATE TABLE PAZNIC
(
    cod_angajat NUMBER(10) PRIMARY KEY REFERENCES ANGAJAT(cod_angajat),
    cod_zona_de_parcare NUMBER(10) REFERENCES ZONA_DE_PARCARE(cod_zona_de_parcare)
);

CREATE TABLE RECEPTIONIST
(
    cod_angajat NUMBER(10) PRIMARY KEY REFERENCES ANGAJAT(cod_angajat),
    cod_hotel NUMBER(10) REFERENCES HOTEL(cod_hotel)
);

CREATE TABLE ADMINISTRATOR
(
    cod_angajat NUMBER(10) PRIMARY KEY REFERENCES ANGAJAT(cod_angajat),
    cod_hotel NUMBER(10) REFERENCES HOTEL(cod_hotel)
);
```

Script Output x | Query Result x

| Task completed in 0.101 seconds

Table PAZNIC created.

Table RECEPTIONIST created.

Table ADMINISTRATOR created.

```

SQL Worksheet History
Worksheet Query Builder

cod_angajat NUMBER(10) PRIMARY KEY REFERENCES ANGAJAT(cod_angajat),
cod_hotel NUMBER(10) REFERENCES HOTEL(cod_hotel)
);

CREATE TABLE CHELNER
(
    cod_angajat NUMBER(10) PRIMARY KEY REFERENCES ANGAJAT(cod_angajat),
    cod_restaurant NUMBER(10) REFERENCES RESTAURANT(cod_restaurant)
);

CREATE TABLE BUCATAR
(
    cod_angajat NUMBER(10) PRIMARY KEY REFERENCES ANGAJAT(cod_angajat),
    cod_restaurant NUMBER(10) REFERENCES RESTAURANT(cod_restaurant)
);

```

Script Output | Query Result | Task completed in 0.063 seconds

Table ADMINISTRATOR created.

Table CHELNER created.

Table BUCATAR created.

Crearea tabelelor in SQL si inserarea de date coerente in fiecare dintre acestea

```

INSERT INTO LOCATIE(cod_locatie, adresa)
VALUES(SECVENTA_LOCATIE.NEXTVAL, 'Zona 1');
INSERT INTO LOCATIE(cod_locatie, adresa)
VALUES(SECVENTA_LOCATIE.NEXTVAL, 'Zona 2');
INSERT INTO LOCATIE(cod_locatie, adresa)
VALUES(SECVENTA_LOCATIE.NEXTVAL, 'Zona 3');
INSERT INTO LOCATIE(cod_locatie, adresa)
VALUES(SECVENTA_LOCATIE.NEXTVAL, 'Zona 4');
INSERT INTO LOCATIE(cod_locatie, adresa)
VALUES(SECVENTA_LOCATIE.NEXTVAL, 'Zona 5');
INSERT INTO LOCATIE(cod_locatie, adresa)
VALUES(SECVENTA_LOCATIE.NEXTVAL, 'Zona 6');
INSERT INTO LOCATIE(cod_locatie, adresa)
VALUES(SECVENTA_LOCATIE.NEXTVAL, 'Zona 7');
INSERT INTO LOCATIE(cod_locatie, adresa)
VALUES(SECVENTA_LOCATIE.NEXTVAL, 'Centru');
INSERT INTO LOCATIE(cod_locatie, adresa)
VALUES(SECVENTA_LOCATIE.NEXTVAL, 'Periferie');

```

```

INSERT INTO HOTEL(cod_hotel, denumire, numar_stele, an_infiintare, cod_locatie)
VALUES(SECVENTA_HOTEL.NEXTVAL, 'Continental', 3, 1965, 1);
INSERT INTO HOTEL(cod_hotel, denumire, numar_stele, an_infiintare, cod_locatie)
VALUES(SECVENTA_HOTEL.NEXTVAL, 'Europa', 4, 1993, 1);
INSERT INTO HOTEL(cod_hotel, denumire, numar_stele, an_infiintare, cod_locatie)
VALUES(SECVENTA_HOTEL.NEXTVAL, 'Atena', 3, 1997, 1);
INSERT INTO HOTEL(cod_hotel, denumire, numar_stele, an_infiintare, cod_locatie)
VALUES(SECVENTA_HOTEL.NEXTVAL, 'Sofia', 2, 1983, 2);
INSERT INTO HOTEL(cod_hotel, denumire, numar_stele, an_infiintare, cod_locatie)
VALUES(SECVENTA_HOTEL.NEXTVAL, 'Mures', 4, 1975, 2);
INSERT INTO HOTEL(cod_hotel, denumire, numar_stele, an_infiintare, cod_locatie)
VALUES(SECVENTA_HOTEL.NEXTVAL, 'Banat', 2, 1968, 2);
INSERT INTO HOTEL(cod_hotel, denumire, numar_stele, an_infiintare, cod_locatie)
VALUES(SECVENTA_HOTEL.NEXTVAL, 'Transilvania', 3, 1970, 3);
INSERT INTO HOTEL(cod_hotel, denumire, numar_stele, an_infiintare, cod_locatie)
VALUES(SECVENTA_HOTEL.NEXTVAL, 'Moldova', NULL, NULL, 3);
INSERT INTO HOTEL(cod_hotel, denumire, numar_stele, an_infiintare, cod_locatie)
VALUES(SECVENTA_HOTEL.NEXTVAL, 'Dobrogea', 5, NULL, 9);
INSERT INTO HOTEL(cod_hotel, denumire, numar_stele, an_infiintare, cod_locatie)
VALUES(SECVENTA_HOTEL.NEXTVAL, 'Basarabia', 1, 1950, 9);
INSERT INTO HOTEL(cod_hotel, denumire, numar_stele, an_infiintare, cod_locatie)
VALUES(SECVENTA_HOTEL.NEXTVAL, 'Bucovina', NULL, NULL, 9);
INSERT INTO HOTEL(cod_hotel, denumire, numar_stele, an_infiintare, cod_locatie)
VALUES(SECVENTA_HOTEL.NEXTVAL, 'Transnistria', NULL, NULL, 8);
INSERT INTO HOTEL(cod_hotel, denumire, numar_stele, an_infiintare, cod_locatie)
VALUES(SECVENTA_HOTEL.NEXTVAL, 'Venetia', NULL, 1963, 8);
INSERT INTO HOTEL(cod_hotel, denumire, numar_stele, an_infiintare, cod_locatie)
VALUES(SECVENTA_HOTEL.NEXTVAL, 'Dunare', 3, NULL, 8);
INSERT INTO HOTEL(cod_hotel, denumire, numar_stele, an_infiintare, cod_locatie)
VALUES(SECVENTA_HOTEL.NEXTVAL, 'Carpati', NULL, 1968, 6);

```

```

INSERT INTO CAMERA(cod_camera, cod_hotel, numar)
VALUES(SECVENTA_CAMERA.NEXTVAL, 1, 1);
INSERT INTO CAMERA(cod_camera, cod_hotel, numar, etaj)
VALUES(SECVENTA_CAMERA.NEXTVAL, 1, 2, 1);
INSERT INTO CAMERA(cod_camera, cod_hotel, numar)
VALUES(SECVENTA_CAMERA.NEXTVAL, 1, 3);
INSERT INTO CAMERA(cod_camera, cod_hotel, numar, etaj)
VALUES(SECVENTA_CAMERA.NEXTVAL, 1, 4, 2);
INSERT INTO CAMERA(cod_camera, cod_hotel, numar)
VALUES(SECVENTA_CAMERA.NEXTVAL, 1, 5);
INSERT INTO CAMERA(cod_camera, cod_hotel, numar)
VALUES(SECVENTA_CAMERA.NEXTVAL, 2, 1);
INSERT INTO CAMERA(cod_camera, cod_hotel, numar, etaj)
VALUES(SECVENTA_CAMERA.NEXTVAL, 2, 2, 3);
INSERT INTO CAMERA(cod_camera, cod_hotel, numar, etaj)
VALUES(SECVENTA_CAMERA.NEXTVAL, 2, 3, 0);
INSERT INTO CAMERA(cod_camera, cod_hotel, numar, etaj)
VALUES(SECVENTA_CAMERA.NEXTVAL, 2, 4, 7);
INSERT INTO CAMERA(cod_camera, cod_hotel, numar)
VALUES(SECVENTA_CAMERA.NEXTVAL, 2, 5);
INSERT INTO CAMERA(cod_camera, cod_hotel, numar, etaj)
VALUES(SECVENTA_CAMERA.NEXTVAL, 3, 1, 1);

```

```

INSERT INTO FACILITATE(cod_facilitate, denumire)
VALUES(SECVENTA_FACILITATE.NEXTVAL, 'Camera Standard');
INSERT INTO FACILITATE(cod_facilitate, denumire)
VALUES(SECVENTA_FACILITATE.NEXTVAL, 'Camera Dubla');
INSERT INTO FACILITATE(cod_facilitate, denumire)
VALUES(SECVENTA_FACILITATE.NEXTVAL, 'Camera Tripla');
INSERT INTO FACILITATE(cod_facilitate, denumire)
VALUES(SECVENTA_FACILITATE.NEXTVAL, 'Apartament');
INSERT INTO FACILITATE(cod_facilitate, denumire)
VALUES(SECVENTA_FACILITATE.NEXTVAL, 'Balcon');
INSERT INTO FACILITATE(cod_facilitate, denumire)
VALUES(SECVENTA_FACILITATE.NEXTVAL, 'Aer Conditionat');
INSERT INTO FACILITATE(cod_facilitate, denumire)
VALUES(SECVENTA_FACILITATE.NEXTVAL, 'Room Service');

```

```

INSERT INTO CLIENT(cod_client, nume, prenume, CNP, numar_telefon, adresa_mail)
VALUES(SECVENTA_CLIENT.NEXTVAL, 'Popescu', 'Ion', '1234567890123', '0701234567', 'popescu.ion@mail.ro');
INSERT INTO CLIENT(cod_client, nume, prenume, CNP, numar_telefon, adresa_mail)
VALUES(SECVENTA_CLIENT.NEXTVAL, 'Ionescu', 'Mircea', '3210987654321', '0798765432', 'mircea_ionescu@mail.com');
INSERT INTO CLIENT(cod_client, nume, prenume, CNP, numar_telefon, adresa_mail)
VALUES(SECVENTA_CLIENT.NEXTVAL, 'Andrei', 'Ionescu', '0001112223334', '0700111222', 'andreiionescu@mail.ro');
INSERT INTO CLIENT(cod_client, nume, prenume, CNP, numar_telefon, adresa_mail)
VALUES(SECVENTA_CLIENT.NEXTVAL, 'Mihai', 'Paul', '9998887776665', '0799888777', 'paul.mihai@mail.com');
INSERT INTO CLIENT(cod_client, nume, prenume, CNP, numar_telefon, adresa_mail)
VALUES(SECVENTA_CLIENT.NEXTVAL, 'Bogdan', 'Mircea', '0000011111222', '0777777777', NULL);
INSERT INTO CLIENT(cod_client, nume, prenume, CNP, numar_telefon, adresa_mail)
VALUES(SECVENTA_CLIENT.NEXTVAL, 'Georgescu', 'Gheorghe', '0010011111222', '0777377777', 'georgescu_gheorghe@mail.ro');
INSERT INTO CLIENT(cod_client, nume, prenume, CNP, numar_telefon, adresa_mail)
VALUES(SECVENTA_CLIENT.NEXTVAL, 'Vlad', 'Teodor', '2020202020202', '0755000555', NULL);
INSERT INTO CLIENT(cod_client, nume, prenume, CNP, numar_telefon, adresa_mail)
VALUES(SECVENTA_CLIENT.NEXTVAL, 'Alexandru', 'Sebastian', '1313131313131', '0744333432', NULL);
INSERT INTO CLIENT(cod_client, nume, prenume, CNP, numar_telefon, adresa_mail)
VALUES(SECVENTA_CLIENT.NEXTVAL, 'Mihai', 'Florin', '1717171717171', '0700000000', NULL);
INSERT INTO CLIENT(cod_client, nume, prenume, CNP, numar_telefon, adresa_mail)
VALUES(SECVENTA_CLIENT.NEXTVAL, 'Andreeescu', 'Robert', '1919191919191', '0711222333', NULL);

INSERT INTO CAMERA_presinta_FACILITATE(cod_camera, cod_hotel, cod_facilitate)
VALUES(1, 1, 1);
INSERT INTO CAMERA_presinta_FACILITATE(cod_camera, cod_hotel, cod_facilitate)
VALUES(1, 1, 6);
INSERT INTO CAMERA_presinta_FACILITATE(cod_camera, cod_hotel, cod_facilitate)
VALUES(1, 1, 7);
INSERT INTO CAMERA_presinta_FACILITATE(cod_camera, cod_hotel, cod_facilitate)
VALUES(9, 2, 3);
INSERT INTO CAMERA_presinta_FACILITATE(cod_camera, cod_hotel, cod_facilitate)
VALUES(9, 2, 5);
INSERT INTO CAMERA_presinta_FACILITATE(cod_camera, cod_hotel, cod_facilitate)
VALUES(9, 2, 6);
INSERT INTO CAMERA_presinta_FACILITATE(cod_camera, cod_hotel, cod_facilitate)
VALUES(9, 2, 7);
INSERT INTO CAMERA_presinta_FACILITATE(cod_camera, cod_hotel, cod_facilitate)
VALUES(10, 2, 4);
INSERT INTO CAMERA_presinta_FACILITATE(cod_camera, cod_hotel, cod_facilitate)
VALUES(10, 2, 5);
INSERT INTO CAMERA_presinta_FACILITATE(cod_camera, cod_hotel, cod_facilitate)
VALUES(10, 2, 6);
INSERT INTO CAMERA_presinta_FACILITATE(cod_camera, cod_hotel, cod_facilitate)
VALUES(8, 2, 2);
INSERT INTO CAMERA_presinta_FACILITATE(cod_camera, cod_hotel, cod_facilitate)
VALUES(8, 2, 7);
INSERT INTO CAMERA_presinta_FACILITATE(cod_camera, cod_hotel, cod_facilitate)
VALUES(7, 2, 7);
INSERT INTO CAMERA_presinta_FACILITATE(cod_camera, cod_hotel, cod_facilitate)
VALUES(7, 2, 6);
INSERT INTO CAMERA_presinta_FACILITATE(cod_camera, cod_hotel, cod_facilitate)
VALUES(7, 2, 1);

INSERT INTO CLIENT_rezerva_CAMERA(cod_client, cod_camera, cod_hotel, data_incep, data_sfarsit)
VALUES(1, 1, 1, TO_DATE('02-09-2023', 'dd-mm-yyyy'), TO_DATE('08-09-2023', 'dd-mm-yyyy'));
INSERT INTO CLIENT_rezerva_CAMERA(cod_client, cod_camera, cod_hotel, data_incep, data_sfarsit)
VALUES(2, 2, 1, TO_DATE('07-05-2023', 'dd-mm-yyyy'), TO_DATE('13-05-2023', 'dd-mm-yyyy'));
INSERT INTO CLIENT_rezerva_CAMERA(cod_client, cod_camera, cod_hotel, data_incep, data_sfarsit)
VALUES(3, 3, 1, TO_DATE('26-04-2023', 'dd-mm-yyyy'), TO_DATE('02-05-2023', 'dd-mm-yyyy'));
INSERT INTO CLIENT_rezerva_CAMERA(cod_client, cod_camera, cod_hotel, data_incep, data_sfarsit)
VALUES(4, 4, 1, TO_DATE('23-03-2023', 'dd-mm-yyyy'), TO_DATE('01-04-2023', 'dd-mm-yyyy'));
INSERT INTO CLIENT_rezerva_CAMERA(cod_client, cod_camera, cod_hotel, data_incep, data_sfarsit)
VALUES(5, 5, 1, TO_DATE('23-10-2023', 'dd-mm-yyyy'), TO_DATE('01-11-2023', 'dd-mm-yyyy'));
INSERT INTO CLIENT_rezerva_CAMERA(cod_client, cod_camera, cod_hotel, data_incep, data_sfarsit)
VALUES(5, 7, 2, TO_DATE('07-09-2023', 'dd-mm-yyyy'), TO_DATE('08-09-2023', 'dd-mm-yyyy'));
INSERT INTO CLIENT_rezerva_CAMERA(cod_client, cod_camera, cod_hotel, data_incep, data_sfarsit)
VALUES(6, 7, 2, TO_DATE('10-03-2023', 'dd-mm-yyyy'), TO_DATE('14-03-2023', 'dd-mm-yyyy'));
INSERT INTO CLIENT_rezerva_CAMERA(cod_client, cod_camera, cod_hotel, data_incep, data_sfarsit)
VALUES(6, 8, 2, TO_DATE('15-04-2023', 'dd-mm-yyyy'), TO_DATE('15-05-2023', 'dd-mm-yyyy'));
INSERT INTO CLIENT_rezerva_CAMERA(cod_client, cod_camera, cod_hotel, data_incep, data_sfarsit)
VALUES(9, 9, 2, TO_DATE('17-03-2023', 'dd-mm-yyyy'), TO_DATE('19-05-2023', 'dd-mm-yyyy'));
INSERT INTO CLIENT_rezerva_CAMERA(cod_client, cod_camera, cod_hotel, data_incep, data_sfarsit)
VALUES(9, 9, 2, TO_DATE('01-01-2023', 'dd-mm-yyyy'), TO_DATE('01-02-2023', 'dd-mm-yyyy'));

```



```

INSERT INTO CONTRACT(cod_contract, data_inceput, data_sfarsit)
VALUES(SECVENTA_CONTRACT.NEXTVAL, TO_DATE('02-09-2023', 'dd-mm-yyyy'), TO_DATE('02-12-2023', 'dd-mm-yyyy'));
INSERT INTO CONTRACT(cod_contract, data_inceput, data_sfarsit)
VALUES(SECVENTA_CONTRACT.NEXTVAL, TO_DATE('01-01-2024', 'dd-mm-yyyy'), TO_DATE('01-01-2025', 'dd-mm-yyyy'));
INSERT INTO CONTRACT(cod_contract, data_inceput, data_sfarsit)
VALUES(SECVENTA_CONTRACT.NEXTVAL, TO_DATE('01-01-2023', 'dd-mm-yyyy'), TO_DATE('01-07-2023', 'dd-mm-yyyy'));
INSERT INTO CONTRACT(cod_contract, data_inceput, data_sfarsit)
VALUES(SECVENTA_CONTRACT.NEXTVAL, TO_DATE('01-01-2023', 'dd-mm-yyyy'), TO_DATE('01-08-2023', 'dd-mm-yyyy'));
INSERT INTO CONTRACT(cod_contract, data_inceput, data_sfarsit)
VALUES(SECVENTA_CONTRACT.NEXTVAL, TO_DATE('01-09-2023', 'dd-mm-yyyy'), TO_DATE('01-01-2024', 'dd-mm-yyyy'));
INSERT INTO CONTRACT(cod_contract, data_inceput, data_sfarsit)
VALUES(SECVENTA_CONTRACT.NEXTVAL, TO_DATE('01-03-2023', 'dd-mm-yyyy'), TO_DATE('01-04-2023', 'dd-mm-yyyy'));
INSERT INTO CONTRACT(cod_contract, data_inceput, data_sfarsit)
VALUES(SECVENTA_CONTRACT.NEXTVAL, TO_DATE('01-04-2023', 'dd-mm-yyyy'), TO_DATE('12-05-2023', 'dd-mm-yyyy'));
INSERT INTO CONTRACT(cod_contract, data_inceput, data_sfarsit)
VALUES(SECVENTA_CONTRACT.NEXTVAL, TO_DATE('26-04-2023', 'dd-mm-yyyy'), TO_DATE('02-06-2023', 'dd-mm-yyyy'));
INSERT INTO CONTRACT(cod_contract, data_inceput, data_sfarsit)
VALUES(SECVENTA_CONTRACT.NEXTVAL, TO_DATE('01-09-2023', 'dd-mm-yyyy'), TO_DATE('01-01-2025', 'dd-mm-yyyy'));

INSERT INTO FEL_DE_MANCARE(cod_fel_de_mancare, denumire, pret)
VALUES(SECVENTA_FEL_DE_MANCARE.NEXTVAL, 'Ciorba De Legume', 11);
INSERT INTO FEL_DE_MANCARE(cod_fel_de_mancare, denumire, pret)
VALUES(SECVENTA_FEL_DE_MANCARE.NEXTVAL, 'Ciorba De Perisoare', 12);
INSERT INTO FEL_DE_MANCARE(cod_fel_de_mancare, denumire, pret)
VALUES(SECVENTA_FEL_DE_MANCARE.NEXTVAL, 'Ciorba De Pui', 13);
INSERT INTO FEL_DE_MANCARE(cod_fel_de_mancare, denumire, pret)
VALUES(SECVENTA_FEL_DE_MANCARE.NEXTVAL, 'Supa De Legume', 7);
INSERT INTO FEL_DE_MANCARE(cod_fel_de_mancare, denumire, pret)
VALUES(SECVENTA_FEL_DE_MANCARE.NEXTVAL, 'Cartofi Prajiti', 15);
INSERT INTO FEL_DE_MANCARE(cod_fel_de_mancare, denumire, pret)
VALUES(SECVENTA_FEL_DE_MANCARE.NEXTVAL, 'Piuire', 12);
INSERT INTO FEL_DE_MANCARE(cod_fel_de_mancare, denumire, pret)
VALUES(SECVENTA_FEL_DE_MANCARE.NEXTVAL, 'Spaghete Carbonara', 22);
INSERT INTO FEL_DE_MANCARE(cod_fel_de_mancare, denumire, pret)
VALUES(SECVENTA_FEL_DE_MANCARE.NEXTVAL, 'Pizza', 27);
INSERT INTO FEL_DE_MANCARE(cod_fel_de_mancare, denumire, pret)
VALUES(SECVENTA_FEL_DE_MANCARE.NEXTVAL, 'Limonada', 7);
INSERT INTO FEL_DE_MANCARE(cod_fel_de_mancare, denumire, pret)
VALUES(SECVENTA_FEL_DE_MANCARE.NEXTVAL, 'Somon', 24);
INSERT INTO FEL_DE_MANCARE(cod_fel_de_mancare, denumire, pret)
VALUES(SECVENTA_FEL_DE_MANCARE.NEXTVAL, 'Sniitel De Pui', 14);
INSERT INTO FEL_DE_MANCARE(cod_fel_de_mancare, denumire, pret)
VALUES(SECVENTA_FEL_DE_MANCARE.NEXTVAL, 'Coca Cola', 7);
INSERT INTO FEL_DE_MANCARE(cod_fel_de_mancare, denumire, pret)
VALUES(SECVENTA_FEL_DE_MANCARE.NEXTVAL, 'Pepsi', 7);
INSERT INTO FEL_DE_MANCARE(cod_fel_de_mancare, denumire, pret)
VALUES(SECVENTA_FEL_DE_MANCARE.NEXTVAL, 'Sprite', 7);
INSERT INTO FEL_DE_MANCARE(cod_fel_de_mancare, denumire, pret)
VALUES(SECVENTA_FEL_DE_MANCARE.NEXTVAL, 'Fanta', 7);

INSERT INTO RESTAURANT(cod_restaurant, cod_hotel, cod_locatie, denumire, numar_stele, an_infiintare)
VALUES(SECVENTA_RESTAURANT.NEXTVAL, 1, 5, 'Roma', 3, 1995);
INSERT INTO RESTAURANT(cod_restaurant, cod_hotel, cod_locatie, denumire, numar_stele, an_infiintare)
VALUES(SECVENTA_RESTAURANT.NEXTVAL, 1, 2, 'Milano', 2, NULL);
INSERT INTO RESTAURANT(cod_restaurant, cod_hotel, cod_locatie, denumire, numar_stele, an_infiintare)
VALUES(SECVENTA_RESTAURANT.NEXTVAL, 1, 5, 'Savoia', NULL, 2000);
INSERT INTO RESTAURANT(cod_restaurant, cod_hotel, cod_locatie, denumire, numar_stele, an_infiintare)
VALUES(SECVENTA_RESTAURANT.NEXTVAL, 7, 3, 'Varsovia', 4, 2004);
INSERT INTO RESTAURANT(cod_restaurant, cod_hotel, cod_locatie, denumire, numar_stele, an_infiintare)
VALUES(SECVENTA_RESTAURANT.NEXTVAL, 1, 5, 'Napoli', NULL, 2008);
INSERT INTO RESTAURANT(cod_restaurant, cod_hotel, cod_locatie, denumire, numar_stele, an_infiintare)
VALUES(SECVENTA_RESTAURANT.NEXTVAL, 4, 7, 'Madrid', NULL, 1993);
INSERT INTO RESTAURANT(cod_restaurant, cod_hotel, cod_locatie, denumire, numar_stele, an_infiintare)
VALUES(SECVENTA_RESTAURANT.NEXTVAL, 1, 5, 'Paris', NULL, NULL);
INSERT INTO RESTAURANT(cod_restaurant, cod_hotel, cod_locatie, denumire, numar_stele, an_infiintare)
VALUES(SECVENTA_RESTAURANT.NEXTVAL, 2, 1, 'Lisabona', 3, NULL);

```



```

INSERT INTO ANGAJAT(cod_angajat, nume, prenume, CNP, numar_telefon, adresa_mail, salariu)
VALUES(SECVENTA_ANGAJAT.NEXTVAL, 'Popescu', 'Florin', '0000000000001', '0700000001', NULL, 4000);
INSERT INTO ANGAJAT(cod_angajat, nume, prenume, CNP, numar_telefon, adresa_mail, salariu)
VALUES(SECVENTA_ANGAJAT.NEXTVAL, 'Stoica', 'Marius', '0000000000002', '0700000002', 'stoicamarius@mail.com', 4100);
INSERT INTO ANGAJAT(cod_angajat, nume, prenume, CNP, numar_telefon, adresa_mail, salariu)
VALUES(SECVENTA_ANGAJAT.NEXTVAL, 'Stan', 'Ioan', '0000000000003', '0700000003', NULL, 4200);
INSERT INTO ANGAJAT(cod_angajat, nume, prenume, CNP, numar_telefon, adresa_mail, salariu)
VALUES(SECVENTA_ANGAJAT.NEXTVAL, 'Vasilescu', 'George', '0000000000004', '0700000004', 'vasilescugeorge@mail.ro', 5000);
INSERT INTO ANGAJAT(cod_angajat, nume, prenume, CNP, numar_telefon, adresa_mail, salariu)
VALUES(SECVENTA_ANGAJAT.NEXTVAL, 'Vasilescu', 'Adrian', '0000000000005', '0700000005', NULL, 4700);
INSERT INTO ANGAJAT(cod_angajat, nume, prenume, CNP, numar_telefon, adresa_mail, salariu)
VALUES(SECVENTA_ANGAJAT.NEXTVAL, 'Marinescu', 'Cristian', '0000000000007', '0700000007', NULL, 4300);
INSERT INTO ANGAJAT(cod_angajat, nume, prenume, CNP, numar_telefon, adresa_mail, salariu)
VALUES(SECVENTA_ANGAJAT.NEXTVAL, 'Florinescu', 'Cristian', '0000000000010', '0700000010', NULL, 4900);
INSERT INTO ANGAJAT(cod_angajat, nume, prenume, CNP, numar_telefon, adresa_mail, salariu)
VALUES(SECVENTA_ANGAJAT.NEXTVAL, 'Popa', 'Ion', '0000000000011', '0700000011', NULL, 4950);
INSERT INTO ANGAJAT(cod_angajat, nume, prenume, CNP, numar_telefon, adresa_mail, salariu)
VALUES(SECVENTA_ANGAJAT.NEXTVAL, 'Lupu', 'Daria', '0000000000012', '0700000012', NULL, 5500);
INSERT INTO ANGAJAT(cod_angajat, nume, prenume, CNP, numar_telefon, adresa_mail, salariu)
VALUES(SECVENTA_ANGAJAT.NEXTVAL, 'Adamescu', 'Luca', '0000000000013', '0700000013', NULL, 5500);
INSERT INTO ANGAJAT(cod_angajat, nume, prenume, CNP, numar_telefon, adresa_mail, salariu)
VALUES(SECVENTA_ANGAJAT.NEXTVAL, 'Ionescu', 'Mirela', '0000000000014', '0700000014', NULL, 5250);
INSERT INTO ANGAJAT(cod_angajat, nume, prenume, CNP, numar_telefon, adresa_mail, salariu)
VALUES(SECVENTA_ANGAJAT.NEXTVAL, 'Paraschiv', 'Darius', '0000000000015', '0700000015', 'paraschivdarius@mail.com', 5150);
INSERT INTO ANGAJAT(cod_angajat, nume, prenume, CNP, numar_telefon, adresa_mail, salariu)
VALUES(SECVENTA_ANGAJAT.NEXTVAL, 'Serban', 'Stefan', '0000000000016', '0700000016', NULL, 5050);
INSERT INTO ANGAJAT(cod_angajat, nume, prenume, CNP, numar_telefon, adresa_mail, salariu)
VALUES(SECVENTA_ANGAJAT.NEXTVAL, 'Victor', 'Camelia', '0000000000017', '0700000017', 'camelia.victor@mail.ro', 4750);
INSERT INTO ANGAJAT(cod_angajat, nume, prenume, CNP, numar_telefon, adresa_mail, salariu)
VALUES(SECVENTA_ANGAJAT.NEXTVAL, 'Dan', 'Oana', '0000000000018', '0700000018', 'dan_oana@mail.com', 4850);
INSERT INTO ANGAJAT(cod_angajat, nume, prenume, CNP, numar_telefon, adresa_mail, salariu)
VALUES(SECVENTA_ANGAJAT.NEXTVAL, 'Cosmin', 'Ciprian', '0000000000019', '0700000019', 'ciprian.cosmin@mail.ro', 5000);
INSERT INTO ANGAJAT(cod_angajat, nume, prenume, CNP, numar_telefon, adresa_mail, salariu)
VALUES(SECVENTA_ANGAJAT.NEXTVAL, 'Popescu', 'Iulian', '0000000000020', '0700000020', NULL, 5000);
INSERT INTO ANGAJAT(cod_angajat, nume, prenume, CNP, numar_telefon, adresa_mail, salariu)
VALUES(SECVENTA_ANGAJAT.NEXTVAL, 'Popescu', 'Lucian', '0000000000021', '0700000021', NULL, 5750);
INSERT INTO ANGAJAT(cod_angajat, nume, prenume, CNP, numar_telefon, adresa_mail, salariu)
VALUES(SECVENTA_ANGAJAT.NEXTVAL, 'Popescu', 'Raluca', '0000000000022', '0700000022', NULL, 6050);
INSERT INTO ANGAJAT(cod_angajat, nume, prenume, CNP, numar_telefon, adresa_mail, salariu)
VALUES(SECVENTA_ANGAJAT.NEXTVAL, 'Tudor', 'Serban', '0000000000023', '0700000023', 'serban.tudor@mail.com', 6350);
INSERT INTO ANGAJAT(cod_angajat, nume, prenume, CNP, numar_telefon, adresa_mail, salariu)
VALUES(SECVENTA_ANGAJAT.NEXTVAL, 'Octavian', 'Silviu', '0000000000024', '0700000024', NULL, 7050);
INSERT INTO ANGAJAT(cod_angajat, nume, prenume, CNP, numar_telefon, adresa_mail, salariu)
VALUES(SECVENTA_ANGAJAT.NEXTVAL, 'Popescu', 'Valentin', '0000000000025', '0700000025', NULL, 6750);
INSERT INTO ANGAJAT(cod_angajat, nume, prenume, CNP, numar_telefon, adresa_mail, salariu)
VALUES(SECVENTA_ANGAJAT.NEXTVAL, 'Emanuel', 'Anca', '0000000000026', '0700000026', 'emanuel.anca@mail.com', 6550);
INSERT INTO ANGAJAT(cod_angajat, nume, prenume, CNP, numar_telefon, adresa_mail, salariu)
VALUES(SECVENTA_ANGAJAT.NEXTVAL, 'Sorin', 'Darius', '0000000000027', '0700000027', 'sorin.darius@mail.ro', 6350);

```

```

INSERT INTO PAZNIC(cod_angajat, cod_zona_de_parcare)
VALUES(1, 1);
INSERT INTO PAZNIC(cod_angajat, cod_zona_de_parcare)
VALUES(2, NULL);
INSERT INTO PAZNIC(cod_angajat, cod_zona_de_parcare)
VALUES(3, 2);
INSERT INTO PAZNIC(cod_angajat, cod_zona_de_parcare)
VALUES(4, 2);
INSERT INTO PAZNIC(cod_angajat, cod_zona_de_parcare)
VALUES(5, 3);

INSERT INTO RECEPTIONIST(cod_angajat, cod_hotel)
VALUES(6, NULL);
INSERT INTO RECEPTIONIST(cod_angajat, cod_hotel)
VALUES(7, 2);
INSERT INTO RECEPTIONIST(cod_angajat, cod_hotel)
VALUES(8, 1);
INSERT INTO RECEPTIONIST(cod_angajat, cod_hotel)
VALUES(9, NULL);
INSERT INTO RECEPTIONIST(cod_angajat, cod_hotel)
VALUES(10, 1);

INSERT INTO CHELNER(cod_angajat, cod_restaurant)
VALUES(11, 1);
INSERT INTO CHELNER(cod_angajat, cod_restaurant)
VALUES(12, 1);
INSERT INTO CHELNER(cod_angajat, cod_restaurant)
VALUES(13, 2);
INSERT INTO CHELNER(cod_angajat, cod_restaurant)
VALUES(14, 2);
INSERT INTO CHELNER(cod_angajat, cod_restaurant)
VALUES(15, 4);

INSERT INTO BUCATAR(cod_angajat, cod_restaurant)
VALUES(16, 1);
INSERT INTO BUCATAR(cod_angajat, cod_restaurant)
VALUES(17, NULL);
INSERT INTO BUCATAR(cod_angajat, cod_restaurant)
VALUES(18, 1);
INSERT INTO BUCATAR(cod_angajat, cod_restaurant)
VALUES(19, NULL);
INSERT INTO BUCATAR(cod_angajat, cod_restaurant)
VALUES(20, 1);

INSERT INTO ADMINISTRATOR(cod_angajat, cod_hotel)
VALUES(21, 1);
INSERT INTO ADMINISTRATOR(cod_angajat, cod_hotel)
VALUES(22, 2);
INSERT INTO ADMINISTRATOR(cod_angajat, cod_hotel)
VALUES(23, 3);
INSERT INTO ADMINISTRATOR(cod_angajat, cod_hotel)
VALUES(24, 4);
INSERT INTO ADMINISTRATOR(cod_angajat, cod_hotel)
VALUES(25, NULL);

```

SQL Worksheet History

Worksheet Query Builder

```
INSERT INTO LOCATIE(cod_locatie, adresa)
VALUES (SEVENTA_LOCATIE.NEXTVAL, 'Zona 1');
INSERT INTO LOCATIE(cod_locatie, adresa)
VALUES (SEVENTA_LOCATIE.NEXTVAL, 'Zona 2');
INSERT INTO LOCATIE(cod_locatie, adresa)
VALUES (SEVENTA_LOCATIE.NEXTVAL, 'Zona 3');
INSERT INTO LOCATIE(cod_locatie, adresa)
VALUES (SEVENTA_LOCATIE.NEXTVAL, 'Zona 4');
INSERT INTO LOCATIE(cod_locatie, adresa)
VALUES (SEVENTA_LOCATIE.NEXTVAL, 'Zona 5');
INSERT INTO LOCATIE(cod_locatie, adresa)
VALUES (SEVENTA_LOCATIE.NEXTVAL, 'Zona 6');
INSERT INTO LOCATIE(cod_locatie, adresa)
VALUES (SEVENTA_LOCATIE.NEXTVAL, 'Zona 7');
INSERT INTO LOCATIE(cod_locatie, adresa)
VALUES (SEVENTA_LOCATIE.NEXTVAL, 'Centru');
INSERT INTO LOCATIE(cod_locatie, adresa)
VALUES (SEVENTA_LOCATIE.NEXTVAL, 'Periferie');

INSERT INTO HOTEL(cod hotel, denumire, numar stele
```

Script Output x Query Result x

Task completed in 0.103 seconds

1 row inserted.

1 row inserted.

SQL Worksheet History

Worksheet Query Builder

```
INSERT INTO HOTEL(cod_hotel, denumire, numar_stele, an_infiintare, cod_locatie)
VALUES(SECVENTA_HOTEL.NEXTVAL, 'Continental', 3, 1965, 1);
INSERT INTO HOTEL(cod_hotel, denumire, numar_stele, an_infiintare, cod_locatie)
VALUES(SECVENTA_HOTEL.NEXTVAL, 'Europa', 4, 1993, 1);
INSERT INTO HOTEL(cod_hotel, denumire, numar_stele, an_infiintare, cod_locatie)
VALUES(SECVENTA_HOTEL.NEXTVAL, 'Atena', 3, 1997, 1);
INSERT INTO HOTEL(cod_hotel, denumire, numar_stele, an_infiintare, cod_locatie)
VALUES(SECVENTA_HOTEL.NEXTVAL, 'Sofia', 2, 1983, 2);
INSERT INTO HOTEL(cod_hotel, denumire, numar_stele, an_infiintare, cod_locatie)
VALUES(SECVENTA_HOTEL.NEXTVAL, 'Mures', 4, 1975, 2);
INSERT INTO HOTEL(cod_hotel, denumire, numar_stele, an_infiintare, cod_locatie)
VALUES(SECVENTA_HOTEL.NEXTVAL, 'Banat', 2, 1968, 2);
INSERT INTO HOTEL(cod_hotel, denumire, numar_stele, an_infiintare, cod_locatie)
VALUES(SECVENTA_HOTEL.NEXTVAL, 'Transilvania', 3, 1970, 3);
INSERT INTO HOTEL(cod_hotel, denumire, numar_stele, an_infiintare, cod_locatie)
VALUES(SECVENTA_HOTEL.NEXTVAL, 'Moldova', NULL, NULL, 3);
INSERT INTO HOTEL(cod_hotel, denumire, numar_stele, an_infiintare, cod_locatie)
VALUES(SECVENTA_HOTEL.NEXTVAL, 'Dobrogea', 5, NULL, 9);
INSERT INTO HOTEL(cod_hotel, denumire, numar_stele, an_infiintare, cod_locatie)
VALUES(SECVENTA_HOTEL.NEXTVAL, 'Basarabia', 1, 1950, 9);
INSERT INTO HOTEL(cod_hotel, denumire, numar_stele, an_infiintare, cod_locatie)
VALUES(SECVENTA_HOTEL.NEXTVAL, 'Bucovina', NULL, NULL, 9);
INSERT INTO HOTEL(cod_hotel, denumire, numar_stele, an_infiintare, cod_locatie)
VALUES(SECVENTA_HOTEL.NEXTVAL, 'Transnistria', NULL, NULL, 8);
INSERT INTO HOTEL(cod_hotel, denumire, numar_stele, an_infiintare, cod_locatie)
VALUES(SECVENTA_HOTEL.NEXTVAL, 'Venetia', NULL, 1963, 8);
INSERT INTO HOTEL(cod_hotel, denumire, numar_stele, an_infiintare, cod_locatie)
VALUES(SECVENTA_HOTEL.NEXTVAL, 'Dunare', 3, NULL, 8);
INSERT INTO HOTEL(cod_hotel, denumire, numar_stele, an_infiintare, cod_locatie)
VALUES(SECVENTA_HOTEL.NEXTVAL, 'Carpati', NULL, 1968, 6);
```

Script Output x | Query Result x

Task completed in 0.116 seconds

1 row inserted.

1 row inserted.

SQL Worksheet | History

Worksheet | Query Builder

```
INSERT INTO HOTEL(cod_hotel, denumire, numar_stele, an_infiintare, cod_locatie)
VALUES(SECVENTA_HOTEL.NEXTVAL, 'Dunare', 3, NULL, 8);
INSERT INTO HOTEL(cod_hotel, denumire, numar_stele, an_infiintare, cod_locatie)
VALUES(SECVENTA_HOTEL.NEXTVAL, 'Carpati', NULL, 1968, 6);

INSERT INTO CAMERA(cod_camera, cod_hotel, numar)
VALUES(SECVENTA_CAMERA.NEXTVAL, 1, 1);
INSERT INTO CAMERA(cod_camera, cod_hotel, numar, etaj)
VALUES(SECVENTA_CAMERA.NEXTVAL, 1, 2, 1);
INSERT INTO CAMERA(cod_camera, cod_hotel, numar)
VALUES(SECVENTA_CAMERA.NEXTVAL, 1, 3);
INSERT INTO CAMERA(cod_camera, cod_hotel, numar, etaj)
VALUES(SECVENTA_CAMERA.NEXTVAL, 1, 4, 2);
INSERT INTO CAMERA(cod_camera, cod_hotel, numar)
VALUES(SECVENTA_CAMERA.NEXTVAL, 1, 5);
INSERT INTO CAMERA(cod_camera, cod_hotel, numar)
VALUES(SECVENTA_CAMERA.NEXTVAL, 2, 1);
INSERT INTO CAMERA(cod_camera, cod_hotel, numar, etaj)
VALUES(SECVENTA_CAMERA.NEXTVAL, 2, 2, 3);
INSERT INTO CAMERA(cod_camera, cod_hotel, numar, etaj)
VALUES(SECVENTA_CAMERA.NEXTVAL, 2, 3, 0);
INSERT INTO CAMERA(cod_camera, cod_hotel, numar, etaj)
VALUES(SECVENTA_CAMERA.NEXTVAL, 2, 4, 7);
INSERT INTO CAMERA(cod_camera, cod_hotel, numar)
VALUES(SECVENTA_CAMERA.NEXTVAL, 2, 5);
INSERT INTO CAMERA(cod_camera, cod_hotel, numar, etaj)
VALUES(SECVENTA_CAMERA.NEXTVAL, 3, 1, 1);

INSERT INTO FACILITATE(cod_facilitate, denumire)
VALUES(SECVENTA_FACILITATE.NEXTVAL, 'Camera Standard');
INSERT INTO FACILITATE(cod_facilitate, denumire)
```

Script Output | Query Result

Task completed in 0.094 seconds

1 row inserted.

1 row inserted.

SQL Worksheet History

Worksheet Query Builder

```

VALUES (SECVENTA_CAMERA.NEXTVAL, 2, 2, 3);
INSERT INTO CAMERA(cod_camera, cod_hotel, numar, etaj)
VALUES (SECVENTA_CAMERA.NEXTVAL, 2, 3, 0);
INSERT INTO CAMERA(cod_camera, cod_hotel, numar, etaj)
VALUES (SECVENTA_CAMERA.NEXTVAL, 2, 4, 7);
INSERT INTO CAMERA(cod_camera, cod_hotel, numar)
VALUES (SECVENTA_CAMERA.NEXTVAL, 2, 5);
INSERT INTO CAMERA(cod_camera, cod_hotel, numar, etaj)
VALUES (SECVENTA_CAMERA.NEXTVAL, 3, 1, 1);

INSERT INTO FACILITATE(cod_facilitate, denumire)
VALUES (SECVENTA_FACILITATE.NEXTVAL, 'Camera Standard');
INSERT INTO FACILITATE(cod_facilitate, denumire)
VALUES (SECVENTA_FACILITATE.NEXTVAL, 'Camera Dubla');
INSERT INTO FACILITATE(cod_facilitate, denumire)
VALUES (SECVENTA_FACILITATE.NEXTVAL, 'Camera Tripla');
INSERT INTO FACILITATE(cod_facilitate, denumire)
VALUES (SECVENTA_FACILITATE.NEXTVAL, 'Apartament');
INSERT INTO FACILITATE(cod_facilitate, denumire)
VALUES (SECVENTA_FACILITATE.NEXTVAL, 'Balcon');
INSERT INTO FACILITATE(cod_facilitate, denumire)
VALUES (SECVENTA_FACILITATE.NEXTVAL, 'Aer Conditionat');
INSERT INTO FACILITATE(cod_facilitate, denumire)
VALUES (SECVENTA_FACILITATE.NEXTVAL, 'Room Service');

INSERT INTO CLIENT(cod_client, nume, prenume, CNP, numar_telefon, adresa_mail)
VALUES (SECVENTA_CLIENT.NEXTVAL, 'Popescu', 'Ion', '1234567890123', '0701234567', 'popescu@popescu.ro');
INSERT INTO CLIENT(cod_client, nume, prenume, CNP, numar_telefon, adresa_mail)
VALUES (SECVENTA_CLIENT.NEXTVAL, 'Ionescu', 'Mircea', '3210987654321', '0798765432', 'mirescu@ionescu.ro');
INSERT INTO CLIENT(cod_client, nume, prenume, CNP, numar_telefon, adresa_mail)
VALUES (SECVENTA_CLIENT.NEXTVAL, 'Andrei', 'Ionescu', '0001112223334', '0700111222', 'andrei.i@ionescu.ro');

```

Script Output Query Result | Task completed in 0.062 seconds

1 row inserted.

1 row inserted.

SQL Worksheet | History

Worksheet | Query Builder

```

VALUES(SECVENTA_FACILITATE.NEXTVAL, 'Aer Conditionat');
INSERT INTO FACILITATE(cod_facilitate, denumire)
VALUES(SECVENTA_FACILITATE.NEXTVAL, 'Room Service');

INSERT INTO CLIENT(cod_client, nume, prenume, CNP, numar_telefon, adresa_mail)
VALUES(SECVENTA_CLIENT.NEXTVAL, 'Popescu', 'Ion', '1234567890123', '0701234567', 'popescu.ion@mail.ro');
INSERT INTO CLIENT(cod_client, nume, prenume, CNP, numar_telefon, adresa_mail)
VALUES(SECVENTA_CLIENT.NEXTVAL, 'Ionescu', 'Mircea', '3210987654321', '0798765432', 'mircea_ionescu@mail.com');
INSERT INTO CLIENT(cod_client, nume, prenume, CNP, numar_telefon, adresa_mail)
VALUES(SECVENTA_CLIENT.NEXTVAL, 'Andrei', 'Ionescu', '0001112223334', '0700111222', 'andreionescu@mail.ro');
INSERT INTO CLIENT(cod_client, nume, prenume, CNP, numar_telefon, adresa_mail)
VALUES(SECVENTA_CLIENT.NEXTVAL, 'Mihai', 'Paul', '9998887776665', '0799888777', 'paul.mihai@mail.com');
INSERT INTO CLIENT(cod_client, nume, prenume, CNP, numar_telefon, adresa_mail)
VALUES(SECVENTA_CLIENT.NEXTVAL, 'Bogdan', 'Mircea', '000001111222', '0777777777', NULL);
INSERT INTO CLIENT(cod_client, nume, prenume, CNP, numar_telefon, adresa_mail)
VALUES(SECVENTA_CLIENT.NEXTVAL, 'Georgescu', 'Gheorghe', '001001111222', '0777377777', 'georgescu_gheorghe@mail.ro');
INSERT INTO CLIENT(cod_client, nume, prenume, CNP, numar_telefon, adresa_mail)
VALUES(SECVENTA_CLIENT.NEXTVAL, 'Vlad', 'Teodor', '2020202020202', '0755000555', NULL);
INSERT INTO CLIENT(cod_client, nume, prenume, CNP, numar_telefon, adresa_mail)
VALUES(SECVENTA_CLIENT.NEXTVAL, 'Alexandru', 'Sebastian', '1313131313131', '0744333432', NULL);
INSERT INTO CLIENT(cod_client, nume, prenume, CNP, numar_telefon, adresa_mail)
VALUES(SECVENTA_CLIENT.NEXTVAL, 'Mihailescu', 'Florin', '1717171717171', '0700000000', NULL);
INSERT INTO CLIENT(cod_client, nume, prenume, CNP, numar_telefon, adresa_mail)
VALUES(SECVENTA_CLIENT.NEXTVAL, 'Andreeescu', 'Robert', '1919191919191', '0711222333', NULL);

INSERT INTO CAMERA_presinta_FACILITATE(cod_camera, cod_hotel, cod_facilitate)
VALUES(1, 1, 1);
INSERT INTO CAMERA_presinta_FACILITATE(cod_camera, cod_hotel, cod_facilitate)
VALUES(1, 1, 6);
INSERT INTO CAMERA_presinta_FACILITATE(cod_camera, cod_hotel, cod_facilitate)
VALUES(1, 1, 7);

1 row inserted.

1 row inserted.

```

Script Output | Query Result

Task completed in 0.222 seconds

SQL Worksheet History

Worksheet Query Builder

```
INSERT INTO CAMERA_presinta_FACILITATE(cod_camera, cod_hotel, cod_facilitate)
VALUES(1, 1, 1);
INSERT INTO CAMERA_presinta_FACILITATE(cod_camera, cod_hotel, cod_facilitate)
VALUES(1, 1, 6);
INSERT INTO CAMERA_presinta_FACILITATE(cod_camera, cod_hotel, cod_facilitate)
VALUES(1, 1, 7);
INSERT INTO CAMERA_presinta_FACILITATE(cod_camera, cod_hotel, cod_facilitate)
VALUES(9, 2, 3);
INSERT INTO CAMERA_presinta_FACILITATE(cod_camera, cod_hotel, cod_facilitate)
VALUES(9, 2, 5);
INSERT INTO CAMERA_presinta_FACILITATE(cod_camera, cod_hotel, cod_facilitate)
VALUES(9, 2, 6);
INSERT INTO CAMERA_presinta_FACILITATE(cod_camera, cod_hotel, cod_facilitate)
VALUES(9, 2, 7);
INSERT INTO CAMERA_presinta_FACILITATE(cod_camera, cod_hotel, cod_facilitate)
VALUES(10, 2, 4);
INSERT INTO CAMERA_presinta_FACILITATE(cod_camera, cod_hotel, cod_facilitate)
VALUES(10, 2, 5);
INSERT INTO CAMERA_presinta_FACILITATE(cod_camera, cod_hotel, cod_facilitate)
VALUES(10, 2, 6);
INSERT INTO CAMERA_presinta_FACILITATE(cod_camera, cod_hotel, cod_facilitate)
VALUES(8, 2, 2);
INSERT INTO CAMERA_presinta_FACILITATE(cod_camera, cod_hotel, cod_facilitate)
VALUES(8, 2, 7);
INSERT INTO CAMERA_presinta_FACILITATE(cod_camera, cod_hotel, cod_facilitate)
VALUES(7, 2, 7);
INSERT INTO CAMERA_presinta_FACILITATE(cod_camera, cod_hotel, cod_facilitate)
VALUES(7, 2, 6);
INSERT INTO CAMERA_presinta_FACILITATE(cod_camera, cod_hotel, cod_facilitate)
VALUES(7, 2, 1);
```

Script Output X | Query Result X

Task completed in 0.116 seconds

1 row inserted.

1 row inserted.

SQL Worksheet History

Worksheet Query Builder

```

VALUES (7, 2, 6);
INSERT INTO CAMERA_preszinta_FACILITATE(cod_camera, cod_hotel, cod_facilitate)
VALUES (7, 2, 1);

INSERT INTO CLIENT_rezerva_CAMERA(cod_client, cod_camera, cod_hotel, data_inceput, data_sfarsit)
VALUES(1, 1, 1, TO_DATE('02-09-2023', 'dd-mm-yyyy'), TO_DATE('08-09-2023', 'dd-mm-yyyy'));
INSERT INTO CLIENT_rezerva_CAMERA(cod_client, cod_camera, cod_hotel, data_inceput, data_sfarsit)
VALUES(2, 2, 1, TO_DATE('07-05-2023', 'dd-mm-yyyy'), TO_DATE('13-05-2023', 'dd-mm-yyyy'));
INSERT INTO CLIENT_rezerva_CAMERA(cod_client, cod_camera, cod_hotel, data_inceput, data_sfarsit)
VALUES(3, 3, 1, TO_DATE('26-04-2023', 'dd-mm-yyyy'), TO_DATE('02-05-2023', 'dd-mm-yyyy'));
INSERT INTO CLIENT_rezerva_CAMERA(cod_client, cod_camera, cod_hotel, data_inceput, data_sfarsit)
VALUES(4, 4, 1, TO_DATE('23-03-2023', 'dd-mm-yyyy'), TO_DATE('01-04-2023', 'dd-mm-yyyy'));
INSERT INTO CLIENT_rezerva_CAMERA(cod_client, cod_camera, cod_hotel, data_inceput, data_sfarsit)
VALUES(5, 5, 1, TO_DATE('23-10-2023', 'dd-mm-yyyy'), TO_DATE('01-11-2023', 'dd-mm-yyyy'));
INSERT INTO CLIENT_rezerva_CAMERA(cod_client, cod_camera, cod_hotel, data_inceput, data_sfarsit)
VALUES(5, 7, 2, TO_DATE('07-09-2023', 'dd-mm-yyyy'), TO_DATE('08-09-2023', 'dd-mm-yyyy'));
INSERT INTO CLIENT_rezerva_CAMERA(cod_client, cod_camera, cod_hotel, data_inceput, data_sfarsit)
VALUES(6, 7, 2, TO_DATE('10-03-2023', 'dd-mm-yyyy'), TO_DATE('14-03-2023', 'dd-mm-yyyy'));
INSERT INTO CLIENT_rezerva_CAMERA(cod_client, cod_camera, cod_hotel, data_inceput, data_sfarsit)
VALUES(6, 8, 2, TO_DATE('15-04-2023', 'dd-mm-yyyy'), TO_DATE('15-05-2023', 'dd-mm-yyyy'));
INSERT INTO CLIENT_rezerva_CAMERA(cod_client, cod_camera, cod_hotel, data_inceput, data_sfarsit)
VALUES(9, 9, 2, TO_DATE('17-03-2023', 'dd-mm-yyyy'), TO_DATE('19-05-2023', 'dd-mm-yyyy'));
INSERT INTO CLIENT_rezerva_CAMERA(cod_client, cod_camera, cod_hotel, data_inceput, data_sfarsit)
VALUES(9, 9, 2, TO_DATE('01-01-2023', 'dd-mm-yyyy'), TO_DATE('01-02-2023', 'dd-mm-yyyy'));

INSERT INTO ZONA_DE_PARCARE(cod_zona_de_parcare, cod_hotel, cod_locatie)
VALUES(SECVENTA_ZONA_DE_PARCARE.NEXTVAL, 1, 1);
INSERT INTO ZONA_DE_PARCARE(cod_zona_de_parcare, cod_hotel, cod_locatie)
VALUES(SECVENTA_ZONA_DE_PARCARE.NEXTVAL, 2, 2);
INSERT INTO ZONA_DE_PARCARE(cod_zona_de_parcare, cod_hotel, cod_locatie)
VALUES(SECVENTA_ZONA_DE_PARCARE.NEXTVAL, 3, 4);

1 row inserted.

1 row inserted.

```

Script Output X | Task completed in 0.074 seconds

SQL Worksheet | History

Worksheet | Query Builder

```

VALUES(6, 8, 2, TO_DATE('15-04-2023', 'dd-mm-yyyy'), TO_DATE('15-05-2023', 'dd-mm-yyyy'));
INSERT INTO CLIENT_rezerva_CAMERA(cod_client, cod_camera, cod_hotel, data_inceput, data_sfarsit)
VALUES(9, 9, 2, TO_DATE('17-03-2023', 'dd-mm-yyyy'), TO_DATE('19-05-2023', 'dd-mm-yyyy'));
INSERT INTO CLIENT_rezerva_CAMERA(cod_client, cod_camera, cod_hotel, data_inceput, data_sfarsit)
VALUES(9, 9, 2, TO_DATE('01-01-2023', 'dd-mm-yyyy'), TO_DATE('01-02-2023', 'dd-mm-yyyy'));

INSERT INTO ZONA_DE_PARCARE(cod_zona_de_parcare, cod_hotel, cod_locatie)
VALUES(SECVENTA_ZONA_DE_PARCARE.NEXTVAL, 1, 1);
INSERT INTO ZONA_DE_PARCARE(cod_zona_de_parcare, cod_hotel, cod_locatie)
VALUES(SECVENTA_ZONA_DE_PARCARE.NEXTVAL, 2, 2);
INSERT INTO ZONA_DE_PARCARE(cod_zona_de_parcare, cod_hotel, cod_locatie)
VALUES(SECVENTA_ZONA_DE_PARCARE.NEXTVAL, 3, 4);
INSERT INTO ZONA_DE_PARCARE(cod_zona_de_parcare, cod_hotel, cod_locatie)
VALUES(SECVENTA_ZONA_DE_PARCARE.NEXTVAL, 1, 1);
INSERT INTO ZONA_DE_PARCARE(cod_zona_de_parcare, cod_hotel, cod_locatie)
VALUES(SECVENTA_ZONA_DE_PARCARE.NEXTVAL, 4, 3);
INSERT INTO ZONA_DE_PARCARE(cod_zona_de_parcare, cod_hotel, cod_locatie)
VALUES(SECVENTA_ZONA_DE_PARCARE.NEXTVAL, 5, 5);
INSERT INTO ZONA_DE_PARCARE(cod_zona_de_parcare, cod_hotel, cod_locatie)
VALUES(SECVENTA_ZONA_DE_PARCARE.NEXTVAL, 6, 5);
INSERT INTO ZONA_DE_PARCARE(cod_zona_de_parcare, cod_hotel, cod_locatie)
VALUES(SECVENTA_ZONA_DE_PARCARE.NEXTVAL, 1, 5);
INSERT INTO ZONA_DE_PARCARE(cod_zona_de_parcare, cod_hotel, cod_locatie)
VALUES(SECVENTA_ZONA_DE_PARCARE.NEXTVAL, 7, 6);
INSERT INTO ZONA_DE_PARCARE(cod_zona_de_parcare, cod_hotel, cod_locatie)
VALUES(SECVENTA_ZONA_DE_PARCARE.NEXTVAL, 8, 6);

INSERT INTO LOC_DE_PARCARE(cod_loc_de_parcare, cod_zona_de_parcare, numar)
VALUES(SECVENTA_LOC_DE_PARCARE.NEXTVAL, 1, 1);
INSERT INTO LOC_DE_PARCARE(cod_loc_de_parcare, cod_zona_de_parcare, numar)
```

Script Output | Query Result

| Task completed in 0.262 seconds

1 row inserted.

1 row inserted.

SQL Worksheet History

Worksheet Query Builder

```
INSERT INTO LOC_DE_PARCARE(cod_loc_de_parcare, cod_zona_de_parcare, numar)
VALUES (SECVENTA_LOC_DE_PARCARE.NEXTVAL, 1, 1);
INSERT INTO LOC_DE_PARCARE(cod_loc_de_parcare, cod_zona_de_parcare, numar)
VALUES (SECVENTA_LOC_DE_PARCARE.NEXTVAL, 1, 2);
INSERT INTO LOC_DE_PARCARE(cod_loc_de_parcare, cod_zona_de_parcare, numar)
VALUES (SECVENTA_LOC_DE_PARCARE.NEXTVAL, 1, 3);
INSERT INTO LOC_DE_PARCARE(cod_loc_de_parcare, cod_zona_de_parcare, numar)
VALUES (SECVENTA_LOC_DE_PARCARE.NEXTVAL, 1, 4);
INSERT INTO LOC_DE_PARCARE(cod_loc_de_parcare, cod_zona_de_parcare, numar)
VALUES (SECVENTA_LOC_DE_PARCARE.NEXTVAL, 1, 5);
INSERT INTO LOC_DE_PARCARE(cod_loc_de_parcare, cod_zona_de_parcare, numar)
VALUES (SECVENTA_LOC_DE_PARCARE.NEXTVAL, 1, 6);
INSERT INTO LOC_DE_PARCARE(cod_loc_de_parcare, cod_zona_de_parcare, numar)
VALUES (SECVENTA_LOC_DE_PARCARE.NEXTVAL, 2, 1);
INSERT INTO LOC_DE_PARCARE(cod_loc_de_parcare, cod_zona_de_parcare, numar)
VALUES (SECVENTA_LOC_DE_PARCARE.NEXTVAL, 2, 2);
INSERT INTO LOC_DE_PARCARE(cod_loc_de_parcare, cod_zona_de_parcare, numar)
VALUES (SECVENTA_LOC_DE_PARCARE.NEXTVAL, 2, 3);
INSERT INTO LOC_DE_PARCARE(cod_loc_de_parcare, cod_zona_de_parcare, numar)
VALUES (SECVENTA_LOC_DE_PARCARE.NEXTVAL, 2, 4);
INSERT INTO LOC_DE_PARCARE(cod_loc_de_parcare, cod_zona_de_parcare, numar)
VALUES (SECVENTA_LOC_DE_PARCARE.NEXTVAL, 3, 1);
INSERT INTO LOC_DE_PARCARE(cod_loc_de_parcare, cod_zona_de_parcare, numar)
VALUES (SECVENTA_LOC_DE_PARCARE.NEXTVAL, 3, 2);
INSERT INTO LOC_DE_PARCARE(cod_loc_de_parcare, cod_zona_de_parcare, numar)
VALUES (SECVENTA_LOC_DE_PARCARE.NEXTVAL, 3, 3);
INSERT INTO LOC_DE_PARCARE(cod_loc_de_parcare, cod_zona_de_parcare, numar)
VALUES (SECVENTA_LOC_DE_PARCARE.NEXTVAL, 3, 4);
INSERT INTO LOC_DE_PARCARE(cod_loc_de_parcare, cod_zona_de_parcare, numar)
VALUES (SECVENTA_LOC_DE_PARCARE.NEXTVAL, 3, 5);
```

Script Output x | Query Result x

Query Result: Task completed in 0.1 seconds

1 row inserted.

1 row inserted.

SQL Worksheet | History

0.15800001 seconds

Worksheet | Query Builder

```

INSERT INTO LOC_DE_PARCARE(cod_loc_de_parcare, cod_zona_de_parcare, numar)
VALUES(SECVENTA_LOC_DE_PARCARE.NEXTVAL, 3, 1);
INSERT INTO LOC_DE_PARCARE(cod_loc_de_parcare, cod_zona_de_parcare, numar)
VALUES(SECVENTA_LOC_DE_PARCARE.NEXTVAL, 3, 2);
INSERT INTO LOC_DE_PARCARE(cod_loc_de_parcare, cod_zona_de_parcare, numar)
VALUES(SECVENTA_LOC_DE_PARCARE.NEXTVAL, 3, 3);
INSERT INTO LOC_DE_PARCARE(cod_loc_de_parcare, cod_zona_de_parcare, numar)
VALUES(SECVENTA_LOC_DE_PARCARE.NEXTVAL, 3, 4);
INSERT INTO LOC_DE_PARCARE(cod_loc_de_parcare, cod_zona_de_parcare, numar)
VALUES(SECVENTA_LOC_DE_PARCARE.NEXTVAL, 3, 5);

INSERT INTO FURNIZOR(cod_furnizor, nume)
VALUES(SECVENTA_FURNIZOR.NEXTVAL, 'Brutaria Iberia S.R.L.');
INSERT INTO FURNIZOR(cod_furnizor, nume)
VALUES(SECVENTA_FURNIZOR.NEXTVAL, 'Macelaria Creta S.R.L.');
INSERT INTO FURNIZOR(cod_furnizor, nume)
VALUES(SECVENTA_FURNIZOR.NEXTVAL, 'Apa Dorna S.R.L.');
INSERT INTO FURNIZOR(cod_furnizor, nume)
VALUES(SECVENTA_FURNIZOR.NEXTVAL, 'Brutaria Pavel S.R.L.');
INSERT INTO FURNIZOR(cod_furnizor, nume)
VALUES(SECVENTA_FURNIZOR.NEXTVAL, 'Macelarie Centru S.R.L.');
INSERT INTO FURNIZOR(cod_furnizor, nume)
VALUES(SECVENTA_FURNIZOR.NEXTVAL, 'Apa Carpatica S.R.L.');

INSERT INTO CONTRACT(cod_contract, data_inceput, data_sfarsit)
VALUES(SECVENTA_CONTRACT.NEXTVAL, TO_DATE('02-09-2023', 'dd-mm-yyyy'), TO_DATE('02-12-2023', 'dd-mm-yyyy'));
INSERT INTO CONTRACT(cod_contract, data_inceput, data_sfarsit)
VALUES(SECVENTA_CONTRACT.NEXTVAL, TO_DATE('01-01-2024', 'dd-mm-yyyy'), TO_DATE('01-01-2025', 'dd-mm-yyyy'));
INSERT INTO CONTRACT(cod_contract, data_inceput, data_sfarsit)
VALUES(SECVENTA_CONTRACT.NEXTVAL, TO_DATE('01-01-2023', 'dd-mm-yyyy'), TO_DATE('01-07-2023', 'dd-mm-yyyy'));
INSERT INTO CONTRACT(cod_contract, data_inceput, data_sfarsit)
```

Script Output | Query Result

Task completed in 0.158 seconds

1 row inserted.

1 row inserted.

SQL Worksheet History

Worksheet Query Builder

```

INSERT INTO FURNIZOR(cod_furnizor, nume)
VALUES(SECVENTA_FURNIZOR.NEXTVAL, 'Apa Dorna S.R.L.');
INSERT INTO FURNIZOR(cod_furnizor, nume)
VALUES(SECVENTA_FURNIZOR.NEXTVAL, 'Brutaria Pavel S.R.L.');
INSERT INTO FURNIZOR(cod_furnizor, nume)
VALUES(SECVENTA_FURNIZOR.NEXTVAL, 'Macelarie Centru S.R.L.');
INSERT INTO FURNIZOR(cod_furnizor, nume)
VALUES(SECVENTA_FURNIZOR.NEXTVAL, 'Apa Carpatica S.R.L.');

INSERT INTO CONTRACT(cod_contract, data_inceput, data_sfarsit)
VALUES(SECVENTA_CONTRACT.NEXTVAL, TO_DATE('02-09-2023', 'dd-mm-yyyy'), TO_DATE('02-12-2023', 'dd-mm-yyyy'));
INSERT INTO CONTRACT(cod_contract, data_inceput, data_sfarsit)
VALUES(SECVENTA_CONTRACT.NEXTVAL, TO_DATE('01-01-2024', 'dd-mm-yyyy'), TO_DATE('01-01-2025', 'dd-mm-yyyy'));
INSERT INTO CONTRACT(cod_contract, data_inceput, data_sfarsit)
VALUES(SECVENTA_CONTRACT.NEXTVAL, TO_DATE('01-01-2023', 'dd-mm-yyyy'), TO_DATE('01-07-2023', 'dd-mm-yyyy'));
INSERT INTO CONTRACT(cod_contract, data_inceput, data_sfarsit)
VALUES(SECVENTA_CONTRACT.NEXTVAL, TO_DATE('01-01-2023', 'dd-mm-yyyy'), TO_DATE('01-08-2023', 'dd-mm-yyyy'));
INSERT INTO CONTRACT(cod_contract, data_inceput, data_sfarsit)
VALUES(SECVENTA_CONTRACT.NEXTVAL, TO_DATE('01-09-2023', 'dd-mm-yyyy'), TO_DATE('01-01-2024', 'dd-mm-yyyy'));
INSERT INTO CONTRACT(cod_contract, data_inceput, data_sfarsit)
VALUES(SECVENTA_CONTRACT.NEXTVAL, TO_DATE('01-03-2023', 'dd-mm-yyyy'), TO_DATE('01-04-2023', 'dd-mm-yyyy'));
INSERT INTO CONTRACT(cod_contract, data_inceput, data_sfarsit)
VALUES(SECVENTA_CONTRACT.NEXTVAL, TO_DATE('01-04-2023', 'dd-mm-yyyy'), TO_DATE('12-05-2023', 'dd-mm-yyyy'));
INSERT INTO CONTRACT(cod_contract, data_inceput, data_sfarsit)
VALUES(SECVENTA_CONTRACT.NEXTVAL, TO_DATE('26-04-2023', 'dd-mm-yyyy'), TO_DATE('02-06-2023', 'dd-mm-yyyy'));
INSERT INTO CONTRACT(cod_contract, data_inceput, data_sfarsit)
VALUES(SECVENTA_CONTRACT.NEXTVAL, TO_DATE('01-09-2023', 'dd-mm-yyyy'), TO_DATE('01-01-2025', 'dd-mm-yyyy'));

INSERT INTO FEL_DE_MANCARE(cod_fel_de_mancare, denumire, pret)
VALUES(SECVENTA_FEL_DE_MANCARE.NEXTVAL, 'Ciorba De Legume', 11);
INSERT INTO FEL_DE_MANCARE(cod_fel_de_mancare, denumire, pret)


```

Script Output x | Query Result x

Task completed in 0.19 seconds

1 row inserted.

1 row inserted.

SQL Worksheet History

Worksheet Query Builder

```
INSERT INTO FEL_DE_MANCARE(cod_fel_de_mancare, denumire, pret)
VALUES(SECVENTA_FEL_DE_MANCARE.NEXTVAL, 'Ciorba De Legume', 11);
INSERT INTO FEL_DE_MANCARE(cod_fel_de_mancare, denumire, pret)
VALUES(SECVENTA_FEL_DE_MANCARE.NEXTVAL, 'Ciorba De Perisoare', 12);
INSERT INTO FEL_DE_MANCARE(cod_fel_de_mancare, denumire, pret)
VALUES(SECVENTA_FEL_DE_MANCARE.NEXTVAL, 'Ciorba De Pui', 13);
INSERT INTO FEL_DE_MANCARE(cod_fel_de_mancare, denumire, pret)
VALUES(SECVENTA_FEL_DE_MANCARE.NEXTVAL, 'Supa De Legume', 7);
INSERT INTO FEL_DE_MANCARE(cod_fel_de_mancare, denumire, pret)
VALUES(SECVENTA_FEL_DE_MANCARE.NEXTVAL, 'Cartofi Prajiti', 15);
INSERT INTO FEL_DE_MANCARE(cod_fel_de_mancare, denumire, pret)
VALUES(SECVENTA_FEL_DE_MANCARE.NEXTVAL, 'Piure', 12);
INSERT INTO FEL_DE_MANCARE(cod_fel_de_mancare, denumire, pret)
VALUES(SECVENTA_FEL_DE_MANCARE.NEXTVAL, 'Spaghete Carbonara', 22);
INSERT INTO FEL_DE_MANCARE(cod_fel_de_mancare, denumire, pret)
VALUES(SECVENTA_FEL_DE_MANCARE.NEXTVAL, 'Pizza', 27);
INSERT INTO FEL_DE_MANCARE(cod_fel_de_mancare, denumire, pret)
VALUES(SECVENTA_FEL_DE_MANCARE.NEXTVAL, 'Limonada', 7);
INSERT INTO FEL_DE_MANCARE(cod_fel_de_mancare, denumire, pret)
VALUES(SECVENTA_FEL_DE_MANCARE.NEXTVAL, 'Somon', 24);
INSERT INTO FEL_DE_MANCARE(cod_fel_de_mancare, denumire, pret)
VALUES(SECVENTA_FEL_DE_MANCARE.NEXTVAL, 'Snitel De Pui', 14);
INSERT INTO FEL_DE_MANCARE(cod_fel_de_mancare, denumire, pret)
VALUES(SECVENTA_FEL_DE_MANCARE.NEXTVAL, 'Coca Cola', 7);
INSERT INTO FEL_DE_MANCARE(cod_fel_de_mancare, denumire, pret)
VALUES(SECVENTA_FEL_DE_MANCARE.NEXTVAL, 'Pepsi', 7);
INSERT INTO FEL_DE_MANCARE(cod_fel_de_mancare, denumire, pret)
VALUES(SECVENTA_FEL_DE_MANCARE.NEXTVAL, 'Sprite', 7);
INSERT INTO FEL_DE_MANCARE(cod_fel_de_mancare, denumire, pret)
VALUES(SECVENTA_FEL_DE_MANCARE.NEXTVAL, 'Fanta', 7);
```

Script Output X Query Result X

redo undo save print Task completed in 0.114 seconds

1 row inserted.

1 row inserted.

SQL Worksheet | History

Worksheet | Query Builder

```

VALUES (SECVENTA_FEL_DE_MANCARE.NEXTVAL, 'Coca Cola', 7);
INSERT INTO FEL_DE_MANCARE(cod_fel_de_mancare, denumire, pret)
VALUES (SECVENTA_FEL_DE_MANCARE.NEXTVAL, 'Pepsi', 7);
INSERT INTO FEL_DE_MANCARE(cod_fel_de_mancare, denumire, pret)
VALUES (SECVENTA_FEL_DE_MANCARE.NEXTVAL, 'Sprite', 7);
INSERT INTO FEL_DE_MANCARE(cod_fel_de_mancare, denumire, pret)
VALUES (SECVENTA_FEL_DE_MANCARE.NEXTVAL, 'Fanta', 7);

INSERT INTO RESTAURANT(cod_restaurant, cod_hotel, cod_locatie, denumire, numar_stele, an_infiintare)
VALUES (SECVENTA_RESTAURANT.NEXTVAL, 1, 5, 'Roma', 3, 1995);
INSERT INTO RESTAURANT(cod_restaurant, cod_hotel, cod_locatie, denumire, numar_stele, an_infiintare)
VALUES (SECVENTA_RESTAURANT.NEXTVAL, 1, 2, 'Milano', 2, NULL);
INSERT INTO RESTAURANT(cod_restaurant, cod_hotel, cod_locatie, denumire, numar_stele, an_infiintare)
VALUES (SECVENTA_RESTAURANT.NEXTVAL, 1, 5, 'Savoia', NULL, 2000);
INSERT INTO RESTAURANT(cod_restaurant, cod_hotel, cod_locatie, denumire, numar_stele, an_infiintare)
VALUES (SECVENTA_RESTAURANT.NEXTVAL, 7, 3, 'Varsovia', 4, 2004);
INSERT INTO RESTAURANT(cod_restaurant, cod_hotel, cod_locatie, denumire, numar_stele, an_infiintare)
VALUES (SECVENTA_RESTAURANT.NEXTVAL, 1, 5, 'Napoli', NULL, 2008);
INSERT INTO RESTAURANT(cod_restaurant, cod_hotel, cod_locatie, denumire, numar_stele, an_infiintare)
VALUES (SECVENTA_RESTAURANT.NEXTVAL, 4, 7, 'Madrid', NULL, 1993);
INSERT INTO RESTAURANT(cod_restaurant, cod_hotel, cod_locatie, denumire, numar_stele, an_infiintare)
VALUES (SECVENTA_RESTAURANT.NEXTVAL, 1, 5, 'Paris', NULL, NULL);
INSERT INTO RESTAURANT(cod_restaurant, cod_hotel, cod_locatie, denumire, numar_stele, an_infiintare)
VALUES (SECVENTA_RESTAURANT.NEXTVAL, 2, 1, 'Lisabona', 3, NULL);

INSERT INTO RESTAURANT_serveste_FEL_DE_MANCARE(cod_restaurant, cod_fel_de_mancare)
VALUES (1, 1);
INSERT INTO RESTAURANT_serveste_FEL_DE_MANCARE(cod_restaurant, cod_fel_de_mancare)
VALUES (1, 2);
INSERT INTO RESTAURANT_serveste_FEL_DE_MANCARE(cod_restaurant, cod_fel_de_mancare)
VALUES (2, 3);

```

Script Output | Query Result

Task completed in 0.207 seconds

1 row inserted.

1 row inserted.

SQL Worksheet History

Worksheet Query Builder

```

VALUES(1, 1);
INSERT INTO RESTAURANT_serveste_FEL_DE_MANCARE(cod_restaurant, cod_fel_de_mancare)
VALUES(1, 2);
INSERT INTO RESTAURANT_serveste_FEL_DE_MANCARE(cod_restaurant, cod_fel_de_mancare)
VALUES(2, 3);
INSERT INTO RESTAURANT_serveste_FEL_DE_MANCARE(cod_restaurant, cod_fel_de_mancare)
VALUES(2, 4);
INSERT INTO RESTAURANT_serveste_FEL_DE_MANCARE(cod_restaurant, cod_fel_de_mancare)
VALUES(2, 5);
INSERT INTO RESTAURANT_serveste_FEL_DE_MANCARE(cod_restaurant, cod_fel_de_mancare)
VALUES(1, 5);
INSERT INTO RESTAURANT_serveste_FEL_DE_MANCARE(cod_restaurant, cod_fel_de_mancare)
VALUES(1, 7);
INSERT INTO RESTAURANT_serveste_FEL_DE_MANCARE(cod_restaurant, cod_fel_de_mancare)
VALUES(3, 2);
INSERT INTO RESTAURANT_serveste_FEL_DE_MANCARE(cod_restaurant, cod_fel_de_mancare)
VALUES(3, 5);
INSERT INTO RESTAURANT_serveste_FEL_DE_MANCARE(cod_restaurant, cod_fel_de_mancare)
VALUES(4, 2);
INSERT INTO RESTAURANT_serveste_FEL_DE_MANCARE(cod_restaurant, cod_fel_de_mancare)
VALUES(5, 2);
INSERT INTO RESTAURANT_serveste_FEL_DE_MANCARE(cod_restaurant, cod_fel_de_mancare)
VALUES(6, 3);
INSERT INTO RESTAURANT_serveste_FEL_DE_MANCARE(cod_restaurant, cod_fel_de_mancare)
VALUES(8, 4);
INSERT INTO RESTAURANT_serveste_FEL_DE_MANCARE(cod_restaurant, cod_fel_de_mancare)
VALUES(1, 6);
INSERT INTO RESTAURANT_serveste_FEL_DE_MANCARE(cod_restaurant, cod_fel_de_mancare)
VALUES(8, 6);
INSERT INTO RESTAURANT_serveste_FEL_DE_MANCARE(cod_restaurant, cod_fel_de_mancare)
VALUES(1, 10);
INSERT INTO RESTAURANT_serveste_FEL_DE_MANCARE(cod_restaurant, cod_fel_de_mancare)
VALUES(2, 8);

INSERT INTO FURNIZOR semneaza cu RESTAURANT mai multe CONTRACTE(cod_restaurant, cod_furnizo

```

Script Output X | Query Result X

| Task completed in 0.122 seconds

1 row inserted.

1 row inserted.

SQL Worksheet History

Worksheet Query Builder

```
VALUES (2, 6);

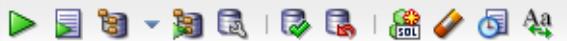
INSERT INTO FURNIZOR_semneaza_cu_RESTAURANT_mai_multe_CONTRACTE(cod_restaurant, cod_furnizor, cod_contract)
VALUES(1, 1, 1);
INSERT INTO FURNIZOR_semneaza_cu_RESTAURANT_mai_multe_CONTRACTE(cod_restaurant, cod_furnizor, cod_contract)
VALUES(1, 1, 2);
INSERT INTO FURNIZOR_semneaza_cu_RESTAURANT_mai_multe_CONTRACTE(cod_restaurant, cod_furnizor, cod_contract)
VALUES(1, 1, 3);
INSERT INTO FURNIZOR_semneaza_cu_RESTAURANT_mai_multe_CONTRACTE(cod_restaurant, cod_furnizor, cod_contract)
VALUES(1, 1, 4);
INSERT INTO FURNIZOR_semneaza_cu_RESTAURANT_mai_multe_CONTRACTE(cod_restaurant, cod_furnizor, cod_contract)
VALUES(2, 1, 5);
INSERT INTO FURNIZOR_semneaza_cu_RESTAURANT_mai_multe_CONTRACTE(cod_restaurant, cod_furnizor, cod_contract)
VALUES(2, 1, 6);
INSERT INTO FURNIZOR_semneaza_cu_RESTAURANT_mai_multe_CONTRACTE(cod_restaurant, cod_furnizor, cod_contract)
VALUES(2, 2, 1);
INSERT INTO FURNIZOR_semneaza_cu_RESTAURANT_mai_multe_CONTRACTE(cod_restaurant, cod_furnizor, cod_contract)
VALUES(2, 2, 6);
INSERT INTO FURNIZOR_semneaza_cu_RESTAURANT_mai_multe_CONTRACTE(cod_restaurant, cod_furnizor, cod_contract)
VALUES(3, 4, 1);
INSERT INTO FURNIZOR_semneaza_cu_RESTAURANT_mai_multe_CONTRACTE(cod_restaurant, cod_furnizor, cod_contract)
VALUES(3, 4, 2);
INSERT INTO FURNIZOR_semneaza_cu_RESTAURANT_mai_multe_CONTRACTE(cod_restaurant, cod_furnizor, cod_contract)
VALUES(3, 5, 3);
INSERT INTO FURNIZOR_semneaza_cu_RESTAURANT_mai_multe_CONTRACTE(cod_restaurant, cod_furnizor, cod_contract)
VALUES(3, 6, 4);
INSERT INTO FURNIZOR_semneaza_cu_RESTAURANT_mai_multe_CONTRACTE(cod_restaurant, cod_furnizor, cod_contract)
VALUES(3, 6, 5);
INSERT INTO FURNIZOR_semneaza_cu_RESTAURANT_mai_multe_CONTRACTE(cod_restaurant, cod_furnizor, cod_contract)
VALUES(3, 6, 6);
INSERT INTO FURNIZOR_semneaza_cu_RESTAURANT_mai_multe_CONTRACTE(cod_restaurant, cod_furnizor, cod_contract)
VALUES(4, 5, 5);
INSERT INTO FURNIZOR_semneaza_cu_RESTAURANT_mai_multe_CONTRACTE(cod_restaurant, cod_furnizor, cod_contract)
VALUES(4, 5, 6);
```

Script Output x | Query Result x

Script Output | Task completed in 0.084 seconds

```
1 row inserted.

1 row inserted.
```



Worksheet Query Builder

```

INSERT INTO ANGAJAT(cod_angajat, nume, prenume, CNP, numar_telefon, adresa_mail, salariu)
VALUES(SECVENTA_ANGAJAT.NEXTVAL, 'Popescu', 'Florin', '000000000001', '0700000001', NULL, 4000);
INSERT INTO ANGAJAT(cod_angajat, nume, prenume, CNP, numar_telefon, adresa_mail, salariu)
VALUES(SECVENTA_ANGAJAT.NEXTVAL, 'Stoica', 'Mirius', '000000000002', '0700000002', 'stoiceamarius@mail.com', 4100);
INSERT INTO ANGAJAT(cod_angajat, nume, prenume, CNP, numar_telefon, adresa_mail, salariu)
VALUES(SECVENTA_ANGAJAT.NEXTVAL, 'Stan', 'Ioan', '000000000003', '0700000003', NULL, 4200);
INSERT INTO ANGAJAT(cod_angajat, nume, prenume, CNP, numar_telefon, adresa_mail, salariu)
VALUES(SECVENTA_ANGAJAT.NEXTVAL, 'Vasilescu', 'George', '000000000004', 'vasilescugeorge@mail.ro', 5000);
INSERT INTO ANGAJAT(cod_angajat, nume, prenume, CNP, numar_telefon, adresa_mail, salariu)
VALUES(SECVENTA_ANGAJAT.NEXTVAL, 'Vasilescu', 'Adrian', '000000000005', '0700000005', NULL, 4700);
INSERT INTO ANGAJAT(cod_angajat, nume, prenume, CNP, numar_telefon, adresa_mail, salariu)
VALUES(SECVENTA_ANGAJAT.NEXTVAL, 'Radu', 'Andrei', '000000000006', '0700000006', NULL, 4500);
INSERT INTO ANGAJAT(cod_angajat, nume, prenume, CNP, numar_telefon, adresa_mail, salariu)
VALUES(SECVENTA_ANGAJAT.NEXTVAL, 'Marin', 'Iacob', '000000000009', '0700000009', NULL, 5200);
INSERT INTO ANGAJAT(cod_angajat, nume, prenume, CNP, numar_telefon, adresa_mail, salariu)
VALUES(SECVENTA_ANGAJAT.NEXTVAL, 'Marinescu', 'Cristian', '000000000007', '0700000007', NULL, 4300);
INSERT INTO ANGAJAT(cod_angajat, nume, prenume, CNP, numar_telefon, adresa_mail, salariu)
VALUES(SECVENTA_ANGAJAT.NEXTVAL, 'Marinescu', 'Dumitru', '000000000008', '0700000008', 'dumitru.marinescu@mail.com', 5300);
INSERT INTO ANGAJAT(cod_angajat, nume, prenume, CNP, numar_telefon, adresa_mail, salariu)
VALUES(SECVENTA_ANGAJAT.NEXTVAL, 'Lupa', 'Daria', '000000000012', '0700000012', NULL, 5500);
INSERT INTO ANGAJAT(cod_angajat, nume, prenume, CNP, numar_telefon, adresa_mail, salariu)
VALUES(SECVENTA_ANGAJAT.NEXTVAL, 'Adamescu', 'Luca', '000000000013', '0700000013', NULL, 5500);
INSERT INTO ANGAJAT(cod_angajat, nume, prenume, CNP, numar_telefon, adresa_mail, salariu)
VALUES(SECVENTA_ANGAJAT.NEXTVAL, 'Ionescu', 'Mirela', '000000000014', '0700000014', NULL, 5250);
INSERT INTO ANGAJAT(cod_angajat, nume, prenume, CNP, numar_telefon, adresa_mail, salariu)
VALUES(SECVENTA_ANGAJAT.NEXTVAL, 'Paraschiv', 'Darius', '000000000015', '0700000015', 'paraschivdarius@mail.com', 5150);
INSERT INTO ANGAJAT(cod_angajat, nume, prenume, CNP, numar_telefon, adresa_mail, salariu)
VALUES(SECVENTA_ANGAJAT.NEXTVAL, 'Serban', 'Stefan', '000000000016', '0700000016', NULL, 5050);
INSERT INTO ANGAJAT(cod_angajat, nume, prenume, CNP, numar_telefon, adresa_mail, salariu)
VALUES(SECVENTA_ANGAJAT.NEXTVAL, 'Victor', 'Camelia', '000000000017', '0700000017', 'camelia.victor@mail.ro', 4750);
INSERT INTO ANGAJAT(cod_angajat, nume, prenume, CNP, numar_telefon, adresa_mail, salariu)
VALUES(SECVENTA_ANGAJAT.NEXTVAL, 'Dan', 'Oana', '000000000018', '0700000018', 'dan_oana@mail.com', 4850);
INSERT INTO ANGAJAT(cod_angajat, nume, prenume, CNP, numar_telefon, adresa_mail, salariu)
VALUES(SECVENTA_ANGAJAT.NEXTVAL, 'Cosmin', 'Ciprian', '000000000019', '0700000019', 'ciprian.cosmin@mail.ro', 5000);
INSERT INTO ANGAJAT(cod_angajat, nume, prenume, CNP, numar_telefon, adresa_mail, salariu)
VALUES(SECVENTA_ANGAJAT.NEXTVAL, 'Popescu', 'Iulian', '000000000020', '0700000020', NULL, 5000);
INSERT INTO ANGAJAT(cod_angajat, nume, prenume, CNP, numar_telefon, adresa_mail, salariu)
VALUES(SECVENTA_ANGAJAT.NEXTVAL, 'Popescu', 'Lucian', '000000000021', '0700000021', NULL, 5750);
INSERT INTO ANGAJAT(cod_angajat, nume, prenume, CNP, numar_telefon, adresa_mail, salariu)
VALUES(SECVENTA_ANGAJAT.NEXTVAL, 'Popescu', 'Raluca', '000000000022', '0700000022', NULL, 6050);
INSERT INTO ANGAJAT(cod_angajat, nume, prenume, CNP, numar_telefon, adresa_mail, salariu)
VALUES(SECVENTA_ANGAJAT.NEXTVAL, 'Tudor', 'Serban', '000000000023', '0700000023', 'serban.tudor@mail.com', 6350);
INSERT INTO ANGAJAT(cod_angajat, nume, prenume, CNP, numar_telefon, adresa_mail, salariu)
VALUES(SECVENTA_ANGAJAT.NEXTVAL, 'Octavian', 'Silviu', '000000000024', '0700000024', NULL, 7050);
INSERT INTO ANGAJAT(cod_angajat, nume, prenume, CNP, numar_telefon, adresa_mail, salariu)
VALUES(SECVENTA_ANGAJAT.NEXTVAL, 'Valentin', 'Catalin', '000000000025', '0700000025', NULL, 6750);
INSERT INTO ANGAJAT(cod_angajat, nume, prenume, CNP, numar_telefon, adresa_mail, salariu)
VALUES(SECVENTA_ANGAJAT.NEXTVAL, 'Emanuel', 'Anca', '000000000026', '0700000026', 'emanuel.anca@mail.com', 6550);
INSERT INTO ANGAJAT(cod_angajat, nume, prenume, CNP, numar_telefon, adresa_mail, salariu)
VALUES(SECVENTA_ANGAJAT.NEXTVAL, 'Sorin', 'Darius', '000000000027', '0700000027', 'sorin.darius@mail.ro', 6350);

```

Script Output

Query Result



Task completed in 0.128 seconds

1 row inserted.

1 row inserted.

1 row inserted.

1 row inserted.



Worksheet

Query Builder

```

VALUES (SECVENTA_ANGAJAT.NEXTVAL, 'Sorin', 'Darius', '000000000027', '0700000027', 'sorin.darius@mail.ro', 6350);

INSERT INTO PAZNIC(cod_angajat, cod_zona_de_parcare)
VALUES(1, 1);
INSERT INTO PAZNIC(cod_angajat, cod_zona_de_parcare)
VALUES(2, NULL);
INSERT INTO PAZNIC(cod_angajat, cod_zona_de_parcare)
VALUES(3, 2);
INSERT INTO PAZNIC(cod_angajat, cod_zona_de_parcare)
VALUES(4, 2);
INSERT INTO PAZNIC(cod_angajat, cod_zona_de_parcare)
VALUES(5, 3);

INSERT INTO RECEPTIONIST(cod_angajat, cod_hotel)
VALUES(6, NULL);
INSERT INTO RECEPTIONIST(cod_angajat, cod_hotel)
VALUES(7, 2);
INSERT INTO RECEPTIONIST(cod_angajat, cod_hotel)
VALUES(8, 1);
INSERT INTO RECEPTIONIST(cod_angajat, cod_hotel)
VALUES(9, NULL);
INSERT INTO RECEPTIONIST(cod_angajat, cod_hotel)
VALUES(10, 1);

INSERT INTO CHELNER(cod_angajat, cod_restaurant)
VALUES(11, 1);
INSERT INTO CHELNER(cod_angajat, cod_restaurant)
VALUES(12, 1);
INSERT INTO CHELNER(cod_angajat, cod_restaurant)
VALUES(13, 2);
INSERT INTO CHELNER(cod_angajat, cod_restaurant)
VALUES(14, 2);
INSERT INTO CHELNER(cod_angajat, cod_restaurant)
VALUES(15, 4);

INSERT INTO BUCATAR(cod_angajat, cod_restaurant)
VALUES(16, 1);
INSERT INTO BUCATAR(cod_angajat, cod_restaurant)
VALUES(17, NULL);
INSERT INTO BUCATAR(cod_angajat, cod_restaurant)
VALUES(18, 1);
INSERT INTO BUCATAR(cod_angajat, cod_restaurant)
VALUES(19, NULL);
INSERT INTO BUCATAR(cod_angajat, cod_restaurant)
VALUES(20, 1);

INSERT INTO ADMINISTRATOR(cod_angajat, cod_hotel)
VALUES(21, 1);
INSERT INTO ADMINISTRATOR(cod_angajat, cod_hotel)
VALUES(22, 2);
INSERT INTO ADMINISTRATOR(cod_angajat, cod_hotel)
VALUES(23, 3);
INSERT INTO ADMINISTRATOR(cod_angajat, cod_hotel)
VALUES(24, 4);
INSERT INTO ADMINISTRATOR(cod_angajat, cod_hotel)
VALUES(25, NULL);

```

Script Output X

Query Result X



Task completed in 0.213 seconds

1 row inserted.

1 row inserted.

1 row inserted.

1 row inserted.

SQL Worksheet History

Worksheet Query Builder

```
SELECT * FROM LOCATIE;
```

Script Output x Explain Plan x Query Result x

SQL | All Rows Fetched: 9 in 0.004 seconds

COD_LOCATIE	ADRESA
1	Zona 1
2	Zona 2
3	Zona 3
4	Zona 4
5	Zona 5
6	Zona 6
7	Zona 7
8	Centru
9	Periferie

SQL Worksheet History

Worksheet Query Builder

```
SELECT * FROM HOTEL;
```

Script Output x Explain Plan x Query Result x

SQL | All Rows Fetched: 15 in 0.006 seconds

COD_HOTEL	DENUMIRE	NUMAR_STELE	AN_INFIINTARE	COD_LOCATIE
1	Continental	3	1965	1
2	Europa	4	1993	1
3	Atena	3	1997	1
4	Sofia	2	1983	2
5	Mures	4	1975	2
6	Banat	2	1968	2
7	Transilvania	3	1970	3
8	Moldova	(null)	(null)	3
9	Dobrogea	5	(null)	9
10	Basarabia	1	1950	9
11	Bucovina	(null)	(null)	9
12	Transnistria	(null)	(null)	8
13	Venetia	(null)	1963	8
14	Dunare	3	(null)	8
15	Carpati	(null)	1968	6

SQL Worksheet History

Worksheet Query Builder

```
SELECT * FROM CAMERA;
```

Script Output x Explain Plan x Query Result x

SQL | All Rows Fetched: 11 in 0 seconds

COD_CAMERA	COD_HOTEL	NUMAR	ETAJ
1	1	1	0
2	2	1	1
3	3	1	3
4	4	1	4
5	5	1	5
6	6	2	1
7	7	2	2
8	8	2	3
9	9	2	4
10	10	2	5
11	11	3	1

SQL Worksheet History

Worksheet Query Builder

```
SELECT * FROM FACILITATE;
```

Script Output x Explain Plan x Query Result x

SQL | All Rows Fetched: 7 in 0.006 seconds

COD_FACILITATE	DENUMIRE
1	Camera Standard
2	Camera Dubla
3	Camera Tripla
4	Apartament
5	Balcon
6	Aer Conditionat
7	Room Service

SQL Worksheet | History

Worksheet | Query Builder

```
SELECT * FROM CLIENT;
```

Script Output | Explain Plan | Query Result | SQL | All Rows Fetched: 10 in 0.008 seconds

	COD_CLIENT	NUME	PRENUME	CNP	NUMAR_TELEFON	ADRESA_MAIL
1	1	Popescu	Ion	1234567890123	0701234567	popescu.ion@mail.ro
2	2	Ionescu	Mircea	3210987654321	0798765432	mircea.ionescu@mail.com
3	3	Andrei	Ionescu	0001112223334	0700111222	andrei.ionescu@mail.ro
4	4	Mihai	Paul	9998887776665	0799888777	paul.mihai@mail.com
5	5	Boqdan	Mircea	0000011111222	0777777777	(null)
6	6	Georgescu	Gheorghe	0010011111222	0777377777	georgescu.gheorghe@mail.ro
7	7	Vlad	Teodor	2020202020202	0755000555	(null)
8	8	Alexandru	Sebastian	1313131313131	0744333432	(null)
9	9	Mihailescu	Florin	1717171717171	07000000000	(null)
10	10	Andreeescu	Robert	1919191919191	0711222333	(null)

SQL Worksheet | History

Worksheet | Query Builder

```
SELECT * FROM CAMERA_presinta_FACILITATE;
```

Script Output | Explain Plan | Query Result | SQL | All Rows Fetched: 15 in 0.005 seconds

	COD_CAMERA	COD_HOTEL	COD_FACILITATE
1	1	1	1
2	1	1	6
3	1	1	7
4	9	2	3
5	9	2	5
6	9	2	6
7	9	2	7
8	10	2	4
9	10	2	5
10	10	2	6
11	8	2	2
12	8	2	7
13	7	2	7
14	7	2	6
15	7	2	1

SQL Worksheet | History

Worksheet | Query Builder

```
SELECT * FROM CLIENT_rezerva_CAMERA;
```

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

	COD_CLIENT	COD_CAMERA	COD_HOTEL	DATA_INCEPUT	DATA_SFARSIT
1	1	1	1	1 02-SEP-23	08-SEP-23
2	2	2	2	1 07-MAY-23	13-MAY-23
3	3	3	3	1 26-APR-23	02-MAY-23
4	4	4	4	1 23-MAR-23	01-APR-23
5	5	5	5	1 23-OCT-23	01-NOV-23
6	5	7	7	2 07-SEP-23	08-SEP-23
7	6	7	7	2 10-MAR-23	14-MAR-23
8	6	8	8	2 15-APR-23	15-MAY-23
9	9	9	9	2 17-MAR-23	19-MAY-23
10	9	9	9	2 01-JAN-23	01-FEB-23

SQL Worksheet | History

Worksheet | Query Builder

```
SELECT * FROM ZONA_DE_PARCARE;
```

Script Output | Explain Plan | Query Result | SQL | All Rows Fetched: 10 in 0 seconds

	COD_ZONA_DE_PARCARE	COD_HOTEL	COD_LOCATIE
1	1	1	1
2	2	2	2
3	3	3	4
4	4	1	1
5	5	4	3
6	6	5	5
7	7	6	5
8	8	1	5
9	9	7	6
10	10	8	6

SQL Worksheet | History

Worksheet | Query Builder

```
SELECT * FROM LOC_DE_PARCARE;
```

Script Output | Explain Plan | Query Result | SQL | All Rows Fetched: 15 in 0 seconds

	COD_LOC_DE_PARCARE	COD_ZONA_DE_PARCARE	NUMAR
1	1	1	1
2	2	1	2
3	3	1	3
4	4	1	4
5	5	1	5
6	6	1	6
7	7	2	1
8	8	2	2
9	9	2	3
10	10	2	4
11	11	3	1
12	12	3	2
13	13	3	3
14	14	3	4
15	15	3	5

SQL Worksheet | History

Worksheet | Query Builder

```
SELECT * FROM FURNIZOR;
```

Script Output | Explain Plan | Query Result | SQL | All Rows Fetched: 6 in 0.01 seconds

	COD_FURNIZOR	NUME
1	1	Brutaria Iberia S.R.L.
2	2	Macelaria Creta S.R.L.
3	3	Apa Dorna S.R.L.
4	4	Brutaria Pavel S.R.L.
5	5	Macelarie Centru S.R.L.
6	6	Apa Carpatica S.R.L.

SQL Worksheet | History

Worksheet | Query Builder

```
SELECT * FROM CONTRACT;
```

Script Output | Explain Plan | Query Result | SQL | All Rows Fetched: 9 in 0.006 seconds

	COD_CONTRACT	DATA_INCEPUT	DATA_SFARSIT
1	1	02-SEP-23	02-DEC-23
2	2	01-JAN-24	01-JAN-25
3	3	01-JAN-23	01-JUL-23
4	4	01-JAN-23	01-AUG-23
5	5	01-SEP-23	01-JAN-24
6	6	01-MAR-23	01-APR-23
7	7	01-APR-23	12-MAY-23
8	8	26-APR-23	02-JUN-23
9	9	01-SEP-23	01-JAN-25

SQL Worksheet History

Worksheet Query Builder

```
SELECT * FROM FEL_DE_MANCARE;
```

Script Output Explain Plan Query Result

All Rows Fetched: 15 in 0.008 seconds

	COD_FEL_DE_MANCARE	DENUMIRE	PRET
1	1	Ciorba De Legume	11
2	2	Ciorba De Perisoare	12
3	3	Ciorba De Pui	13
4	4	Supa De Legume	7
5	5	Cartofi Prajiti	15
6	6	Piure	12
7	7	Spaqlente Carbonara	22
8	8	Pizza	27
9	9	Limonada	7
10	10	Somon	24
11	11	Snitel De Pui	14
12	12	Coca Cola	7
13	13	Pepsi	7
14	14	Sprite	7
15	15	Fanta	7

SQL Worksheet History

Worksheet Query Builder

```
SELECT * FROM RESTAURANT;
```

Script Output Explain Plan Query Result

All Rows Fetched: 8 in 0.004 seconds

	COD_RESTAURANT	COD_HOTEL	COD_LOCATIE	DENUMIRE	NUMAR_STELE	AN_INFIINTARE
1	1	1	5	Roma	3	1995
2	2	1	2	Milano	2	(null)
3	3	1	5	Savoia	(null)	2000
4	4	7	3	Varsovia	4	2004
5	5	1	5	Napoli	(null)	2008
6	6	4	7	Madrid	(null)	1993
7	7	1	5	Paris	(null)	(null)
8	8	2	1	Lisabona	3	(null)

SQL Worksheet History

Worksheet Query Builder

```
SELECT * FROM RESTAURANT_serveste_FEL_DE_MANCARE;
```

Script Output Explain Plan Query Result

All Rows Fetched: 17 in 0.008 seconds

	COD_RESTAURANT	COD_FEL_DE_MANCARE
1	1	1
2	1	2
3	2	3
4	2	4
5	2	5
6	1	5
7	1	7
8	3	2
9	3	5
10	4	2
11	5	2
12	6	3
13	8	4
14	1	6
15	8	6
16	1	10
17	2	8

SQL Worksheet History

Worksheet Query Builder

```
SELECT * FROM FURNIZOR_semneaza_cu_RESTAURANT_mai_multe_CONTRACTE;
```

Script Output Explain Plan Query Result

SQL | All Rows Fetched: 16 in 0.007 seconds

COD_RESTAURANT	COD_FURNIZOR	COD_CONTACT
1	1	1
2	1	2
3	1	3
4	1	4
5	2	5
6	2	6
7	2	1
8	2	6
9	3	1
10	3	2
11	3	3
12	3	4
13	3	5
14	3	6
15	4	5
16	4	6

SQL Worksheet History

Worksheet Query Builder

```
SELECT * FROM ANGAJAT;
```

Script Output Explain Plan Query Result

SQL | All Rows Fetched: 27 in 0.012 seconds

COD_ANGAJAT	NUME	PRENUME	CNP	NUMAR_TELEFON	ADRESA_MAIL	SALARIU
1	Popescu	Florin	0000000000000001	070000000001	(null)	4000
2	Stoica	Marius	0000000000000002	070000000002	stoicamarius@mail.com	4100
3	Stan	Ioan	0000000000000003	070000000003	(null)	4200
4	Vasilescu	George	0000000000000004	070000000004	vasilescugeorge@mail.ro	5000
5	Vasilescu	Adrian	0000000000000005	070000000005	(null)	4700
6	Radu	Andrei	0000000000000006	070000000006	(null)	4500
7	Marinescu	Cristian	0000000000000007	070000000007	(null)	4300
8	Marinescu	Dumitru	0000000000000008	070000000008	dumitru.marinescu@mail.com	5300
9	Marin	Iacob	0000000000000009	070000000009	(null)	5200
10	Florinescu	Cristian	0000000000000010	070000000010	(null)	4900
11	Popa	Ion	0000000000000011	070000000011	(null)	4950
12	Lupu	Daria	0000000000000012	070000000012	(null)	5500
13	Adamescu	Luca	0000000000000013	070000000013	(null)	5500
14	Ionescu	Mirela	0000000000000014	070000000014	(null)	5250
15	Paraschiv	Darius	0000000000000015	070000000015	paraschivdarius@mail.com	5150
16	Serban	Stefan	0000000000000016	070000000016	(null)	5050
17	Victor	Camelia	0000000000000017	070000000017	camelia.victor@mail.ro	4750
18	Dan	Oana	0000000000000018	070000000018	dan.oana@mail.com	4850
19	Cosmin	Ciprian	0000000000000019	070000000019	ciprian.cosmin@mail.ro	5000
20	Popescu	Iulian	0000000000000020	070000000020	(null)	5000
21	Popescu	Lucian	0000000000000021	070000000021	(null)	5750
22	Popescu	Raluca	0000000000000022	070000000022	(null)	6050
23	Tudor	Serban	0000000000000023	070000000023	serban.tudor@mail.com	6350
24	Octavian	Silviu	0000000000000024	070000000024	(null)	7050
25	Valentin	Catalin	0000000000000025	070000000025	(null)	6750
26	Emanuel	Anca	0000000000000026	070000000026	emanuel.anca@mail.com	6550
27	Sorin	Darius	0000000000000027	070000000027	sorin.darius@mail.ro	6350

SQL Worksheet History

Worksheet Query Builder

```
SELECT * FROM PAZNIC;
```

Script Output Explain Plan Query Result

All Rows Fetched: 5 in 0.015 seconds

	COD_ANGAJAT	COD_ZONA_DE_PARCARE
1	1	1
2	2	(null)
3	3	2
4	4	2
5	5	3

SQL Worksheet History

Worksheet Query Builder

```
SELECT * FROM ADMINISTRATOR;
```

Script Output Explain Plan Query Result

All Rows Fetched: 5 in 0.006 seconds

	COD_ANGAJAT	COD_HOTEL
1	21	1
2	22	2
3	23	3
4	24	4
5	25	(null)

SQL Worksheet History

Worksheet Query Builder

```
SELECT * FROM RECEPTIONIST;
```

Script Output Explain Plan Query Result

All Rows Fetched: 5 in 0.005 seconds

	COD_ANGAJAT	COD_HOTEL
1	6	(null)
2	7	2
3	8	1
4	9	(null)
5	10	1

SQL Worksheet History

Worksheet Query Builder

```
SELECT * FROM BUCATAR;
```

Script Output Explain Plan Query Result

All Rows Fetched: 5 in 0.007 seconds

	COD_ANGAJAT	COD_RESTAURANT
1	16	1
2	17	(null)
3	18	1
4	19	(null)
5	20	1

SQL Worksheet History

Worksheet Query Builder

```
SELECT * FROM CHELNER;
```

Script Output Explain Plan Query Result

All Rows Fetched: 5 in 0.006 seconds

	COD_ANGAJAT	COD_RESTAURANT
1	11	1
2	12	1
3	13	2
4	14	2
5	15	4

Formularea in limbaj natural si implementarea a 5 cereri SQL complexe

Sa se afiseze numele hotelurilor care au macar un paznic ce lucreaza la o zona de parcare apartenenta de hotel si care se afla in locatia cu adresa 'Zona 1'.
(subcereri sincronizate in care intervin cel putin 3 tabele)

```
SELECT h.denumire FROM HOTEL h WHERE
(SELECT COUNT(*) FROM PAZNIC p JOIN ZONA_DE_PARCARE z ON (p.cod_zona_de_parcare = z.cod_zona_de_parcare)
JOIN LOCATIE l ON (z.cod_locatie = l.cod_locatie) WHERE INITCAP(l.adresa) = 'Zona 1' AND z.cod_hotel = h.cod_hotel) >= 1;
```

Sa se determine salariul mediu al paznicilor ce lucreaza intr-o zona de parcare ce apartine hotelului in care se afla camera cu id-ul 1.
(subcereri nesincronizate in clauza FROM)

```
SELECT AVG(a1.salariu) FROM (SELECT * FROM ANGAJAT a JOIN PAZNIC p ON (a.cod_angajat = p.cod_angajat) JOIN ZONA_DE_PARCARE z ON
(p.cod_zona_de_parcare = z.cod_zona_de_parcare) JOIN HOTEL h ON (z.cod_hotel = h.cod_hotel) JOIN camera c ON
(c.cod_hotel = h.cod_hotel) AND c.cod_camera = 1) a1;
```

Sa se afiseze salariile medii care depasesc 6000 ale tuturor administratorilor cu aceeasi initiala a numelui care lucreaza intr-un hotel aflat in locatia cu adresa 'Zona 1'.
(grupari de date cu subcereri nesincronizate in care intervin cel putin 3 tabele, functii grup, filtrare la nivel de grupuri, in cadrul aceleiasi cereri)

```
SELECT AVG(subcerere.salariu) FROM (SELECT ang.salariu, ang.nume FROM ANGAJAT ang JOIN ADMINISTRATOR a
ON (ang.cod_angajat = a.cod_angajat) JOIN HOTEL h ON (a.cod_hotel = h.cod_hotel) JOIN
LOCATIE l ON (h.cod_locatie = l.cod_locatie) WHERE INITCAP(l.adresa) = 'Zona 1') subcerere
GROUP BY SUBSTR(subcerere.nume, 1, 1) HAVING AVG(subcerere.salariu) > 6000;
```

Sa se afiseze pentru fiecare client numele, prenumele si adresa sa de mail in ordine crescatoare lexicografic dupa nume si descrescatoare lexicografic dupa prenume.
De asemenea, pentru fiecare adresa de mail se va afisa un text ce mentioneaza daca adresa exista sau nu pentru fiecare client, iar in cazul existentei sale va mentiona daca adresa este sau nu neaparat in limba romana.
(ordonari si utilizarea functiilor NVL si DECODE in cadrul aceleiasi cereri, utilizarea a cel putin unei expresii CASE)

```
SELECT nume, prenume, adresa_mail,
DECODE(
CASE
WHEN adresa_mail LIKE '%.ro' THEN 1
WHEN adresa_mail LIKE '%.com' THEN 2
ELSE 3
END,
1, 'Clientul are o adresa de mail in romana.',
2, 'Clientul are o adresa de mail care nu este neaparat in romana.',
3, NVL(adresa_mail, 'Clientul nu are o adresa de mail.')
)
FROM CLIENT
ORDER BY NUME, PRENUME DESC;
```

Sa se afiseze salariul, numele si prenumele tuturor angajatilor care au ca nume 'Popescu' si care contin subsecventa 'uc' in numele lor complet.
Rezultatele vor fi ordonate descrescator dupa salariul angajatului.
(utilizarea a cel putin 2 functii pe siruri de caractere)

```
SELECT salariu, nume, prenume FROM ANGAJAT
WHERE INITCAP(nume) = 'Popescu' AND INSTR(CONCAT(nume, prenume), 'uc') != 0
ORDER BY salariu DESC;
```

Sa se afiseze durata in luni a tuturor sejururilor care incep pe o zi de 7 si care rezerva o camera aflata intr-un hotel din locatia cu codul 1.
(utilizarea a cel putin 2 functii pe date calendaristice, utilizarea a cel putin 1 bloc de cerere, clauza WITH)

```
WITH SEJUR AS
(
  SELECT r.data_inceput, r.data_sfarsit FROM CLIENT_rezerva_CAMERA r
  JOIN HOTEL h ON(r.cod_hotel = h.cod_hotel) AND h.cod_locatie = 1
)
SELECT MONTHS_BETWEEN(s.data_sfarsit, s.data_inceput) FROM SEJUR s
WHERE EXTRACT(DAY FROM s.data_inceput) = 7;
```

Sa se afiseze numele hotelurilor care au macar un paznic ce lucreaza la o zona de parcare apartenenta de hotel si care se afla in locatia cu adresa 'Zona 1'.
 (subcereri sincronizate in care intervin cel putin 3 tabele)

```
SELECT h.denumire FROM HOTEL h WHERE
  (SELECT COUNT(*) FROM PAZNIC p JOIN ZONA_DE_PARCARE z ON (p.cod_zona_de_parcare = z.cod_zona_de_parcare)
  JOIN LOCATIE l ON (z.cod_locatie = l.cod_locatie) WHERE INITCAP(l.adresa) = 'Zona 1' AND z.cod_hotel = h.cod_hotel) >= 1;
```

Script Output x | Query Result x | All Rows Fetched: 1 in 0.029 seconds

DENUMIRE
1 Continental

Sa se determine salariul mediu al paznicilor ce lucreaza intr-o zona de parcare ce apartine hotelului in care se afla camera cu id-ul 1.
 (subcereri nesincronizate in clauza FROM)

```
SELECT AVG(al.salariu) FROM (SELECT * FROM ANGAJAT a JOIN PAZNIC p ON (a.cod_angajat = p.cod_angajat) JOIN ZONA_DE_PARCARE z ON
(p.cod_zona_de_parcare = z.cod_zona_de_parcare) JOIN HOTEL h ON (z.cod_hotel = h.cod_hotel) JOIN camera c ON
(c.cod_hotel = h.cod_hotel) AND c.cod_camera = 1) al;
```

Script Output x | Query Result x | All Rows Fetched: 1 in 0.037 seconds

AVG(A1.SALARIU)
1 4000

Sa se afiseze salariile medii care depasesc 6000 ale tuturor administratorilor cu aceeasi initiala a numelui care lucreaza intr-un hotel aflat in locatia cu adresa 'Zona 1'.
 (grupari de date cu subcereri nesincronizate in care intervin cel putin 3 tabele, functii grup, filtrare la nivel de grupuri, in cadrul aceleiasi cereri)

```
SELECT AVG(subcerere.salariu) FROM (SELECT ang.salariu, ang.nume FROM ANGAJAT ang JOIN ADMINISTRATOR a
ON (ang.cod_angajat = a.cod_angajat) JOIN HOTEL h ON (a.cod_hotel = h.cod_hotel) JOIN
LOCATIE l ON (h.cod_locatie = l.cod_locatie) WHERE INITCAP(l.adresa) = 'Zona 1') subcerere
GROUP BY SUBSTR(subcerere.nume, 1, 1) HAVING AVG(subcerere.salariu) > 6000;
```

Script Output x | Query Result x | All Rows Fetched: 1 in 0.012 seconds

AVG(SUBCERERE.SALARIU)
1 6350

Worksheet Query Builder

Sa se afiseze pentru fiecare **client** numele, prenumele si adresa sa de mail **in** ordine crescatoare lexicografic dupa nume si descrescatoare lexicografic dupa prenume.

De asemenea, pentru fiecare adresa de mail se va afisa un text **ce** mentioneaza daca adresa exista sau nu pentru fiecare **client**, iar **in** cazul existentei sale va mentiona daca adresa este sau nu neaparat **in** limba romana. (ordonari si utilizarea functiilor NVL si DECODE **in** cadrul aceleiasi cereri, utilizarea a cel putin unei expresii CASE)

```

SELECT nume, prenume, adresa_mail,
DECODE(
    CASE
        WHEN adresa_mail LIKE '%.ro' THEN 1
        WHEN adresa_mail LIKE '%.com' THEN 2
        ELSE 3
    END,
    1, 'Clientul are o adresa de mail in romana.',
    2, 'Clientul are o adresa de mail care nu este neaparat in romana.',
    3, NVL(adresa_mail, 'Clientul nu are o adresa de mail.')
)
FROM CLIENT
ORDER BY NUME, PRENUME DESC;

```

Script Output | Query Result | All Rows Fetched: 10 in 0.007 seconds

	NUME	PRENUME	ADRESA_MAIL	DECODE(CASEWHENADRESA_MAILLIKE'%.ROTHEN1WHENADRESA_MAILLIKE'%.COMTHEN2ELSE3END,1,CLIENTULAREADRESADEMAILINROMANA.,2,CLIENTULAREADRESADEMAILCARENUESTE)
1	Alexandru	Sebastian	(null)	Clientul nu are o adresa de mail.
2	Andreescu	Robert	(null)	Clientul nu are o adresa de mail.
3	Andrei	Ionescu	andreiionescu@mail.ro	Clientul are o adresa de mail in romana.
4	Boqdan	Mircea	(null)	Clientul nu are o adresa de mail.
5	Georgescu	Gheorghe	georgescu gheorghe@mail.ro	Clientul are o adresa de mail in romana.
6	Ionescu	Mircea	mircea ionescu@mail.com	Clientul are o adresa de mail care nu este neaparat in romana.
7	Mihai	Paul	paul.mihai@mail.com	Clientul are o adresa de mail care nu este neaparat in romana.
8	Mihailescu	Florin	(null)	Clientul nu are o adresa de mail.
9	Popescu	Ion	popescu.ion@mail.ro	Clientul are o adresa de mail in romana.
10	Vlad	Teodor	(null)	Clientul nu are o adresa de mail.

Worksheet Query Builder

Sa se afiseze salariul, numele si prenumele tuturor angajatilor care au ca nume '**Popescu**' si care contin subsecventa '**uc**' **in** numele lor complet.

Rezultatele vor fi ordonate descrescator dupa salariul angajatului.

(utilizarea a cel putin 2 functii pe siruri de caractere)

```

SELECT salariu, nume, prenume FROM ANGAJAT
WHERE INITCAP(nume) = 'Popescu' AND INSTR(LOWER(CONCAT(nume, prenume)), 'uc') != 0
ORDER BY salariu DESC;

```

Script Output | Query Result | All Rows Fetched: 2 in 0.02 seconds

	SALARIU	NUME	PRENUME
1	6050	Popescu	Raluca
2	5750	Popescu	Lucian

Sa se afiseze durata **in** luni a tuturor sejururilor care incep pe o zi de **7** si care rezerva o camera aflata intr-un hotel din locatia cu codul **1**.
(utilizarea a cel putin **2** functii pe **date** calendaristice, utilizarea a cel putin **1** bloc de cerere, clauza **WITH**)

```
WITH SEJUR AS
(
    SELECT r.data_inceput, r.data_sfarsit FROM CLIENT_rezerva_CAMERA r
    JOIN HOTEL h ON(r.cod_hotel = h.cod_hotel) AND h.cod_locatie = 1
)
SELECT MONTHS_BETWEEN(s.data_sfarsit, s.data_inceput) FROM SEJUR s
WHERE EXTRACT(DAY FROM s.data_inceput) = 7;
```

Script Output | Query Result | All Rows Fetched: 2 in 0.031 seconds

MONTHS_BETWEEN(S.DATA_SFARSIT,S.DATA_INCEPUT)
1 0.1935483870967741935483870967741935483871
2 0.0322580645161290322580645161290322580645

**O cerere ce utilizeaza
operatia Outer-Join pe minimum 4 tabele, o cerere ce utilizeaza operatia
Division si o cerere care implementeaza analiza Top-n**

Operatia Outer-Join pe minimum 4 tabele:

Sa se afiseze codul angajatilor ce ocupa functia de bucatar, denumirea restaurantului in care lucreaza, denumirea hotelului de care apartine restaurantul, numarul de stele ale hotelului si adresa hotelului. In cazul in care unul dintre aceste campuri nu exista se va afisa 'Nu exista'. In cazul numarului de stele, daca acesta nu exista, se va afisa '-1'.

```
SELECT b.cod_angajat "Cod Angajat", NVL(r.denumire, 'Nu exista') "Denumire Restaurant", NVL(h.denumire, 'Nu exista') "Denumire Hotel",  
NVL(h.numar_stele, -1) "Numar Stele Hotel", NVL(l.adresa, 'Nu exista') "Adresa Hotel"  
FROM BUCATAR b LEFT JOIN RESTAURANT r ON (b.cod_restaurant = r.cod_restaurant) LEFT JOIN  
HOTEL h ON (r.cod_hotel = h.cod_hotel) LEFT JOIN LOCATIE l ON (h.cod_locatie = l.cod_locatie);
```

Analiza Top-n:

Sa se afiseze primii 15 cei mai bine platiti angajati, excluzand angajatii ce au cel mai mare salariu dintre toti.

```
SELECT *  
FROM  
(  
    SELECT t.*, ROWNUM numar  
    FROM  
    (  
        SELECT *  
        FROM ANGAJAT a  
        WHERE a.salariu < (SELECT MAX(a1.salariu) FROM ANGAJAT a1)  
        ORDER BY a.salariu DESC  
    ) t  
)  
WHERE numar <= 15;
```

Operatia Division:

Sa se afiseze toate felurile de mancare (cod si denumire) care sunt servite doar in restaurantul cu denumirea 'Roma'.

```
SELECT sol.cod_fel_de_mancare, sol.denumire FROM FEL_DE_MANCARE sol  
WHERE NOT EXISTS (  
    SELECT m.cod_fel_de_mancare FROM FEL_DE_MANCARE m  
    WHERE NOT EXISTS (  
        SELECT s.cod_fel_de_mancare FROM RESTAURANT_serveste_FEL_DE_MANCARE s  
        JOIN RESTAURANT r ON (s.cod_restaurant = r.cod_restaurant)  
        WHERE INITCAP(r.denumire) = 'Roma'  
        AND s.cod_fel_de_mancare = m.cod_fel_de_mancare  
    )  
    AND sol.cod_fel_de_mancare = m.cod_fel_de_mancare  
);
```

Worksheet | Query Builder

Operatia Outer-Join pe minimum 4 tabele:

Sa se afiseze codul angajatilor ce ocupa functia de bucatar, denumirea restaurantului in care lucreaza, denumirea hotelului de care apartine restaurantul, numarul de stele ale hotelului si adresa hotelului. In cazul in care unul dintre aceste campuri nu exista se va afisa 'Nu exista'. In cazul numarului de stele, daca acesta nu exista, se va afisa '-1'.

```

SELECT b.cod_angajat "Cod Angajat", NVL(r.denumire, 'Nu exista') "Denumire Restaurant", NVL(h.denumire, 'Nu exista') "Denumire Hotel",
NVL(h.numar_stele, -1) "Numar Stele Hotel", NVL(l.adresa, 'Nu exista') "Adresa Hotel"
FROM BUCATAR b LEFT JOIN RESTAURANT r ON (b.cod_restaurant = r.cod_restaurant) LEFT JOIN
HOTEL h ON (r.cod_hotel = h.cod_hotel) LEFT JOIN LOCATIE l ON (h.cod_locatie = l.cod_locatie);

```

Script Output | Query Result | All Rows Fetched: 5 in 0.012 seconds

	Cod Angajat	Denumire Restaurant	Denumire Hotel	Numar Stele Hotel	Adresa Hotel
1	16	Roma	Continental	3	Zona 1
2	18	Roma	Continental	3	Zona 1
3	20	Roma	Continental	3	Zona 1
4	17	Nu exista	Nu exista	-1	Nu exista
5	19	Nu exista	Nu exista	-1	Nu exista

Worksheet | Query Builder

Analiza Top-n:

Sa se afiseze primii 15 cei mai bine platiti angajati, excluzand angajatii ce au cel mai mare salariu dintre toti.

```

SELECT *
FROM
(
  SELECT t.*, ROWNUM numar
  FROM
  (
    SELECT *
    FROM ANGAJAT a
    WHERE a.salariu < (SELECT MAX(al.salariu) FROM ANGAJAT al)
    ORDER BY a.salariu DESC
  ) t
)
WHERE numar <= 15;

```

Script Output | Query Result | All Rows Fetched: 15 in 0.009 seconds

	COD_ANGAJAT	NUME	PRENUME	CNP	NUMAR_TELEFON	ADRESA_MAIL	SALARIU	NUMAR
1	25	Valentin	Catalin	0000000000025	0700000025	(null)	6750	1
2	26	Emmanuel	Anca	0000000000026	0700000026	emmanuel.anca@mail.com	6550	2
3	23	Tudor	Serban	0000000000023	0700000023	serban.tudor@mail.com	6350	3
4	27	Sorin	Darius	0000000000027	0700000027	sorin.darius@mail.ro	6350	4
5	22	Popescu	Raluca	0000000000022	0700000022	(null)	6050	5
6	21	Popescu	Lucian	0000000000021	0700000021	(null)	5750	6
7	13	Adamescu	Luca	0000000000013	0700000013	(null)	5500	7
8	12	Lupu	Daria	0000000000012	0700000012	(null)	5500	8
9	8	Marinescu	Dumitru	0000000000008	0700000008	dumitru.marinescu@mail.com	5300	9
10	14	Ionescu	Mirela	0000000000014	0700000014	(null)	5250	10
11	9	Marin	Iacob	0000000000009	0700000009	(null)	5200	11
12	15	Paraschiv	Darius	0000000000015	0700000015	paraschivdarius@mail.com	5150	12
13	16	Serban	Stefan	0000000000016	0700000016	(null)	5050	13
14	10	Gheorghe	Ciniana	0000000000010	0700000010	gheorghe.gheorghe@mail.com	5000	14

Operatia Division:

Sa se afiseze toate felurile de mancare (cod si denumire) care sunt servite doar **in** restaurantul cu denumirea 'Roma'.

```

SELECT sol.cod_fel_de_mancare, sol.denumire FROM FEL_DE_MANCARE sol
WHERE NOT EXISTS (
    SELECT m.cod_fel_de_mancare FROM FEL_DE_MANCARE m
    WHERE NOT EXISTS (
        SELECT s.cod_fel_de_mancare FROM RESTAURANT_serveste_FEL_DE_MANCARE s
        JOIN RESTAURANT r ON (s.cod_restaurant = r.cod_restaurant)
        WHERE INITCAP(r.denumire) = 'Roma'
        AND s.cod_fel_de_mancare = m.cod_fel_de_mancare
    )
    AND sol.cod_fel_de_mancare = m.cod_fel_de_mancare
);

```

Script Output x | Query Result x | All Rows Fetched: 6 in 0.031 seconds

COD_FEL_DE_MANCARE	DENUMIRE
1	Ciorba De Lequme
2	Ciorba De Perisoare
3	Cartofi Prajiti
4	Piure
5	Spaghete Carbonara
6	Somon

Implementarea a 3 operatii de actualizare si de suprimare a datelor utilizand subcereri

Actualizare:

Sa se mareasca salariul cu 5% tuturor administratorilor care lucreaza intr-un hotel.

```
UPDATE ANGAJAT  
SET salariu = salariu * 0.05  
WHERE cod_angajat IN  
(SELECT a.cod_angajat FROM ANGAJAT a JOIN ADMINISTRATOR a1 ON (a.cod_angajat = a1.cod_angajat) WHERE a1.cod_hotel IS NOT NULL);
```

Sa se extinda toate rezervarile din hotelurile din locatia cu denumirea 'Zona 1' cu 30 de zile.

```
UPDATE CLIENT_rezerva_CAMERA  
SET data_sfarsit = data_sfarsit + 30  
WHERE cod_hotel IN  
(SELECT h.cod_hotel FROM HOTEL h JOIN LOCATIE l ON (h.cod_locatie = l.cod_locatie) WHERE INITCAP(l.adresa) = 'Zona 1');
```

Sa se seteze numarul de stele la 2 pentru toate hotelurile ce nu au un numar setat si care se afla in locatia 3.

```
UPDATE HOTEL  
SET numar_stele = 2  
WHERE numar_stele IS NULL AND cod_hotel IN  
(SELECT cod_hotel FROM HOTEL WHERE cod_locatie = 3);
```

Suprimare:

Sa se stearga toate rezervarile care au fost realizate pe cea mai apropiata data de 1 iunie 2023, dar dupa 1 iunie 2023.

```
DELETE FROM CLIENT_rezerva_CAMERA r  
WHERE r.data_inceput = (SELECT MIN(r1.data_inceput) FROM CLIENT_rezerva_CAMERA r1  
WHERE r1.data_inceput > TO_DATE('01-06-2023', 'DD-MM-YYYY'));
```

Sa se stearga toti receptionistii ce lucreaza intr-un hotel cu mai putin de 3 stele sau care nu lucreaza deloc.

```
DELETE FROM RECEPTIONIST r  
WHERE r.cod_angajat IN  
(  
(SELECT r1.cod_angajat FROM RECEPTIONIST r1 WHERE r1.cod_hotel IS NULL)  
UNION  
(SELECT r1.cod_angajat FROM RECEPTIONIST r1 JOIN HOTEL h ON (r1.cod_hotel = h.cod_hotel) AND h.numar_stele < 3)  
)
```

Sa se stearga felurile de mancare ce nu sunt servite in niciun restaurant.

```
DELETE FROM FEL_DE_MANCARE m  
WHERE m.cod_fel_de_mancare NOT IN  
(SELECT s.cod_fel_de_mancare FROM RESTAURANT_serveste_FEL_DE_MANCARE s);
```

Worksheet Query Builder Run Statement (Ctrl+Enter)

Actualizare:

```
Sa se mareasca salariul cu 5% tuturor administratorilor care lucreaza intr-un hotel.
```

```
UPDATE ANGAJAT
SET salariu = salariu * 0.05
WHERE cod_angajat IN
(SELECT a.cod_angajat FROM ANGAJAT a JOIN ADMINISTRATOR al ON (a.cod_angajat = al.cod_angajat) WHERE al.cod_hotel IS NOT NULL);
```

Script Output Query Result Task completed in 0.08 seconds

4 rows updated.

Worksheet Query Builder

```
Sa se extinda toate rezervarile din hotelurile din locatia cu denumirea 'Zona 1' cu 30 de zile.
```

```
UPDATE CLIENT_rezerva_CAMERA
SET data_sfarsit = data_sfarsit + 30
WHERE cod_hotel IN
(SELECT h.cod_hotel FROM HOTEL h JOIN LOCATIE l ON (h.cod_locatie = l.cod_locatie) WHERE INITCAP(l.adresa) = 'Zona 1');
```

Script Output Query Result Task completed in 0.038 seconds

10 rows updated.

Worksheet Query Builder

```
Sa se seteze numarul de stele la 2 pentru toate hotelurile ce nu au un numar setat si care se afla in locatia 3.
```

```
UPDATE HOTEL
SET numar_stele = 2
WHERE numar_stele IS NULL AND cod_hotel IN
(SELECT cod_hotel FROM HOTEL WHERE cod_locatie = 3);
```

Script Output Query Result Task completed in 0.063 seconds

1 row updated.

Suprimare:

Sa se stearga toate rezervarile care au fost realizate pe cea mai apropiata **data** de **1 iunie 2023**, dar dupa **1 iunie 2023**.

```
DELETE FROM CLIENT_rezerva_CAMERA r
WHERE r.data_inceput = (SELECT MIN(r1.data_inceput) FROM CLIENT_rezerva_CAMERA r1
WHERE r1.data_inceput > TO_DATE('01-06-2023', 'DD-MM-YYYY'));
```

Script Output | Query Result | Task completed in 0.035 seconds

1 row deleted.

Sa se stearga toti receptionistii **ce** lucreaza intr-un hotel cu mai putin de **3** stele sau care nu lucreaza deloc.

```
DELETE FROM RECEPTIONIST r
WHERE r.cod_angajat IN
(
  (SELECT r1.cod_angajat FROM RECEPTIONIST r1 WHERE r1.cod_hotel IS NULL)
UNION
  (SELECT r1.cod_angajat FROM RECEPTIONIST r1 JOIN HOTEL h ON (r1.cod_hotel = h.cod_hotel) AND h.numar_stele < 3)
);
```

Script Output | Query Result | Task completed in 0.055 seconds

2 rows deleted.

Sa se stearga felurile de mancare **ce** nu sunt servite **in** niciun restaurant.

```
DELETE FROM FEL_DE_MANCARE m
WHERE m.cod_fel_de_mancare NOT IN
  (SELECT s.cod_fel_de_mancare FROM RESTAURANT_serveste_FEL_DE_MANCARE s);
```

Script Output | Query Result | Task completed in 0.035 seconds

6 rows deleted.