

Companie Aeriana - Proiect Baze de Date

Maria Preda, Grupa 151

Contents

| | | |
|----|--|----|
| 1 | Modelul real | 3 |
| 2 | Prezentarea constrangerilor (restrictii, reguli) impuse asupra modelului | 3 |
| 3 | Descrierea entitatilor, incluzand precizarea cheii primare | 4 |
| 4 | Descrierea relatiilor, incluzand precizarea cardinalitatii acestora | 6 |
| 5 | Descrierea atributelor, incluzand tipul de date si eventualele constrangeri, valori implicite, valori posibile ale atributelor. | 7 |
| 6 | Diagrama E/R | 13 |
| 7 | Diagrama conceptuala | 14 |
| 8 | Enumerarea schemelor relationale corespunzatoare diagramei conceptuale la punctul 7. | 15 |
| 9 | Realizarea normalizarii pana la forma normala 3 ($FN1 - FN3$) | 17 |
| 10 | Crearea unei secvente ce va fi utilizata in inserarea inregistrarilor in tabele (punctul 11). | 18 |
| 11 | Crearea tabelelor in SQL si inserarea de date coerente in fiecare dintre acestea (minimum 5 inregistrari in fiecare tabel neasociativ; minimum 10 inregistrari in tabelele asociative) . | 19 |
| 12 | Formulati in limbaj natural si implementati 5 cereri SQL complexe | 67 |
| 13 | Implementarea a 3 operatii de actualizare si de suprimare a datelor utilizand subcereri | 74 |
| 14 | Formulati in limbaj natural si implementati in SQL: o cerere ce utilizeaza operatia outer-join pe minimum 4 tabele, o cerere ce utilizează operatia division si o cerere care implementează analiza top-n | 80 |
| 15 | Optimizarea unei cereri, aplicand regulile de optimizare ce deriva din proprietatile operatorilor algebrei relationale. Cererea va fi exprimata prin expresie algebrica, arbore algebric si limbaj (SQL), atat anterior cat si ulterior optimizarii. | 83 |
| 16 | Realizarea normalizarii BCNF, FN4, FN5. Aplicarea denormalizarii | 88 |

1 Modelul real

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

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

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

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

2 Prezentarea constrangerilor (restrictii, reguli) impuse asupra modelului

- Un aeroport are o singura locatie, iar o locatie are un singur aeroport.
- Un aeroport are minim un terminal, dar poate avea mai multe.
- Un terminal are minim o poarta, dar poate avea mai multe.
- O poarta este conectata la minim o pista, iar o pista este conectata la 0 sau mai multe porti.
- De la un aeroport exista mai multe rute posibile pe care se poate pleca, in timp ce o ruta pleaca de la un singur aeroport si ajunge la un singur aeroport.
- Un zbor are o singura ruta, iar o ruta poate avea mai multe zboruri.
- Un zbor porneste de pe o singura pista.
- Un zbor ajunge pe o singura pista.
- De pe o pista pot pleca mai multe zboruri.
- In cadrul fiecarui zbor sunt repartizati mai multi angajati ai companie.
- Fiecare zbor se realizeaza cu un singur avion, in timp ce cu un avion se pot realiza mai multe zboruri.

- Un avion contine mai multe locuri.
- Un loc apartine unei singure clase, iar o clasa poate avea mai multe locuri.
- O clasa poate avea mai multe facilitati sau niciuna.
- O facilitate apartine minim unei clase.
- Un pasager face minim o rezervare, in timp ce o rezervare poate fi facuta de un singur pasager.
- In cadrul unui zbor pot exista mai multe rezervari, iar in cadrul unei rezervari pot fi alese mai multe locuri.
- In cadrul unei rezervari pot fi alese locuri din mai multe zboruri.

3 Descrierea entitatilor, incluzand precizarea cheii primare

Pentru modelul de date referitor la Compania Aeriana, structurile LOCATIE, AEROPORT, AVION, RUTA, TERMINAL, LOC, POARTA, CLASA, ZBOR, PISTA, FACILITATE, REZERVARE, ANGAJAT, PILOT, INSOTITOR DE BORD, INGINER, PASAGER reprezinta entitati.

Vom prezenta entitatile modelului de date, dand o descriere completa a fiecaruia. De asemenea, pentru fiecare entitate se va preciza cheia primara.

Toate entitatile pe care le vom prezenta sunt independente, cu exceptia PILOT, INSOTITOR DE BORD.

LOCATIE = entitate ce defineste un punct geografic precis, avand o adresa, latitudinea, longitudinea, orasul, tara, limba oficiala si moneda oficiala. Cheia primara a entitatii este cod_locatie.

AEROPORT = entitate ce desemneaza un spatiu destinat decolarilor si aterizarilor avioanelor. Aceasta include numele aeroportului, daca este destinat si zborurilor internationale din cadrul acestuia. Cheia primara a entitatii este cod_aeroport.

TERMINAL = aceasta entitate descrie informatiile referitoare la terminalele aeroportului, precum capacitatea. Cheia primara a entitatii este cod_terminal.

POARTA = aceasta entitate descrie informatiile referitoare la portile de imbarcare si debarcare din terminalul aeroportului. Aceste informatii includ capacitatea si pozitia sa in terminal. Cheia primara a entitatii este cod_poarta.

PISTA = entitate ce descrie informatiile referitoare la pistele unui aeroport, de pe care se decoleaza si aterizeaza, incluzand lungimea, latimea, si orientarea pistei. Cheia primara a acestei entitati este cod_pista.

RUTA = entitate ce descrie toate legaturile intre aeroporturi, precum si informatii legate de acestea: escala, distanta si timp. Cheia primara a acestei entitati este cod_ruta.

ZBOR = entitate ce descrie informatiile cu privire la zborurile disponibile pe o anumita ruta, incluzand data, ora la care decoleaza si cea la care aterizeaza, precum si eventualele intarzieri. Cheia primara a acestei entitati este cod_zbor.

AVION = entitate ce descrie informatiile cu privire la aeronavele pe care le detine Compania Aeriana, precum numarul de locuri, modelul avionului, daca se pot efectua zboruri intercontinentale si dimensiunea acestuia. Cheia primara a acestei entitati este cod_avion.

LOC = entitate ce descrie locurile dintr-un avion, precizandu-le pozitia si numarul. Cheia primara a acestei entitati este cod_loc.

CLASA = entitate ce descrie clasa careia apartin locurile dintr-un avion, precum pretul. Cheia primara a acestei entitati este cod_clasa.

FACILITATE = entitate ce descrie serviciile pe care le ofera clasa respectiva, masa, daca include bagaj de cala, dar si eventualul pret suplimentar. Cheia primara a acestei entitati este cod_facilitate.

ANGAJAT = persoana fizica, angajata de compania aeriana, desemnata pentru diferite zboruri, fie pentru a pilota aeronava, fie ca insotitor de bord, fie ca inginer ce realizeaza revizia. Aceasta entitate are un nume, un prenume, o data a nasterii si una a angajarii. Cheia primara a acestei entitati este cod_angajat.

PILOT = subentitate a entitatii ANGAJAT, ce contine informatii specifice pilotilor. Cheia primara a acestei entitati este cod_angajat.

INSOTITOR_DE_BORD = subentitate a entitatii ANGAJAT, ce contine informatii specifice insotitorilor de bord. Cheia primara a acestei entitati este cod_angajat.

INGINER = subentitate a entitatii ANGAJAT, ce contine informatii specifice inginerilor. Cheia primara a acestei entitati este cod_angajat.

REZERVARE = entitate ce descrie rezervarile facute de pasageri pentru un anumit zbor (numar de locuri, pretul total). Cheia primara a acestei entitati este cod_rezervare.

PASAGER = persoana fizica, client al Companiei Aeriene, care zboara de la un aeroport la altul. Entitatea descrie pasagerii, avand ca atribute nume, prenume, nationalitate si numarul pasaportului. Cheia primara a acestei entitati este cod_pasager.

4 Descrierea relatiilor, incluzand precizarea cardinalitatii acestora

Vom prezenta relatiile modelului de date, dand o descriere completa a fiecareia. De fapt, denumirile acestor legaturi sunt sugestive, reflectand continutul acestora si entitatile pe care le leaga. Pentru fiecare relatie se va preciza cardinalitatea minima si maxima.

ORAS_apartine_TARA = relatie care leaga entitatile TARA si ORAS, reflectand legatura dintre acestea (fiecare oras este situat intr-o tara). Ea are cardinalitatea minima 1:1 (o tara trebuie sa aiba minim un oras) si cardinalitatea maxima n:1 (o tara poate avea mai multe orase, dar un oras poate apartine unei singure tari).

ORAS_situat_la_LOCATIE = relatie care leaga entitatile ORAS si LOCATIE, reflectand legatura dintre acestea (o anumita locatie se afla intr-un oras). Ea are cardinalitatea minima 1:1 (intr-un oras trebuie sa existe cel putin o locatie) si cardinalitatea maxima 1:n (un oras poate avea mai multe locatii, dar o locatie nu se poate afla in mai multe orase).

AEROPORT_situat_la_LOCATIE = relatie care leaga entitatile AEROPORT si LOCATIE, reflectand legatura dintre acestea (fiecare aeroport se afla la o anumita locatie). Ea are cardinalitatea minima 1:1, iar cardinalitatea maxima 1:1 (un aeroport se afla la o singura locatie, iar intr-o locatie poate exista un singur aeroport).

AEROPORT_are_TERMINAL = relatie care leaga entitatile AEROPORT si TERMINAL, reflectand faptul ca fiecare aeroport are mai terminale. Ea are cardinalitatea minima 1:1 (orice aeroport trebuie sa aiba minim un terminal) si cardinalitatea maxima 1:n (un terminal apartine unui singur aeroport, dar un aeroport poate avea mai multe terminale).

TERMINAL_deserveste_POARTA = relatie care leaga entitatile TERMINAL si POARTA, reflectand legatura dintre acestea (o poarta se afla intr-un terminal). Ea are cardinalitatea minima 1:1 si cardinalitatea maxima 1:n (intr-un terminal pot exista mai multe porti, dar o poarta poate apartine unui singur terminal).

POARTA_CONECTATA_la_PISTA = relatie care leaga entitatile POARTA si PISTA, reflectand legatura dintre acestea (oamenii ajung la pista cu ajutorul portii). Ea are cardinalitatea minima 1:1 si cardinalitatea maxima n:n (pe o poarta pot fi accesate mai multe piste, iar o pista poate fi accesata de la mai multe porti).

RUTA_are_AEROPORT = relatie care leaga entitatile AEROPORT si RUTA, reflectand rutele ce exista intre aeroporturi (de unde se poate pleca sau de unde se poate veni pe un aeroport). Ea are cardinalitatea minima 2:1 si cardinalitatea maxima 2:n (de la un aeroport exista mai multe rute posibile, iar o ruta este intre doua aeroporturi).

ZBOR_zboara_pe_RUTA = relatie care leaga entitatile ZBOR si RUTA, reflectand legatura dintre acestea (un zbor se desfasoara pe una dintre rutele existente). Cardinalitatea minima este 1:1 si cardinalitatea maxima este n:1 (pe o ruta exista mai multe zboruri, dar un zbor se desfasoara pe o singura ruta).

ANGAJAT_lucreaza_ZBOR = relatie care leaga entitatile ANGAJAT si ZBOR, reflectand legatura dintre acestea (un angajat este repartizat pentru a lucra pe parcursul unui zbor). Cardinalitatea minima este 4:1 (in cadrul fiecarui zbor, lucreaza 4 angajati: pilot, copilot si doi insotitori de bord, iar un angajat lucreaza in timpul a minim un zbor) si cardinalitatea maxima este 4:n (un angajat poate lucra in timpul mai multor zboruri).

AVION_zboara_ZBOR = relatie care leaga entitatile AVION si ZBOR, reflectand legatura dintre acestea (avionul este utilizat pentru zbor). Ea are cardinalitatea minima 1:1 si cardinalitatea maxima 1:n (un avion trebuie sa zboare in cadrul cel putin unui zbor, iar in cadrul unui zbor nu poate fi folosit mai mult de un avion).

AVION_contine_LOC = relatie care leaga entitatile AVION si LOC, reflectand legatura dintre acestea (avionul are locuri pentru clienti). Ea are cardinalitatea minima 1:1 si cardinalitatea maxima 1:n (un avion poate avea mai multe locuri, dar un loc poate apartine unui singur avion).

LOC_apartine_CLASA = relatie care leaga entitatile CLASA si LOC, reflectand legatura dintre acestea (locurile apartin unor clase). Ea are cardinalitatea minima 1:1 si cardinalitatea maxima n:1 (o clasa poate avea mai multe locuri, dar un loc poate apartine unei singure clase).

CLASA_ofera_FACILITATE = relatie care leaga entitatile CLASA si FACILITATE, reflectand legatura dintre acestea (clasele pot avea diferite facilitati). Ea are cardinalitatea minima 1:0 si cardinalitatea maxima n:m (o clasa poate avea mai multe facilitati, iar o facilitate poate apartine mai multor clase).

PASAGER_cumpara_REZERVARE = relatie care leaga entitatile PASAGER si REZERVARE, reflectand legatura dintre acestea (un pasager face o rezervare). Ea are cardinalitatea minima 1:0 (un pasager poate sa nu aiba rezervare, daca este facuta de altcineva) si cardinalitatea maxima n:m (un pasager poate face mai multe rezervari, iar o rezervare poate fi facuta pentru mai multi pasageri).

REZERVAREA_rezerva_LOC_in_ZBOR = relatie de tip 3 care leaga entitatile REZERVARE, LOC si ZBOR, reflectand locurile pe care le contine o rezervare facuta in cadrul unui zbor. Denumirea acestei relatii va fi rezerva.

5 Descrierea atributelor, incluzand tipul de date si eventualele constrangeri, valori implicite, valori posibile ale atributelor.

Entitatea LOCATIE are ca attribute:

cod_locatie = variabila de tip intreg, de lungime maxim 5, care reprezinta codul unei locatii.
adresa = variabila de tip caracter, de lungime maxima 50, care reprezinta adresa locatiei.
tara = variabila de tip caracter, de lungime maxima 30, care reprezinta tara in care se afla.
oras = variabila de tip caracter, de lungime maxima 30, care reprezinta oras in care se afla.
limba_oficiala = variabila de tip caracter, de lungime maxima 25, care reprezinta limba cea mai utilizata in tara respectiva.
moneda_oficiala = variabila de tip caracter, de lungime 3, care reprezinta prescurtarea monedei oficiale.

Entitatea AEROPORT are ca atribute:

cod_aeroport = variabila de tip intreg, de lungime maxima 5, care reprezinta codul unui aeroport.
cod_locatie = variabila de tip intreg, de lungime maxima 5, care reprezinta codul locatiei la care se afla aeroportul.
nume = variabila de tip caracter, de lungime maxima 50, care reprezinta numele aeroportului.
zbor_international = variabila de tip caracter, care ia valorile da sau nu, de lungime 2, care ofera informatii despre tipul de aeroport, daca exista zboruri intercontinentale care pornesc de pe acel aeroport.

Entitatea TERMINAL are ca atribute:

Cod_terminal = variabila de tip intreg, de lungime maxima 5, care reprezinta codul terminalului.
Cod_aeroport = variabila de tip intreg, de lungime maxima 5, care reprezinta codul aeroportului in care se afla terminalul.
Capacitate = variabila de tip intreg, de lungime maxima 5, care reprezinta numarul de persoane care se pot afla in cladire la un moment dat.
Pozitie = variabila de tip caracter, de lungime maxima 10, care reprezinta pozitia terminalului in cadrul aeroportului (pozitia geografica).

Entitatea POARTA are ca atribute:

Cod_poarta = variabila de tip intreg, de lungime maxima 5, care reprezinta codul portii.
Cod_terminal = variabila de tip intreg, de lungime maxima 5, care reprezinta codul terminalului in care se afla poarta.
Capacitate = variabila de tip intreg, de lungime maxima 5, care reprezinta numarul de persoane care se pot afla in zona portii la un moment dat.
Pozitie = variabila de tip caracter, de lungime maxima 50, care reprezinta pozitia portii in cadrul terminalului.

Entitatea PISTA are ca atribute:

Cod_pista = variabila de tip intreg, de lungime maxima 5, care reprezinta codul pistei.

Lungime = variabila de tip intreg, de lungime maxima 4, care reprezinta lungimea pistei.
Latime = variabila de tip intreg, de lungime maxima 2, care reprezinta latimea pistei.
Orientare = variabila de tip caracter, de lungime maxima 30, care precizeaza orientarea pistei.

Entitatea ZBOR are ca atribute:

Cod_zbor = variabila de tip intreg, de lungime maxima 5, care reprezinta codul zborului.
Data = variabila de tip data calendaristica, care reprezinta data zborului.
Ora_decolare = variabila de tip caracter, de lungime maxima 5, care reprezinta ora decolarii.
Intarziere = variabila de tip intreg, de lungime maxim 3, care reprezinta intarzierea zborului, in minute.
Cod_avion = variabila de tip intreg, de lungime maxima 5, care reprezinta codul avionului cu care se realizeaza zborul.
Cod_pista1 = variabila de tip intreg, de lungime maxima 5, care reprezinta codul pistei de pe care decoleaza.
Cod_pista2 = variabila de tip intreg, de lungime maxima 5, care reprezinta codul pistei pe care aterizeaza.
Cod_ruta = variabila de tip intreg, de lungime maxima 5, care reprezinta codul rutei pe care se realizeaza zborul.

Entitatea RUTA are ca atribute:

Cod_ruta = variabila de tip intreg, de lungime maxima 5, care reprezinta codul rutei.
Escala = variabila de tip caracter, de lungime maxima 50, care reprezinta numele orasului si al tarii in care este facuta escala sau nu daca nu exista.
Distanta = variabila de tip intreg, de lungime maxima 5, care reprezinta dinstanta rutei, in kilometri.
Timp = variabila de tip intreg, de lungime maxima 4, care reprezinta timpul mediu facut pe acea ruta, in minute.

Entitatea AVION are ca atribute:

Cod_avion = variabila de tip intreg, de lungime maxima 5, care reprezinta codul avionului.
Numar_locuri = variabila de tip intreg, de lungime maxima 3, care reprezinta numarul de locuri din avion, destinate pasagerilor.
Inaltime = variabila de tip intreg, de lungime maxima 2, care reprezinta inaltimea avionului.
Lungime = variabila de tip intreg, de lungime maxima 2, care reprezinta lungimea avionului.
Zbor_intercontinental = variabila de tip caracter, care ia valorile da si nu, de lungime 2, care indica daca acel avion poate efectua zboruri intercontinentale.
Model = variabila de tip caracter, de lungime maxima 20, care reprezinta modelul avionului.

Entitatea LOC are ca atribute:

Cod_loc = variabila de tip intreg, de lungime maxima 5, care reprezinta codul locului.

Cod_avion = variabila de tip intreg, de lungime maxima 5, care reprezinta codul avionului in care se afla locul.

Numar_loc = variabila de tip intreg, de lungime maxima 3, care reprezinta numarul locului, ca un nume al acestuia.

Pozitie = variabila de tip caracter, de lungime maxima 9, care descrie pozitia locului.

Cod_clasa = variabila de tip intreg, de lungime maxima 5, care reprezinta codul clasei in care se incadreaza locul.

Entitatea CLASA are ca attribute:

Cod_clasa = variabila de tip intreg, de lungime maxima 5, care reprezinta codul clasei.

Pret = variabila de tip intreg, de lungime maxima 4, care reprezinta pretul de baza al biletelor de la acea clasa.

Entitatea FACILITATE are ca attribute:

Cod_facilitate = variabila de tip intreg, de lungime maxima 5, care reprezinta codul facilitatii.

Nume_facilitate = variabila de tip caracter, de lungime maxima 25, care reprezinta denumirea facilitatii.

Pret_suplimentar = variabila de tip intreg, de lungime maxima 3, care reprezinta pretul facilitatii, in caz ca aceasta nu este una standard.

Standard = variabila de tip standard, ce poate lua valorile da si nu, de lungime 2, care precizeaza daca este o facilitate standard a clasei.

Entitatea REZERVARE are ca attribute:

Cod_rezervare = variabila de tip intreg, de lungime maxima 5, care reprezinta codul rezervarii.

Pret_total = variabila de tip intreg, de lungime maxima 6, care reprezinta pretul total al biletelor de pe o rezervare, cu tot cu eventualele costuri suplimentare.

Entitatea PASAGER are ca attribute:

Cod_pasager = variabila de tip intreg, de lungime maxima 5, care reprezinta codul pasagerului.

Nume = variabila de tip caracter, de lungime maxima 25, care reprezinta numele pasagerului.

Prenume = variabila de tip caracter, de lungime maxima 25, care reprezinta prenumele pasagerului.

Nationalitate = variabila de tip caracter, de lungime maxima 25, care reprezinta nationalitatea pasagerului.

Numar_pasaport = variabila de tip intreg, de lungime maxima 15, care reprezinta numarul pasaportului pasagerului.

Entitatea ANGAJAT are ca attribute:

Cod_angajat = variabila de tip intreg, de lungime maxima 5, care reprezinta codul angajatului.
Nume = variabila de tip caracter, de lungime maxima 25, care reprezinta numele angajatului.
Prenume = variabila de tip caracter, de lungime maxima 25, care reprezinta prenumele angajatului.
Data_nasterii = variabila de tip data calendaristica, care reprezinta data nasterii angajatului.
Data_angajarii = variabila de tip data calendaristica, care reprezinta data la care persoana a fost angajata in cadrul companiei.
Nationalitate = variabila de tip caracter, de lungime maxima 25, care descrie nationalitatea angajatului.

Subentitatea PILOT are ca attribute:

Cod_angajat = variabila de tip intreg, de lungime maxima 5, care reprezinta codul pilotului.
Educatie = variabila de tip caracter, de lungime maxima 30, care denumeste scoala de pilotaj urmata.
Experienta = variabila de tip intreg, de lungime maxima 2, care reprezinta anii de experienta.

Subentitatea INSOTITOR_DE_BORD are ca attribute:

Cod_angajat = variabila de tip intreg, de lungime maxima 5, care reprezinta codul insotitorului de bord.
Experienta = variabila de tip intreg, de lungime maxima 2, care reprezinta anii de experienta.
Limba_straina = variabila de tip caracter, de lungime maxima 30, care reprezinta limba straina pe care o poate vorbi fluent.

Subentitatea INGINER are ca attribute:

Cod_angajat = variabila de tip intreg, de lungime maxima 5, care reprezinta codul inginerului.
Educatie = variabila de tip caracter, de lungime maxima 30, care denumeste facultatea urmata.

Relatia RUTA_are_AEROPORT are ca attribute:

Cod_ruta = variabila de tip intreg, de lungime maxima 5, care reprezinta codul inginerului. variabila de tip intreg, de lungime maxima 5, care reprezinta codul rutei si care intra in componenta cheii primare.
Cod_aeroport1 = variabila de tip intreg, de lungime maxima 5, care reprezinta codul aeroportului de pe care se decoleaza si care intra in componenta cheii primare.
Cod_aeroport2 = variabila de tip intreg, de lungime maxima 5, care reprezinta codul aeroportului pe care se aterizeaza si care intra in componenta cheii primare.

Relatia CLASA_ofera_FACILITATE are ca attribute:

Cod_clasa = variabila de tip intreg, de lungime maxima 5, care reprezinta codul clasei, care intra in componenta cheii primare.
Cod_facilitate = variabila de tip intreg, de lungime maxima 5, care reprezinta codul facilitatii, care intra in componenta cheii primare.

Relatia PASAGER_cumpara_REZERVARE are ca atribute:

Cod_rezervare = variabila de tip intreg, de lungime maxima 5, care reprezinta codul rezervarii, care intra in componenta cheii primare.

Cod_pasager = variabila de tip intreg, de lungime maxima 5, care reprezinta codul pasagerului, care intra in componenta cheii primare.

Numar_locuri = variabila de tip intreg, de lungime maxima 2, care reprezinta numarul de locuri pe care un pasager le cumpara in cadrul unei rezervari.

Relatia REZERVAREA_rezerva_LOC_in_ZBOR are ca atribute:

Cod_rezervare = variabila de tip intreg, de lungime maxima 5, care reprezinta codul rezervarii, care intra in componenta cheii primare.

Cod_loc = variabila de tip intreg, de lungime maxima 5, care reprezinta codul locului din rezervare, care intra in componenta cheii primare.

Cod_zbor = variabila de tip intreg, de lungime maxima 5, care reprezinta codul zborului, care intra in componenta cheii primare.

Data = variabila de tip data, care descrie data la care a fost facuta rezervarea.

Relatia ANGAJAT_lucreaza_ZBOR are ca atribute:

Cod_angajat = variabila de tip intreg, de lungime maxima 5, care reprezinta codul angajatului, care intra in componenta cheii primare.

Cod_zbor = variabila de tip intreg, de lungime maxima 5, care reprezinta codul zborului, care intra in componenta cheii primare.

Ore = variabila de tip intreg, de lungime maxima 2, care reprezinta numarul de ore pe care le lucreaza angajatul in cadrul zborului respectiv.

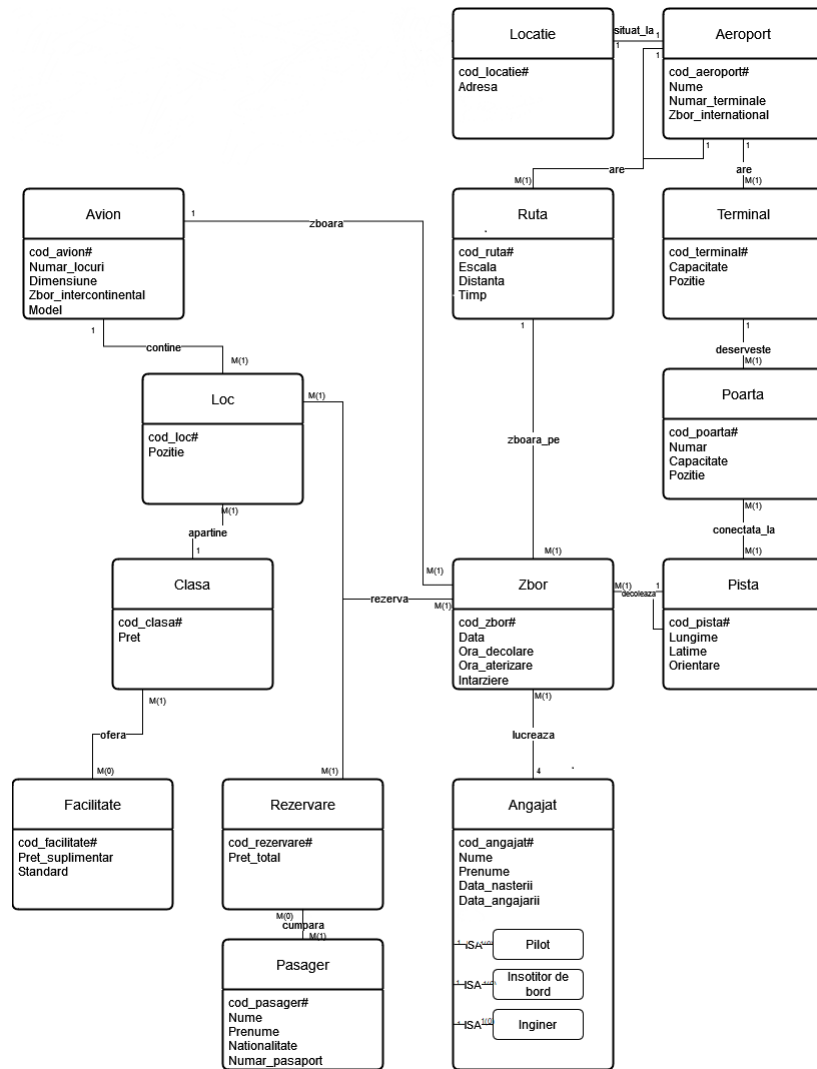
Spor = variabila de tip intreg, de lungime maxima 3, care reprezinta banii pe care ii primeste pentru zborul respectiv.

Relatia PISTA_CONECTATA_POARTA are ca atribute:

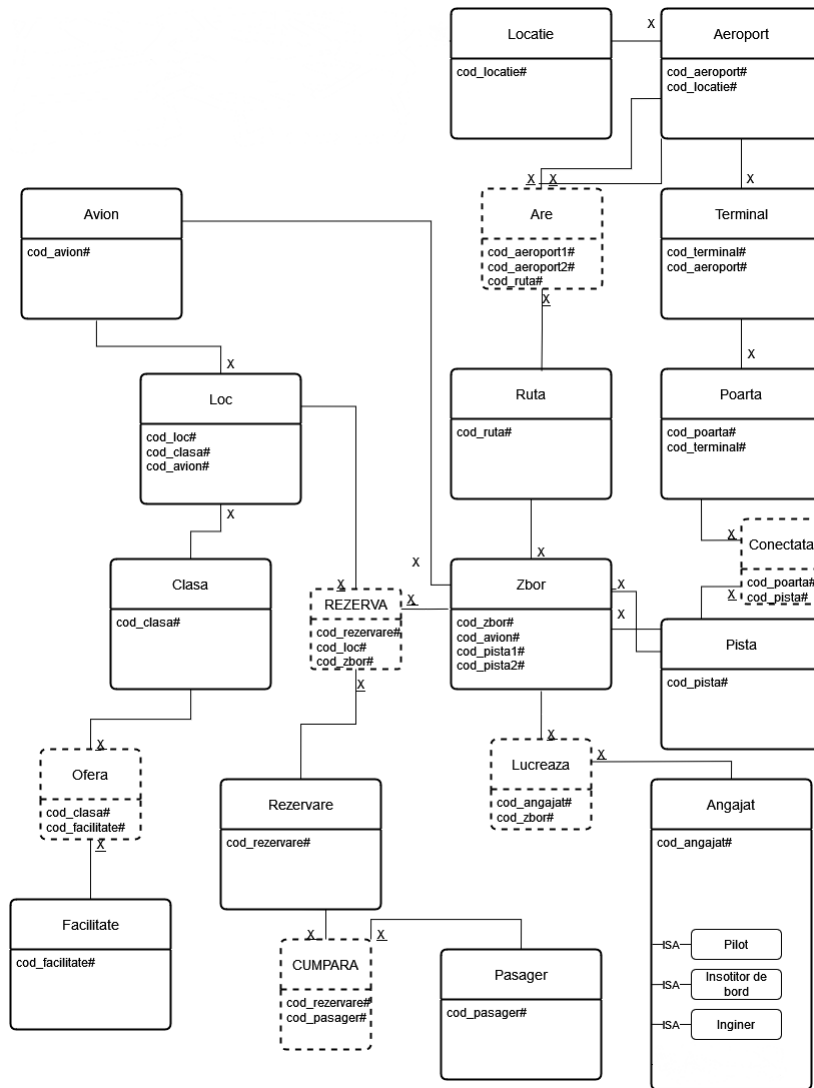
Cod_pista = variabila de tip intreg, de lungime maxima 5, care reprezinta codul pistei, care intra in componenta cheii primare.

Cod_poarta = variabila de tip intreg, de lungime maxima 5, care reprezinta codul portii, care intra in componenta cheii primare.

6 Diagrama E/R



7 Diagrama conceptuala



8 Enumerarea schemelor relationale corespunzatoare diagramei conceptuale la punctul 7.

LOCATIE (cod_locatie#, adresa, tara, oras, limba_oficiala, moneda_oficiala)

AEROPORT (cod_aeroport#, cod_locatie, nume, zbor_international)

TERMINAL (cod_terminal#, cod_aeroport, capacitate, pozitie)

POARTA (cod_poarta#, cod_terminal, capacitate, pozitie)

PISTA (cod_pista#, lungime, latime, orientare)

RUTA (cod_ruta#, escala, distanta, timp)

AVION (cod_avion#, numar_locuri, inaltime, lungime, zbor_intercontinental, model)

LOC (cod_loc#, cod_avion#, pozitie, numar_loc, cod_clasa)

CLASA (cod_clasa#, pret)

FACILITATE (cod_facilitate#, nume_facilitate, pret_suplimentar, standard)

REZERVARE (cod_rezervare#, pret_total)

PASAGER (cod_pasager#, nume, prenume, nationalitate, numar_pasaport)

ANGAJAT (cod_angajat#, nume, prenume, data_nasterii, data_angajarii, nationalitate)

ZBOR (cod_zbor#, data, ora_decolare, intarziere, cod_avion, cod_pista1, cod_pista2, cod_ruta)

PILOT (cod_angajat#, educatie, experienta)

INSOTITOR_DE_BORD (cod_angajat#, experienta, limba_straina)

INGINER (cod_angajat#, educatie)

ARE (cod_aeroport1#, cod_aeroport2#, cod_ruta#)

OFERA (cod_clasa#, cod_facilitate#)

CUMPARA (cod_rezervare#, cod_pasager#, numar_locuri)

REZERVA (cod_rezervare#, cod_loc#, cod_zbor#, data)

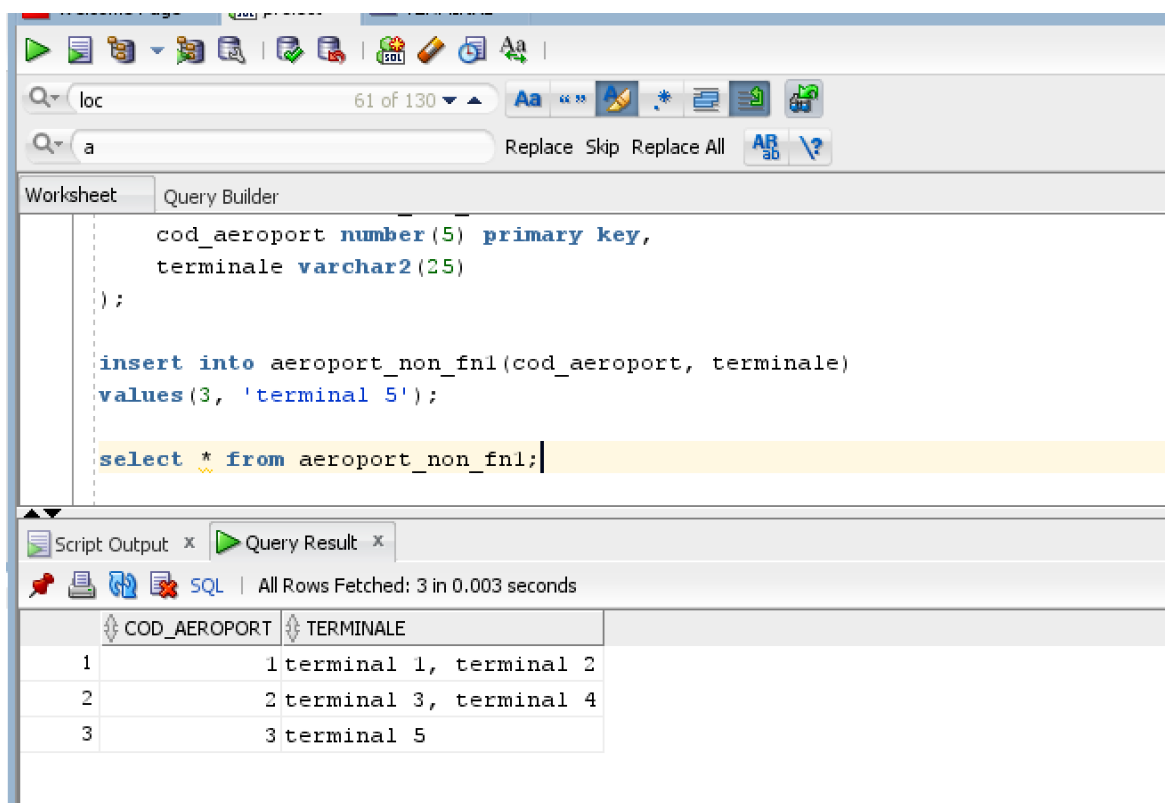
LUCREAZA (cod_angajat#, cod_zbor#, ore, spor)

CONECTATA (cod_poarta#, cod_pista#).

9 Realizarea normalizarii pana la forma normala 3 (FN1 – FN3)

In acest moment, diagrama conceptuala de la punctul 7 este deja in forma normala 3, asa ca vom prezenta atat trei exemple de relatii, care nu se afla in formele normale, cat si modul in care le putem normaliza. Acestea sunt independente intre ele si privesc compania aeriana.

Compania aeriana realizeaza zboruri intre aeroporturi, fiecare aeroport putand avea mai multe terminale. Daca am pastra aceste informatii sub forma:



The screenshot shows a SQL query editor with the following content:

```
cod_aeroport number(5) primary key,  
terminale varchar2(25)  
);  
  
insert into aeroport_non_fn1(cod_aeroport, terminale)  
values(3, 'terminal 5');  
  
select * from aeroport_non_fn1;
```

Below the query editor, the 'Query Result' tab is active, displaying the following data:

| | COD_AEROPORT | TERMINALE |
|---|--------------|------------------------|
| 1 | 1 | terminal 1, terminal 2 |
| 2 | 2 | terminal 3, terminal 4 |
| 3 | 3 | terminal 5 |

Atunci, nu s-ar mai afla in forma normala 1. Pentru a rezolva aceasta problema, putem proceda ca in cadrul diagramei conceptuale. In tabelul AEROPORT vom pastra informatiile comune tuturor terminalelor din cadrul unui aeroport, iar, pentru informatiile specifice, vom avea un tabel separat, care va avea cod_aeroport cheie externa.

Fiecare dintre relatiile din diagrama este compusa din attribute carora le corespunde o valoare indivizibila (atomica).

Mai departe, vom prezenta un exemplu de relatie ce nu se afla in forma normala 2. In primul rand, pentru a fi in forma normala 2, trebuie sa fie in forma normala 1. Pe deasupra, orice atribut ce nu contribuie la cheia

primara trebuie sa depinda de intreaga cheia primara, nu doar de o parte a acesteia.

Sa luam ca exemplu tabelul LUCREAZA din diagrama de la punctul 7. Orele pe care le lucreaza un angajat pentru un zbor depind atat de timpul jobului, cat si de zborul in sine (durata). De asemenea, bonusul depinde atat de angajat (tipul jobului, pregatirea, experienta), cat si de durata zborului. Asadar, in acest moment, este in FN2.

Sa presupunem ca bonusul ar fi depins doar de tipul jobului, adica toti angajatii de acelasi fel ar fi primit un bonus egal, indiferent de durata zborului. In acest caz, ar fi fost in forma normala 1, dar nu si in forma normala 2. Pentru normalizarea acestei relatii, am fi putut face un tabel separat in care sa avem codul tipul jobului si bonusul.

In continuare, vom presupune ca bonusul ar fi depins doar de orele pe care le lucreaza un angajat, ceea ce ar insemna ca relatia nu s-ar afla in forma normala 3. Astfel, bonusul ar fi calculat strict pe baza orelor, fara sa conteze codul angajatului sau codul zborului. Se poate observa ca acest lucru nu se intampla in diagrama pentru niciun atribut, nu doar pentru acesta.

Totusi, continuand presupunerea facuta, pentru a aduce aceasta relatie in forma normala 3, ar trebui sa avem un tabel suplimentar, in care sa avem numarul de ore si, pentru fiecare numar de ore lucrate in parte, bonusul aferent. Acest lucru nu este necesar in cadrul acestui proiect, intrucat bonusul depinde de mai multi factori, nu doar de numarul orelor lucrate.

10 Crearea unei secvente ce va fi utilizata in inserarea inregistrarilor in tabele (punctul 11).

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

11 Crearea tabelelor in SQL si inserarea de date coerente in fiecare dintre acestea (minimum 5 inregistrari in fiecare tabel neasociativ; minimum 10 inregistrari in tabelele asociative) .

```
CREATE TABLE LOCATIE(
```

```
cod_locatie number(5),  
adresa varchar2(50),  
tara varchar2(30) NOT NULL,  
oras varchar(30) NOT NULL,  
limba_oficiala varchar2(25),  
moneda_oficiala char(3),  
CONSTRAINT pk_locatie PRIMARY KEY(cod_locatie)
```

```
);
```

```
CREATE TABLE AEROPORT(
```

```
cod_aeroport number(5),  
cod_locatie number(5),  
nume varchar2(50),  
zbor_international char(2),  
CONSTRAINT pk_aeroport PRIMARY KEY(cod_aeroport),  
CONSTRAINT fk_aeroport FOREIGN KEY(cod_locatie) REFERENCES LOCATIE(cod_locatie)
```

```
);
```

```
CREATE TABLE TERMINAL(
```

```
cod_terminal number(5),  
cod_aeroport number(5),  
capacitate number(5),  
pozitie varchar(10),  
CONSTRAINT pk_terminal PRIMARY KEY(cod_terminal),  
CONSTRAINT fk_terminal FOREIGN KEY(cod_aeroport) REFERENCES AEROPORT(cod_aeroport)
```

);

CREATE TABLE POARTA(

cod_poarta number(5),
cod_terminal number(5),
capacitate number(5),
pozitie varchar(50),
CONSTRAINT pk_poarta PRIMARY KEY(cod_poarta),
CONSTRAINT fk_poarta FOREIGN KEY(cod_terminal) REFERENCES TERMINAL(cod_terminal)

);

CREATE TABLE PISTA(

cod_pista number(5),
lungime number(4),
latime number(2),
orientare varchar(30),
CONSTRAINT pk_pista PRIMARY KEY(cod_pista)

);

CREATE TABLE RUTA(

cod_ruta number(5),
escala varchar2(50),
distanta number(5),
timp number(4),
CONSTRAINT pk_ruta PRIMARY KEY(cod_ruta)

);

CREATE TABLE AVION(

```

cod_avion number(5),
numar_locuri number(3),
inaltime number(2),
lungime number(2),
zbor_intercontinental char(2),
model varchar2(20),
CONSTRAINT pk_avion PRIMARY KEY(cod_avion)

);

```

```

CREATE TABLE LOC(

```

```

cod_loc number(5),
cod_avion number(5),
pozitie varchar2(9),
numar_loc number(3),
cod_clasa number(5),
CONSTRAINT pk_loc PRIMARY KEY(cod_loc, cod_avion),
CONSTRAINT fk_loc FOREIGN KEY(cod_clasa) REFERENCES CLASA(cod_clasa)

);

```

```

CREATE TABLE CLASA(

```

```

cod_clasa number(5),
pret number(4),
CONSTRAINT pk_clasa PRIMARY KEY(cod_clasa)

);

```

```

CREATE TABLE FACILITATE(

```

```

cod_facilitate number(5),
nume_facilitate varchar2(25),
pret_suplimentar number(3),
standard char(2),

```

```
CONSTRAINT pk_facilitate PRIMARY KEY(cod_facilitate)
```

```
);
```

```
CREATE TABLE REZERVARE(
```

```
cod_rezervare number(5),
```

```
pret_total number(5),
```

```
CONSTRAINT pk_rezervare PRIMARY KEY(cod_rezervare),
```

```
CONSTRAINT fk_rezervare FOREIGN KEY(cod_pasager) REFERENCES PASAGER(cod_pasager)
```

```
);
```

```
CREATE TABLE PASAGER(
```

```
cod_pasager number(5),
```

```
nume varchar2(25),
```

```
prenume varchar2(25),
```

```
nationalitate varchar2(25),
```

```
numar_pasaport number(8),
```

```
CONSTRAINT pk_pasager PRIMARY KEY(cod_pasager)
```

```
);
```

```
CREATE TABLE ANGAJAT(
```

```
cod_angajat number(5),
```

```
nume varchar2(25),
```

```
prenume varchar2(25),
```

```
data_nasterii date,
```

```
data_angajarii date,
```

```
nationalitate varchar2(25),
```

```
CONSTRAINT pk_angajat PRIMARY KEY(cod_angajat)
```

```
);
```

```
CREATE TABLE PILOT(
```

```
cod_angajat number(5),  
educatie varchar2(50),  
experienta number(2),  
CONSTRAINT pk_pilot FOREIGN KEY (cod_angajat) REFERENCES ANGAJAT(cod_angajat)
```

```
);
```

```
CREATE TABLE INSOTITOR_DE_BORD(
```

```
cod_angajat number(5),  
experienta number(2),  
limba_straina varchar2(25),  
CONSTRAINT pk_insotitor FOREIGN KEY (cod_angajat) REFERENCES ANGAJAT(cod_angajat)
```

```
);
```

```
CREATE TABLE INGINER(
```

```
cod_angajat number(5),  
educatie varchar2(50),  
CONSTRAINT fk_inginer FOREIGN KEY (cod_angajat) REFERENCES ANGAJAT(cod_angajat)
```

```
);
```

```
CREATE TABLE ZBOR(
```

```
cod_zbor number(5),  
data date,  
ora_decolare time,  
intarziere number(),  
cod_avion number(5),  
cod_pista1 number(5),  
cod_pista2 number(5),
```

```

cod_ruta number(5),
CONSTRAINT pk_zbor PRIMARY KEY(cod_zbor),
CONSTRAINT fk_zbor1 FOREIGN KEY(cod_pista1) REFERENCES PISTA(cod_pista),
CONSTRAINT fk_zbor2 FOREIGN KEY(cod_pista2) REFERENCES PISTA(cod_pista),
CONSTRAINT fk_zbor3 FOREIGN KEY(cod_ruta) REFERENCES RUTA(cod_ruta)

```

```
);
```

```
CREATE TABLE ARE(
```

```

cod_aeroport1 number(5),
cod_aeroport2 number(5),
cod_ruta number(5),
CONSTRAINT pk_are PRIMARY KEY(cod_aeroport1, cod_aeroport2, cod_ruta),
CONSTRAINT fk_are1 FOREIGN KEY(cod_aeroport1) REFERENCES AEROPORT(cod_aeroport),
CONSTRAINT fk_are2 FOREIGN KEY(cod_aeroport2) REFERENCES AEROPORT(cod_aeroport),
CONSTRAINT fk_are3 FOREIGN KEY(cod_ruta) REFERENCES RUTA(cod_ruta)

```

```
);
```

```
CREATE TABLE OFERA(
```

```

cod_clasa number(5),
cod_facilitate number(5),
CONSTRAINT pk_ofera PRIMARY KEY(cod_clasa, cod_facilitate),
CONSTRAINT fk_ofera1 FOREIGN KEY(cod_clasa) REFERENCES CLASA(cod_clasa),
CONSTRAINT fk_ofera2 FOREIGN KEY(cod_facilitate) REFERENCES FACILITATE(cod_facilitate)

```

```
);
```

```
CREATE TABLE CUMPARA(
```

```

cod_rezervare number(5),
cod_pasager number(5),
numar_locuri number(2),
CONSTRAINT pk_cumpara PRIMARY KEY(cod_rezervare, cod_pasager),

```



```
CONSTRAINT fk_cumpara1 FOREIGN KEY(cod_rezervare) REFERENCES REZERVARE(cod_rezervare),  
CONSTRAINT fk_cumpara2 FOREIGN KEY(cod_pasager) REFERENCES PASAGER(cod_pasager)
```

```
);
```

```
CREATE TABLE REZERVA(
```

```
cod_rezervare number(5),  
cod_loc number(5),  
cod_zbor number(5),  
data_data,  
CONSTRAINT pk_rezerva PRIMARY KEY(cod_rezervare, cod_loc, cod_zbor),  
CONSTRAINT fk_rezerva1 FOREIGN KEY(cod_rezervare) REFERENCES REZERVARE(cod_rezervare),  
CONSTRAINT fk_rezerva2 FOREIGN KEY(cod_loc) REFERENCES LOC(cod_loc),  
CONSTRAINT fk_rezerva3 FOREIGN KEY(cod_zbor) REFERENCES ZBOR(cod_zbor)
```

```
);
```

```
CREATE TABLE LUCREAZA(
```

```
cod_angajat number(5),  
cod_zbor number(5),  
ore number(2),  
spor number(4),  
CONSTRAINT pk_lucreaza PRIMARY KEY(cod_angajat, cod_zbor),  
CONSTRAINT fk_lucreaza1 FOREIGN KEY(cod_angajat) REFERENCES ANGAJAT(cod_angajat),  
CONSTRAINT fk_lucreaza2 FOREIGN KEY(cod_zbor) REFERENCES ZBOR(cod_zbor)
```

```
);
```

```
CREATE TABLE CONECTATA(
```

```
cod_poarta number(5),  
cod_pista number(5),  
CONSTRAINT pk_CONECTATA PRIMARY KEY(cod_poarta, cod_pista),  
CONSTRAINT fk_CONECTATA1 FOREIGN KEY(cod_poarta) REFERENCES POARTA(cod_poarta),
```

CONSTRAINT fk_CONECTATA2 FOREIGN KEY(cod_pista) REFERENCES PISTA(cod_pista)

);

--PENTRU LOCATIE

INSERT INTO LOCATIE(cod_locatie, adresa, tara, oras, limba_oficiala, moneda_oficiala)
VALUES (CHEIE.NEXTVAL, 'JFK Access Rd, NY 11430', 'Statele Unite ale Americii', 'New York',
'engleza', 'USD');

INSERT INTO LOCATIE(cod_locatie, adresa, tara, oras, limba_oficiala, moneda_oficiala)
VALUES (CHEIE.NEXTVAL, 'Longford TW6', 'Regatul Unit', 'Londra', 'engleza', 'GBP');

INSERT INTO LOCATIE(cod_locatie, adresa, tara, oras, limba_oficiala, moneda_oficiala)
VALUES (CHEIE.NEXTVAL, '1 Sky Plaza Rd', 'Hong Kong', 'Hong Kong', null, 'HKD');

INSERT INTO LOCATIE(cod_locatie, adresa, tara, oras, limba_oficiala, moneda_oficiala)
VALUES (CHEIE.NEXTVAL, '95700 Roissy-en-France', 'Franta', 'Paris', 'franceza', 'EUR');

INSERT INTO LOCATIE(cod_locatie, adresa, tara, oras, limba_oficiala, moneda_oficiala)
VALUES (CHEIE.NEXTVAL, '60547 Frankfurt', 'Germania', 'Frankfurt', 'germana', 'EUR');

INSERT INTO LOCATIE(cod_locatie, adresa, tara, oras, limba_oficiala, moneda_oficiala)
VALUES (CHEIE.NEXTVAL, '272 Gonghang-ro, Jung-gu, Incheon', 'Coreea de Sud', 'Seul', 'coreeana',
'KRW');

INSERT INTO LOCATIE(cod_locatie, adresa, tara, oras, limba_oficiala, moneda_oficiala)
VALUES (CHEIE.NEXTVAL, 'Delhi 110037', 'India', 'New Delhi', 'hindi', 'INR');

INSERT INTO LOCATIE(cod_locatie, adresa, tara, oras, limba_oficiala, moneda_oficiala)
VALUES (CHEIE.NEXTVAL, 'Mascot NSW 2020', 'Australia', 'Sydney', 'engleza', 'AUD');

INSERT INTO LOCATIE(cod_locatie, adresa, tara, oras, limba_oficiala, moneda_oficiala)
VALUES (CHEIE.NEXTVAL, '6301 Silver Dart Dr, Mississauga, L5P 1B2', 'Canada', 'Toronto', 'en-
gleza', 'CAD');

INSERT INTO LOCATIE(cod_locatie, adresa, tara, oras, limba_oficiala, moneda_oficiala)
VALUES (CHEIE.NEXTVAL, 'Calea Bucurestilor', 'Romania', 'Bucuresti', 'romana', 'RON');

INSERT INTO LOCATIE(cod_locatie, adresa, tara, oras, limba_oficiala, moneda_oficiala)
VALUES (CHEIE.NEXTVAL, 'Strada Aeroportului 1', 'Romania', 'Iasi', 'romana', 'RON');

```
INSERT INTO LOCATIE(cod_locatie, adresa, tara, oras, limba_oficiala, moneda_oficiala)
VALUES (CHEIE.NEXTVAL, 'Strada Aeroportului', 'Romania', 'Craiova', 'romana', 'RON');
```

```
INSERT INTO LOCATIE(cod_locatie, adresa, tara, oras, limba_oficiala, moneda_oficiala)
VALUES (CHEIE.NEXTVAL, 'Via dell Aeroporto di Fiumicino', 'Italia', 'Roma', 'italiana', 'EUR');
```

SQL Worksheet History

Search: chei Replace Skip Replace All Replace Completed: 0 Skipped, 176 Replaced

Worksheet Query Builder

```
INSERT INTO LOCATIE(cod_locatie, adresa, tara, oras, limba_oficiala, moneda_oficiala)
VALUES (CHEIE.NEXTVAL, 'Strada Aeroportului 1', 'Romania', 'Iasi', 'romana', 'RON');
```

Script Output x Query Result x

SQL All Rows Fetched: 13 in 0.012 seconds

| COD_LOCATIE | ADRESA | TARA | ORAS | LIMBA_OFICIALA | MONEDA_OFICIALA |
|-------------|--|----------------------------|------------------|----------------|-----------------|
| 1 | 10 JFK Access Rd, NY 11430 | Statele Unite ale Americii | New York | engleza | USD |
| 2 | 20 Longford TW6 | Regatul Unit | Londra | engleza | GBP |
| 3 | 30 1 Sky Plaza Rd | Hong Kong | Hong Kong (null) | | HKD |
| 4 | 40 95700 Roissy-en-France | Franta | Paris | franceza | EUR |
| 5 | 50 60547 Frankfurt | Germania | Frankfurt | germana | EUR |
| 6 | 60 272 Gonghang-ro, Jung-gu, Incheon | Coreea de Sud | Seul | coreeana | KRW |
| 7 | 70 Delhi 110037 | India | New Delhi | hindi | INR |
| 8 | 80 Mascot NSW 2020 | Australia | Sydney | engleza | AUD |
| 9 | 90 6301 Silver Dart Dr, Mississauga, L5P 1B2 | Canada | Toronto | engleza | CAD |
| 10 | 100 Calea Bucurestilor | Romania | Bucuresti | romana | RON |
| 11 | 110 Strada Aeroportului 1 | Romania | Iasi | romana | RON |
| 12 | 120 Strada Aeroportului | Romania | Craiova | romana | RON |
| 13 | 130 Via dell Aeroporto di Fiumicino | Italia | Roma | italiana | EUR |

-INSERARI PENTRU AEROPORT

```
INSERT INTO AEROPORT(cod_aeroport, cod_locatie, nume, zbor_international)
VALUES (chei.NEXTVAL, 10, 'John F. Kennedy International Airport', 'DA');
```

```
INSERT INTO AEROPORT(cod_aeroport, cod_locatie, nume, zbor_international)
VALUES (chei.NEXTVAL, 20, 'Aeroportul International Heathrow', 'DA');
```

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

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

INSERT INTO AEROPORT(cod_aeroport, cod_locatie, nume, zbor_international)
VALUES (chei.NEXTVAL, 40, 'Aeroportul Internațional Charles de Gaulle', 'DA');

INSERT INTO AEROPORT(cod_aeroport, cod_locatie, nume, zbor_international)
VALUES (chei.NEXTVAL, 50, 'Aeroportul Internațional de la Frankfurt', 'DA');

INSERT INTO AEROPORT(cod_aeroport, cod_locatie, nume, zbor_international)
VALUES (chei.NEXTVAL, 60, 'Aeroportul Internațional Incheon', 'DA');

INSERT INTO AEROPORT(cod_aeroport, cod_locatie, nume, zbor_international)
VALUES (chei.NEXTVAL, 70, 'Aeroportul Internațional Indira Gandhi', 'DA');

INSERT INTO AEROPORT(cod_aeroport, cod_locatie, nume, zbor_international)
VALUES (chei.NEXTVAL, 80, 'Aeroportul Internațional din Sydney', 'DA');

INSERT INTO AEROPORT(cod_aeroport, cod_locatie, nume, zbor_international)
VALUES (chei.NEXTVAL, 90, 'Aeroportul Internațional Pearson Toronto', 'DA');

INSERT INTO AEROPORT(cod_aeroport, cod_locatie, nume, zbor_international)
VALUES (chei.NEXTVAL, 100, 'Aeroportul Internațional Henri Coandă București', 'DA');

INSERT INTO AEROPORT(cod_aeroport, cod_locatie, nume, zbor_international)
VALUES (chei.NEXTVAL, 110, 'Aeroportul Internațional Iași', 'DA');

INSERT INTO AEROPORT(cod_aeroport, cod_locatie, nume, zbor_international)
VALUES (chei.NEXTVAL, 120, 'Aeroportul Internațional Craiova', 'DA');

INSERT INTO AEROPORT(cod_aeroport, cod_locatie, nume, zbor_international)
VALUES (chei.NEXTVAL, 130, 'Aeroportul Internațional Leonardo da Vinci', 'DA');

SQL Worksheet History

Q: chei

Q: CHEIE Replace Skip Replace All Replace Completed: 0 Skipped, 176 Replaced

Worksheet Query Builder

```

INSERT INTO AEROPORT(cod_aeroport, cod_locatie, nume, zbor_international)
VALUES (CHEIE.NEXTVAL, 100, 'Aeroportul International Henri Coanda Bucuresti', 'DA');

INSERT INTO AEROPORT(cod_aeroport, cod_locatie, nume, zbor_international)
VALUES (CHEIE.NEXTVAL, 110, 'Aeroportul din Iasi', 'NU');

```

Script Output x Query Result x

SQL | All Rows Fetched: 13 in 0.006 seconds

| | COD_AEROPORT | COD_LOCATIE | NUME | ZBOR_INTERNATIONAL |
|----|--------------|-------------|---|--------------------|
| 1 | 140 | 10 | John F. Kennedy International Airport | DA |
| 2 | 150 | 20 | Aeroportul International Heathrow | DA |
| 3 | 160 | 30 | Aeroportul International de la Hong Kong | DA |
| 4 | 170 | 40 | Aeroportul International Charles de Gaulle | DA |
| 5 | 180 | 50 | Aeroportul International de la Frankfurt | DA |
| 6 | 190 | 60 | Aeroportul International Incheon | DA |
| 7 | 200 | 70 | Aeroportul International Indira Gandhi | DA |
| 8 | 210 | 80 | Aeroportul International din Sydney | DA |
| 9 | 220 | 90 | Aeroportul International Pearson Toronto | DA |
| 10 | 230 | 100 | Aeroportul International Henri Coanda Bucuresti | DA |
| 11 | 240 | 110 | Aeroportul din Iasi | NU |
| 12 | 250 | 120 | Aeroportul International Craiova | DA |
| 13 | 260 | 130 | Aeroportul International Leonardo da Vinci | DA |

-INSERTURI PENTRU TERMINAL

```

INSERT INTO TERMINAL(cod_terminal, cod_aeroport, capacitate, pozitie)
VALUES (chei.NEXTVAL, 140, 2500, 'SUD');

```

```

INSERT INTO TERMINAL(cod_terminal, cod_aeroport, capacitate, pozitie)
VALUES (chei.NEXTVAL, 140, 1800, 'NORD-VEST');

```

```

INSERT INTO TERMINAL(cod_terminal, cod_aeroport, capacitate, pozitie)
VALUES (chei.NEXTVAL, 140, 3000, 'NORD-EST');

```

```

INSERT INTO TERMINAL(cod_terminal, cod_aeroport, capacitate, pozitie)
VALUES (chei.NEXTVAL, 150, 3000, 'CENTRAL');

```

```

INSERT INTO TERMINAL(cod_terminal, cod_aeroport, capacitate, pozitie)
VALUES (chei.NEXTVAL, 160, 2200, 'EST');

```

```
INSERT INTO TERMINAL(cod_terminal, cod_aeroport, capacitate, pozitie)
VALUES (chei.NEXTVAL, 160, 1500, 'VEST');
```

```
INSERT INTO TERMINAL(cod_terminal, cod_aeroport, capacitate, pozitie)
VALUES (chei.NEXTVAL, 170, 2800, 'NORD');
```

```
INSERT INTO TERMINAL(cod_terminal, cod_aeroport, capacitate, pozitie)
VALUES (chei.NEXTVAL, 170, 2100, 'SUD');
```

```
INSERT INTO TERMINAL(cod_terminal, cod_aeroport, capacitate, pozitie)
VALUES (chei.NEXTVAL, 180, 3200, 'EST');
```

```
INSERT INTO TERMINAL(cod_terminal, cod_aeroport, capacitate, pozitie)
VALUES (chei.NEXTVAL, 180, 2500, 'SUD-VEST');
```

```
INSERT INTO TERMINAL(cod_terminal, cod_aeroport, capacitate, pozitie)
VALUES (chei.NEXTVAL, 190, 2400, 'NORD');
```

```
INSERT INTO TERMINAL(cod_terminal, cod_aeroport, capacitate, pozitie)
VALUES (chei.NEXTVAL, 200, 2800, 'SUD-EST');
```

```
INSERT INTO TERMINAL(cod_terminal, cod_aeroport, capacitate, pozitie)
VALUES (chei.NEXTVAL, 210, 3500, 'CENTRAL');
```

```
INSERT INTO TERMINAL(cod_terminal, cod_aeroport, capacitate, pozitie)
VALUES (chei.NEXTVAL, 220, 1800, 'NORD-VEST');
```

```
INSERT INTO TERMINAL(cod_terminal, cod_aeroport, capacitate, pozitie)
VALUES (chei.NEXTVAL, 230, 2000, 'VEST');
```

```
INSERT INTO TERMINAL(cod_terminal, cod_aeroport, capacitate, pozitie)
VALUES (chei.NEXTVAL, 230, 1500, 'SUD-EST');
```

```
INSERT INTO TERMINAL(cod_terminal, cod_aeroport, capacitate, pozitie)
VALUES (chei.NEXTVAL, 240, 600, 'EST');
```

```
INSERT INTO TERMINAL(cod_terminal, cod_aeroport, capacitate, pozitie)
VALUES (chei.NEXTVAL, 250, 850, 'NORD');
```

```
INSERT INTO TERMINAL(cod_terminal, cod_aeroport, capacitate, pozitie)
VALUES (chei.NEXTVAL, 260, 3000, 'SUD');
```

```
INSERT INTO TERMINAL(cod_terminal, cod_aeroport, capacitate, pozitie)
VALUES (chei.NEXTVAL, 260, 2500, 'CENTRAL');
```

SQL Worksheet | History

Q= chei

Q= CHEIE Replace Skip Replace All Replace Completed: 0 Skipped, 176 Replaced

Worksheet Query Builder

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

Script Output x Query Result x

SQL | All Rows Fetched: 20 in 0.009 seconds

| | COD_TERMINAL | COD_AEROPORT | CAPACITATE | POZITIE |
|----|--------------|--------------|------------|-----------|
| 1 | 270 | 140 | 2500 | SUD |
| 2 | 280 | 140 | 1800 | NORD-VEST |
| 3 | 290 | 140 | 3000 | NORD-EST |
| 4 | 300 | 150 | 3000 | CENTRAL |
| 5 | 310 | 160 | 2200 | EST |
| 6 | 320 | 160 | 1500 | VEST |
| 7 | 330 | 170 | 2800 | NORD |
| 8 | 340 | 170 | 2100 | SUD |
| 9 | 350 | 180 | 3200 | EST |
| 10 | 360 | 180 | 2500 | SUD-VEST |
| 11 | 370 | 190 | 2400 | NORD |
| 12 | 380 | 200 | 2800 | SUD-EST |
| 13 | 390 | 210 | 3500 | CENTRAL |
| 14 | 400 | 220 | 1800 | NORD-VEST |
| 15 | 410 | 230 | 2000 | VEST |
| 16 | 420 | 230 | 1500 | SUD-EST |
| 17 | 430 | 240 | 600 | EST |
| 18 | 440 | 250 | 850 | NORD |
| 19 | 450 | 260 | 3000 | SUD |
| 20 | 460 | 260 | 2500 | CENTRAL |

–INSERTURI PENTRU POARTA

```
INSERT INTO POARTA(cod_poarta, cod_terminal, capacitate, pozitie)
VALUES(CHEIE.NEXTVAL, 270, 400, 'NORD');
```

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

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

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

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

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

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

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

```
INSERT INTO POARTA(cod_poarta, cod_terminal, capacitate, pozitie)
VALUES(CHEIE.NEXTVAL, 320, 400, 'NORD');
```

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

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

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

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



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

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

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

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

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

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

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

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

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

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

```
INSERT INTO POARTA(cod_poarta, cod_terminal, capacitate, pozitie)
VALUES(CHEIE.NEXTVAL, 450, 400, 'NORD');
```

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

SQL Worksheet | History

Q= chei

Q= CHEIE Replace Skip Replace All Replace Completed: 0 Skipped, 176 Replaced

Worksheet Query Builder

Script Output x Query Result x

All Rows Fetched: 25 in 0.003 seconds

| | COD_POARTA | COD_TERMINAL | CAPACITATE | POZITIE |
|----|------------|--------------|------------|-----------|
| 1 | 470 | 270 | 400 | NORD |
| 2 | 480 | 270 | 200 | SUD |
| 3 | 490 | 280 | 350 | NORD |
| 4 | 500 | 290 | 400 | CENTRAL |
| 5 | 510 | 290 | 290 | EST |
| 6 | 520 | 290 | 400 | NORD |
| 7 | 530 | 300 | 380 | VEST |
| 8 | 540 | 310 | 200 | NORD |
| 9 | 550 | 320 | 400 | NORD |
| 10 | 560 | 320 | 340 | SUD |
| 11 | 570 | 330 | 100 | CENTRAL |
| 12 | 580 | 340 | 180 | NORD-EST |
| 13 | 590 | 350 | 400 | NORD-VEST |
| 14 | 600 | 360 | 500 | SUD-EST |
| 15 | 610 | 370 | 400 | NORD |
| 16 | 620 | 380 | 300 | CENTRAL |
| 17 | 630 | 390 | 400 | NORD |
| 18 | 640 | 400 | 100 | EST |
| 19 | 650 | 410 | 400 | NORD |
| 20 | 660 | 410 | 400 | SUD |

SQL Worksheet | History

Q= chei

Q= CHEIE Replace Skip Replace All Replace Completed: 0 Skipped, 176 Replaced

Worksheet Query Builder

Script Output x Query Result x

All Rows Fetched: 25 in 0.003 seconds

| | COD_POARTA | COD_TERMINAL | CAPACITATE | POZITIE |
|----|------------|--------------|------------|-----------|
| 6 | 520 | 290 | 400 | NORD |
| 7 | 530 | 300 | 380 | VEST |
| 8 | 540 | 310 | 200 | NORD |
| 9 | 550 | 320 | 400 | NORD |
| 10 | 560 | 320 | 340 | SUD |
| 11 | 570 | 330 | 100 | CENTRAL |
| 12 | 580 | 340 | 180 | NORD-EST |
| 13 | 590 | 350 | 400 | NORD-VEST |
| 14 | 600 | 360 | 500 | SUD-EST |
| 15 | 610 | 370 | 400 | NORD |
| 16 | 620 | 380 | 300 | CENTRAL |
| 17 | 630 | 390 | 400 | NORD |
| 18 | 640 | 400 | 100 | EST |
| 19 | 650 | 410 | 400 | NORD |
| 20 | 660 | 410 | 400 | SUD |
| 21 | 670 | 420 | 200 | VEST |
| 22 | 680 | 430 | 256 | NORD |
| 23 | 690 | 440 | 420 | SUD-EST |
| 24 | 700 | 450 | 400 | NORD |
| 25 | 710 | 460 | 200 | EST |

-INSERTURI PENTRU PISTA

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

SQL Worksheet History

Search: chei

Replace Skip Replace All Replace Completed: 0 Skipped, 176 Replaced

Worksheet Query Builder

INSERT INTO PISTA(cod pista, lungime, latime, orientare)

Script Output x Query Result x

SQL | All Rows Fetched: 21 in 0.007 seconds

| | COD_PISTA | LUNGIME | LATIME | ORIENTARE |
|----|-----------|---------|--------------|-----------|
| 1 | 720 | 3200 | 45 Nord | |
| 2 | 730 | 3800 | 60 Est | |
| 3 | 740 | 3000 | 50 Nord-Vest | |
| 4 | 750 | 2800 | 55 Sud | |
| 5 | 760 | 3500 | 40 Sud-Est | |
| 6 | 770 | 4000 | 70 Vest | |
| 7 | 780 | 3200 | 45 Nord-Est | |
| 8 | 790 | 3700 | 55 Sud-Vest | |
| 9 | 800 | 3100 | 60 Est-Vest | |
| 10 | 810 | 2900 | 50 Nord-Sud | |
| 11 | 820 | 3300 | 40 Sud-Est | |
| 12 | 830 | 3600 | 70 Vest-Est | |
| 13 | 840 | 3200 | 45 Nord-Vest | |
| 14 | 850 | 3800 | 60 Est-Nord | |
| 15 | 860 | 3000 | 50 Nord-Est | |
| 16 | 870 | 2800 | 55 Sud-Vest | |
| 17 | 880 | 3500 | 40 Sud-Est | |
| 18 | 890 | 4000 | 70 Vest-Sud | |
| 19 | 900 | 3200 | 45 Nord-Est | |
| 20 | 910 | 3100 | 60 Est-Sud | |

–INSERTURI PENTRU RUTA

```
INSERT INTO RUTA(cod_ruta, escala, distanta, timp)
VALUES(CHEIE.NEXTVAL, null, 2500, 250);
```

```
INSERT INTO RUTA(cod_ruta, escala, distanta, timp)
VALUES(CHEIE.NEXTVAL, null, 6000, 530);
```

```
INSERT INTO RUTA(cod_ruta, escala, distanta, timp)
VALUES(CHEIE.NEXTVAL, null, 1000, 210);
```

```
INSERT INTO RUTA(cod_ruta, escala, distanta, timp)
VALUES(CHEIE.NEXTVAL, null, 2000, 430);
```

```
INSERT INTO RUTA(cod_ruta, escala, distanta, timp)
VALUES(CHEIE.NEXTVAL, 'Aeroportul International Charles de Gaulle', 5500, 650);
```

```
INSERT INTO RUTA(cod_ruta, escala, distante, timp)
VALUES(CHEIE.NEXTVAL, null, 800, 150);
```

```
INSERT INTO RUTA(cod_ruta, escala, distante, timp)
VALUES(CHEIE.NEXTVAL, null, 5500, 530);
```

```
INSERT INTO RUTA(cod_ruta, escala, distante, timp)
VALUES(CHEIE.NEXTVAL, null, 1800, 340);
```

```
INSERT INTO RUTA(cod_ruta, escala, distante, timp)
VALUES(CHEIE.NEXTVAL, null, 1500, 370);
```

```
INSERT INTO RUTA(cod_ruta, escala, distante, timp)
VALUES(CHEIE.NEXTVAL, 'Aeroportul International Henri Coanda Bucuresti', 700, 300);
```

SQL Worksheet History

Search: chei
Replace: CHEIE
Replace Completed: 0 Skipped, 176 Replaced

Worksheet Query Builder

```
INSERT INTO RUTA(cod_ruta, escala, distante, timp)
VALUES(CHEIE.NEXTVAL, null, 800, 150);

INSERT INTO RUTA(cod_ruta, escala, distante, timp)
VALUES(CHEIE.NEXTVAL, null, 5500, 530);

INSERT INTO RUTA(cod_ruta, escala, distante, timp)
VALUES(CHEIE.NEXTVAL, null, 1800, 340);
```

Script Output x Query Result x

SQL | All Rows Fetched: 10 in 0.005 seconds

| | COD_RUTA | ESCALA | DISTANTA | TIMP |
|----|----------|---|----------|------|
| 1 | 1470 | (null) | 2500 | 250 |
| 2 | 1480 | (null) | 6000 | 530 |
| 3 | 1490 | (null) | 1000 | 210 |
| 4 | 1500 | (null) | 2000 | 430 |
| 5 | 1510 | Aeroportul International Charles de Gaulle | 5500 | 650 |
| 6 | 1520 | (null) | 800 | 150 |
| 7 | 1530 | (null) | 5500 | 530 |
| 8 | 1540 | (null) | 1800 | 340 |
| 9 | 1550 | (null) | 1500 | 370 |
| 10 | 1560 | Aeroportul International Henri Coanda Bucuresti | 700 | 300 |

–INSERTURI PENTRU AVION

```
INSERT INTO AVION (cod_avion, numar_locuri, inaltime, lungime, zbor_intercontinental, model)
VALUES (CHEIE.NEXTVAL, 150, 10, 15, 'DA', 'Boeing 747');
```

```
INSERT INTO AVION (cod_avion, numar_locuri, inaltime, lungime, zbor_intercontinental, model)
VALUES (CHEIE.NEXTVAL, 100, 8, 12, 'NU', 'Airbus A320');
```

```
INSERT INTO AVION (cod_avion, numar_locuri, inaltime, lungime, zbor_intercontinental, model)
VALUES (CHEIE.NEXTVAL, 200, 12, 18, 'DA', 'Boeing 787');
```

```
INSERT INTO AVION (cod_avion, numar_locuri, inaltime, lungime, zbor_intercontinental, model)
VALUES (CHEIE.NEXTVAL, 80, 6, 10, 'NU', 'Embraer E190');
```

```
INSERT INTO AVION (cod_avion, numar_locuri, inaltime, lungime, zbor_intercontinental, model)
VALUES (CHEIE.NEXTVAL, 250, 15, 20, 'DA', 'Airbus A380');
```

```
INSERT INTO AVION (cod_avion, numar_locuri, inaltime, lungime, zbor_intercontinental, model)
VALUES (CHEIE.NEXTVAL, 120, 9, 14, 'NU', 'Boeing 737');
```

```
INSERT INTO AVION (cod_avion, numar_locuri, inaltime, lungime, zbor_intercontinental, model)
VALUES (CHEIE.NEXTVAL, 180, 11, 16, 'DA', 'Airbus A350');
```

```
INSERT INTO AVION (cod_avion, numar_locuri, inaltime, lungime, zbor_intercontinental, model)
VALUES (CHEIE.NEXTVAL, 90, 7, 11, 'NU', 'Embraer E195');
```

SQL Worksheet History

Search: chei

Replace: Replace Skip Replace All Replace Completed: 0 Skipped, 176 Replaced

Worksheet Query Builder

```

INSERT INTO AVION (cod_avion, numar_locuri, inaltime, lungime, zbor_intercontinental, model)
VALUES (CHEIE.NEXTVAL, 80, 6, 10, 'NU', 'Embraer E190');

INSERT INTO AVION (cod_avion, numar_locuri, inaltime, lungime, zbor_intercontinental, model)
VALUES (CHEIE.NEXTVAL, 250, 15, 20, 'DA', 'Airbus A380');

INSERT INTO AVION (cod_avion, numar_locuri, inaltime, lungime, zbor_intercontinental, model)
VALUES (CHEIE.NEXTVAL, 120, 9, 14, 'NU', 'Boeing 737');

INSERT INTO AVION (cod_avion, numar_locuri, inaltime, lungime, zbor_intercontinental, model)
VALUES (CHEIE.NEXTVAL, 180, 11, 16, 'DA', 'Airbus A350');

```

Script Output x Query Result x

SQL All Rows Fetched: 8 in 0.013 seconds

| | COD_AVION | NUMAR_LOCURI | INALTIME | LUNGIME | ZBOR_INTERCONTINENTAL | MODEL |
|---|-----------|--------------|----------|---------|-----------------------|-------|
| 1 | 980 | 150 | 10 | 15 DA | Boeing 747 | |
| 2 | 990 | 100 | 8 | 12 NU | Airbus A320 | |
| 3 | 1000 | 200 | 12 | 18 DA | Boeing 787 | |
| 4 | 1010 | 80 | 6 | 10 NU | Embraer E190 | |
| 5 | 1020 | 250 | 15 | 20 DA | Airbus A380 | |
| 6 | 1030 | 120 | 9 | 14 NU | Boeing 737 | |
| 7 | 1040 | 180 | 11 | 16 DA | Airbus A350 | |
| 8 | 1050 | 90 | 7 | 11 NU | Embraer E195 | |

–INSERTURI PENTRU ZBOR

```

INSERT INTO ZBOR(cod_zbor, data, ora_decolare, intarziere, cod_avion, cod_pista1, cod_pista2, cod_ruta)
VALUES (CHEIE.NEXTVAL, TO_DATE('2023-05-17', 'YYYY-MM-DD'), '08:00', 10, 980, 820, 780,
1470);

```

```

INSERT INTO ZBOR(cod_zbor, data, ora_decolare, intarziere, cod_avion, cod_pista1, cod_pista2, cod_ruta)
VALUES (CHEIE.NEXTVAL, TO_DATE('2023-05-18', 'YYYY-MM-DD'), '10:00', 0, 1000, 720, 840,
1480);

```

```

INSERT INTO ZBOR(cod_zbor, data, ora_decolare, intarziere, cod_avion, cod_pista1, cod_pista2, cod_ruta)
VALUES (CHEIE.NEXTVAL, TO_DATE('2023-05-19', 'YYYY-MM-DD'), '10:10', 0, 1030, 790, 870,
1490);

```

```

INSERT INTO ZBOR(cod_zbor, data, ora_decolare, intarziere, cod_avion, cod_pista1, cod_pista2, cod_ruta)
VALUES (CHEIE.NEXTVAL, TO_DATE('2023-05-19', 'YYYY-MM-DD'), '14:00', 45, 1040, 720, 850,
1500);

```



```
INSERT INTO ZBOR(cod_zbor, data, ora_decolare, intarziere, cod_avion, cod_pista1, cod_pista2, cod_ruta)
VALUES (CHEIE.NEXTVAL, TO_DATE('2023-05-21', 'YYYY-MM-DD'), '11:20', 10, 1040, 750, 820,
1510);
```

```
INSERT INTO ZBOR(cod_zbor, data, ora_decolare, intarziere, cod_avion, cod_pista1, cod_pista2, cod_ruta)
VALUES (CHEIE.NEXTVAL, TO_DATE('2023-05-22', 'YYYY-MM-DD'), '18:00', 0, 1020, 750, 910,
1520);
```

```
INSERT INTO ZBOR(cod_zbor, data, ora_decolare, intarziere, cod_avion, cod_pista1, cod_pista2, cod_ruta)
VALUES (CHEIE.NEXTVAL, TO_DATE('2023-05-23', 'YYYY-MM-DD'), '10:00', 0, 1010, 830, 750,
1530);
```

```
INSERT INTO ZBOR(cod_zbor, data, ora_decolare, intarziere, cod_avion, cod_pista1, cod_pista2, cod_ruta)
VALUES (CHEIE.NEXTVAL, TO_DATE('2023-05-24', 'YYYY-MM-DD'), '17:00', 0, 980, 910, 770,
1540);
```

```
INSERT INTO ZBOR(cod_zbor, data, ora_decolare, intarziere, cod_avion, cod_pista1, cod_pista2, cod_ruta)
VALUES (CHEIE.NEXTVAL, TO_DATE('2023-05-25', 'YYYY-MM-DD'), '09:00', 0, 840, 750, 810,
1550);
```

```
INSERT INTO ZBOR(cod_zbor, data, ora_decolare, intarziere, cod_avion, cod_pista1, cod_pista2, cod_ruta)
VALUES (CHEIE.NEXTVAL, TO_DATE('2023-05-26', 'YYYY-MM-DD'), '11:00', 0, 780, 880, 800,
1560);
```

SQL Worksheet | History

Search: **CHEIE** Replace Skip Replace All Replace Completed: 0 Skipped, 176 Replaced

Worksheet Query Builder

```

INSERT INTO ZBOR(cod_zbor, data, ora_decolare, intarziere, cod_avion, cod_pista1, cod_pista2, cod_ruta)
VALUES (CHEIE.NEXTVAL, TO_DATE('2023-05-24', 'YYYY-MM-DD'), '17:00', 0, 980, 910, 770, 1540);

INSERT INTO ZBOR(cod_zbor, data, ora_decolare, intarziere, cod_avion, cod_pista1, cod_pista2, cod_ruta)
VALUES (CHEIE.NEXTVAL, TO_DATE('2023-05-25', 'YYYY-MM-DD'), '08:00', 0, 840, 750, 810, 1550);

```

Script Output x Query Result x

SQL All Rows Fetched: 10 in 0.013 seconds

| | COD_ZBOR | DATA | ORA_DECOLARE | INTARZIERE | COD_AVION | COD_PISTA1 | COD_PISTA2 | COD_RUTA |
|----|----------|-----------|--------------|------------|-----------|------------|------------|----------|
| 1 | 1570 | 17-MAY-23 | 08:00 | 10 | 980 | 820 | 780 | 1470 |
| 2 | 1580 | 18-MAY-23 | 10:00 | 0 | 1000 | 720 | 840 | 1480 |
| 3 | 1590 | 19-MAY-23 | 10:10 | 0 | 1030 | 790 | 870 | 1490 |
| 4 | 1600 | 19-MAY-23 | 14:00 | 45 | 1040 | 720 | 850 | 1500 |
| 5 | 1610 | 21-MAY-23 | 11:20 | 10 | 1040 | 750 | 820 | 1510 |
| 6 | 1620 | 22-MAY-23 | 18:00 | 0 | 1020 | 750 | 910 | 1520 |
| 7 | 1630 | 23-MAY-23 | 10:00 | 0 | 1010 | 830 | 750 | 1530 |
| 8 | 1640 | 24-MAY-23 | 17:00 | 0 | 980 | 910 | 770 | 1540 |
| 9 | 1650 | 25-MAY-23 | 09:00 | 0 | 840 | 750 | 810 | 1550 |
| 10 | 1660 | 26-MAY-23 | 11:00 | 0 | 780 | 880 | 800 | 1560 |

–INSERTURI PENTRU REZERVARE

```

INSERT INTO REZERVARE(cod_rezervare, pret_total)
VALUES(CHEIE.NEXTVAL,515);

```

```

INSERT INTO REZERVARE(cod_rezervare, pret_total)
VALUES(CHEIE.NEXTVAL,1650);

```

```

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

```

```

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

```

```

INSERT INTO REZERVARE(cod_rezervare, pret_total)

```

VALUES(CHEIE.NEXTVAL,3120);

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

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

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

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

INSERT INTO REZERVARE(cod_rezervare, pret_total)
VALUES(CHEIE.NEXTVAL,1200);

The screenshot shows an SQL Worksheet application with a toolbar at the top. The main window is divided into two panes. The top pane, titled 'Worksheet', contains a SQL script with the following lines:

```
VALUES(CHEIE.NEXTVAL,1100);  
  
INSERT INTO REZERVARE(cod_rezervare, pret_total)  
  
VALUES(CHEIE.NEXTVAL,1100);  
  
INSERT INTO REZERVARE(cod_rezervare, pret_total)  
  
VALUES(CHEIE.NEXTVAL,1200);
```

The bottom pane, titled 'Query Result', shows the results of the script. It displays a table with two columns: 'COD_REZERVARE' and 'PRET_TOTAL'. The table contains 10 rows of data, numbered 1 through 10 in the first column. The status bar at the bottom indicates 'All Rows Fetched: 10 in 0.008 seconds'.

| | COD_REZERVARE | PRET_TOTAL |
|----|---------------|------------|
| 1 | 1670 | 515 |
| 2 | 1680 | 1650 |
| 3 | 1690 | 1255 |
| 4 | 1700 | 2420 |
| 5 | 1710 | 3120 |
| 6 | 1720 | 1115 |
| 7 | 1730 | 1010 |
| 8 | 1740 | 1100 |
| 9 | 1750 | 1100 |
| 10 | 1760 | 1200 |

-INSERTURI PENTRU LOC

```
INSERT INTO LOC(cod_loc, cod_avion, pozitie, numar_loc, cod_clasa)
VALUES(CHEIE.NEXTVAL, 990, 'Centru', 14, 930);
```

```
INSERT INTO LOC(cod_loc, cod_avion, pozitie, numar_loc, cod_clasa)
VALUES(CHEIE.NEXTVAL, 990, 'Geam', 18, 930);
```

```
INSERT INTO LOC(cod_loc, cod_avion, pozitie, numar_loc, cod_clasa)
VALUES(CHEIE.NEXTVAL, 1000, 'Margine', 14, 950);
```

```
INSERT INTO LOC(cod_loc, cod_avion, pozitie, numar_loc, cod_clasa)
VALUES(CHEIE.NEXTVAL, 1010, 'Central', 20, 950);
```

```
INSERT INTO LOC(cod_loc, cod_avion, pozitie, numar_loc, cod_clasa)
VALUES(CHEIE.NEXTVAL, 1020, 'Geam', 21, 960);
```

```
INSERT INTO LOC(cod_loc, cod_avion, pozitie, numar_loc, cod_clasa)
VALUES(CHEIE.NEXTVAL, 1030, 'Geam', 67, 970);
```

```
INSERT INTO LOC(cod_loc, cod_avion, pozitie, numar_loc, cod_clasa)
VALUES(CHEIE.NEXTVAL, 1040, 'Central', 114, 940);
```

```
INSERT INTO LOC(cod_loc, cod_avion, pozitie, numar_loc, cod_clasa)
VALUES(CHEIE.NEXTVAL, 1050, 'Geam', 22, 970);
```

```
INSERT INTO LOC(cod_loc, cod_avion, pozitie, numar_loc, cod_clasa)
VALUES(CHEIE.NEXTVAL, 1050, 'Margine', 22, 940);
```

```
INSERT INTO LOC(cod_loc, cod_avion, pozitie, numar_loc, cod_clasa)
VALUES(CHEIE.NEXTVAL, 1050, 'Margine', 12, 940);
```

```
INSERT INTO LOC(cod_loc, cod_avion, pozitie, numar_loc, cod_clasa)
VALUES(CHEIE.NEXTVAL, 1050, 'Geam', 35, 960);
```

```
INSERT INTO LOC(cod_loc, cod_avion, pozitie, numar_loc, cod_clasa)
VALUES(CHEIE.NEXTVAL, 1050, 'Geam', 1, 930);
```

```
INSERT INTO LOC(cod_loc, cod_avion, pozitie, numar_loc, cod_clasa)
VALUES(CHEIE.NEXTVAL, 1050, 'Central', 98, 960);
```

```
INSERT INTO LOC(cod_loc, cod_avion, pozitie, numar_loc, cod_clasa)
VALUES(CHEIE.NEXTVAL, 1030, 'Central', 2, 950);
```

```
INSERT INTO LOC(cod_loc, cod_avion, pozitie, numar_loc, cod_clasa)
VALUES(CHEIE.NEXTVAL, 1010, 'Margine', 26, 950);
```

SQL Worksheet History

Search: chei

Replace: Replace Skip Replace All Replace Completed: 0 Skipped, 176 Replaced

Worksheet Query Builder

```
VALUES(CHEIE.NEXTVAL, 1050, 'Central', 98, 960);
INSERT INTO LOC(cod_loc, cod_avion, pozitie, numar_loc, cod_clasa)
VALUES(CHEIE.NEXTVAL, 1030, 'Central', 2, 950);
INSERT INTO LOC(cod_loc, cod_avion, pozitie, numar_loc, cod_clasa)
```

Script Output x Query Result x

SQL All Rows Fetched: 15 in 0.009 seconds

| | COD_LOC | COD_AVION | POZITIE | NUMAR_LOC | COD_CLASA |
|----|---------|-----------|---------|-----------|-----------|
| 1 | 1060 | 990 | Centru | 14 | 930 |
| 2 | 1070 | 990 | Geam | 18 | 930 |
| 3 | 1080 | 1000 | Margine | 14 | 950 |
| 4 | 1090 | 1010 | Central | 20 | 950 |
| 5 | 1100 | 1020 | Geam | 21 | 960 |
| 6 | 1110 | 1030 | Geam | 67 | 970 |
| 7 | 1120 | 1040 | Central | 114 | 940 |
| 8 | 1130 | 1050 | Geam | 22 | 970 |
| 9 | 1140 | 1050 | Margine | 22 | 940 |
| 10 | 1150 | 1050 | Margine | 12 | 940 |
| 11 | 1160 | 1050 | Geam | 35 | 960 |
| 12 | 1170 | 1050 | Geam | 1 | 930 |
| 13 | 1180 | 1050 | Central | 98 | 960 |
| 14 | 1190 | 1030 | Central | 2 | 950 |
| 15 | 1200 | 1010 | Margine | 26 | 950 |

–INSERTURI PENTRU CLASA

```
INSERT INTO CLASA(cod_clasa, pret)
VALUES (chei.NEXTVAL, 500);
```

```
INSERT INTO CLASA(cod_clasa, pret)
VALUES (chei.NEXTVAL, 800);
```

```
INSERT INTO CLASA(cod_clasa, pret)
VALUES (chei.NEXTVAL, 1200);
```

```
INSERT INTO CLASA(cod_clasa, pret)
VALUES (chei.NEXTVAL, 1100);
```

```
INSERT INTO CLASA(cod_clasa, pret)
VALUES (chei.NEXTVAL, 1000);
```

The screenshot shows an SQL Worksheet interface. At the top, there's a search bar with 'chei' and a replace bar with 'CHEIE'. Below the replace bar, the 'Worksheet' tab is active, showing a SQL script with four INSERT statements. The 'Query Result' tab is also visible, showing a table with 5 rows and 2 columns: COD_CLASA and PRET. The table contains the following data:

| | COD_CLASA | PRET |
|---|-----------|------|
| 1 | 930 | 500 |
| 2 | 940 | 800 |
| 3 | 950 | 1200 |
| 4 | 960 | 1100 |
| 5 | 970 | 1000 |

–INSERTURI PENTRU FACILITATE

```
INSERT INTO FACILITATE(cod_facilitate, nume_facilitate, pret_suplimentar, standard)
VALUES(chei.NEXTVAL, 'WiFi', 10, 'DA');
```

```
INSERT INTO FACILITATE(cod_facilitate, nume_facilitate, pret_suplimentar, standard)
VALUES(chei.NEXTVAL, 'Masa calda', 15, 'DA');
```

```
INSERT INTO FACILITATE(cod_facilitate, nume_facilitate, pret_suplimentar, standard)
VALUES(chei.NEXTVAL, 'Divertisment la bord', 5, 'DA');
```

```
INSERT INTO FACILITATE(cod_facilitate, nume_facilitate, pret_suplimentar, standard)
VALUES(chei.NEXTVAL, 'Prioritate la Imbarcare', 8, 'NU');
```

```
INSERT INTO FACILITATE(cod_facilitate, nume_facilitate, pret_suplimentar, standard)
VALUES(chei.NEXTVAL, 'Bagaj de cala', 12, 'NU');
```

```
INSERT INTO FACILITATE(cod_facilitate, nume_facilitate, pret_suplimentar, standard)
VALUES(chei.NEXTVAL, 'Acces la salonul VIP', 20, 'DA');
```

The screenshot displays an SQL Worksheet interface. The top section shows a search bar with 'chei' and a replace bar with 'CHEIE'. Below this, the 'Worksheet' tab is active, showing six INSERT statements. The 'Query Result' tab is also visible, showing a table with 6 rows and 4 columns: COD_FACILITATE, NUME_FACILITATE, PRET_SUPLIMENTAR, and STANDARD. The table contains the following data:

| | COD_FACILITATE | NUME_FACILITATE | PRET_SUPLIMENTAR | STANDARD |
|---|----------------|-------------------------|------------------|----------|
| 1 | 1210 | WiFi | 10 | DA |
| 2 | 1220 | Masa calda | 15 | DA |
| 3 | 1230 | Divertisment la bord | 5 | DA |
| 4 | 1240 | Prioritate la Imbarcare | 8 | NU |
| 5 | 1250 | Bagaj de cala | 12 | NU |
| 6 | 1260 | Acces la salonul VIP | 20 | DA |

-INSERTURI PENTRU PASAGER

```
INSERT INTO PASAGER(cod_pasager, nume, prenume, nationalitate, numar_pasaport)
VALUES(chei.NEXTVAL, 'Popescu', 'Ion', 'Română', 12345678);
```

```
INSERT INTO PASAGER(cod_pasager, nume, prenume, nationalitate, numar_pasaport)
VALUES(chei.NEXTVAL, 'Ionescu', 'Maria', 'Română', 87654321);
```

```
INSERT INTO PASAGER(cod_pasager, nume, prenume, nationalitate, numar_pasaport)
VALUES(chei.NEXTVAL, 'Smith', 'John', 'Engleză', 98765432);
```

```
INSERT INTO PASAGER(cod_pasager, nume, prenume, nationalitate, numar_pasaport)
VALUES(chei.NEXTVAL, 'Müller', 'Hans', 'Germană', 54321678);
```

```
INSERT INTO PASAGER(cod_pasager, nume, prenume, nationalitate, numar_pasaport)
VALUES(chei.NEXTVAL, 'Rossi', 'Giovanni', 'Italiană', 87654321);
```

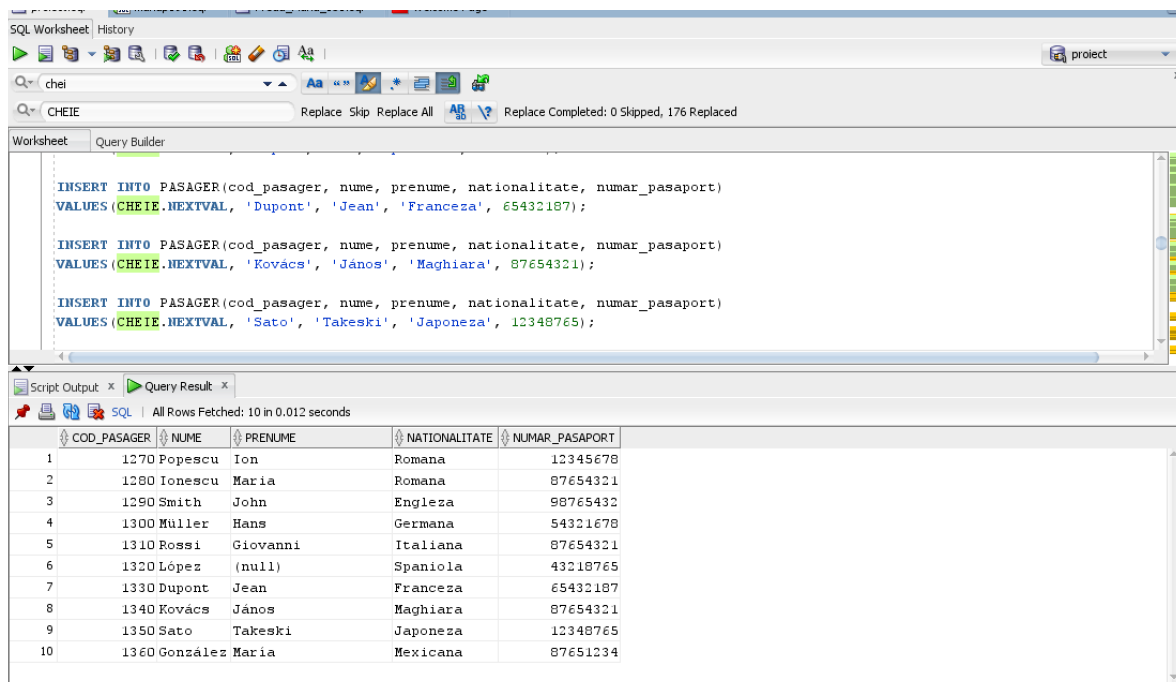
```
INSERT INTO PASAGER(cod_pasager, nume, prenume, nationalitate, numar_pasaport)
VALUES(chei.NEXTVAL, 'López', null, 'Spaniolă', 43218765);
```

```
INSERT INTO PASAGER(cod_pasager, nume, prenume, nationalitate, numar_pasaport)
VALUES(chei.NEXTVAL, 'Dupont', 'Jean', 'Franceză', 65432187);
```

```
INSERT INTO PASAGER(cod_pasager, nume, prenume, nationalitate, numar_pasaport)
VALUES(chei.NEXTVAL, 'Kovács', 'János', 'Maghiară', 87654321);
```

```
INSERT INTO PASAGER(cod_pasager, nume, prenume, nationalitate, numar_pasaport)
VALUES(chei.NEXTVAL, 'Sato', 'Takeshi', 'Japoneză', 12348765);
```

```
INSERT INTO PASAGER(cod_pasager, nume, prenume, nationalitate, numar_pasaport)
VALUES(chei.NEXTVAL, 'González', 'María', 'Mexicană', 87651234);
```

–INSERTURI PENTRU PILOT

```

INSERT INTO PILOT(cod_angajat, educatie, experienta)
VALUES(1370, 'Universitatea Politehnica Bucuresti', 5);

```

```

INSERT INTO PILOT(cod_angajat, educatie, experienta)
VALUES(1400, 'ETH Zurich', 8);

```

```

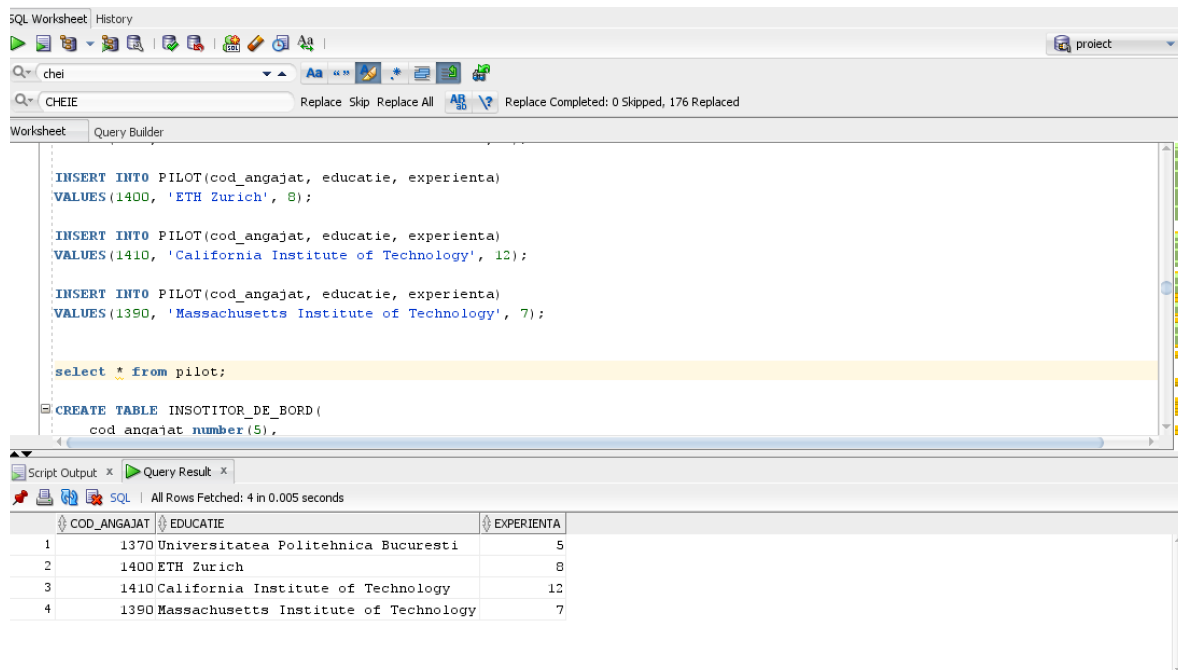
INSERT INTO PILOT(cod_angajat, educatie, experienta)
VALUES(1410, 'California Institute of Technology', 12);

```

```

INSERT INTO PILOT(cod_angajat, educatie, experienta)
VALUES(1390, 'Massachusetts Institute of Technology', 7);

```



–INSERTURI PENTRU ANGAJAT

```

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

```

```

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

```

```

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

```

```

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

```

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

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

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

```
INSERT INTO ANGAJAT(cod_angajat, nume, prenume, data_nasterii, data_angajarii, nationalitate)
VALUES(CHEIE.NEXTVAL, 'Kim', 'Min-ji', TO_DATE('29-07-1993', 'dd-mm-yyyy'), TO_DATE('12-
08-2020', 'dd-mm-yyyy'), 'Coreeana');
```

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

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

SQL Worksheet History

Search: chei

Replace: Replace All

Replace Completed: 0 Skipped, 176 Replaced

Worksheet Query Builder

```

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

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

```

Script Output x Query Result x

SQL | All Rows Fetched: 10 in 0.005 seconds

| | COD_ANGAJAT | NUME | PRENUME | DATA_NASTERII | DATA_ANGAJARII | NATIONALITATE |
|----|-------------|----------|---------|---------------|----------------|---------------|
| 1 | 1370 | Popescu | Ion | 01-JAN-90 | 01-JAN-20 | Romana |
| 2 | 1380 | Ionescu | Andrei | 15-FEB-85 | 01-JUL-15 | Romana |
| 3 | 1390 | Popa | Maria | 10-MAY-92 | 15-MAR-21 | Italiana |
| 4 | 1400 | Muller | Hans | 20-JUN-88 | 10-SEP-17 | Germana |
| 5 | 1410 | Smith | John | 03-DEC-95 | 25-MAY-19 | Engleza |
| 6 | 1420 | Lopez | Maria | 08-SEP-91 | 18-NOV-18 | Spaniola |
| 7 | 1430 | Dupont | Jean | 12-APR-87 | 05-FEB-16 | Franceza |
| 8 | 1440 | Kim | Min-ji | 29-JUL-93 | 12-AUG-20 | Coreeana |
| 9 | 1450 | Muhammad | Ahmed | 17-NOV-94 | 30-JUL-19 | Pachistaneza |
| 10 | 1460 | James | Anna | 17-OCT-94 | 20-JUL-19 | Engleza |

–INSERTURI PENTRU INSOTITOR_DE_BORD

```

INSERT INTO INSOTITOR_DE_BORD(cod_angajat, experienta, limba_straina)
VALUES(1440, 3, 'Engleza');

```

```

INSERT INTO INSOTITOR_DE_BORD(cod_angajat, experienta, limba_straina)
VALUES(1450, 5, 'Franceza');

```

```

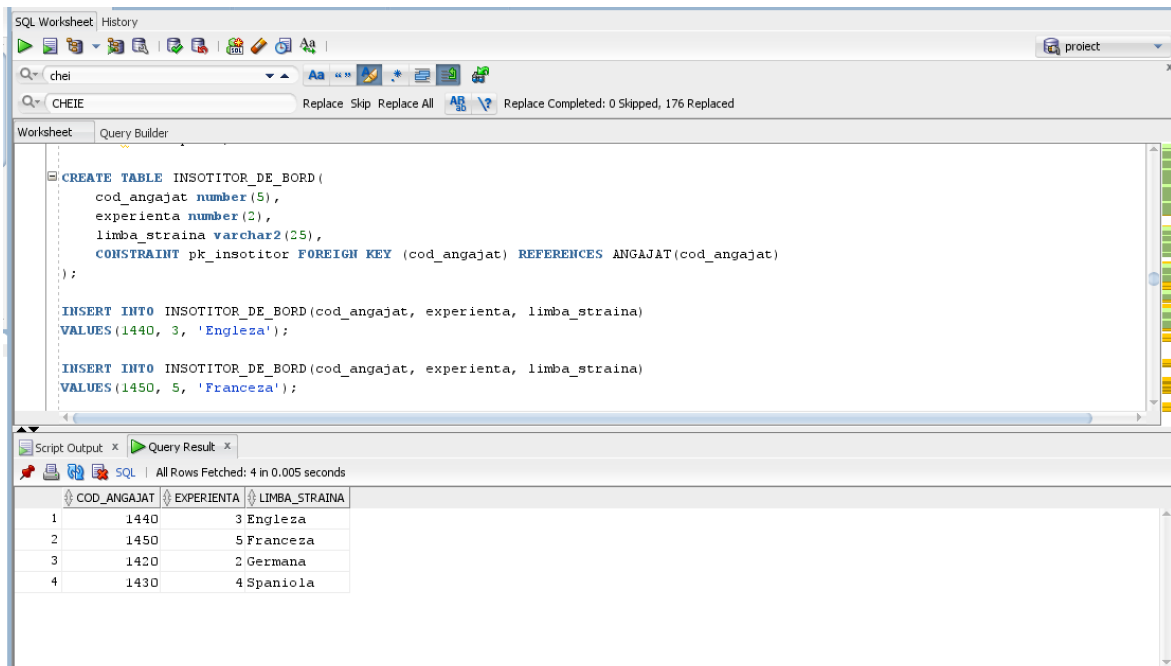
INSERT INTO INSOTITOR_DE_BORD(cod_angajat, experienta, limba_straina)
VALUES(1420, 2, 'Germana');

```

```

INSERT INTO INSOTITOR_DE_BORD(cod_angajat, experienta, limba_straina)
VALUES(1430, 4, 'Spaniola');

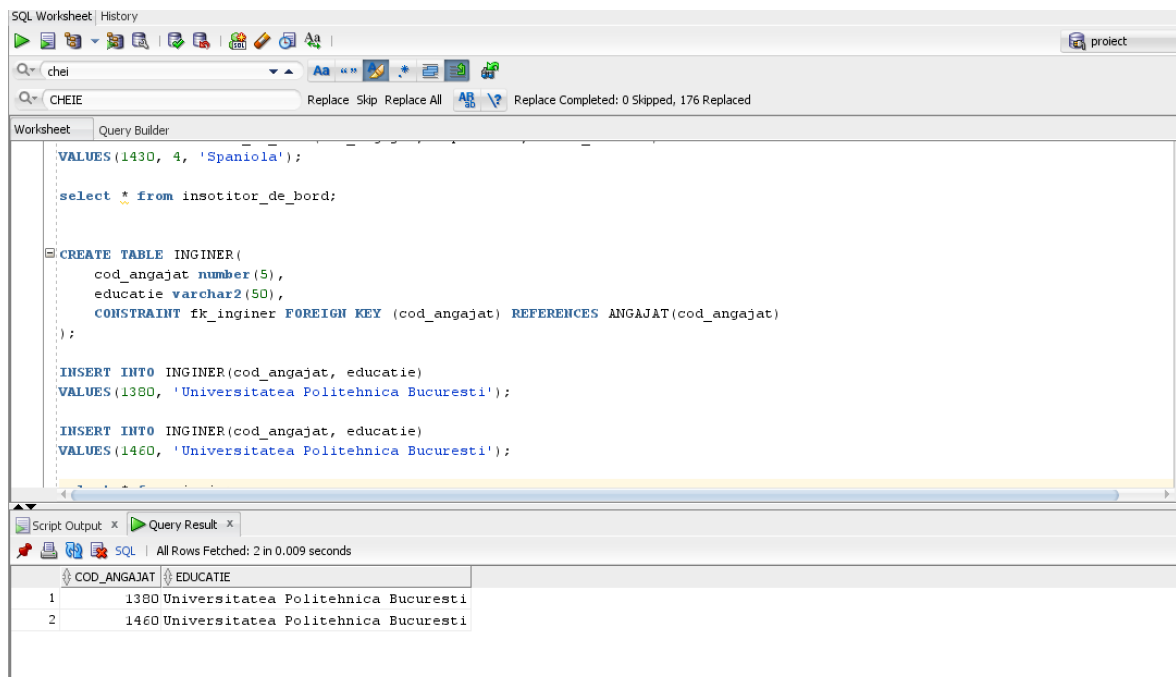
```



–INSERTURI PENTRU INGINER

INSERT INTO INGINER(cod_angajat, educatie)
VALUES(1380, 'Universitatea Politehnica Bucuresti');

INSERT INTO INGINER(cod_angajat, educatie)
VALUES(1460, 'Universitatea Politehnica Bucuresti');



–INSERTURI PENTRU ARE

INSERT INTO ARE(cod_aeroport1, cod_aeroport2, cod_ruta)
VALUES(140,210,1480);

INSERT INTO ARE(cod_aeroport1, cod_aeroport2, cod_ruta)
VALUES(170,190,1470);

INSERT INTO ARE(cod_aeroport1, cod_aeroport2, cod_ruta)
VALUES(170,230,1490);

INSERT INTO ARE(cod_aeroport1, cod_aeroport2, cod_ruta)
VALUES(140,220,1500);

```
INSERT INTO ARE(cod_aeroporto1, cod_aeroporto2, cod_ruta)
VALUES(150,190,1510);
```

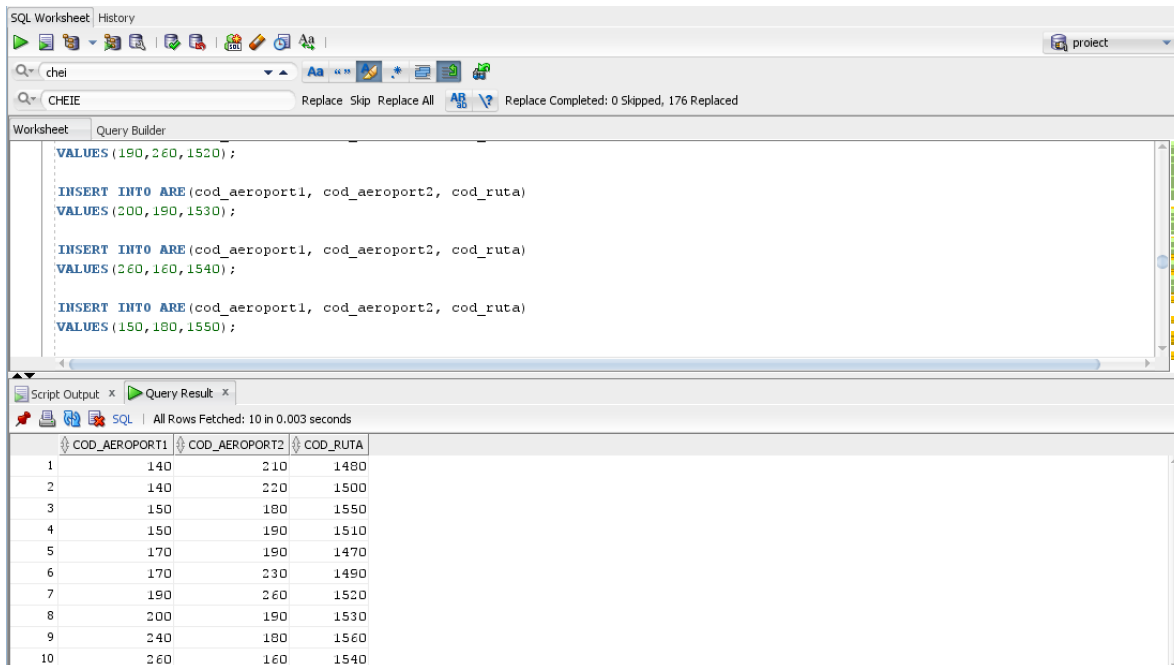
```
INSERT INTO ARE(cod_aeroporto1, cod_aeroporto2, cod_ruta)
VALUES(190,260,1520);
```

```
INSERT INTO ARE(cod_aeroporto1, cod_aeroporto2, cod_ruta)
VALUES(200,190,1530);
```

```
INSERT INTO ARE(cod_aeroporto1, cod_aeroporto2, cod_ruta)
VALUES(260,160,1540);
```

```
INSERT INTO ARE(cod_aeroporto1, cod_aeroporto2, cod_ruta)
VALUES(150,180,1550);
```

```
INSERT INTO ARE(cod_aeroporto1, cod_aeroporto2, cod_ruta)
VALUES(240,180,1560);
```



The screenshot shows an SQL Worksheet interface with a toolbar at the top containing icons for running queries, saving, and other database operations. Below the toolbar, there is a search bar with the text 'chei' and a replace section with 'CHEIE' and a status 'Replace Completed: 0 Skipped, 176 Replaced'. The main area is divided into 'Worksheet' and 'Query Builder' tabs. The 'Worksheet' tab is active, displaying a SQL script with several INSERT statements. Below the script, there is a 'Script Output' and 'Query Result' section. The 'Query Result' section shows a table with 10 rows and 3 columns: COD_AEROPORT1, COD_AEROPORT2, and COD_RUTA. The table contains the following data:

| | COD_AEROPORT1 | COD_AEROPORT2 | COD_RUTA |
|----|---------------|---------------|----------|
| 1 | 140 | 210 | 1480 |
| 2 | 140 | 220 | 1500 |
| 3 | 150 | 180 | 1550 |
| 4 | 150 | 190 | 1510 |
| 5 | 170 | 190 | 1470 |
| 6 | 170 | 230 | 1490 |
| 7 | 190 | 260 | 1520 |
| 8 | 200 | 190 | 1530 |
| 9 | 240 | 180 | 1560 |
| 10 | 260 | 160 | 1540 |

–INSERTURI PENTRU CUMPARA

```
INSERT INTO CUMPARA(cod_rezervare, cod_pasager, numar_locuri)
VALUES(1670, 1270, 1);
```

```
INSERT INTO CUMPARA(cod_rezervare, cod_pasager, numar_locuri)
VALUES(1680, 1280, 2);
```

```
INSERT INTO CUMPARA(cod_rezervare, cod_pasager, numar_locuri)
VALUES(1690, 1290, 1);
```

```
INSERT INTO CUMPARA(cod_rezervare, cod_pasager, numar_locuri)
VALUES(1700, 1300, 2);
```

```
INSERT INTO CUMPARA(cod_rezervare, cod_pasager, numar_locuri)
VALUES(1710, 1310, 3);
```

```
INSERT INTO CUMPARA(cod_rezervare, cod_pasager, numar_locuri)
VALUES(1720, 1320, 1);
```

```
INSERT INTO CUMPARA(cod_rezervare, cod_pasager, numar_locuri)
VALUES(1730, 1330, 1);
```

```
INSERT INTO CUMPARA(cod_rezervare, cod_pasager, numar_locuri)
VALUES(1740, 1340, 1);
```

```
INSERT INTO CUMPARA(cod_rezervare, cod_pasager, numar_locuri)
VALUES(1750, 1340, 1);
```

```
INSERT INTO CUMPARA(cod_rezervare, cod_pasager, numar_locuri)
VALUES(1760, 1340, 1);
```


SQL Worksheet | History

Search: chei

Replace: CHEIE | Replace | Skip | Replace All | Replace Completed: 0 Skipped, 176 Replaced

Worksheet | Query Builder

```

INSERT INTO CUMPARA(cod_rezervare, cod_pasager, numar_locuri)
VALUES (1720, 1320, 1);

INSERT INTO CUMPARA(cod_rezervare, cod_pasager, numar_locuri)
VALUES (1730, 1330, 1);

INSERT INTO CUMPARA(cod_rezervare, cod_pasager, numar_locuri)
VALUES (1740, 1340, 1);

INSERT INTO CUMPARA(cod_rezervare, cod_pasager, numar_locuri)
VALUES (1750, 1340, 1);

```

Script Output | Query Result | All Rows Fetched: 10 in 0.005 seconds

| | COD_REZERVARE | COD_PASAGER | NUMAR_LOCURI |
|----|---------------|-------------|--------------|
| 1 | 1670 | 1270 | 1 |
| 2 | 1680 | 1280 | 2 |
| 3 | 1690 | 1290 | 1 |
| 4 | 1700 | 1300 | 2 |
| 5 | 1710 | 1310 | 3 |
| 6 | 1720 | 1320 | 1 |
| 7 | 1730 | 1330 | 1 |
| 8 | 1740 | 1340 | 1 |
| 9 | 1750 | 1340 | 1 |
| 10 | 1760 | 1340 | 1 |

–INSERTURI PENTRU OFERA

```

INSERT INTO OFERA(cod_clasa, cod_facilitate)
VALUES(950,1210);

```

```

INSERT INTO OFERA(cod_clasa, cod_facilitate)
VALUES(970,1210);

```

```

INSERT INTO OFERA(cod_clasa, cod_facilitate)
VALUES(930,1220);

```

```

INSERT INTO OFERA(cod_clasa, cod_facilitate)
VALUES(960,1220);

```

```

INSERT INTO OFERA(cod_clasa, cod_facilitate)
VALUES(950,1230);

```

```
INSERT INTO OFERA(cod_clasa, cod_facilitate)
VALUES(950,1240);
```

```
INSERT INTO OFERA(cod_clasa, cod_facilitate)
VALUES(970,1250);
```

```
INSERT INTO OFERA(cod_clasa, cod_facilitate)
VALUES(960,1250);
```

```
INSERT INTO OFERA(cod_clasa, cod_facilitate)
VALUES(950,1250);
```

```
INSERT INTO OFERA(cod_clasa, cod_facilitate)
VALUES(950,1260);
```

```
INSERT INTO OFERA(cod_clasa, cod_facilitate)
VALUES(940,1220);
```

```
INSERT INTO OFERA(cod_clasa, cod_facilitate)
VALUES(940,1210);
```

SQL Worksheet | History

Search: chei | Replace | Skip | Replace All | Replace Completed: 0 Skipped, 176 Replaced

Worksheet | Query Builder

```
VALUES (940, 1220);

INSERT INTO OFERA (cod_clasa, cod_facilitate)
VALUES (940, 1210);

select * from ofera;

CREATE TABLE PASAGER (
```

Script Output | Query Result | All Rows Fetched: 12 in 0.008 seconds

| | COD_CLASA | COD_FACILITATE |
|----|-----------|----------------|
| 1 | 930 | 1220 |
| 2 | 940 | 1210 |
| 3 | 940 | 1220 |
| 4 | 950 | 1210 |
| 5 | 950 | 1230 |
| 6 | 950 | 1240 |
| 7 | 950 | 1250 |
| 8 | 950 | 1260 |
| 9 | 960 | 1220 |
| 10 | 960 | 1250 |
| 11 | 970 | 1210 |
| 12 | 970 | 1250 |

–INSERTURI PENTRU LUCREAZA

```
INSERT INTO LUCREAZA(cod_angajat, cod_zbor, ore, spor)
VALUES(1370, 1570, 5, 230);
```

```
INSERT INTO LUCREAZA(cod_angajat, cod_zbor, ore, spor)
VALUES(1400, 1580, 9, 370);
```

```
INSERT INTO LUCREAZA(cod_angajat, cod_zbor, ore, spor)
VALUES(1400, 1590, 4, 200);
```

```
INSERT INTO LUCREAZA(cod_angajat, cod_zbor, ore, spor)
VALUES(1400, 1600, 8, 325);
```

```
INSERT INTO LUCREAZA(cod_angajat, cod_zbor, ore, spor)
VALUES(1370, 1610, 12, 430);
```

```
INSERT INTO LUCREAZA(cod_angajat, cod_zbor, ore, spor)
VALUES(1410, 1620, 3, 100);
```

```
INSERT INTO LUCREAZA(cod_angajat, cod_zbor, ore, spor)
VALUES(1410, 1630, 9, 350);
```

```
INSERT INTO LUCREAZA(cod_angajat, cod_zbor, ore, spor)
VALUES(1390, 1640, 8, 300);
```

```
INSERT INTO LUCREAZA(cod_angajat, cod_zbor, ore, spor)
VALUES(1390, 1650, 7, 200);
```

```
INSERT INTO LUCREAZA(cod_angajat, cod_zbor, ore, spor)
VALUES(1390, 1660, 5, 100);
```

```
INSERT INTO LUCREAZA(cod_angajat, cod_zbor, ore, spor)
VALUES(1440, 1570, 5, 170);
```

```
INSERT INTO LUCREAZA(cod_angajat, cod_zbor, ore, spor)
VALUES(1440, 1580, 9, 240);
```

```
INSERT INTO LUCREAZA(cod_angajat, cod_zbor, ore, spor)
VALUES(1450, 1590, 4, 120);
```

```
INSERT INTO LUCREAZA(cod_angajat, cod_zbor, ore, spor)
VALUES(1450, 1600, 8, 160);
```

```
INSERT INTO LUCREAZA(cod_angajat, cod_zbor, ore, spor)
VALUES(1450, 1610, 12, 400);
```

```
INSERT INTO LUCREAZA(cod_angajat, cod_zbor, ore, spor)
VALUES(1420, 1620, 3, 70);
```

```
INSERT INTO LUCREAZA(cod_angajat, cod_zbor, ore, spor)
VALUES(1420, 1630, 9, 220);
```

```
INSERT INTO LUCREAZA(cod_angajat, cod_zbor, ore, spor)
VALUES(1440, 1640, 8, 100);
```

```
INSERT INTO LUCREAZA(cod_angajat, cod_zbor, ore, spor)
VALUES(1430, 1650, 7, 150);
```

```
INSERT INTO LUCREAZA(cod_angajat, cod_zbor, ore, spor)
VALUES(1430, 1660, 5, 200);
```

```
INSERT INTO LUCREAZA(cod_angajat, cod_zbor, ore, spor)
VALUES(1380, 1570, 1, 15);
```

```
INSERT INTO LUCREAZA(cod_angajat, cod_zbor, ore, spor)
VALUES(1380, 1580, 2, null);
```

```
INSERT INTO LUCREAZA(cod_angajat, cod_zbor, ore, spor)
VALUES(1380, 1590, 1, 14);
```

```
INSERT INTO LUCREAZA(cod_angajat, cod_zbor, ore, spor)
VALUES(1460, 1600, 2, null);
```

```
INSERT INTO LUCREAZA(cod_angajat, cod_zbor, ore, spor)
VALUES(1460, 1610, 2, null);
```

```
INSERT INTO LUCREAZA(cod_angajat, cod_zbor, ore, spor)
VALUES(1380, 1620, 1, null);
```

```
INSERT INTO LUCREAZA(cod_angajat, cod_zbor, ore, spor)
VALUES(1460, 1630, 2, 35);
```

```
INSERT INTO LUCREAZA(cod_angajat, cod_zbor, ore, spor)
VALUES(1460, 1640, 2, null);
```

```
INSERT INTO LUCREAZA(cod_angajat, cod_zbor, ore, spor)
VALUES(1460, 1650, 2, 40);
```

```
INSERT INTO LUCREAZA(cod_angajat, cod_zbor, ore, spor)
VALUES(1460, 1660, 1, null);
```

SQL Worksheet History

Q= chei

Q= CHEIE Replace Skip Replace All Replace Completed: 0 Skipped, 176 Replaced

Worksheet Query Builder

Script Output x Query Result x

SQL | All Rows Fetched: 30 in 0.003 seconds

| | COD_ANGAJAT | COD_ZBOR | ORE | SPOR |
|----|-------------|----------|-----|------|
| 1 | 1370 | 1570 | 5 | 230 |
| 2 | 1400 | 1580 | 9 | 370 |
| 3 | 1400 | 1590 | 4 | 200 |
| 4 | 1400 | 1600 | 8 | 325 |
| 5 | 1370 | 1610 | 12 | 430 |
| 6 | 1410 | 1620 | 3 | 100 |
| 7 | 1410 | 1630 | 9 | 350 |
| 8 | 1390 | 1640 | 8 | 300 |
| 9 | 1390 | 1650 | 7 | 200 |
| 10 | 1390 | 1660 | 5 | 100 |
| 11 | 1440 | 1570 | 5 | 170 |
| 12 | 1440 | 1580 | 9 | 240 |
| 13 | 1450 | 1590 | 4 | 120 |
| 14 | 1450 | 1600 | 8 | 160 |
| 15 | 1450 | 1610 | 12 | 400 |
| 16 | 1420 | 1620 | 3 | 70 |
| 17 | 1420 | 1630 | 9 | 220 |
| 18 | 1440 | 1640 | 8 | 100 |
| 19 | 1430 | 1650 | 7 | 150 |
| 20 | 1430 | 1660 | 5 | 200 |

SQL Worksheet | History

Q: chei

Q: CHEIE

Replace Skip Replace All

Replace Completed: 0 Skipped, 176 Replaced

Worksheet Query Builder

Script Output x Query Result x

SQL | All Rows Fetched: 30 in 0.003 seconds

| | COD_ANGAJAT | COD_ZBOR | ORE | SPOR |
|----|-------------|----------|-----|--------|
| 11 | 1440 | 1570 | 5 | 170 |
| 12 | 1440 | 1580 | 9 | 240 |
| 13 | 1450 | 1590 | 4 | 120 |
| 14 | 1450 | 1600 | 8 | 160 |
| 15 | 1450 | 1610 | 12 | 400 |
| 16 | 1420 | 1620 | 3 | 70 |
| 17 | 1420 | 1630 | 9 | 220 |
| 18 | 1440 | 1640 | 8 | 100 |
| 19 | 1430 | 1650 | 7 | 150 |
| 20 | 1430 | 1660 | 5 | 200 |
| 21 | 1380 | 1570 | 1 | 15 |
| 22 | 1380 | 1580 | 2 | (null) |
| 23 | 1380 | 1590 | 1 | 14 |
| 24 | 1460 | 1600 | 2 | (null) |
| 25 | 1460 | 1610 | 2 | (null) |
| 26 | 1380 | 1620 | 1 | (null) |
| 27 | 1460 | 1630 | 2 | 35 |
| 28 | 1460 | 1640 | 2 | (null) |
| 29 | 1460 | 1650 | 2 | 40 |
| 30 | 1460 | 1660 | 1 | (null) |

–INSERTURI PENTRU CONECTATA

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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



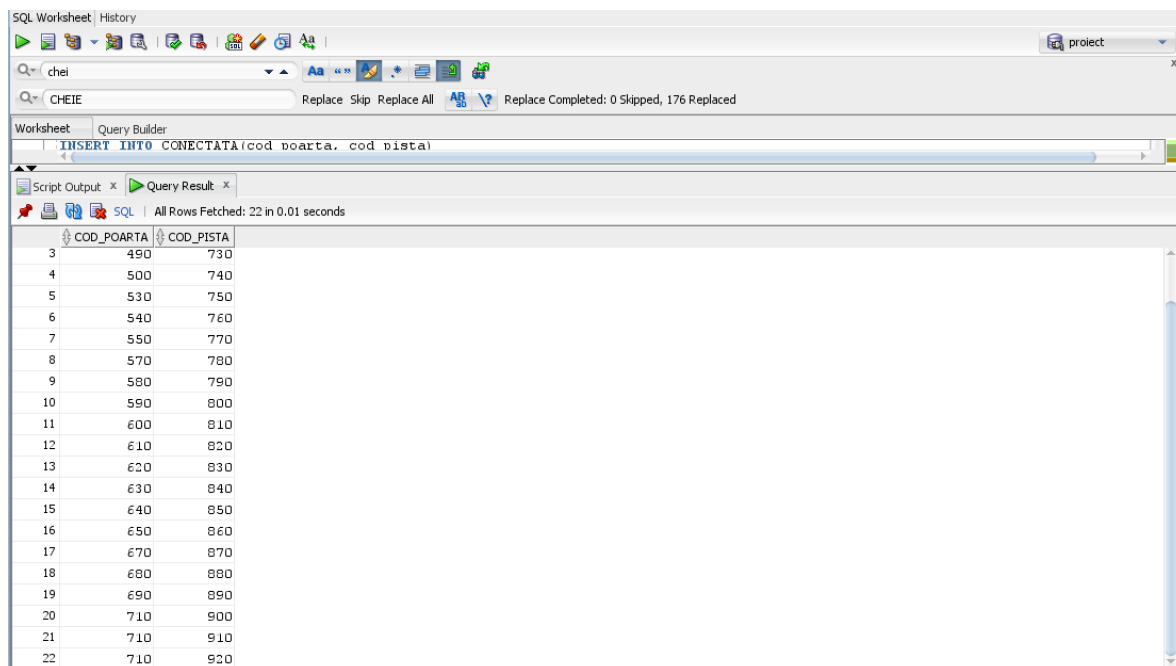
```
INSERT INTO CONECTATA(cod_poarta, cod_pista)
VALUES(680,880);
```

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

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

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

```
INSERT INTO CONECTATA(cod_poarta, cod_pista)
VALUES(710,920);
```



SQL Worksheet | History

Q: chei

Q: CHEIE

Replace Skip Replace All

Replace Completed: 0 Skipped, 176 Replaced

Worksheet | Query Builder

INSERT INTO CONECTATA(cod_poarta, cod_pista)

Script Output | Query Result

SQL | All Rows Fetched: 22 in 0.01 seconds

| | COD_POARTA | COD_PISTA |
|----|------------|-----------|
| 3 | 490 | 730 |
| 4 | 500 | 740 |
| 5 | 530 | 750 |
| 6 | 540 | 760 |
| 7 | 550 | 770 |
| 8 | 570 | 780 |
| 9 | 580 | 790 |
| 10 | 590 | 800 |
| 11 | 600 | 810 |
| 12 | 610 | 820 |
| 13 | 620 | 830 |
| 14 | 630 | 840 |
| 15 | 640 | 850 |
| 16 | 650 | 860 |
| 17 | 670 | 870 |
| 18 | 680 | 880 |
| 19 | 690 | 890 |
| 20 | 710 | 900 |
| 21 | 710 | 910 |
| 22 | 710 | 920 |

–INSERTURI PENTRU REZERVA

```
INSERT INTO REZERVA(cod_rezervare, cod_loc, cod_zbor, data)
VALUES (1750, 1060, 1570, TO_DATE('2023-05-17', 'YYYY-MM-DD'));
```

```
INSERT INTO REZERVA(cod_rezervare, cod_loc, cod_zbor, data)
VALUES (1760, 1060, 1580, TO_DATE('2023-05-18', 'YYYY-MM-DD'));
```

```
INSERT INTO REZERVA(cod_rezervare, cod_loc, cod_zbor, data)
VALUES (1670, 1090, 1600, TO_DATE('2023-04-15', 'YYYY-MM-DD'));
```

```
INSERT INTO REZERVA(cod_rezervare, cod_loc, cod_zbor, data)
VALUES (1680, 1100, 1610, TO_DATE('2023-03-20', 'YYYY-MM-DD'));
```

```
INSERT INTO REZERVA(cod_rezervare, cod_loc, cod_zbor, data)
VALUES (1680, 1110, 1610, TO_DATE('2023-03-20', 'YYYY-MM-DD'));
```

```
INSERT INTO REZERVA(cod_rezervare, cod_loc, cod_zbor, data)
VALUES (1690, 1150, 1620, TO_DATE('2022-10-10', 'YYYY-MM-DD'));
```

```
INSERT INTO REZERVA(cod_rezervare, cod_loc, cod_zbor, data)
VALUES (1700, 1160, 1630, TO_DATE('2022-09-15', 'YYYY-MM-DD'));
```

```
INSERT INTO REZERVA(cod_rezervare, cod_loc, cod_zbor, data)
VALUES (1700, 1150, 1640, TO_DATE('2022-09-15', 'YYYY-MM-DD'));
```

```
INSERT INTO REZERVA(cod_rezervare, cod_loc, cod_zbor, data)
VALUES (1710, 1190, 1650, TO_DATE('2021-11-15', 'YYYY-MM-DD'));
```

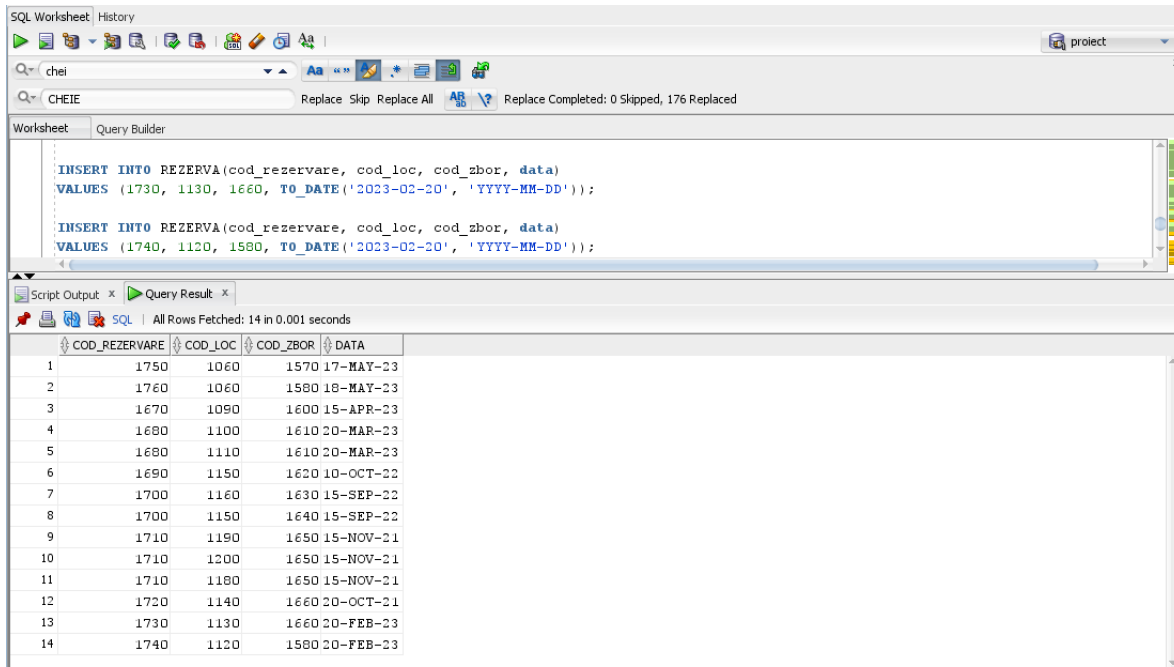
```
INSERT INTO REZERVA(cod_rezervare, cod_loc, cod_zbor, data)
VALUES (1710, 1200, 1650, TO_DATE('2021-11-15', 'YYYY-MM-DD'));
```

```
INSERT INTO REZERVA(cod_rezervare, cod_loc, cod_zbor, data)
VALUES (1710, 1180, 1650, TO_DATE('2021-11-15', 'YYYY-MM-DD'));
```

```
INSERT INTO REZERVA(cod_rezervare, cod_loc, cod_zbor, data)
VALUES (1720, 1140, 1660, TO_DATE('2021-10-20', 'YYYY-MM-DD'));
```

```
INSERT INTO REZERVA(cod_rezervare, cod_loc, cod_zbor, data)
VALUES (1730, 1130, 1660, TO_DATE('2023-02-20', 'YYYY-MM-DD'));
```

```
INSERT INTO REZERVA(cod_rezervare, cod_loc, cod_zbor, data)
VALUES (1740, 1120, 1580, TO_DATE('2023-02-20', 'YYYY-MM-DD'));
```



The screenshot shows an SQL Worksheet interface. The top section contains two SQL statements:


```
INSERT INTO REZERVA(cod_rezervare, cod_loc, cod_zbor, data)
VALUES (1730, 1130, 1660, TO_DATE('2023-02-20', 'YYYY-MM-DD'));

INSERT INTO REZERVA(cod_rezervare, cod_loc, cod_zbor, data)
VALUES (1740, 1120, 1580, TO_DATE('2023-02-20', 'YYYY-MM-DD'));
```

 Below the statements, the 'Query Result' tab is active, displaying a table with 14 rows. The table has four columns: COD_REZERVARE, COD_LOC, COD_ZBOR, and DATA. The data represents a list of reservations with their respective codes and dates.

| | COD_REZERVARE | COD_LOC | COD_ZBOR | DATA |
|----|---------------|---------|----------|-----------|
| 1 | 1750 | 1060 | 1570 | 17-MAY-23 |
| 2 | 1760 | 1060 | 1580 | 18-MAY-23 |
| 3 | 1670 | 1090 | 1600 | 15-APR-23 |
| 4 | 1680 | 1100 | 1610 | 20-MAR-23 |
| 5 | 1680 | 1110 | 1610 | 20-MAR-23 |
| 6 | 1690 | 1150 | 1620 | 10-OCT-22 |
| 7 | 1700 | 1160 | 1630 | 15-SEP-22 |
| 8 | 1700 | 1150 | 1640 | 15-SEP-22 |
| 9 | 1710 | 1190 | 1650 | 15-NOV-21 |
| 10 | 1710 | 1200 | 1650 | 15-NOV-21 |
| 11 | 1710 | 1180 | 1650 | 15-NOV-21 |
| 12 | 1720 | 1140 | 1660 | 20-OCT-21 |
| 13 | 1730 | 1130 | 1660 | 20-FEB-23 |
| 14 | 1740 | 1120 | 1580 | 20-FEB-23 |

12 Formulati in limbaj natural si implementati 5 cereri SQL complexe

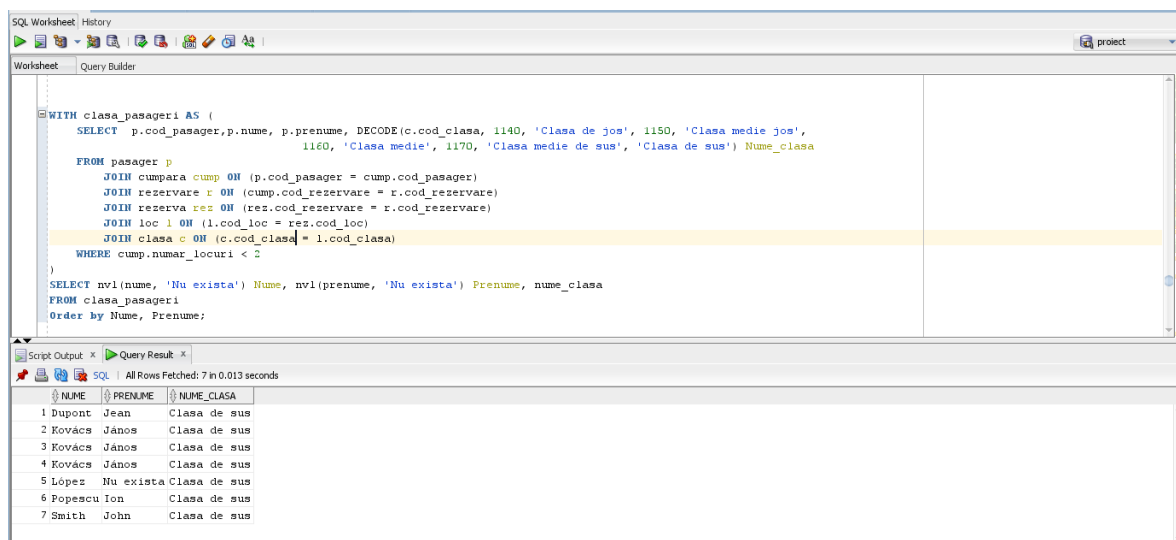
*/*Urmatoarea cerere SQL ia toti pasagerii care au cumparat un singur bilet, afisandu-le numele si prenumele (in caz ca unul dintre aceste attribute este null, va fi inlocuit cu 'Nu exista'). De asemenea, se va afisa si clasa in care se incadreaza biletul. Cum clasele nu aveau nume, vom folosi DECODE pentru a verifica pentru fiecare codul clasei si a il inlocui cu treapta clasei. Am folosit DECODE si NVL in aceeași cerere, precum si with*/*

```

WITH clasa_pasageri AS (
    SELECT p.cod_pasager, p.num, p.prenume, DECODE(c.cod_clasa, 1140, 'Clasa de jos',
                                                    1150, 'Clasa medie jos',
                                                    1160, 'Clasa medie',
                                                    1170, 'Clasa medie de sus',
                                                    'Clasa de sus') Nume_clasa

    FROM pasager p
        JOIN cumpara cump ON (p.cod_pasager = cump.cod_pasager)
        JOIN rezervare r ON (cump.cod_rezervare = r.cod_rezervare)
        JOIN rezerva rez ON (rez.cod_rezervare = r.cod_rezervare)
        JOIN loc l ON (l.cod_loc = rez.cod_loc)
        JOIN clasa c ON (c.cod_clasa = l.cod_clasa)
    WHERE cump.numar_locuri = 1
)
SELECT nvl(num, 'Nu exista') Num, nvl(prenume, 'Nu exista') Prenume, nume_clasa
FROM clasa_pasageri
Order by Num, Prenume;

```



The screenshot shows an SQL Worksheet with a query in the Query Builder and its results in the Query Result pane. The query is identical to the one in the previous block. The results pane shows 7 rows of data.

| | NUME | PRENUME | NUME_CLASA |
|---|---------|-----------|--------------|
| 1 | Dupont | Jean | Clasa de sus |
| 2 | Kovács | János | Clasa de sus |
| 3 | Kovács | János | Clasa de sus |
| 4 | Kovács | János | Clasa de sus |
| 5 | López | Nu exista | Clasa de sus |
| 6 | Popescu | Ion | Clasa de sus |
| 7 | Smith | John | Clasa de sus |

*/*Vom afisa pozitia locului, numar locului si numele pasagerului care l-a rezervat, pentru locurile care nu se afla la geam, iar numarul lor este mai mare decat ziua din data zborului. Am folosit o subcerere nesincronizata in care intervin mai mult de 3 tabele in clauza from, o functie pentru data, ordonari si o*

functie pentru siruri de caractere/*

```
SELECT pozitie, numar_loc, nume
FROM (SELECT rez.cod_zbor cod, p.numa nume, l.pozitie pozitie, l.numar_loc numar_loc
      FROM pasager p
      JOIN cumpara cump ON (p.cod_pasager = cump.cod_pasager)
      JOIN rezervare r ON (cump.cod_rezervare = r.cod_rezervare)
      JOIN rezerva rez ON (rez.cod_rezervare = r.cod_rezervare)
      JOIN loc l ON (l.cod_loc = rez.cod_loc)
      WHERE LOWER(pozitie) <> 'geam') cer
JOIN zbor z ON (z.cod_zbor = cer.cod)
WHERE numar_loc > EXTRACT(day FROM z.data);
```

The screenshot shows an SQL Worksheet interface with a query editor and a results pane. The query is as follows:

```
/*Vom afisa pozitia locului, numar locului si numele pasagerului care l-a rezervat, pentru locurile care nu se afla la geam, iar numarul lor este mai mare decat ziua din data zborului*/

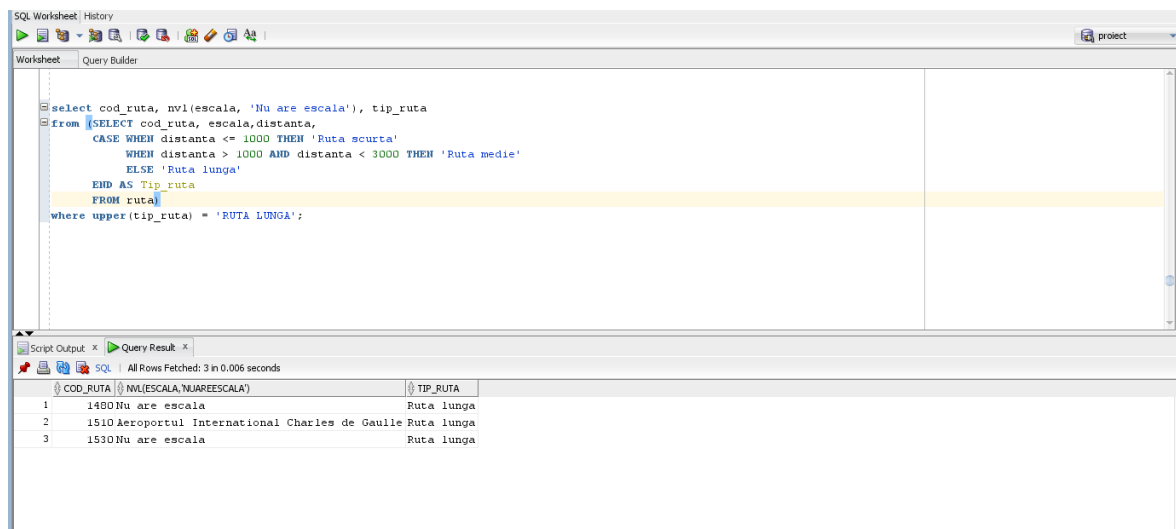
select pozitie, numar_loc, nume
from (select rez.cod_zbor cod, p.numa nume, l.pozitie pozitie, l.numar_loc numar_loc
      from pasager p
      JOIN cumpara cump ON (p.cod_pasager = cump.cod_pasager)
      JOIN rezervare r ON (cump.cod_rezervare = r.cod_rezervare)
      JOIN rezerva rez ON (rez.cod_rezervare = r.cod_rezervare)
      JOIN loc l ON (l.cod_loc = rez.cod_loc)
      where lower(pozitie) <> 'geam') cer
join zbor z ON (z.cod_zbor = cer.cod)
where numar_loc > extract(day from z.data);
```

The results pane shows the following data:

| | POZITIE | NUMAR_LOC | NUME |
|---|---------|-----------|---------|
| 1 | Central | 20 | Popescu |
| 2 | Central | 98 | Rossi |
| 3 | Margine | 26 | Rossi |
| 4 | Central | 114 | Kovacs |
| 5 | Centru | 19 | Kovacs |
| 6 | Centru | 19 | Kovacs |

*/*Vom afisa codul rutelor si aeroportul in care au escala (daca nu au, atunci vom inlocui numele aeroportului cu 'Nu are escala') si tipul rutei pentru zborurile pe ruta lunga. Daca ruta are mai putin de 1000 de km, atunci este scurta, daca are intre 1000 si 3000, atunci este medie, iar daca are mai mult de 3000 este lunga. Am folosit NVL, CASE, o functie pentru siruri de caractere.*/*

```
SELECT cod_ruta, NVL(escala, 'Nu are escala')
FROM (SELECT cod_ruta, escala,distanta,
CASE WHEN distanta <= 1000 THEN 'Ruta scurta'
      WHEN distanta > 1000 AND distanta < 3000 THEN 'Ruta medie'
      ELSE 'Ruta lunga'
END AS Tip_ruta
FROM ruta)
WHERE UPPER(tip_ruta) = 'RUTA LUNGA';
```



/ In cadrul acestei cereri SQL, vom obtine numele si prenumele angajatilor*

care lucreaza in cadrul zborurilor care sunt legate de aeroporturi din tara de care sunt legate cele mai multe zboruri. Pentru a calcula aceasta tara, am verificat pentru fiecare in parte daca numarul de zboruri de care este legata este egal cu maximul dintre numerele zborurilor de care este legata fiecare tara. Am folosit o cerere sincronizata in care intervin mai mult de trei tabele, grupari de date cu subcereri nesincronizate in care intervin cel putin 3 tabele, functii grup, filtrare la nivel de grupuri

**/*

```

select ang.nume, ang.prenume, zb.cod_zbor

from zbor zb join lucreaza lu on (lu.cod_zbor = zb.cod_zbor)

        join angajat ang on(ang.cod_angajat = lu.cod_angajat)

where lu.cod_zbor in(SELECT Z.cod_zbor

        FROM LOCATIE L

        JOIN AEROPORT A ON L.cod_locatie = A.cod_locatie

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

                ar.cod_aeroport2 = A.cod_aeroport

        JOIN RUTA R ON R.cod_ruta = ar.cod_ruta

        JOIN ZBOR Z ON ar.cod_ruta = Z.cod_ruta

        WHERE L.tara IN (

                SELECT L.tara

        FROM LOCATIE L

        JOIN AEROPORT A ON L.cod_locatie = A.cod_locatie

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

        A.cod_aeroport = ar.cod_aeroport2

        JOIN RUTA R ON ar.cod_ruta = R.cod_ruta

        GROUP BY L.tara

        HAVING COUNT(*) = (

                SELECT MAX(numar_zboruri)

```

```

FROM (

    SELECT L.tara, COUNT(*) AS numar_zboruri

    FROM LOCATIE L

    JOIN AEROPORT A ON L.cod_locatie =
A.cod_locatie

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

        A.cod_aeroport = ar.cod_aeroport2

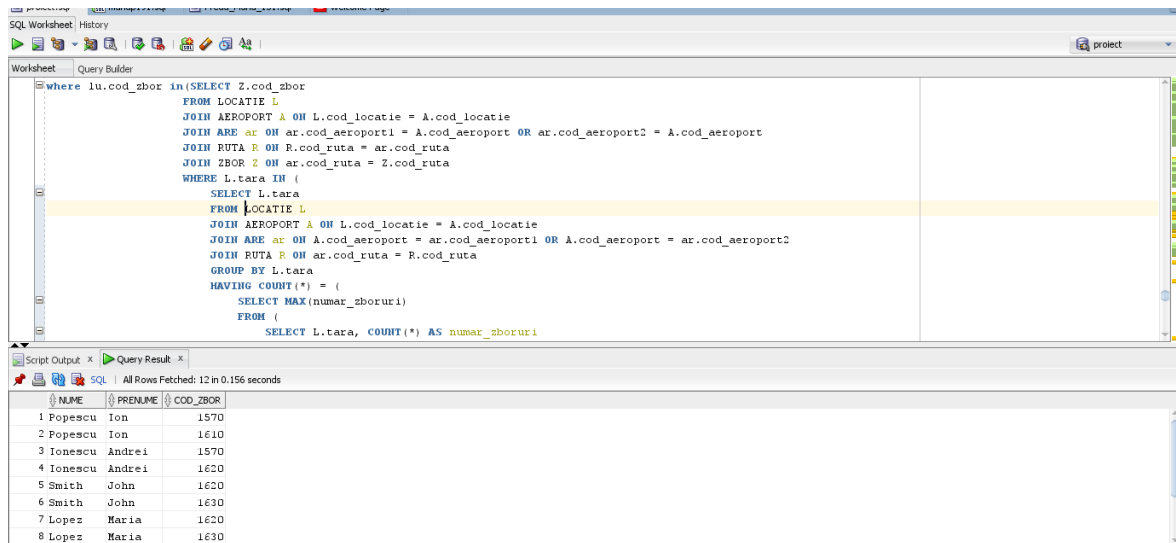
    JOIN RUTA R ON ar.cod_ruta = R.cod_ruta

    GROUP BY L.tara

) Subcerere

));

```

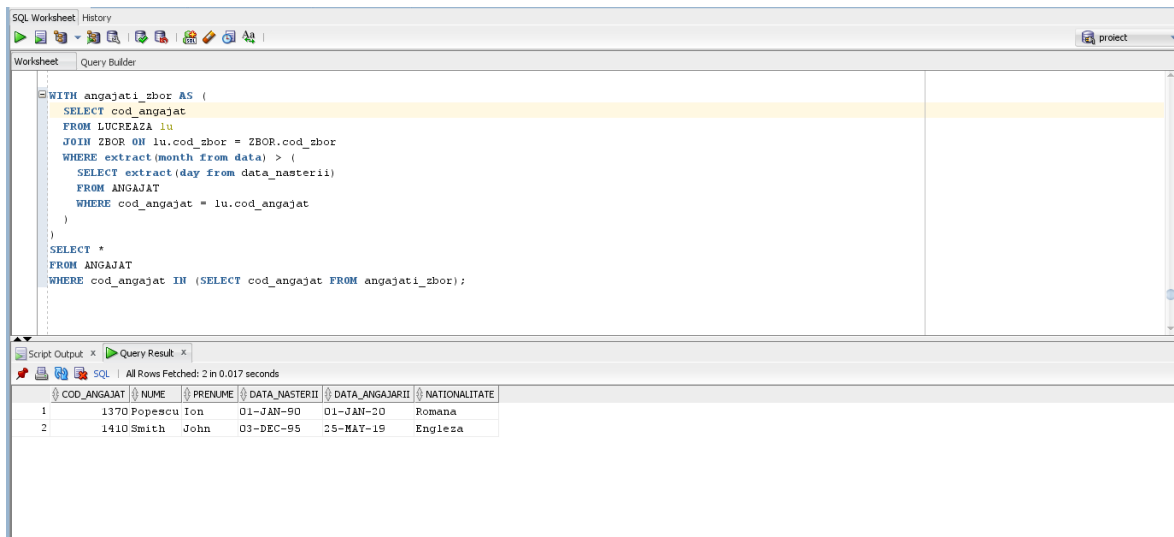


The screenshot shows a SQL IDE interface. The top pane displays a query in the Query Builder, which is a subquery for a larger query. The subquery selects the number of flights (numar_zboruri) for each country (L.tara) based on specific location, airport, and route criteria. The bottom pane shows the results of the query, which is a table with 8 rows and 3 columns: NUME, PRENUME, and COD_ZBOR.

| | NUME | PRENUME | COD_ZBOR |
|---|---------|---------|----------|
| 1 | Popescu | Ion | 1570 |
| 2 | Popescu | Ion | 1610 |
| 3 | Ionescu | Andrei | 1570 |
| 4 | Ionescu | Andrei | 1620 |
| 5 | Smith | John | 1620 |
| 6 | Smith | John | 1630 |
| 7 | Lopez | Maria | 1620 |
| 8 | Lopez | Maria | 1630 |

*/*In urmatoarea cerere SQL, vom afisa toate datele angajatilor, care au o luna in care au lucrat care este mai mare decat ziua nasterii lor. Am folosit din nou with, inca o functie pe date calendaristice si inca o subcerere sincronizata*/*

```
WITH angajati_zbor AS (  
  
    SELECT cod_angajat  
  
    FROM LUCREAZA lu  
  
    JOIN ZBOR ON lu.cod_zbor = ZBOR.cod_zbor  
  
    WHERE extract(month from data) > (  
  
        SELECT extract(day from data_nasterii)  
  
        FROM ANGAJAT  
  
        WHERE cod_angajat = lu.cod_angajat  
  
    )  
  
)  
  
SELECT *  
  
FROM ANGAJAT  
  
WHERE cod_angajat IN (SELECT cod_angajat FROM angajati_zbor);
```



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

*/*In primul update, vom modifica prenumele angajatei cu cel mai mic cod dintre angajatii cu prenumele 'Maria' in 'Daria'*/*

```
UPDATE angajat
```

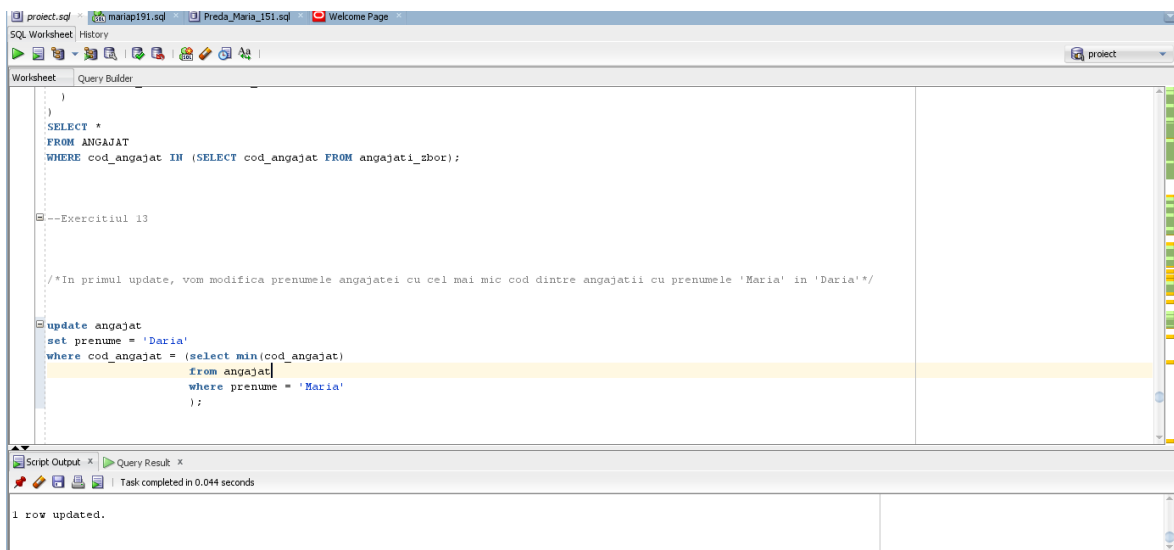
```
SET prenume = 'Daria'
```

```
WHERE cod_angajat = (SELECT MIN(cod_angajat)
```

```
FROM angajat
```

```
WHERE prenume = 'Maria'
```

```
);
```



*/*In al doilea update, vom concatena prenumele pasagerului care are rezervarea cu cele mai multe locuri cu '-Alexander'*/*

UPDATE pasager

SET prenume = prenume || '-Alexander'

WHERE cod_pasager = (SELECT cod_pasager

FROM cumpara

WHERE numar_locuri = (SELECT MAX(numar_locuri)

FROM cumpara));



*/*In cadrul ultimului update, vom modifica in 19 numerele locurilor care au fost rezervate de cele mai multe ori*/*

UPDATE loc

SET numar_loc = 19

WHERE cod_loc IN (SELECT l.cod_loc

FROM loc l JOIN rezerva r ON l.cod_loc = r.cod_loc

JOIN zbor z ON r.cod_zbor = z.cod_zbor

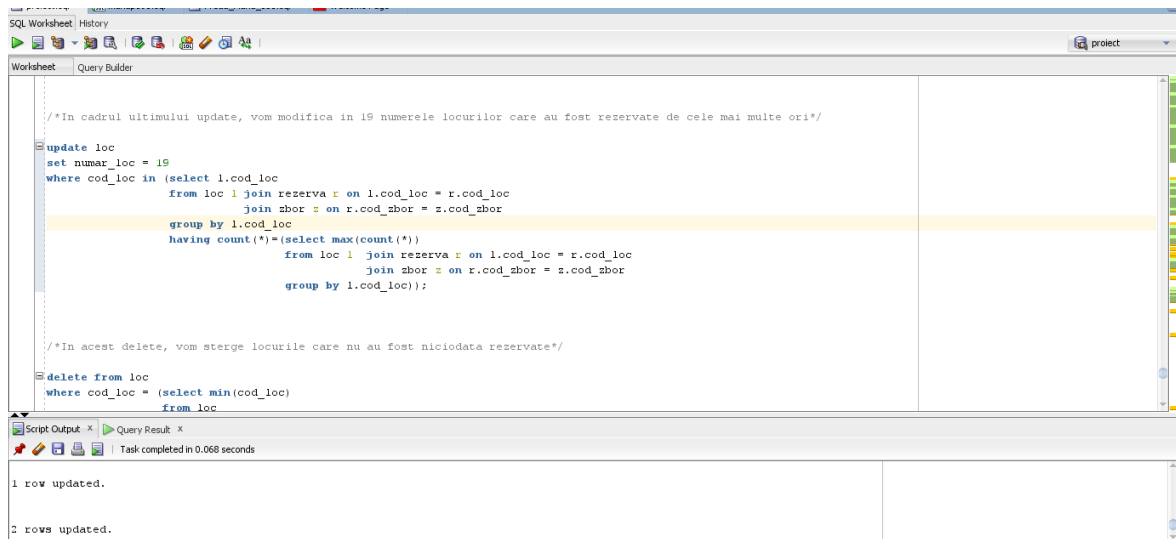
GROUP BY l.cod_loc

HAVING COUNT(*) = (SELECT MAX(COUNT(*))

FROM loc l JOIN rezerva r ON l.cod_loc = r.cod_loc

JOIN zbor z ON r.cod_zbor = z.cod_zbor

GROUP BY l.cod_loc));



*/*In acest delete, vom sterge locurile care nu au fost niciodata rezervate*/*

DELETE FROM loc

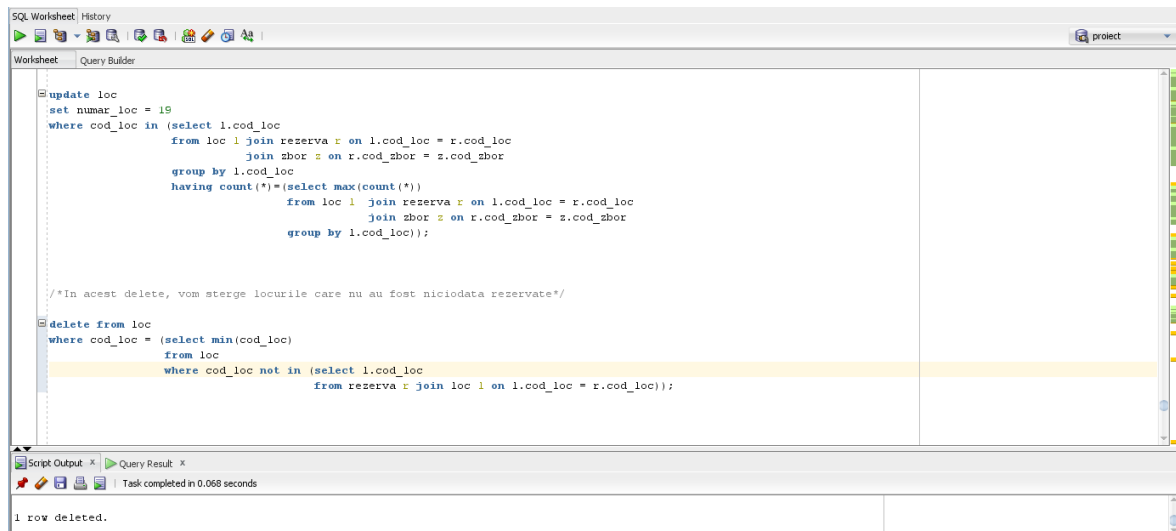
WHERE cod_loc = (SELECT MIN(cod_loc)

FROM loc

WHERE cod_loc NOT IN (SELECT l.cod_loc

FROM rezerva r

JOIN loc l ON l.cod_loc = r.cod_loc));



*/*Intai vom face nula locatie aeroporturilor care nu au zboruri internationale, pentru a putea apoi sa stergem aceasta locatie, fara a afecta tabelul aeroport*/*

UPDATE aeroport

SET cod_locatie = null

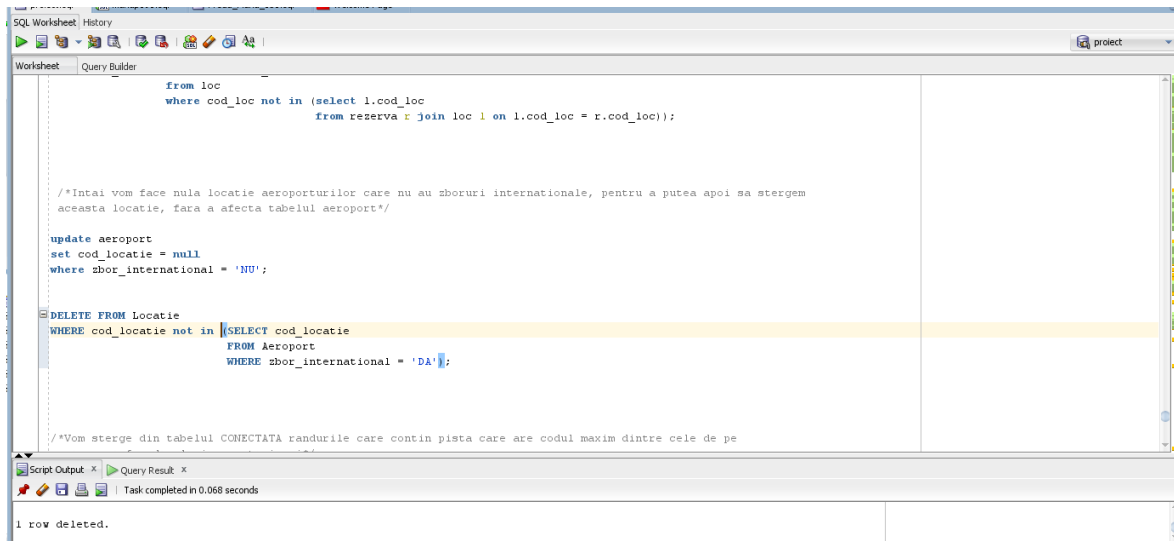
WHERE zbor_international = 'NU';

DELETE FROM Locatie

WHERE cod_locatie NOT IN (SELECT NVL(cod_locatie, 0)

FROM Aeroport

WHERE zbor_international = 'DA');



*/*Vom sterge din tabelul CONECTATA randurile care contin pista care are codul maxim dintre cele de pe care nu se fac decolari sau aterizari*/*

DELETE FROM conectata

WHERE cod_pista = (SELECT MAX(cod_pista)

FROM pista

WHERE cod_pista NOT IN (SELECT cod_pista

FROM pista p

JOIN zbor z ON p.cod_pista = z.cod_pista1

OR p.cod_pista = z.cod_pista2));



14 Formulati in limbaj natural si implementati in SQL: o cerere ce utilizeaza operatia outer-join pe minimum 4 tabele, o cerere ce utilizează operatia division si o cerere care implementează analiza top-n

*/*Vom verifica starea tuturor locurilor de la geam. Daca acestea sunt rezervate, vom afisa numele si prenumele persoanei care a facut rezervarea. Daca nu, vom afisa null.*/*

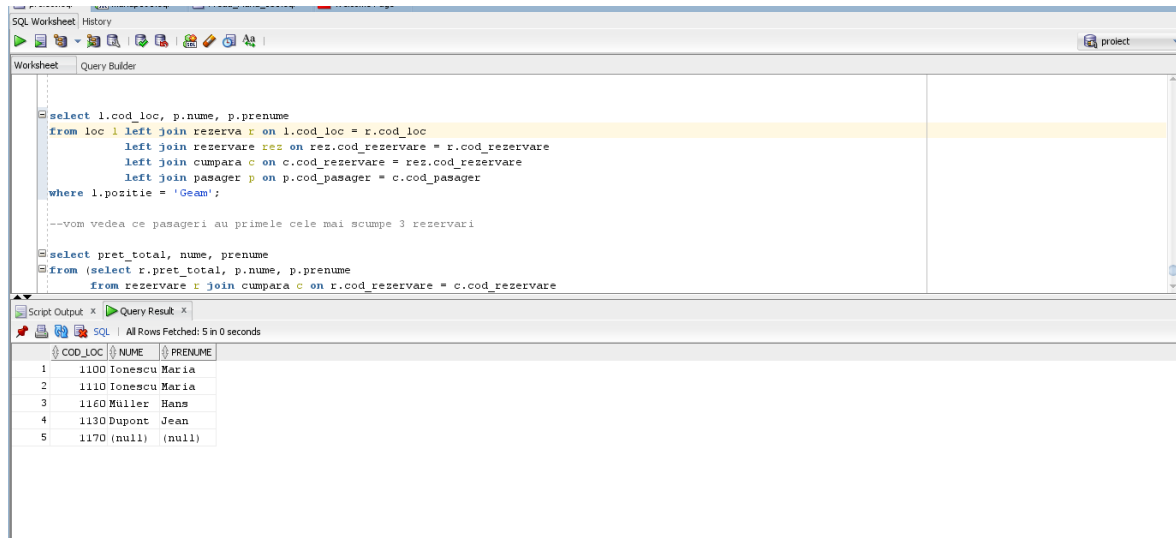
```

select l.cod_loc, p.num, p.prenume
from loc l left join rezerva r on l.cod_loc = r.cod_loc
           left join rezervare rez on rez.cod_rezervare = r.cod_rezervare
           left join cumpara c on c.cod_rezervare = rez.cod_rezervare
           left join pasager p on p.cod_pasager = c.cod_pasager

```



```
where l.pozitie = 'Geam';
```



--vom vedea ce pasageri au primele cele mai scumpe 3 rezervari

```
select pret_total, nume, prenume
from (select r.pret_total, p.nume, p.prenume
      from rezervare r join cumpara c on r.cod_rezervare = c.cod_rezervare
      join pasager p on p.cod_pasager = c.cod_pasager
      order by r.pret_total desc)
where rownum <= 3;
```

The screenshot shows an SQL Worksheet application. The main window displays a SQL query in the 'Query Builder' tab. The query is as follows:

```

left join pasager p on p.cod_pasager = c.cod_pasager
where l.pozitie = 'Gean';

--vom vedea ce pasageri au primele cele mai scumpe 3 rezervari

select pret_total, nume, prenume
from (select r.pret_total, p.nume, p.prenume
      from rezervare r join cumpara c on r.cod_rezervare = c.cod_rezervare
      join pasager p on p.cod_pasager = c.cod_pasager
      order by r.pret_total desc)
where rownum <= 3;

```

Below the query, the 'Query Result' tab shows the output of the query. It displays a table with 3 rows and 3 columns: PRET_TOTAL, NUME, and PRENUME. The data is as follows:

| PRET_TOTAL | NUME | PRENUME |
|------------|---------|--------------------|
| 3100 | Rossi | Giovanni-Alexander |
| 2400 | Müller | Hans |
| 1650 | Ionescu | Maria |

*/*In cadrul acestei cereri, vom folosi division pentru a obtine angajatii care au lucrat doar in cadrul zborurilor in care a lucrat si angajatul cu numele 'James'*/*

```

SELECT ang.nume, ang.prenume

FROM angajat ang JOIN lucreaza lu ON ang.cod_angajat = lu.cod_angajat

WHERE lu.cod_zbor IN (SELECT l.cod_zbor

                      FROM angajat a JOIN lucreaza l ON a.cod_angajat =
                      l.cod_angajat

                      WHERE a.nume = 'James')

```

```

AND ang.nume <> 'James'

```

MINUS

```

SELECT ang.nume, ang.prenume

FROM angajat ang JOIN lucreaza lu ON ang.cod_angajat = lu.cod_angajat

WHERE lu.cod_zbor NOT IN (SELECT l.cod_zbor

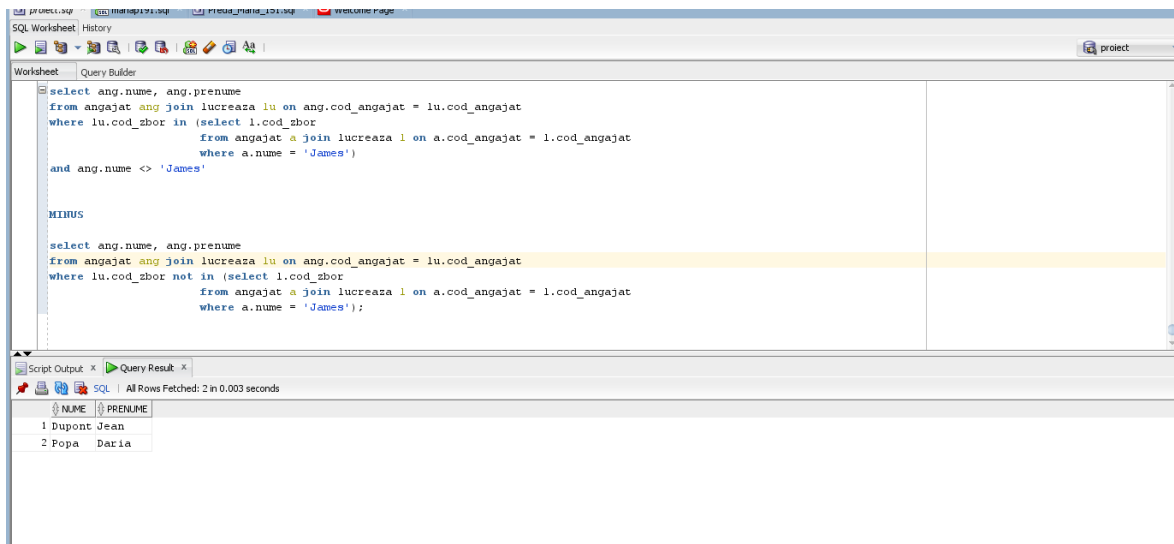
```

```

FROM angajat a JOIN lucreaza l ON a.cod_angajat =
l.cod_angajat

WHERE a.num = 'James')

```



15 Optimizarea unei cereri, aplicand regulile de optimizare ce deriva din proprietatile operatorilor algebrei relationale. Cererea va fi exprimata prin expresie algebrica, arbore algebric si limbaj (SQL), atat anterior cat si ulterior optimizarii.

In cadrul cererii SQL, vom selecta tara, orasul, adresa si numele aeroporturilor in care se afla terminale cu capacitate mai mare de 2000 de persoane, aflandu-se in partea de nord a aeroportului. Se va selecta si capacitatea terminalului. De asemenea, aeroporturile respective trebuie sa fie legate de zboruri internationale.

Cerere SQL:

```

SELECT tara, adresa, oras, nume, t.capacitate
FROM locatie l JOIN aeroport a ON a.cod_locatie = l.cod_locatie
      JOIN terminal t ON a.cod_aeroport = t.cod_aeroport
WHERE zbor_international = 'DA' and capacitate > 2000 and capacitate < 5000

```

Varianta neoptimizata

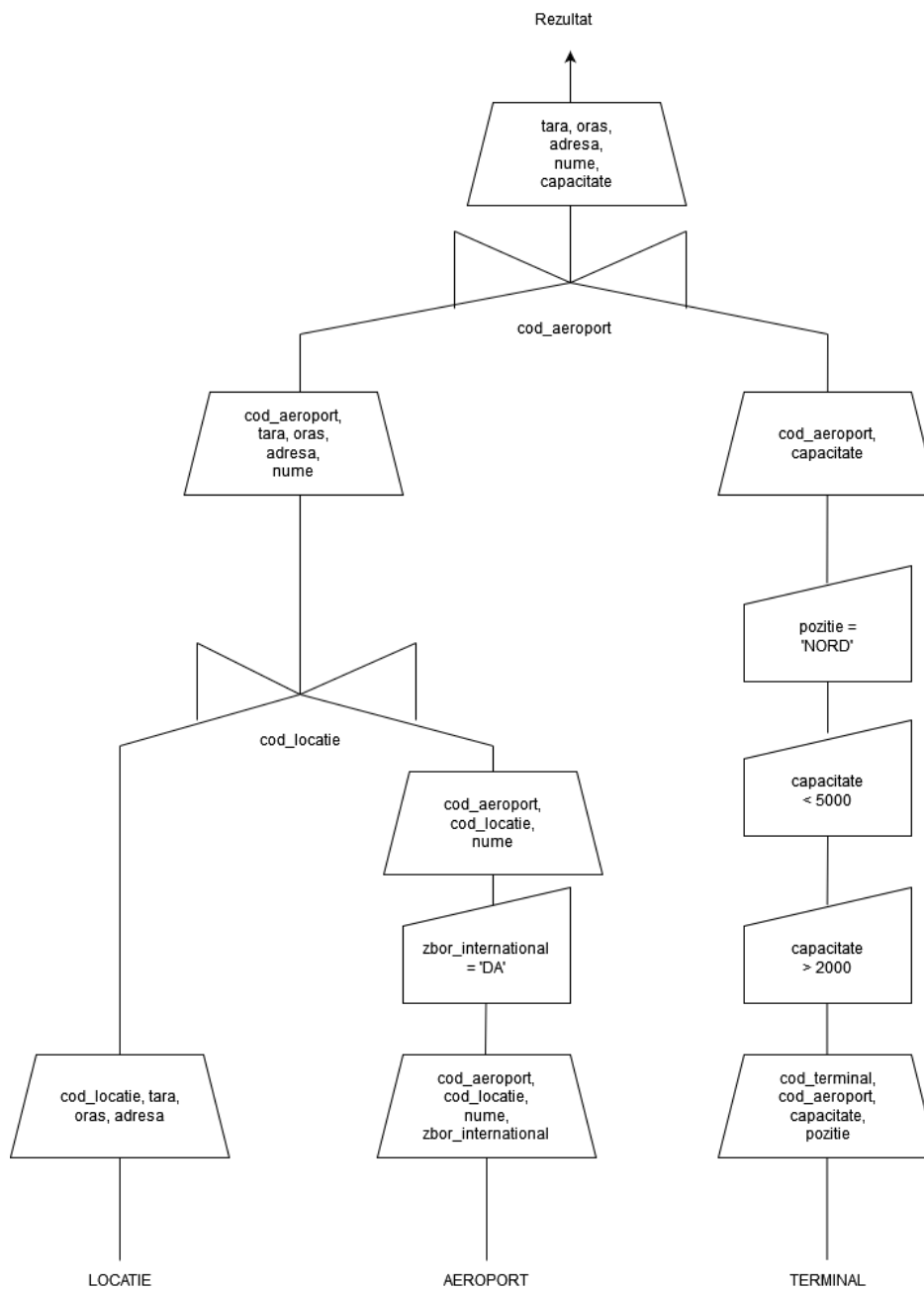
Expresie algebrica:

```

R1 = PROJECT(LOCATIE, cod_locatie, tara, oras, adresa)
R2 = PROJECT(AEROPORT, cod_locatie, cod_aeroport, nume, zbor_international)
R3 = SELECT(R2, zbor_international = 'DA')
R4 = PROJECT(R3, cod_aeroport, cod_locatie, nume)
R5 = JOIN(R1, R4, cod_locatie)
R6 = PROJECT(R5, cod_aeroport, tara, oras, adresa, nume)
R7 = PROJECT(TERMINAL, cod_aeroport, capacitate, pozitie)
R8 = SELECT(R7, capacitate > 2000)
R9 = SELECT(R8, capacitate < 5000)
R10 = SELECT(R9, pozitie = 'NORD')
R11 = PROJECT(R10, cod_aeroport, capacitate)
R12 = JOIN(R5, R11, cod_aeroport)
Rezultat = PROJECT(R12, tara, oras, adresa, nume, capacitate)

```

Arborele algebric:



Varianta optimizata:

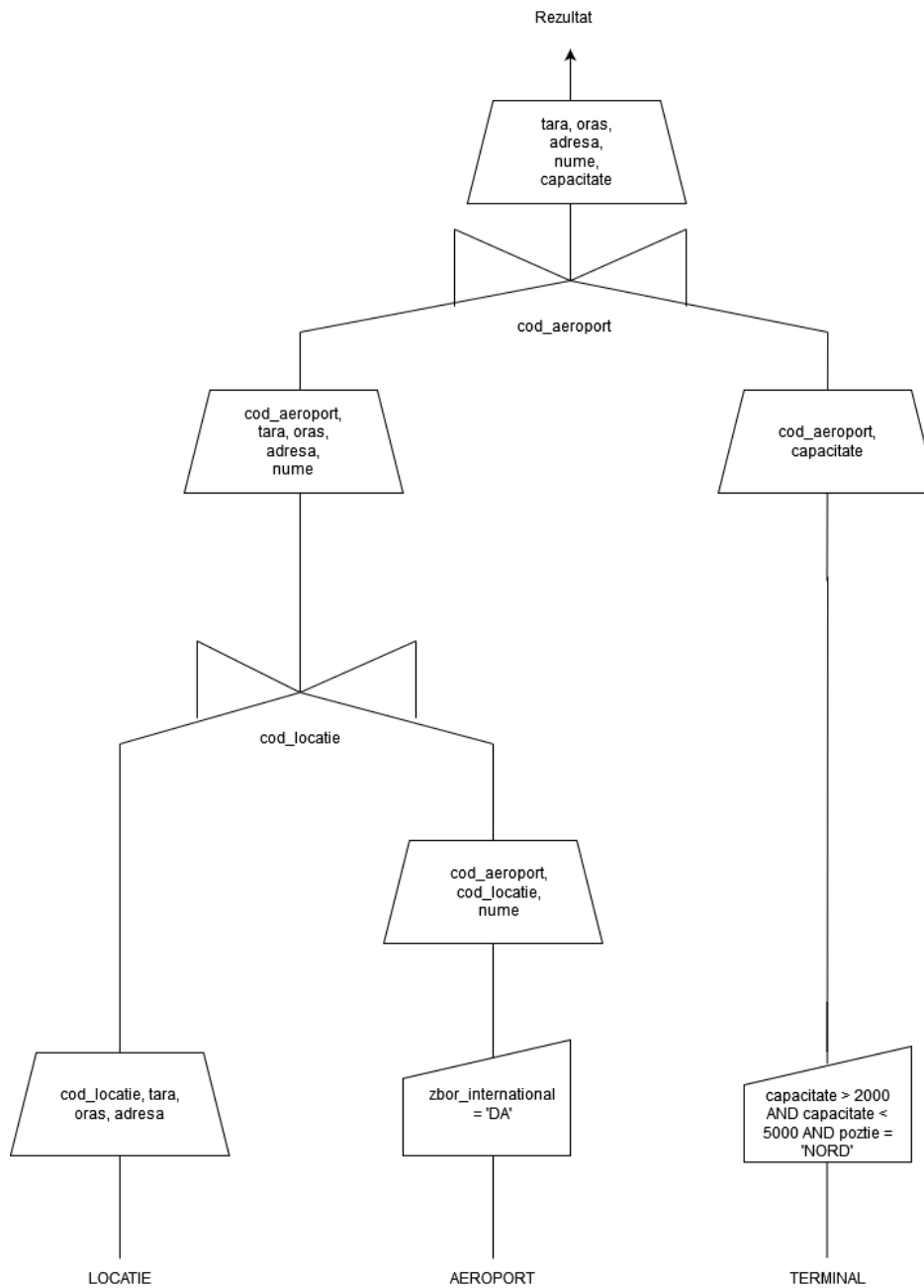
Incepem optimizarea prin aplicarea Proprietatii 4 (compunerea selectiilor) pentru relatiile R8, R9, R10. Astfel, obtinem, ca pas intermediar, relatiile:

```
R1 = PROJECT(LOCATIE, cod_locatie, tara, oras, adresa)
R2 = PROJECT(AEROPORT, cod_locatie, cod_aeroport, nume, zbor_international)
R3 = SELECT(R2, zbor_international = 'DA')
R4 = PROJECT(R3, cod_aeroport, cod_locatie, nume)
R5 = JOIN(R1, R4, cod_locatie)
R6 = PROJECT(R5, cod_aeroport, tara, oras, adresa, nume)
R7 = PROJECT(TERMINAL, cod_aeroport, capacitate, pozitie)
R8 = SELECT(R7, capacitate > 2000 AND capacitate < 5000 AND pozitie = 'NORD')
R9 = PROJECT(R8, cod_aeroport, capacitate)
R10 = JOIN(R5, R9, cod_aeroport)
Rezultat = PROJECT(R10, tara, oras, adresa, nume, capacitate)
```

Mai departe, vom aplica asupra relatiilor R2, R3 si R4 Proprietatea 5 (comutarea selectiei cu proiectia). Facem acelasi lucru si pentru relatiile R7, R8 si R9.

```
R1 = PROJECT(LOCATIE, cod_locatie, tara, oras, adresa)
R2 = SELECT(AEROPORT, zbor_international = 'DA')
R3 = PROJECT(R2, cod_aeroport, cod_locatie, nume)
R4 = JOIN(R1, R3, cod_locatie)
R5 = PROJECT(R4, cod_aeroport, tara, oras, adresa, nume)
R6 = SELECT(TERMINAL, capacitate > 2000 AND capacitate < 5000 AND pozitie = 'NORD')
R7 = PROJECT(R6, cod_aeroport, capacitate)
R8 = JOIN(R4, R7, cod_aeroport)
Rezultat = PROJECT(R8, tara, oras, adresa, nume, capacitate)
Se observa ca aceasta este forma algebrica optimizata.
```

Arbore algebric:



16 Realizarea normalizarii BCNF, FN4, FN5. Aplicarea denormalizarii

Realizarea normalizarii BCNF

In acest moment, se poate observa ca nu exista chei candidat in tabelele din diagrama conceptuala de la punctul 7. De aceea, vom presupune ca am fi introdus si atributul `cnp_pasager` in tabelul asociativ CUMPARA:

CUMPARA(`cod_rezervare#`, `cod_pasager#`, `cnp_pasager`, `nr_locuri`)

In acest caz, intre attributele relatiei exista dependentele:

$\{\text{cod_rezervare\#}, \text{cod_pasager\#}\} \rightarrow \{\text{cnp_pasager}, \text{nr_locuri}\}$

$\{\text{cnp_pasager}\} \rightarrow \{\text{cod_pasager}\}$ (`cnp_pasager` este o cheie candidat, intrucat `cod_pasager` depinde de CNP-ul acestuia)

Aplicam regula Casey-Delobel pentru a aduce relatia in BCNF, cheia candidat transformandu-se in primul tabel in cheie primara, iar in al doilea in cheie externa:

CUMPARA_1(`cod_pasager`, `cnp_pasager#`)

CUMPARA_2(`cod_rezervare#`, `cnp_pasager`, `nr_locuri`)

Astfel, am fi adus aceasta relatie in forma BCFN. Totusi, acest lucru nu este necesar, deoarece nu avem atributul CNP in tabelul CUMPARA.

Tabelul initial

| <code>cod_rezervare</code> | <code>cod_pasager</code> | <code>nr_locuri</code> | <code>cnp_pasager</code> |
|----------------------------|--------------------------|------------------------|--------------------------|
| 1670 | 1270 | 1 | 325326873627 |
| 1680 | 1280 | 2 | 325326873628 |
| 1690 | 1290 | 1 | 325326873629 |
| 1700 | 1300 | 2 | 325326873630 |
| 1710 | 1310 | 3 | 325326873631 |
| 1720 | 1320 | 1 | 325326873632 |
| 1730 | 1330 | 1 | 325326873633 |
| 1740 | 1340 | 1 | 325326873634 |
| 1750 | 1340 | 1 | 325326873635 |
| 1760 | 1340 | 1 | 325326873636 |

CUMPARA_1

| <code>cod_pasager</code> | <code>nume</code> |
|--------------------------|-------------------|
| 1270 | 325326873627 |
| 1280 | 325326873628 |
| 1290 | 325326873629 |
| 1300 | 325326873630 |
| 1310 | 325326873631 |
| 1320 | 325326873632 |
| 1330 | 325326873633 |
| 1340 | 325326873634 |
| 1340 | 325326873635 |
| 1340 | 325326873636 |

CUMPARA_2

| cod_rezervare | nr_locuri | cnp_pasager |
|----------------------|------------------|--------------------|
| 1670 | 1 | 325326873627 |
| 1680 | 2 | 325326873628 |
| 1690 | 1 | 325326873629 |
| 1700 | 2 | 325326873630 |
| 1710 | 3 | 325326873631 |
| 1720 | 1 | 325326873632 |
| 1730 | 1 | 325326873633 |
| 1740 | 1 | 325326873634 |
| 1750 | 1 | 325326873635 |
| 1760 | 1 | 325326873636 |

Realizarea normalizarii FN4

Singura relatie ce ar putea fi normalizata pana la forma normala 4, cu riscul de a pierde date, este relatia REZERVA.(cod rezervare, cod loc, cod zbor, data), in cadrul careia avem urmatoarele dependente multiple:

cod rezervare →→ cod loc

cod rezervare →→ cod zbor

Relatiile rezultate dupa aplicarea lui FN4 sunt:

REZERVA 1(cod rezervare, cod loc)

REZERVA 2(cod rezervare, cod zbor, data)

| cod_rezervare | cod_loc | cod_zbor | data |
|----------------------|----------------|-----------------|-------------|
| 1750 | 1060 | 1570 | 17-MAY-23 |
| 1760 | 1060 | 1580 | 18-MAY-23 |
| 1670 | 1090 | 1600 | 15-APR-23 |
| 1680 | 1100 | 1610 | 20-MAR-23 |
| 1680 | 1110 | 1610 | 20-MAR-23 |
| 1690 | 1150 | 1620 | 10-OCT-22 |
| 1700 | 1160 | 1630 | 15-SEP-22 |
| 1700 | 1150 | 1640 | 15-SEP-22 |
| 1710 | 1190 | 1650 | 15-NOV-21 |
| 1710 | 1200 | 1650 | 15-NOV-21 |
| 1710 | 1180 | 1650 | 15-NOV-21 |
| 1720 | 1140 | 1660 | 20-OCT-21 |
| 1730 | 1130 | 1660 | 20-FEB-23 |
| 1740 | 1120 | 1580 | 20-FEB-23 |

REZERVA_1

| cod_rezervare | cod_loc |
|----------------------|----------------|
| 1750 | 1060 |
| 1760 | 1060 |
| 1670 | 1090 |
| 1680 | 1100 |
| 1680 | 1110 |
| 1690 | 1150 |
| 1700 | 1160 |
| 1700 | 1150 |
| 1710 | 1190 |
| 1710 | 1200 |
| 1710 | 1180 |
| 1720 | 1140 |
| 1730 | 1130 |
| 1740 | 1120 |

REZERVA_2

| cod_rezervare | cod_zbor | data |
|----------------------|-----------------|-------------|
| 1750 | 1570 | 17-MAY-23 |
| 1760 | 1580 | 18-MAY-23 |
| 1670 | 1600 | 15-APR-23 |
| 1680 | 1610 | 20-MAR-23 |
| 1680 | 1610 | 20-MAR-23 |
| 1690 | 1620 | 10-OCT-22 |
| 1700 | 1630 | 15-SEP-22 |
| 1700 | 1640 | 15-SEP-22 |
| 1710 | 1650 | 15-NOV-21 |
| 1710 | 1650 | 15-NOV-21 |
| 1710 | 1650 | 15-NOV-21 |
| 1720 | 1660 | 20-OCT-21 |
| 1730 | 1660 | 20-FEB-23 |
| 1740 | 1580 | 20-FEB-23 |

De asemenea, pentru a oferi un exemplu mai clar, vom modifica unul dintre tabele pentru a putea evidenția necesitatea realizării normalizării FN4.

Vom lua tabelul CONECTATA și vom adăuga la cheia primară și `cod_angajat`, care se ocupă de eliberarea drumului pe pistă pentru conectare \Rightarrow `CONECTATA(cod_pista#, cod_poarta#, cod_angajat#, iluminare)`. Astfel, ar apărea dependentele multiple:

`cod_pista` $\rightarrow \rightarrow$ `cod_angajat`
`cod_pista` $\rightarrow \rightarrow$ `cod_poarta`

Relațiile rezultate după aplicarea lui FN4 sunt:

`CONECTATA_1(cod_pista, cod_poarta)`
`CONECTATA_2(cod_pista, cod_angajat, iluminare)`

Pentru reprezentarea tabelelor, vom lua doar o parte dintre inserturi:

CONECTATA:

| cod_pista | cod_poarta | cod_angajat |
|------------------|-------------------|--------------------|
| 720 | 470 | 4000 |
| 720 | 480 | 4000 |
| 730 | 490 | 4000 |
| 740 | 500 | 4000 |
| 740 | 500 | 4010 |
| 730 | 490 | 4010 |
| 720 | 480 | 4010 |
| 720 | 470 | 4010 |

CONECTATA₁ :

| cod_pista | cod_poarta |
|------------------|-------------------|
| 720 | 470 |
| 720 | 480 |
| 730 | 490 |
| 740 | 500 |
| 740 | 500 |
| 730 | 490 |
| 720 | 480 |
| 720 | 470 |

CONECTATA₂:

| cod_pista | cod_angajat |
|------------------|--------------------|
| 720 | 4000 |
| 720 | 4000 |
| 730 | 4000 |
| 740 | 4000 |
| 740 | 4010 |
| 730 | 4010 |
| 720 | 4010 |
| 720 | 4010 |

Realizarea normalizarii FN5

Pentru ca o relatie sa se afle in FN5, aceasta trebuie sa se afle in FN4 si sa nu aiba dependente ciclice. In acest moment, nu avem dependente ciclice, iar de la punctul anterior am vazut ca deja ne aflam in

FN4. Asadar, vom trata un exemplu de relatie ce nu se afla in FN5.

Vom presupune ca REZERVA nu ar fi o relatie de tip 3, avand in schimb o join-dependenta (ar exista multdependente intre perechile de tabele (REZERVARE, LOC), (REZERVARE ZBOR), (LOC, ZBOR)). Astfel, in loc sa avem tabelul REZERVA, am avea trei tabele asociative T1, T2, T3 (cate unul pentru fiecare multidependenta). Aceasta abordare ar constitui o dependenta ciclica, generand redundanta in baza de date.

Pentru aducerea in forma normala 5, am fi nevoiti sa inlocuim cele trei relatii de tip 2 ((cod_loc#, cod_rezervare#); (cod_loc#, cod_zbor#); (cod_rezervare#, cod_zbor#)) cu o relatie de tip 3 (cod_rezervare#, cod_loc#, cod_zbor#).

Denormalizarea

Mai devreme, pentru a aduce REZERVA in a patra forma normala, am impartit in doua relatii:

REZERVA_1(cod_rezervare, cod_loc)

REZERVA_2(cod_rezervare, cod_zbor)

Deoarece in cadrul unei rezervari pot fi alese mai multe locuri din cadrul mai multor zboruri, in acest moment, o parte dintre date s-au pierdut. Nu putem afla in cadrul carui zbor a fost rezervat fiecare loc. Pentru a rezolva aceasta problema, este necesara aplicarea denormalizarii, ajungand cu tabelul REZ-ERVA in forma initiala. Astfel, vom aplica opusul a ceea ce am facut la realizarea normalizarii FN4, revenind la relatia:

REZERVA(cod_rezervare#, cod_loc#, cod_zbor#).