Proiect Baze de Date

George Radu

1 Introducere

Detaliile din acest proiect se referă la proiectarea unui model de date ce furnizează informații despre gestiunea companiilor de transporturi rutiere de mărfuri din România.

Modelul de date va gestiona informații legate de transporturile de mărfuri ale unor companii client de un depozit la altul. Există firme de pază care se ocupă de paza depozitelor. O firmă de pază alocă fiecărui depozit o echipă de pază.

Transporturile de marfă sunt realizate de o firmă de transport prin intermediul angajaților profesioniști care pot avea mai multe categorii de permise și se pot opri la mai multe popasuri petru odihnă/masă/alimentare. Transportul de marfă începe de la un depozit și se termină la alt depozit, în fiecare depozit accesul camionului este permis dupa o verificare a echipei de pază.

2 Restricții de funcționare

Modelul de date respectă anumite restricții de funcționare.

- Orice depozit are un singur deținător(o companie client ori o firmă de transport) și poate fi folosit pentru depozitarea de mărfuri.
- O companie client sau o firmă de transport poate deține mai multe depozite.
- Unui depozit îi este alocată o unică locație.
- Un depozit este păzit la un moment dat de o singură echipă de pază.

- Un depozit are o listă de inventar cu ce mărfuri au intrat/ieșit împreună cu data și ora.
- Un lot de marfă poate să fie transportat de mai multe ori, nu neapărat î1 același transport, și nu neapărat între aceleași depozite.
- O echipă de pază aparține unei firme de pază, firmă de pază care poate avea mai multe echipe de pază, o firmă putând astfel să păzească mai multe depozite aflate în diferite locații.
- Un angajat poate să fie ori paznic ori șofer.
- Un transport de marfă se realizează prin intermediul unui camion încărcat cu marfă de la un depozit la altul.
- Un șofer poate să oprească în timpul transportului la mai multe popasuri pentru odihnă/mâncare sau alimentare. La popasuri pot veni mai mulți șoferi.
- Un popas se află intr-o unică locație și poate fi o benzinărie, un motel sau un restaurant. Nu ne interesează detalii specifice despre un popas cum ar fi ce mâncare vinde un restaurant, sau cât costa o cameră la un motel.
- Un șofer poate să aibă mai multe permise pentru mai multe categorii, poate chiar să i se anuleze/expire un permis și să trebuiască să îl obțină din nou.
- Un șofer trebuie să conducă un camion, dar poate să conducă mai multe camioane(nu în același timp) în funcție de problemele tehnice legate de camion și necesitățile firmei.
- O firmă de transport deține mai multe camioane, iar un camion este deținut de o singură firmă de transport.
- Transportul de marfă se face între 2 depozite, se transportă mai multe loturi de marfă. Pentru un transport vom reține și data din momentul în care a început transportul. Pentru fiecare transport se va folosi un singur camion condus de un șofer profesionist.

3 Entități

Pentru modelul de date referitor la gestiunea companiilor de transporturi rutiere de mărfuri, structurile: DEPOZIT, LOCATIE, POPAS, FIRMA, CLIENT, PAZA, TRANSPORT, ECHIPA_PAZA, INVENTAR_DEPOZIT, INVENTAR_TRANSPORT, LOT_MARFA, CAMION, ANGAJAT, PAZNIC, SOFER, PERMIS reprezintă entități.

Vom prezenta entitățile modelului de date, realizând o descriere completă a fiecăreia. De asemnea, pentru fiecare entitate se va preciza cheia primară.

Toate entitățile care vor fi precizate sunt independente, cu excepția entităților dependente ECHIPA_PAZA, PERMIS și INVENTAR_TRANSPORT și a subentităților SOFER, PAZNIC, CLIENT, PAZA, TRANSPORT.

- DEPOZIT = entitatea descrie o construcție al cărei proprietar este o firmă client sau de transport, în care pot fi depozitate mărfuri. Cheia primară a entității este id_depozit.
- LOCATIE = entitatea descrie informații despre locația fizică a unei construcții, fie ea depozit sau popas. Cheia primară a entității este id_locatie.
- POPAS = reprezintă un loc în care un șofer se poate opri pentru odihnă. Cheia primară a entității este id_popas.
- FIRMA = entitatea descrie o persoană juridică din România, supraentitate pentru CLIENT, PAZA și TRANSPORT. Cheia primară a entității este id_firma.
- CLIENT = persoană juridică beneficiară a unor servicii de transport oferite de o firmă de transport specializată. Cheia primară a entității este id_firma.
- PAZA = persoana juridică ce asigură securitatea depozitelor prin intermediul unor echipe de pază formate din paznici profesioniști. Cheia primară a entității este id_firma_paza.
- TRANSPORT = persoană juridică responsabilă de transportul de mărfuri în siguranță pentru companii client de la un depozit la altul. Pentru transporturi firma pune la dispoziție șoferilor săi mai multe camioane. Cheia primară a entității este id_firma_transport.

- ECHIPA_PAZA = entitatea descrie informații despre o echipă formată din paznici responsabili de securitatea unui depozit. Cheia primară a entității este id_echipa_paza.
- INVENTAR_DEPOZIT = entitatea descrie informații despre inventarierea unui depozit, ce loturi au intrat în inventariere, când au intrat și când au ieșit. Cheia primară a entității este id_inventar_depozit.
- INVENTAR_TRANSPORT = entitatea descrie informații despre ce lot de marfă a fost implicat intr-un transport între 2 depozite. Cheia primară a entității este cheia compusa dintre id_inventar_transport și id_lot_marfa.
- LOT_MARFA = entitatea descrie informații despre un lot de marfă transportată sau depozitată. Cheia primară a entității este id_marfa.
- CAMION = vehicul de mare tonaj pentru transport marfă. Cheia primară a entității este id_camion.
- ANGAJAT = persoană fizică angajată a unei companii fie de pază fie de transport pentru a-și oferi serviciile în schimbul unei remunerații. Cheia primară a entității este id_angajat.
- PAZNIC = subentitate a entității ANGAJAT, lucrează pentru o companie de pază și este responsabil de securitatea unui depozit din cadrul echipei sale. Cheia primară a entității este id_angajat.
- SOFER = subentitate a entității ANGAJAT, lucrează pentru o firmă de transport, îi este alocat un camion și este responsabil pentru transportul in siguranță a unei remorci cu marfă. Cheia primară a entității este id_angajat.
- PERMIS = entitate ce descrie informații despre ce tipuri de permis are un șofer și când le-a obținut. Cheia primară a entității este una compusă din id_permis + id_angajat.

4 Relații

Se vor prezenta relațiile modelului de date, dând o descriere completă a fiecăreia. Denumirile acestor legături între entități sunt alese sugestiv. Pentru fiecare relație se va preciza cardinalitatea minimă și maximă.

- DEPOZIT_se_află_LOCATIE = relație ce leagă entitățile DEPOZIT și LOCATIE, reflectând legătura dintre acestea(un depozit se află plasat într-o locație unică din țară). Ținând cont de condițiile impuse modelului un depozit trebuie să se afle într-o singură locație. Relația are cardinalitatea minimă 1:1 și cardinalitatea maximă 1:1.
- POPAS_se_află_LOCATIE = relație ce leagă entitățile POPAS și LOCATIE, reflectând legătura dintre acestea(un popas se află plasat într-o locație unică din țară). Ținând cont de condițiile impuse modelului un popas trebuie să se afle într-o singură locație. Relația are cardinalitatea minimă 1:1 și cardinalitatea maximă 1:1.
- FIRMA_detine_DEPOZIT = relație ce leagă entitățile FIRMA și DEPOZIT, reflectând legătura dintre acestea(un depozit este deținut de o firmă). Relația are cardinalitatea minimă 1:1 si cardinalitatea maximă 1:M.
- PAZA_aparține_ECHIPA_PAZA = relație ce leagă entitățile PAZA și ECHIPA_PAZA, reflectând legătura dintre acestea(o echipă de pază aparține unei firme de pază). Relația are cardinalitatea minimă 1:1 si cardinalitatea maximă 1:M.
- ECHIPA_PAZA_păzește_DEPOZIT = relație ce leagă entitățile ECHIPA_PAZA și DEPOZIT, reflectând legătura dintre acestea (o echipă de pază trebuie să păzească un depozit pentru a permite accesuri în depozit doar de staff autorizat). Relația are cardinalitatea minimă 1:1 si cardinalitatea maximă 1:1(Conform condițiilor impuse modelului un depozit va fi păzit la un moment în timp doar de o echipă de pază).
- PAZNIC_aparține_ECHIPA_PAZA = relație ce leagă entitățile PAZNIC și ECHIPA_PAZA, reflectând legătura dintre acestea (un paznic trebuie să facă parte dintr-o echipă de pază trimisă la un depozit, paznicul lucrează pentru firmă de pază de care aparține echipa). Relația are cardinalitatea minimă 1:1 si cardinalitatea maximă M:1(un paznic face parte dintr-o singură echipă, iar o echipă este formată din mai mulți paznici).

- INVENTAR_DEPOZIT_aparține_DEPOZIT = relație ce leagă entitățile INVENTAR_DEPOZIT și DEPOZIT, reflectând legătura dintre acestea (un depozit trebuie să țină inventarul la ce loturi de marfă intră și iese și când s-au întâmplat aceste evenimente). Relația are cardinalitatea minimă 0:1 si cardinalitatea maximă M:1(inițial un depozit abia construit poate să nu conțină niciun record de invetariere, urmând pe parcurs să aibă).
- INVENTAR_DEPOZIT_aparține_LOT_MARFA = relație ce leagă entitățile INVENTAR_DEPOZIT și LOT_MARFA , reflectând legătura dintre acestea(un lot de marfă va fi înregistrat în lista de inventar a unui depozit atunci când va fi adus în depozit fizic). Relația are cardinalitatea minimă 1:1 si cardinalitatea maximă M:1(Același lot de marfă poate să fie transferat între depozite din puncte cheie din țară în funcție de necesitățile firmei client).
- LOT_MARFA_aparține_INVENTAR_TRANSPORT = relație ce leagă entitățile LOT_MARFA și INVENTAR_TRANSPORT, reflectând legătura dintre acestea (pentru fiecare transport de marfă se face un inventar din care fac parte mai multe loturi de marfă, un lot de marfă poate să fie transportat de mai multe ori). Relația are cardinalitatea minimă 1:1 si cardinalitatea maximă 1:M(ținând cont de condițiile impuse modelului entitatea INVENTAR_TRANSPORT va conține doar un lot de marfă și din ce lot face parte).
- CAMION_transportă_INVENTAR_MARFĂ_de_la_DEPOZIT _la_DEPOZIT = relație de tip 3 leagă entitățile DEPOZIT, DEPOZIT, INVENTAR_MARFĂ și CAMION, reflectând ce marfă a fost transportată de la un depozit de plecare la unul de sosire, transportul fiind făcut cu un camion. Denumirea acestei relații va fi transport.
- TRANSPORT_detine_CAMION = relație ce leagă entitățile TRANSPORT și CAMION, reflectând legătura dintre acestea(o firmă de transport trebuie să dețină camioane). Relația are cardinalitatea minimă 1:1 si cardinalitatea maximă 1:M.
- SOFER_conduce_CAMION = relație ce leagă entitățile SOFER și CAMION, reflectând legătura dintre acestea(un șofer de la o firmă de transport trebuie să conducă cel puțin un camion pentru a-și îndeplini saricinile de la muncă). Relația are cardinalitatea minimă 1:1 si cardinalitatea maximă M:M.
- ANGAJAT_lucrează_la_FIRMA = relație ce leagă entitățile ANGAJAT și FIRMA, reflectând legătura dintre acestea(un anagajt/o persoană

fizică este angajat/ă la o firmă, acesta își oferă serviciile în schimbul unei remunerații). Relația are cardinalitatea minimă 1:1 si cardinalitatea maximă M:1.

- SOFER_detine_PERMIS = relație ce leagă entitățile SOFER și PERMIS, reflectând legătura dintre acestea(un șofer pentru a se putea angaja are nevoie de un permis care să îi ofere dreptul de a conduce un camion, șoferul poate să aibă mai multe categorii de permise, în functie de necesitățile sale). Relația are cardinalitatea minimă 1:1 si cardinalitatea maximă 1:M.
- SOFER_istoric_popasuri_POPAS = relație ce leagă entitățile SOFER și POPAS, reflectând legătura dintre acestea(un șofer nu poate să conducă mai mult de 8h conform legii, motiv pentru care trebuie să mai ia pauze de odihnă, pauze pe care le poate lua la un motel sau în parcarea unui restaurant sau benzinării de unde poate să și mănânce și să se întâlnească cu alți șoferi). Relația are cardinalitatea minimă 0:0 si cardinalitatea maximă M:M.

5 Atribute

- Entitatea independentă LOCATIE are ca atribute:
  - id_locatie = variabilă de tip întreg, de lungime maximă 10, care reprezintă id-ul unei locații.
  - județ = variabilă de tip caracter, de lungime maximă 25, care reprezintă numele județului.
  - localitate = variabilă de tip caracter, de lungime maximă 25, care reprezintă numele localității.
  - stradă = variabilă de tip caracter, de lungime maximă 25, care reprezintă numele străzii.
  - nr = variabilă de tip întreg, de lungime maximă 5, care reprezintă numărul locației.
- Entitatea independentă DEPOZIT are ca atribute:
  - id_depozit = variabilă de tip întreg, de lungime maximă 5, care reprezintă id-ul unui depozit.
  - id_locatie = variabilă de tip întreg, de lungime maximă 10, care

reprezintă id-ul locației pentru depozit. Atributul trebuie să corespundă la o valoare a cheii primare din tabelul LOCATIE.

  - id_firma = variabilă de tip întreg, de lungime maximă 12, care reprezintă id-ul firmei care deține depozitul. Atributul trebuie să corespundă la o valoare a cheii primare din tabelul FIRMA.

  - id_echipa_paza = variabilă de tip întreg, de lungime maximă 5, care reprezintă id-ul echipei de pază al depozitului. Atributul trebuie să corespundă la o valoare a cheii primare din tabelul ECHIPA_PAZA.

- Entitatea independentă POPAS are ca atribute:

  - id_popas = variabilă de tip întreg, de lungime maximă 10, care reprezintă id-ul unui popas.

  - id_locatie = variabilă de tip întreg, de lungime maximă 10, care reprezintă id-ul locației pentru popas. Atributul trebuie să corespundă la o valoare a cheii primare din tabelul LOCATIE.

  - nume = variabilă de tip caracter, de lungime maximă 25, care reprezintă numele popasului.

  - tip_popas = variabilă de tip caracter, de lungime maximă 10. Atributul poate lua valorile: BENZINĂRIE, MOTEL, RESTAURANT.

- Entitatea independentă FIRMA are ca atribute:

  - id_firma = variabilă de tip întreg, de lungime maximă 12, care reprezintă id-ul firmei.

  - nume = variabilă de tip caracter, de lungime maximă 25, care reprezintă numele firmei, numele unei firme este unic.

  - tip_firmă = variabilă de tip caracter, de lungime maximă 10, care poate lua valorile: CLIENT, PAZA, TRANSPORT.

- Entitatea ECHIPA_PAZA are ca atribute:

  - id_echipa_paza = variabilă de tip întreg, de lungime maximă 5, care reprezintă id-ul unei echipe din cadrul firmei de pază.

  - id_firma = variabilă de tip întreg, de lungime maximă 12, care reprezintă id-ul firmei de pază. Atributul trebuie să corespundă la o valoare a cheii primare din tabelul PAZA.

- Entitatea INVENTAR_DEPOZIT are ca atribute:
  - id_inventar_depozit = variabilă de tip întreg de lungime maximă 10, care reprezintă id-ul unei intrări de marfă în inventar.
  - id_depozit = variabilă de tip întreg, de lungime maximă 5, care reprezintă id-ul depozitul în care se află marfa. Atributul trebuie să corespundă la o valoare a cheii primare din tabelul .
  - id_lot_marfa = variabilă de tip întreg, de lungime maximă 10, care reprezintă id-ul lotului de marfă care a intrat în inventar. Atributul trebuie să corespundă la o valoare a cheii primare din tabelul LOT_MARFA.
  - data_sosire = variabilă de tip data calendaristică, care reprezintă data la care a intrat marfa în inventariere.
  - ora_sosire = variabilă de tip ora, care reprezintă ora la care a intrat marfa în inventariere.
  - data_plecare = variabilă de tip data calendaristică, care reprezintă data la care a ieșit marfa din inventariere.
  - ora_plecare = variabilă de tip ora, care reprezintă ora la care a ieșit marfa din inventariere.
- Entitatea LOT_MARFA are ca atribute:
  - id_lot_marfa = variabilă de tip întreg, de lungime maximă 10, care reprezintă id-ul unui lot de marfă.
  - nume = variabilă de tip caracter, de lungime maximă 25, care reprezintă numele mărfii din lot.
  - cantitate = variabilă de tip real, de lungime maximă 10, care reprezintă cantitatea lotului de marfă.
- Entitatea INVENTAR_TRANSPORT are ca atribute:
  - id_inventar_transport = variabilă de tip întreg, de lungime maximă 10, care reprezintă id-ul unei intrări de inventar al unui transport.
  - id_lot_marfa = variabilă de tip întreg, de lungime maximă 10, care reprezintă id-ul unui lot de marfă care face parte din inventarul unui transport. Atributul trebuie să corespundă la o valoare a cheii primare din tabelul LOT_MARFA.

  - id_transport = variabilă de tip întreg, de lungime maximă 10, care reprezintă id-ul transportului din care face parte această listare. Atributul trebuie să corespundă la o valoare a cheii primare din tabelul TRANSPORT.

- Entitatea TRANSPORT are ca atribute:
  - id_transport = variabilă de tip întreg, de lungime maximă 10, care reprezintă id-ul unui transport de marfă între 2 depozite.
  - depozit_plecare = variabilă de tip întreg, de lungime maximă 5, care reprezintă id-ul depozitului de unde începe transportul și este preluată marfa. Atributul trebuie să corespundă la o valoare a cheii primare din tabelul DEPOZIT.
  - depozit_destinație = variabilă de tip întreg, de lungime maximă 5, care reprezintă id-ul depozit unde este livrată marfa. Atributul trebuie să corespundă la o valoare a cheii primare din tabelul DEPOZIT.
  - id_camion = variabilă de tip întreg, de lungime maximă 5, care reprezintă id-ul camionului folosit pentru transport. Atributul trebuie să corespundă la o valoare a cheii primare din tabelul CAMION.
  - data_plecare = variabilă de tip data calendaristică, care reprezintă data la care a plecat camionul din depozitul de plecare.
  - ora_plecare = variabilă de tip ora, care reprezintă ora la care a plecat camionul din depozitul de plecare.

- Entitatea CAMION are ca atribute:
  - id_camion = variabilă de tip întreg, de lungime maximă 5, care reprezintă id-ul unui camion.
  - id_firma = variabilă de tip întreg, de lungime maximă 12, care reprezintă id-ul firmei de transport care deține camionul. Atributul trebuie să corespundă la o valoare a cheii primare din tabelul TRANSPORT.
  - marca = variabila de tip caracter, de lungime maxima 10, care reprezinta firma care a produs camionul.
  - nr_inmatriculare = variabila de tip caracter, de lungime maxima 10, reprezinta un identificator unic pentru autovehicule. Acest

atribut poate fi schimbat (de obicei cand se se schimba detinatorul) motiv pentru care nu reprezinta parte din cheia primara.

- Entitatea ANGAJAT are ca atribute:
  - id_angajat = variabilă de tip întreg, de lungime maximă 10, care reprezintă id-ul unui angajat.
  - nume = variabilă de tip caracter, de lungime maximă 25, care reprezintă numele unui angajat.
  - prenume = variabilă de tip caracter, de lungime maximă 25, care reprezintă prenumele unui angajat.
  - nr_telefon = variabilă de tip caracter, de lungime maximă 14, care reprezintă numărul de telefon al unui angajat.
  - data_angajare = variabilă de tip data calendaristică, care reprezintă data în care a fost angajat o persoană.
  - salariu = variabilă de tip real, de lungime maximă 10, care reprezintă salariul anagajatului.
  - id_firma = variabile de tip întreg, de lungime maximă 12, care reprezintă id-ul firmei la care lucrează angajatul. Atributul trebuie să corespundă la o valoare a cheii primare din tabelul FIRMA.
  - tip_angajat = variabilă de tip caracter, de lungime maximă 6, care reprezintă tipul angajatului. Poate să fie PAZNIC sau SOFER.
- Entitatea PAZNIC are aceleași atribute ca supraentitatea din care face parte + atributul id_echipa_paza (= variabilă de tip întreg, de lungime maximă 5, care reprezintă id-ul echipei de pază din care face parte paznicul. Atributul trebuie să corespundă la o valoare a cheii primare din tabelul ECHIPA_PAZA).
- Entitatea SOFER are aceleași atribute ca supraentitatea din care face parte. Nu are alte atribute adiționale.
- Entitatea ISTORIC_CAMIOANE_CONDUSE are ca atribute:
  - id_istoric_camioane_conduse = variabilă de tip întreg, de lungime maximă 10, care reprezintă id-ul unei intrări în istoric.
  - id_angajat = variabilă de tip întreg, de lungime maximă 10, care reprezintă id-ul unui șofer. Atributul trebuie să corespundă la o valoare a cheii primare din tabelul SOFER.

  - id_camion = variabilă de tip întreg, de lungime maximă 5, care reprezintă id-ul camionului condus. Atributul trebuie să corespundă la o valoare a cheii primare din tabelul CAMION.
  - data_inceput = variabilă de tip data calendaristică, care reprezintă data la care a început un șofer să conducă un camion al firmei de transport.
  - data_sfarsit = variabilă de tip data calendaristică, care reprezintă data la care un șofer nu mai conduce un camion.

- Entitatea PERMIS are ca atribute:
  - id_permis = variabilă de tip întreg, de lungime maximă 10, care reprezintă id-ul unui permis.
  - id_angajat = variabilă de tip întreg, de lungime maximă 10, care reprezintă id-ul unui angajat care deține un permis. Atributul trebuie să corespundă la o valoare a cheii primare din tabelul SOFER.
  - categorie = variabilă de tip caracter, de lungime maximă 3, care reprezintă tipul de permis care poate fi deținut de o persoană. Atributul poate lua valorile AM, A1, A2, A, B1, B, BE, C1, C1E, C, CE, D1, D1E, D, DE, Tr, Tb sau Tv.
  - data = variabilă de tip data calendaristică, care reprezintă data la care a fost emis permisul pentru o persoană.
  - ora = variabilă de tip ora, care reprezintă ora la care a fost emis permisul pentru o persoană.

- Entitatea ISTORIC_POPASURI are ca atribute:
  - id_istoric_popas = variabilă de tip întreg, de lungime maximă 20, care reprezintă id-ul unui istoric în popas.
  - id_popas = variabilă de tip întreg, de lungime maximă 10, care reprezintă id-ul unui popas la care s-a oprit un șofer. Atributul trebuie să corespundă la o valoare a cheii primare din tabelul POPAS.
  - id_angajat = variabilă de tip întreg, de lungime maximă 10, care reprezintă id-ul unui șofer care s-a oprit la un popas. Atributul trebuie să corespundă la o valoare a cheii primare din tabelul SOFER.

- data_sosire = variabilă de tip data calendaristică, care reprezintă data la care a ajuns un șofer la un popas.
- ora_sosire = variabilă de tip ora, care reprezintă ora la care a ajuns un șofer la un popas.
- data_plecare = variabilă de tip data calendaristică, care reprezintă data la care a plecat un șofer de la un popas.
- ora_plecare = variabilă de tip ora, care reprezintă ora la care a plecat un șofer de la un popas.

Mai sunt de prezentat următoarele relații care se vor transforma in tabele asociative:

- Relația CAMION_transportă_INVENTAR_MARFĂ_de_la_DEPOZIT _la_DEPOZIT are ca atribute: id_transport, depozit_plecare, depozit_destinație, camion, data_plecare, ora_plecare.
- Relația SOFER_conduce_CAMION are ca atribute: id_istoric_camioane_conduse, id_angajat, id_camion, data_inceput, data_sfarsit.
- Relația SOFER_istoric_popasuri_POPAS are ca atribute: id_istoric_popas, id_popas, id_angajat, data_sosire, ora_sosire, data_plecare, ora_plecare.

6 Diagrama entitate-relație E/R

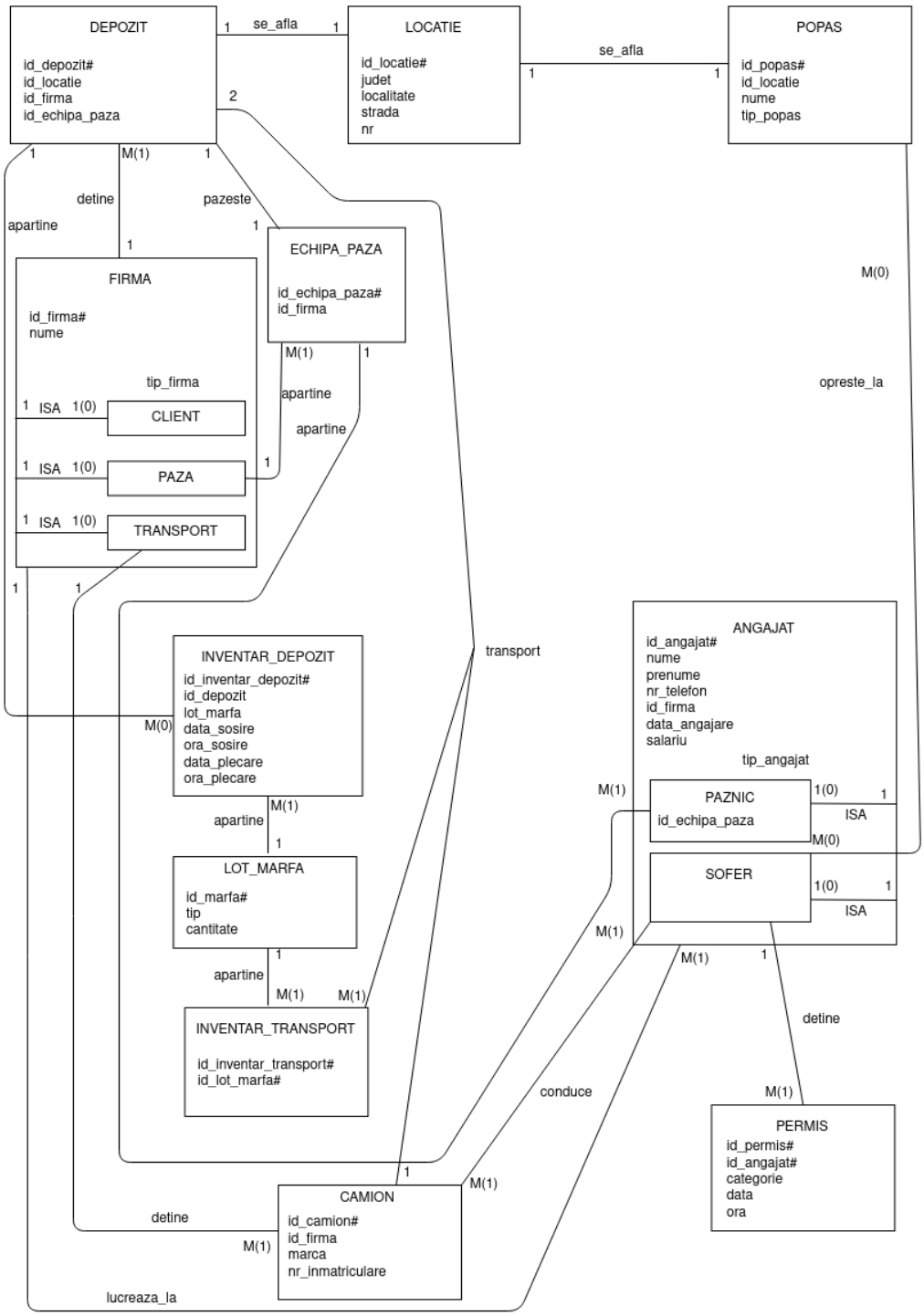


Figura 1

7 Diagrama conceptuală

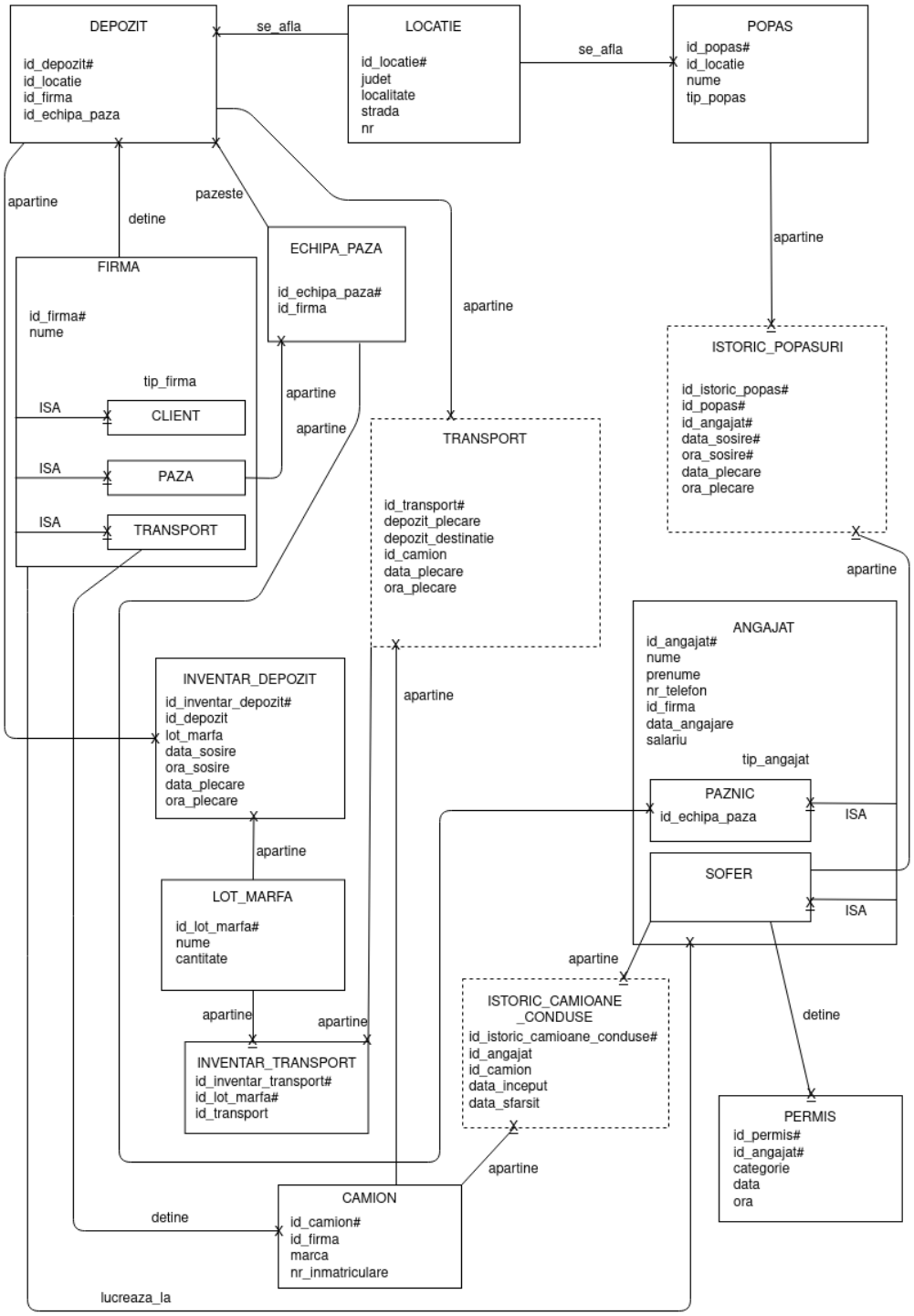


Figura 2

8 Schemele relaționale

Schemele relaționale corespunzătoare diagramei conceptuale din Figura 2 sunt următoarele:

- DEPOZIT(id_depozit#, id_locatie, id_echipa_paza, id_firma)
- LOCATIE(id_locatie#, județ, localitate, stradă, nr)
- POPAS(id_popas#, id_locatie, nume, tip_popas)
- FIRMA(id_firma#, nume, tip_firmă)
- CLIENT(id_firma#)
- PAZA(id_firma#)
- TRANSPORT(id_firma#)
- ECHIPA_PAZA(id_echipa_paza#, id_firma_paza)
- INVENTAR_DEPOZIT(id_inventar_depozit#, id_depozit, id_lot_marfa, data_sosire, ora_sosire, data_plecare, ora_plecare)
- INVENTAR_TRANSPORT(id_inventar_transport#, id_marfa#, id_transport)
- LOT_MARFA(id_lot_marfa#, nume, cantitate)
- CAMION(id_camion#, id_firma, marca, nr_inmatriculare)
- TRANSPORT(id_transport#, depozit_plecare, depozit_destinație, id_camion, data_plecare, ora_plecare)
- ANGAJAT(id_angajat#, nume, prenume, nr_telefon, data_angajare, salariu, id_firma, tip_angajat)
- PAZNIC(id_angajat#, id_echipa_paza)
- SOFER(id_angajat#)
- PERMIS(id_permis#, id_angajat#, categorie, data, ora)
- ISTORIC_CAMIOANE_CONDUSE(id_istoric_camioane_conduse#, id_angajat, id_camion, data_inceput, data_sfarsit)
- ISTORIC_POPASURI(id_istoric_popas#, id_popas#, id_angajat#, data_sosire#, ora_sosire#, data_plecare, ora_plecare)

9 Forme normale

Exemplu 1

Se dă relația *SOFER_detine_PERMIS* în care un șofer poate deține mai multe permise. S-a plecat de relația de tipul(obs. nu sunt trecute toate atributele unui angajat, doar câteva pentru a nu mări tabelul și a face toate datele cât mai lizibile și clare):

id_angajat#	nume	prenume	..	tip_angajat	id_permis#	categorie	data	ora
20000	Ion	Vasile	..	SOFER	1	C, B, A	06-06-2012, 06-06-2013, 06-06-2017	12:00, 12:00, 12:00
20020	Ovidie	Ion	..	SOFER	2	C	06-06-2018	12:00
20100	Constantin	Gheorghe	..	SOFER	3	B, C	06-06-2010, 06-06-2012	12:00, 12:00

Table 1: Relația nu e în FN1

în care se observă că atributelor *categorie*, *data* și *ora* le corespund o listă de valori, i.e. nu le corespund o valoare indivizibilă(atomică), de unde se trage concluzia că relația *SOFER_detine_PERMIS* nu se află în forma normala 1 (FN1). Se aduce relația în FN1 spărgând atributele nonatomice în atribute atomice, proces în care se adaugă mai multe linii pentru fiecare categorie de permis pentru un șofer după cum urmează:

id_angajat#	nume	prenume	..	tip_angajat	id_permis#	categorie	data	ora
20000	Ion	Vasile	..	SOFER	1	A	06-06-2012	12:00
20000	Ion	Vasile	..	SOFER	2	B	06-06-2013	12:00
20000	Ion	Vasile	..	SOFER	3	C	06-06-2017	12:00
20020	Ovidie	Ion	..	SOFER	1	C	06-06-2018	12:00
20100	Constantin	Gheorghe	..	SOFER	1	B	06-06-2010	12:00
20100	Constantin	Gheorghe	..	SOFER	2	C	06-06-2012	12:00

Table 2: Transformarea în FN1

Pentru ca relația *SOFER_detine_PERMIS* să fie în formă normală 2 (FN2) trebuie să fie mai întâi în FN1, lucru realizat anterior, și în plus trebuie să indeplinească condiția ca fiecare atribut care nu este cheie (nu participă la cheia primară) este **dependent de întreaga cheie primară**.

Relația de mai sus nu se află în FN2 întrucât se găsesc atributele *nume, prenume, nr_telefon, .., tip_angajat* care nu sunt chei și trebuie să depindă direct de întreaga cheie primară **id_angajat#** și **id_permis#**. Atributele nu depind direct de întreaga cheie primară, deoarece se observă dependența directă dintre *id_angajat#* și *nume, prenume, .., tip_angajat*, însemnând că

nume, prenume, .., tip_angajat depind direct doar de o parte a cheii primare(i.e. *id_angajat#* nu și de *id_permis#*).

Aplicăm regula *Casey-Delobel* pentru FN2:

Avem relația *R(K1, K2, X, Y)*, unde *K1=id_angajat#*, *K2=id_permis#* care definesc cheia primară compusă, iar X și Y sunt mulțimi de atribute, astfel încât $K1 \rightarrow X$, unde X=\{*nume, prenume, nr_telefon, .., tip_angajat*\} și Y=\{*categorie, data, ora*\}.

Din cauza dependenței funcționale $K1 \rightarrow X$ care arată că R nu este în FN2, se înlocuiește R(*fără pierdere de informație*) prin două proiecții: *(*K1, K2, Y) și *(*K1, X).

Astfel avem în final proiecțiile:

- {id_angajat#} → {nume, prenume, nr_telefon, data_angajare, id_firma, nume_firma, tip_angajat}, *id_angajat#* determinând funcțional atributele menționate anterior
- {id_permis#, id_angajat#} → {categorie, data, ora}

id_angajat#	id_permis#	categorie	data	ora
20000	1	A	06-06-2012	12:00
20000	2	B	06-06-2013	12:00
20000	3	C	06-06-2017	12:00
20020	1	C	06-06-2018	12:00
20100	2	B	06-06-2010	12:00
20100	1	C	06-06-2012	12:00

Table 3: Proiectia *R1(K1, K2, Y)*

id_angajat#	nume	prenume	..	tip_angajat
20000	Ion	Vasile	..	SOFER
20020	Ovidie	Ion	..	SOFER
20100	Constantin	Gheorghe	..	SOFER

Table 4: Proiectia *R2(K1, X)*

Pentru ca relația *SOFER_detine_PERMIS* să fie în formă normală 3 (FN3) trebuie să fie mai întâi în FN2, lucru realizat anterior, și în plus trebuie să îndeplinească condiția ca fiecare atribut care nu este cheie(nu participă la o cheie) **să depindă direct de cheia primară**.

În proiecția *R1(K1, K2, Y)* toate atributele depind direct de cheia primară. *Categoria* reprezintă tipul de autovehicul pe care permisul îi oferă dreptului unui șofer care l-a obținut să conducă autovehicule de categoria respectivă. Data și ora reprezintă când a fost obținut permisul de o persoană(de acum supranumită șofer). Pentru această relație nu există dependențe tranzitive, de unde rezultă că se află în FN3. Se va denumi această relație *SOFER_detine_PERMIS*.

Se verifică acum relația *R2(K1, X)*.

id_angajat#	nume	prenume	nr_telefon	..	id_firma	nume	tip_firma	tip_angajat
20000	Ion	Vasile	0722123456	..	1100	Road Logistics	TRANSPORT	SOFER
20020	Ovidie	Ion	0727124356	..	1100	Road Logistics	TRANSPORT	SOFER
20100	Constantin	Gheorghe	0735123456	..	1400	Lextom Trans Asd	TRANSPORT	SOFER

Table 5: Relația *R2(K1, X)*

Relația R2 este în FN2 după cum a fost demonstrat anterior, însă nu se află în FN3 întrucât se obvservă că atributele *nume(firmă), tip_firmă* depind tranzitiv de cheia primară *id_angajat* prin intermediul atributului *id_firma*.

Pentru a reduce relația în FN3 se aplică regula *Casey-Delobel*. Relația se descompune, prin eliminarea dependențelor funcționale tranzitive, în proiecții.

Avem relația *R(K, X_1, X_2, X_3)*, unde X_2={*nume(firma), tip_firma*} depinde tranzitiv de K=*id_angajat#*(i.e. cheia primara a lui R). Se presupune că $K \to X_1 \to X_2$. (X_1=id_firma si X_3={nume, prenume, nr_telefon, ..})

Din cauza dependenței funcționale $X_1 \to X_2$ care arată că R nu este în FN3, se înlocuiește R (**fără pierdere de informație**) prin două proiecții $R1(K, X_1, X_3)$ și $R2(X_1, X_2)$.

- {id_angajat#} → {nume, prenume, nr_telefon, data_angajare, salariu, id_firma, tip_angajat}
- {id_firma#} → {nume, tip_firma}

id_angajat#	nume	prenume	nr_telefon	..	id_firma	tip_angajat
20000	Ion	Vasile	0722123456	..	1100	SOFER
20020	Ovidie	Ion	0727124356	..	1100	SOFER
20100	Constantin	Gheorghe	0735123456	..	1400	SOFER

Table 6: Proiectia *R2-1(K, X_1, X_3)*

id_firma#	nume	tip_firma
1100	Road Logistics	TRANSPORT
1100	Road Logistics	TRANSPORT
1400	Lextom Trans Asd	TRANSPORT

Table 7: Proiectia *R2-2(K_1, K_2)*

Exemplu 2

Se dă relația *CAMION_transportă_INVENTAR_MARFĂ_de_la_DEPOZIT_la_DEPOZIT* în care prin intermediul unui camion este transportată marfă, marfă pentru care se face o inventariere a transportului de la un depozit la altul. Se pleacă de la prima variantă a relației:

id_ transport#	depozit_ plecare#	id_ locatie	id_ firma	id_echipa _paza	depozit_ destinatie#	id_ locatie	id_ firma	id_echipa _paza
10000	3000	100	4	1100	500	30	2	100
10010	3000	100	4	1100	500	30	2	100
10020	3500	120	10	1200	1000	50	3	200

id_ camion#	id_ firma	marca	nr_ inmatriculare	id_inventar _transport#	id_lot _marfa#	nume	cantitate	data_ plecare	ora_ plecare
15	1100	VOLVO	CT12FSD	100 155	100 105	carne miel carne porc	100 300	6-05-2021	08:00
25	1100	VOLVO	CT90MMM	210	110	carne pui	250	8-05-2021	06:00
35	1200	SCANIA	PH13SCN	265	120	rosii	1000	6-05-2021	08:00

Table 8: Relația nu e în FN1

în care se observă că atributelor *id_inventar_transport, id_lot_marfa, nume(marfa), cantitate* le corespund o listă de valori, i.e. nu le corespund o valoare indivizibilă(atomică), de unde se trage concluzia că relația *CAMION_transportă_INVENTAR_MARFĂ_de_la_DEPOZIT_la_DEPOZIT* nu se află în forma normală 1 (FN1). Se aduce relația în FN1 spărgând atributele nonatomice în atribute atomice, proces în care se adaugă mai multe linii pentru fiecare lot de marfă transportat și detaliile sale după cum urmează:

id_ transport#	depozit_ plecare#	id_ locatie(dp)	id_ firma(dp)	id_echipa _paza(dp)	depozit_ destinatie#	id_ locatie(dd)	id_ firma(dd)	id_echipa _paza(dd)
10000	3000	100	4	1100	500	30	2	100
10000	3000	100	4	1100	500	30	2	100
10010	3000	100	4	1100	500	30	2	100
10020	3500	120	10	1200	1000	50	3	200

id_ camion#	id_ firma(c)	marca	nr_ inmatriculare	id_inventar _transport#	id_lot _marfa#	nume	cantitate	data_ plecare	ora_ plecare
15	1100	VOLVO	CT12FSD	100	100	carne miel	100	6-05-2021	08:00
15	1100	VOLVO	CT12FSD	155	105	carne porc	300	6-05-2021	08:00
25	1100	VOLVO	CT90MMM	210	110	carne pui	250	8-05-2021	06:00
35	1200	SCANIA	PH13SCN	265	120	rosii	1000	6-05-2021	08:00

Table 9: Transformarea în FN1

Obs: Pentru atributele care se repetă s-a adăugat în paranteză la final un alias, a.î. să fie clar de care cheie depinde.

Pentru ca relația *CAMION_transportă_INVENTAR_MARFĂ_de_la_ DEPOZIT_la_DEPOZIT* să fie în formă normală 2 (FN2) trebuie să fie mai întâi în FN1, lucru realizat anterior, și în plus trebuie să indeplinească condiția ca fiecare atribut care nu este cheie (nu participă la cheia primară) este **dependent de întreaga cheie primară**.

Relația de mai sus nu se află în FN2 întrucât se găsesc atributele *id_firma(dp), id_firma(dd),id_firma(c), marca, nr_inmatriculare* care nu sunt chei și trebuie să depindă direct de întreaga cheie primară **id_transport# + depozit_plecare + depozit_destinatie + id_camion + id_inventar_ transport# + id_lot_marfa**. Atributele nu depind direct de întreaga cheie primară, deoarece se observă dependența directă dintre *id_camion* și *id_firma(c), marca, nr_inmatriculare* care nu sunt dependente și de *depozit_destinatie* (de exemplu).

Se mai găsesc următoarele dependențe funcționale:

- dependența directă dintre *depozit_plecare* și *id_locatie(dp), id_firma(dp), id_echipa_paza(dp)* care nu sunt dependente și de *depozit_destinație*.

- dependența directă dintre *depozit_destinatie* și *id_locatie(dd), id_firma(dd), id_echipa_paza(dd)* care nu sunt dependente și de *depozit_plecare*.

Atributele rămase care nu sunt chei *nume, cantitate, data_plecare,*

ora_plecare depind de întreaga cheie primară întrucât se consideră legătura pentru nume de exemplu:

- id_transport → ce marfă a fost transportată într-un transport.
- depozit_plecare → ce marfă a plecat din depozit.
- depozit_destinatie → ce marfă va fi transportată către depozit.
- id_camion → ce marfă va conține camionul.
- id_inventar_transport → ce marfă face parte din inventarierea transportului.

Analog pentru celelalte atribute.

Aplicăm regula *Casey-Delobel* pentru FN2:

Avem relația *R(K1, K2, K3, K4, K5, K6, X, Y, Z, W)*, unde
K1=id_transport#, *K2=depozit_plecare#*, *K3=depozit_destinatie#*,
K4=id_camion#, *K5=id_inventar_transport#*, *K6=id_lot_marfa#*
care definesc cheia primară compusă, iar *X*, *Y*, *Z* și *W* sunt mulțimi de atribute, astfel încât avem dependențele directe:

- *K2*→*X*, unde *X*={*id_locatie(dp), id_firma(dp), id_echipa_paza(dp)*}
- *K3*→*Y*, unde *Y*={*id_locatie(dd), id_firma(dd), id_echipa_paza(dd)*}
- *K4*→*Z*, unde *Z*={*id_firma(c), marca, nr_inmatriculare*}

În plus avem lista de atribute *W*={*nume, cantitate, data_plecare, ora_plecare*}.

Din cauza dependențelor funcționale *K2*→*X*, *K3*→*Y* și *K4*→*Z* care arată că *R* nu este în FN2, se înlocuiește R(*fără pierdere de informație*) prin 4 proiecții: (*K1, K2, K2, K4, K5, K6, Z*), (*K2, X*), (*K3, Y*), (*K4, Z*).

Astfel avem în final proiecțiile:

- {id_transport#, depozit_plecare#, depozit_destinatie#, id_camion#, id_inventar_transport#, id_lot_marfa#} → {nume, cantitate,

data_plecare, ora_plecare}

- {depozit_plecare#} → {id_locatie, id_firma, id_echipa_paza}
- {depozit_destinatie#} → {id_locatie, id_firma, id_echipa_paza}
- {id_camion#} → {id_firma, marca, nr_inmatriculare}

id_ transport#	depozit_ plecare#	depozit_ destinatie#	id_ camion#	id_inventar_ transport#	id_lot_ marfa#	nume	cantitate	data_ plecare	ora_ plecare
10000	3000	500	15	100	100	carne miel	100	6-05-2021	08:00
10000	3000	500	15	155	105	carne porc	300	6-05-2021	08:00
10010	3000	500	25	210	110	carne pui	250	8-05-2021	06:00
10020	3500	1000	35	265	115	rosii	1000	6-05-2021	08:00

Table 10: Proiectia *R1(*K_1*,* K_2*,* K_3*,* K_4*,* K_5*,* K_6*,* W*)*

depozit_plecare#	id_locatie	id_firma	id_echipa_paza
3000	100	4	1100
3500	120	10	1200

Table 11: Proiectia *R2(*K_2*,* X*)*

depozit_destinatie#	id_locatie	id_firma	id_echipa_paza
500	30	2	100
1000	50	3	200

Table 12: Proiectia *R3(*K_3*,* Y*)*

id_camion	id_firma	marca	nr_inmatriculare
15	1100	VOLVO	CT12FSD
25	1100	VOLVO	CT90MMM
35	1200	SCANIA	PH13SCN

Table 13: Proiectia *R4(*K_4*,* Z*)*

Pentru relația *transport* se observă ca a ajuns la 6 chei primare, lucru ineficient în practică, fiind o decizie teribilă, motiv pentru care se înlocuiește cheia primară compusă cu o cheie primară simplă formată doar din *id_transport*.

Relația anterioară se află în forma normală 3 (FN3) daca se află întâi în FN2, lucru realizat anterior, și în plus trebuie să îndeplinească condiția ca fiecare atribut care nu este cheie(nu participă la o cheie) **să depindă direct de cheia primară**.

Relațiile 2-4 prezentate anterior prin proiecții se află deja în FN3. Se verifică acum relația 1: aceasta este în FN2, transformare realizată anterior, dar nu este în FN3, deoarece se observă că atributele *nume* și *cantitate* depind tranzitiv de cheia primară *id_transport* prin intermediul atributului *id_lot_marfa*, atribut care depinde și el tranzitiv prin intermediul atributului *id_inventar_transport*.

Pentru a reduce relația în FN3 se aplică regula *Casey-Delobel*. Relația se descompune, prin eliminarea dependențelor funcționale tranzitive, în proiecții.

Avem relația *R(K, X_1, X_2, X_3, X_4)*, unde K este cheia primară id_transport, X_1=*id_inventar_transport*, X_2=*id_lot_marfa*, X_3={*nume, cantitate*} și X_4={*depozit_plecare, depozit_destinatie, data_plecare, ora_plecare*}. Sunt evidente dependențele tranzitive $K \to X_1 \to X_2$ și $X_1 \to X_2 \to X_3$.

Din cauza acestor dependențe funcționale relația nu se află în FN3, motiv pentru care se înlocuiește R(**fără pierdere de informație**) prin 3 proiecții $R1(K, X_4)$, $R2(X_1, X_2, K)$ și $R3(X_2, X_3)$. În creearea proiecțiilor s-a ținut cont și de cardinalitățile dintre entități.

- {id_transport#} → {depozit_plecare, depozit_destinatie, id_camion, data_plecare, ora_plecare}
- {id_inventar_transport#, id_lot_marfa#} → {id_transport}
- {id_lot_marfa#} → {nume, cantitate}

id_ transport#	depozit_ plecare#	depozit_ destinatie#	id_ camion#	id_inventar_ transport	id_lot_ marfa#	data_ plecare	ora_ plecare
10000	3000	500	15	100	100	6-05-2021	08:00
10000	3000	500	15	155	105	6-05-2021	08:00
10010	3000	500	25	210	110	8-05-2021	06:00
10020	3500	1000	35	265	115	6-05-2021	08:00

Table 14: Proiectia *R1-1(K, X_4)*

id_inventar_transport#	id_lot_marfa	id_transport
100	100	10000
155	105	10000
210	110	10010
265	115	10020

Table 15: Proiectia *R1-2(X_1, X_2, K)*

id_lot_marfa#	nume	cantitate
100	carne miel	100
105	carne porc	300
110	carne pui	250
115	rosii	1000

Table 16: Proiectia *R1-3(X_2, X_3)*

Obs: Datele din următoarele tabele/exemple sunt fictive, nu sunt prezente în baza de date creată. S-a ales această metodă pentru o exemplificare mai bună per exemplu. De asemenea vom avea convenția de a adăuga inainte de fiecare cheie primară un string care să semnifice pentru ce entitate este desemnată cheia primară.

Se consideră că o relație R se află în forma normală **Boyce-Codd** dacă și numai dacă fiecare determinant este o cheie candidat. Un determinant reprezintă un atribut sau o mulțime de atribute neredundante, care constituie un identificator unic pentru un alt atribut sau o altă mulțime de date.

Se dă relația *SOFER_CONDUCE_CAMION* în care un șofer poate conduce mai multe camioane care o preluăm din forma normală 1.

id_istoric camioane_conduse#	id_ angajat#	id_ camion#	id_ firma	marca	nr_ inmatriculare	data_ inceput	data_ sfarsit
ICC1	A1	C1	F1	DAF	PX13ASD	06-06-2019	07-07-2019
ICC2	A2	C2	F1	DAF	PX21QWE	12-04-2019	11-11-2019
ICC3	A3	C3	F2	MAN	PX14HGA	06-05-2019	01-12-2019
ICC3	A4	C4	F3	DAF	PX13ICC	06-01-2019	07-02-2019

Table 17: Relatia *SOFER_CONDUE_CAMION*

În relația anterioară se găsește determinantul *X=nr_inmatriculare* care îl determină funcțional pe *K3=id_camion#* și *Y={id_firma, marca}*, *X* fiind cheie candidat. Astfel aplicăm regula Casey Delobel pentru *R(K1#, K2#, K3#, X, Y, Z)*, unde *K1=id_istoric_camioane_conduse#*, *K2=id_angajat#*,

Z={data_inceput, data_sfarsit}. Astfel se ajunge la proiecțiile: *R1(K1#, K2#, X, Z)* și *R2(X#, K3)* care au adus relația în *BCNF*.

id_istoric camioane_conduse#	id_ angajat#	nr_ inmatriculare	data_ inceput	data_ sfarsit
ICC1	A1	PX13ASD	06-06-2019	07-07-2019
ICC2	A2	PX21QWE	12-04-2019	11-11-2019
ICC3	A3	PX14HGA	06-05-2019	01-12-2019
ICC3	A4	PX13ICC	06-01-2019	07-02-2019

Table 18: R1(K1#, K2#, X, Z)

nr_ inmatriculare#	id_ camion	marca	id_ firma
PX13ASD	C1	DAF	F1
PX21QWE	C2	DAF	F1
PX14HGA	C3	MAN	F2
PX13ICC	C4	DAF	F3

Table 19: R2(X#, K3)

Analog se procedează și pentru relația *transport*. (În exemplu se vor trece doar atributele de folos pentru exemplul dat)

id_depozit _plecare#	id_depozit _destinatie#	id_lot _marfa#	id_ camion#	nr_ inmatriculare
D1	D5	LM1	C1	PX13ASD
D1	D6	LM2	C1	PX13ASD
D2	D5	LM1	C1	PX13ASD
D2	D6	LM2	C1	PX13ASD
D1	D3	LM5	C3	PX14HGA
D1	D3	LM4	C4	PX13ICC

Table 20: Relația *transport*

Se găsește analog determinantul *X=nr_inmatriculare* care determină funcțional pe *K4=id_camion#*. Se mai fac notațiile *K1=id_depozit_plecare#*, *K2=id_depozit_destinatie#*, *K3=id_lot_marfa#*. Astfel se ajunge la proiecțiile *R1(K1#, K2#, K3#, X)* și *R2(X#, K4)*.

id_depozit _plecare#	id_depozit _destinatie#	id_lot _marfa#	nr_ inmatriculare
D1	D5	LM1	PX13ASD
D1	D6	LM2	PX13ASD
D2	D5	LM1	PX13ASD
D2	D6	LM2	PX13ASD
D1	D3	LM5	PX14HGA
D1	D3	LM4	PX13ICC

Table 21: Relația *R1(K1#, K2#, K3#, X)*

nr_ inmatriculare#	id_ camion
PX13ASD	C1
PX13ASD	C1
PX13ASD	C1
PX13ASD	C1
PX14HGA	C3
PX13ICC	C4

Table 22: Relația *R2(X#, K4)*

O relație se află în formă normală 4 (FN4) dacă și numai dacă relația este în BCNF și nu conține relații m:n independente.

Se dă relația anterior transformată(*transport*) în BCNF *R(id_depozit_ destinatie#, id_depozit_plecare#, id_lot_marfa#, nr_inmatriculare#)*

id_depozit _plecare#	id_depozit _destinatie#	id_lot _marfa#	nr_ inmatriculare#
D1	D5	LM1	PX13ASD
D1	D6	LM2	PX13ASD
D2	D5	LM1	PX13ASD
D2	D6	LM2	PX13ASD

Table 23: Relația *transport*

în care se găsesc următoarele dependențe multiple:

- nr_inmatriculare→→id_depozit_plecare
- nr_inmatriculare→→id_depozit_destinatie

- nr_inmatriculare$\rightarrow\rightarrow$id_lot_marfa

și se obțin următoarele rezultate după aplicarea lui FN4:

- *transport1=R1(nr_inmatriculare#, id_depozit_plecare)*
- *transport2=R2(nr_inmatriculare#, id_depozit_destinatie)*
- *transport3=R3(nr_inmatriculare#, id_lot_marfa)*

cu următoarele:

nr_ inmatriculare#	id_depozit _plecare#
PX13ASD	D1
PX13ASD	D2

Table 24: Relația *transport1*

nr_ inmatriculare#	id_depozit _destinatie#
PX13ASD	D5
PX13ASD	D6

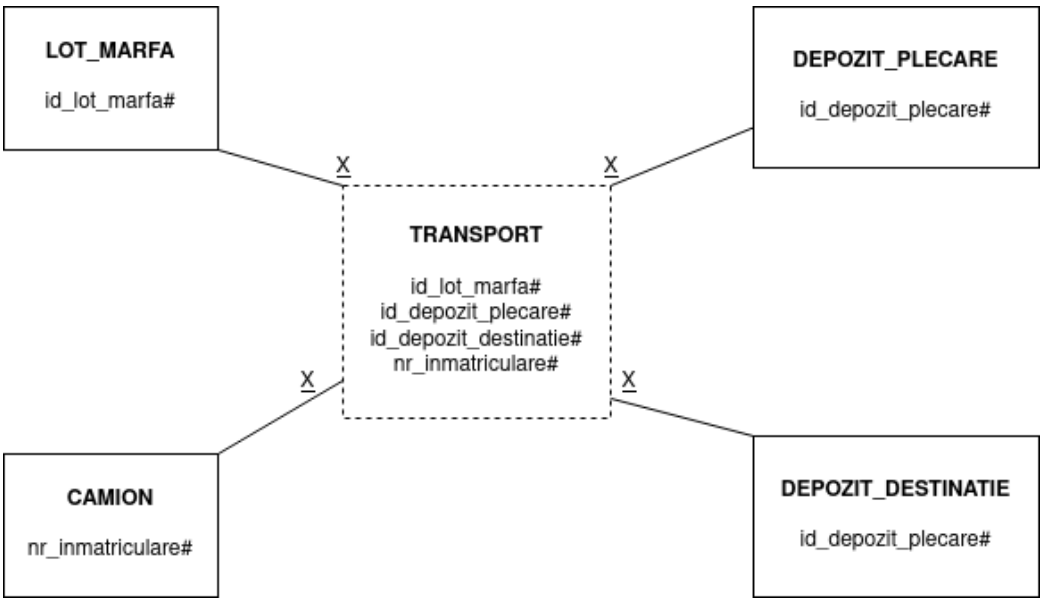
Table 25: Relația *transport2*

nr_ inmatriculare#	id_lot_ marfa#
PX13ASD	LM1
PX13ASD	LM2

Table 26: Relația *transport3*

O relație se află în formă normală 5 (FN5) dacă și numai dacă se află în FN4 și nu conține dependențe ciclice.

Se dă exemplul următor: un lot de marfă poate fi transportat de la mai multe depozite la mai multe depozite și poate să fie transportat de mai multe camioane. Aceasta relație este una de tip 3 și are reprezentarea:



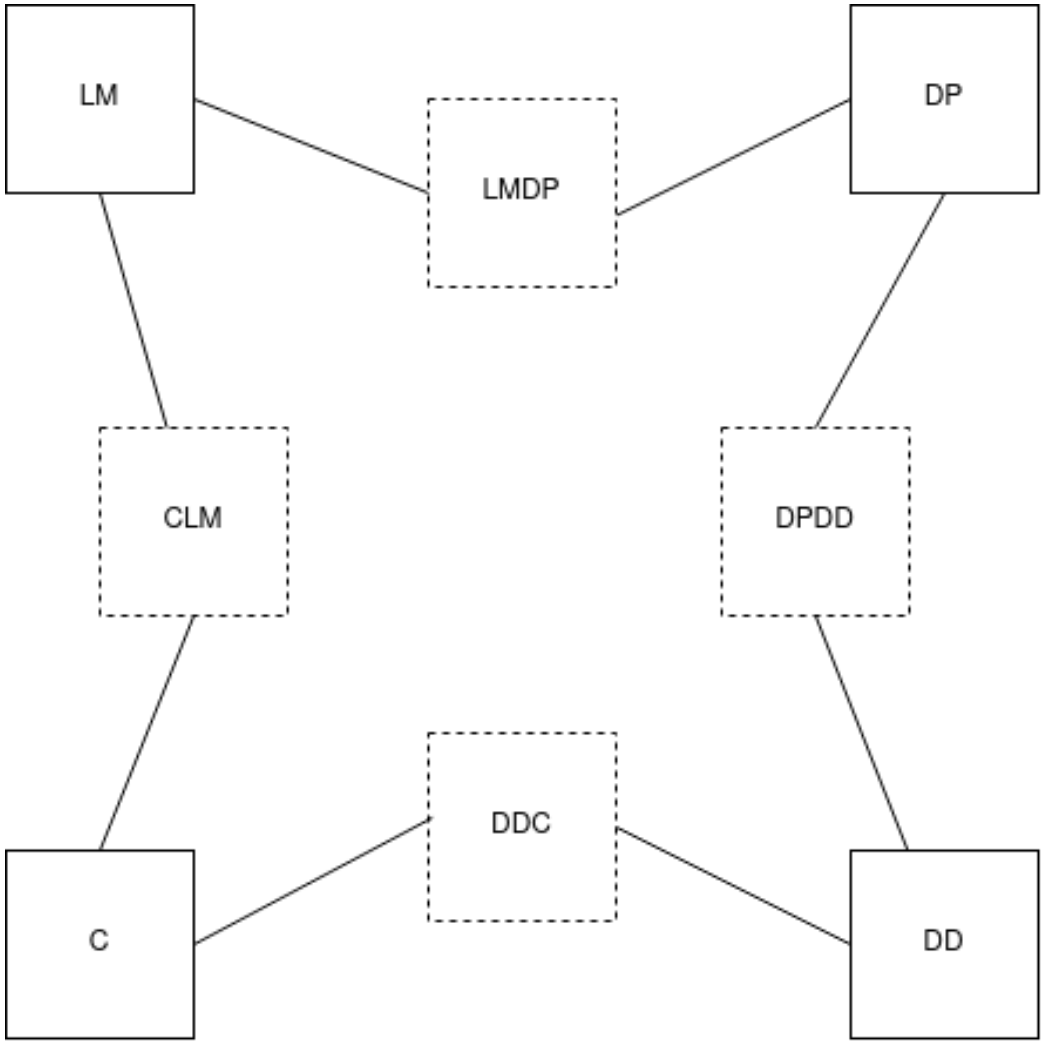
Relatie tip 3

cu posibilele întregistrări:

id_depozit _plecare#	id_depozit _destinatie#	id_lot _marfa#	nr_ inmatriculare#
D1	D5	LM1	PX13ASD
D1	D6	LM2	PX13ASD
D2	D5	LM1	PX13ASD
D2	D6	LM2	PX13ASD

Table 27: Relația *transport*

Această relație de tip 3 poate fi echivalentă cu 4 relații de tip 2 doar dacă aceste relații de tip 2 sunt ciclice.



Relatie tip 2

Relația fiind ciclică, atunci când se va face o interogare cu toate join-urile se va obține un rezultat echivalent cu cel obținut din interogarea relației de tip 3. Anterior au fost menționate condițiile ca o relație să fie în FN5, printre care și condiția de a nu conține dependențe ciclice, motiv pentru care cele 3 relații de tip 2 compun o diagramă care conține dependențe ciclice, deci relația nu se află în FN5. În schimb relația de tip 3 se află în FN5.

10 Denormalizare

Denormalizarea presupune mărirea redundanței pentru a reduce numărul de join-uri care trebuie efectuate pentru rezolvarea unei interogări, astfel se

micșorează timpul de execuție.

→Exemplu când denormalizarea nu este utilă:

În tabelul *LOT_MARFA* în care sunt stocate loturile de marfă din transporturile efectuate. Loturile de marfă au atributele **nume** și **cantitate** care pot conține valori repetitive, aceeași cantitate putând aparținând mai multor loturi de marfă, analog pentru nume. Dacă în baza de date există un tabel separat pentru cantitate și unul pentru nume, fiecare împreună cu id-ul lotului de marfă căruia îi corespunde, este necesar să se aplice procesul de denormalizare în urma căruia atributele nume și cantitate se vor plasa în tabelul *LOT_MARFA* întrucât ar fi ineficient ca în interogări aceste atribute să se afle în tabele separate datorită unui dublu join care ar mări timpul de execuție în cazul în care se dorește aflarea tuturor informațiilor despre un anumit lot de marfă.

→Exemplu când denormalizarea este utilă: În tabelul *ANGAJAT* există atributul **id_firma** care reprezintă o cheie externă pentru o cheie primară din tabelul *FIRMA*, acest atribut desemnând la ce firmă lucrează un angajat. Cu toate acestea se poate afla la ce firmă lucrează un angajat dacă se realizează un join între *ANGAJAT*, *ISTORIC_CAMIOANE_CONDUSE*, *CAMION* și *FIRMA* pentru un angajat de tip șofer și un join între *ANGAJAT*, *ECHIPA_PAZA* și *FIRMA*. Este deja evident că pentru aflarea acestei informații (la ce firmă lucreaza un angajat/ce angajați lucreaza la o firmă) o interogare ar avea un număr de join-uri prea mare pentru ce informație se dorește. Astfel se realizează denormalizarea acestui tabel prin adăugarea atributului **id_firma** pentru a reduce nr. de join-uri din posibilele interogări dorite, lucru care rezultă într-un timp de execuție mai mic. Astfel această denormalizare se dovedește utilă.

11 Optimizarea unei cereri

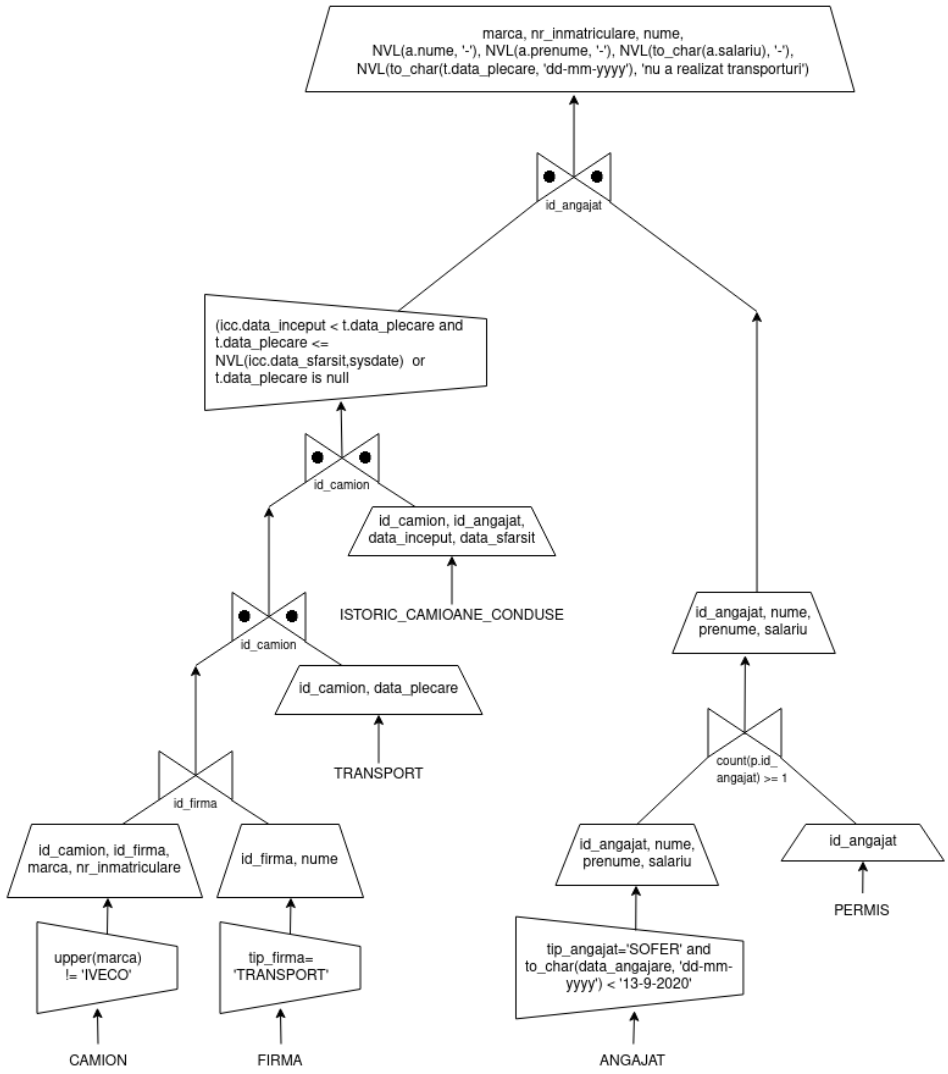
Se cer următoarele informații:

Pentru fiecare camion să se afișeze marca, nr. de înmatriculare, numele firmei, de ce șoferi a fost condus(pentru fiecare șofer să se precizeze numele, prenumele și salariul, în cazul în care un camion nu a fost condus de un șofer să se înlocuiască aceste coloane cu '-'), dar și data de plecare pentru fiecare transport realizat de șofer cu acest camion. Dacă unui angajat i-a fost alocat un camion dar nu a realizat niciun transport in loc de dată să fie afișat mesajul 'nu a realizat transporturi'. Să se afișeze informațiile despre angajați doar dacă sunt șoferi, au fost angajați înainte de data de 13 sept 2020 si dețin cel putin un permis auto, în cazul în care un camion a fost condus de un șofer care nu îndeplinește aceste cerințe se vor înlocui coloanele cu informații despre angajat cu '-'. Să se excludă camioanele de marcă IVECO.

Prima cerere *SQL* implementată:

```
01 | WITH ang AS -- soferii angajati inainte de 13 sept 2020 care detin cel mult
   |     ↪ 2 permise auto
02 |     (SELECT p.id_angajat, a.nume, a.prenume, a.salariu
03 |      FROM angajat a JOIN permis p ON (a.id_angajat = p.id_angajat)
04 |      WHERE a.tip_angajat = 'SOFER'
05 |         and to_char(a.data_angajare, 'dd-mm-yyyy') < '13-9-2020'
06 |      GROUP BY p.id_angajat, a.nume, a.prenume, a.salariu
07 |      HAVING count(p.id_angajat) >= 1  -- care detine cel putin 1 permis
08 |      )
09 |
10 | SELECT c.marca, c.nr_inmatriculare, f.nume,
11 |        NVL(a.nume, '-') "Nume sofer", NVL(a.prenume, '-') "Prenume sofer",
12 |        NVL(to_char(a.salariu), '-') "Salariu sofer",
13 |        NVL(to_char(t.data_plecare, 'dd-mm-yyyy'), 'nu a realizat
   |     ↪ transporturi') "Data transport"
14 | FROM camion c JOIN firma f ON (c.id_firma = f.id_firma)
15 |               FULL OUTER JOIN istoric_camioane_conduse icc ON (c.id_camion =
   |     ↪  icc.id_camion)
16 |               FULL OUTER JOIN ang a ON (icc.id_angajat = a.id_angajat)
17 |               FULL OUTER JOIN transport t ON (c.id_camion = t.id_camion)
18 | WHERE
19 |     upper(c.marca) != 'IVECO' and
20 |     f.tip_firma = 'TRANSPORT' and
21 |     ((icc.data_inceput < t.data_plecare and t.data_plecare <= NVL(icc.
   |     ↪ data_sfarsit, sysdate)
22 |      )   or t.data_plecare is null
23 |     );
```

S-a realizat următorul arbore algebric asociat cererii:



Arbore v1

și expresia algebrică asociată:

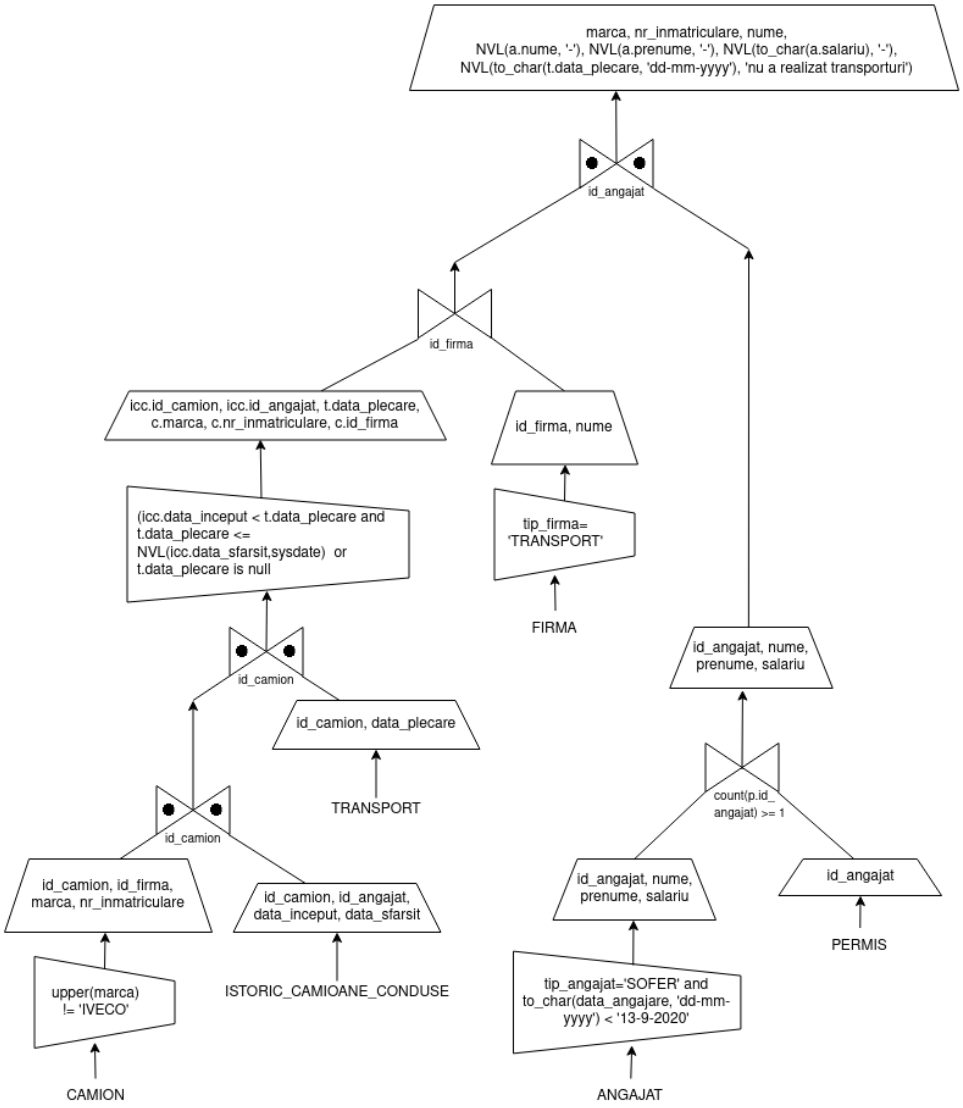
```
R1=SELECT(CAMION, upper(marca)!='IVECO')
R2=PROJECT(R1, id_camion, id_firma, marca, nr_inmatriculare}
R3=SELECT(FIRMA, tip_firma='TRANSPORT')
R4=PROJECT(R3, id_firma, nume)
R5=JOIN(R2, R4, id_firma)
R6=PROJECT(TRANSPORT, id_camion, data_plecare)
R7=FULL OUTER JOIN(R5, R6, id_camion)
R8=PROJECT(ISTORIC_CAMIOANE_CONDUSE, id_camion, id_angajat, data_inceput,
    ↪ data_sfarsit)
R9=FULL OUTER JOIN(R7, R8, id_camion)
R10=SELECT(R9, (icc.data_inceput < t.data_plecare and t.data_plecare <= NVL(
    ↪ icc.data_sfarsit, sysdate)) or t.data_plecare is null)
R11=SELECT(ANGAJAT, tip_angajat = 'SOFER' and to_char(data_angajare, 'dd-mm-
    ↪ yyyy') < '13-9-2020')
R12=PROJECT(R11, id_angajat, nume, prenume, salariu)
R13=PROJECT(PERMIS, id_angajat)
R14=JOIN(R12, R13, count(p.id_angajat) >= 1)
R15=PROJECT(R14, id_angajat, nume, prenume, salariu)
R16=FULL OUTER JOIN(R10, R15, id_angajat)
Rez=R17=PROJECT(R16, marca, nr_inmatriculare, nume, NVL(nume, '-'), NVL(
    ↪ prenume, '-'), NVL(to_char(salariu), '-'), NVL(to_char(t.data_plecare
    ↪ , 'dd-mm-yyyy'), 'nu a realizat transporturi'))
```

Se observă faptul că deja o parte a arborelui este optimizată folosind proprietatea de compunere a proiecțiilor și de compunere a selecțiilor.

Se fac următoarele optimizări:

- se folosește regula de optimizare 1 care spune că selecțiile se execută cât mai devreme posibil pentru R10, selecție care ar putea fi realizată mai devreme, totodată se va folosi și proprietatea 2 care spune că operațiile de join și produs cartezian sunt asociative. În alegerea primelor join-uri vom folosi și regula de optimizare 3 care spune că dacă există mai multe join-uri atunci cel care se execută primul este cel mai restrictiv. Astfel se va face un *FULL OUTER JOIN* între *CAMION* și *ISTORIC_CAMIOANE_CONDUSE* după un alt *FULL OUTER JOIN* între rezultatul anterior și *TRANSPORT* după care vom face selecția care anterior era R10 și o proiecție pentru a elimina atributele nefolositoare.

Și se ajunge la următorul arborele algebric:



Arbore v2 optimizat

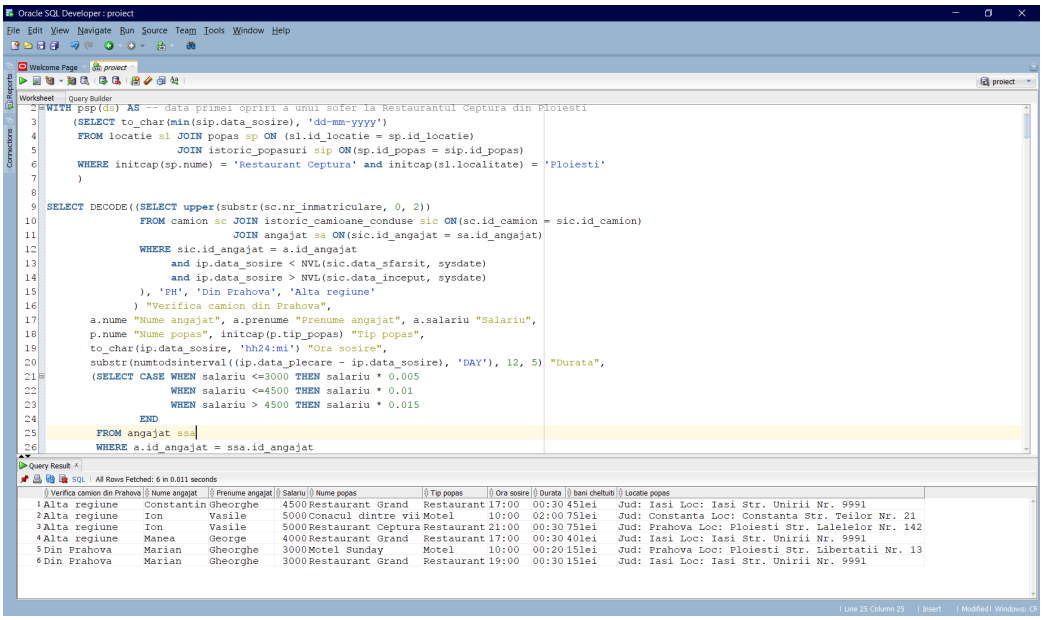
și expresia algebrică asociată:

```
R1=SELECT(CAMION, upper(marca)!='IVECO')
R2=PROJECT(R1, id_camion, id_firma, marca, nr_inmatriculare}
R3=PROJECT(ISTORIC_CAMIOANE_CONDUSE, id_camion, id_angajat, data_inceput,
    ↪ data_sfarsit)
R4=FULL OUTER JOIN(R2, R3, id_camion)
R5=PROJECT(TRANSPORT, id_camion, data_plecare)
R6=FULL OUTER JOIN(R4, R5, id_camion)
R7=SELECT(R6, (icc.data_inceput < t.data_plecare and t.data_plecare <= NVL(
    ↪ icc.data_sfarsit, sysdate)) or t.data_plecare is null)
R8=PROJECT(R7, icc.id_camion, icc.id_angajat, t.data_plecare, c.marca, c.
    ↪ nr_inmatriculare, c.id_firma)
R9=SELECT(FIRMA, tip_firma='TRANSPORT')
R10=PROJECT(R9, id_firma, nume)
R11=JOIN(R8, R10, id_firma)
R12=SELECT(ANGAJAT, tip_angajat = 'SOFER' and to_char(data_angajare, 'dd-mm-
    ↪ yyyy') < '13-9-2020')
R13=PROJECT(R12, id_angajat, nume, prenume, salariu)
R14=PROJECT(PERMIS, id_angajat)
R15=JOIN(R13, R14, count(p.id_angajat) >= 1)
R16=PROJECT(R15, id_angajat, nume, prenume, salariu)
R17=FULL OUTER JOIN(R11, R16, id_angajat)
Rez=R18=PROJECT(R17, marca, nr_inmatriculare, nume, NVL(nume, '-'), NVL(
    ↪ prenume, '-'), NVL(to_char(salariu), '-'), NVL(to_char(t.data_plecare
    ↪ , 'dd-mm-yyyy'), 'nu a realizat transporturi'))
```

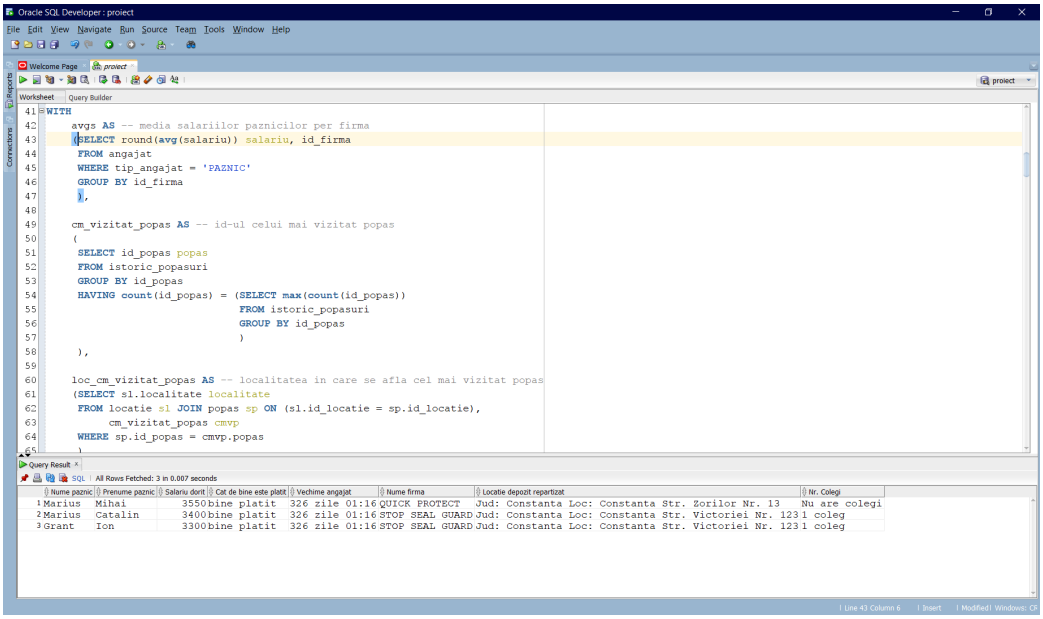
Cererea optimizată finală este:

```
WITH ang AS -- soferii angajati inainte de 13 sept 2020 care detin cel mult
    ↪ 2 permise auto
    (SELECT p.id_angajat, a.nume, a.prenume, a.salariu
     FROM angajat a JOIN permis p ON (a.id_angajat = p.id_angajat)
     WHERE a.tip_angajat = 'SOFER'
        and to_char(a.data_angajare, 'dd-mm-yyyy') < '13-9-2020'
     GROUP BY p.id_angajat, a.nume, a.prenume, a.salariu
     HAVING count(p.id_angajat) >= 1  -- care detine cel putin 1 permis
     ),
     tc AS
     (SELECT icc.id_camion, icc.id_angajat, t.data_plecare, c.marca, c.
    ↪ nr_inmatriculare, c.id_firma
      FROM camion c FULL OUTER JOIN istoric_camioane_conduse icc ON (c.
    ↪ id_camion = icc.id_camion)
                    FULL OUTER JOIN transport t ON (c.id_camion = t.
    ↪ id_camion)
      WHERE c.marca != 'IVECO' and
            (t.data_plecare is null or icc.data_inceput is null or
             (icc.data_inceput < t.data_plecare and t.data_plecare <= NVL(
    ↪ icc.data_sfarsit, sysdate))
            )
      )

SELECT tc.marca, tc.nr_inmatriculare, f.nume,
       NVL(a.nume, '-') "Nume sofer", NVL(a.prenume, '-') "Prenume sofer",
       NVL(to_char(a.salariu), '-') "Salariu sofer",
       NVL(to_char(tc.data_plecare, 'dd-mm-yyyy'), 'nu a realizat
    ↪ transporturi') "Data transport"
FROM tc JOIN firma f ON (tc.id_firma = f.id_firma)
        FULL OUTER JOIN ang a ON (tc.id_angajat = a.id_angajat)
WHERE f.tip_firma = 'TRANSPORT';
```

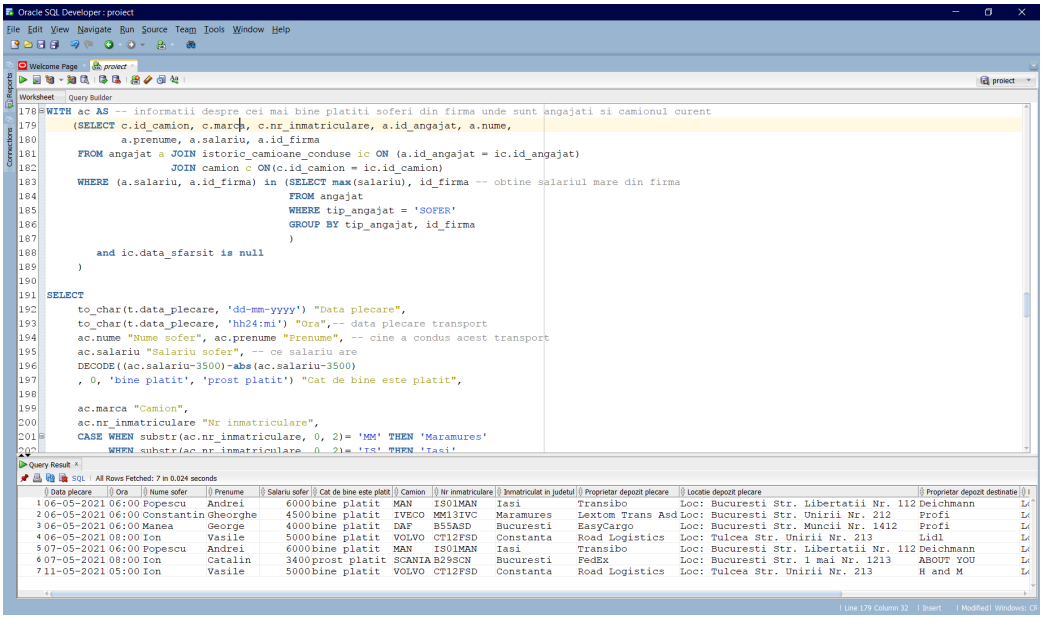
12 Anexe



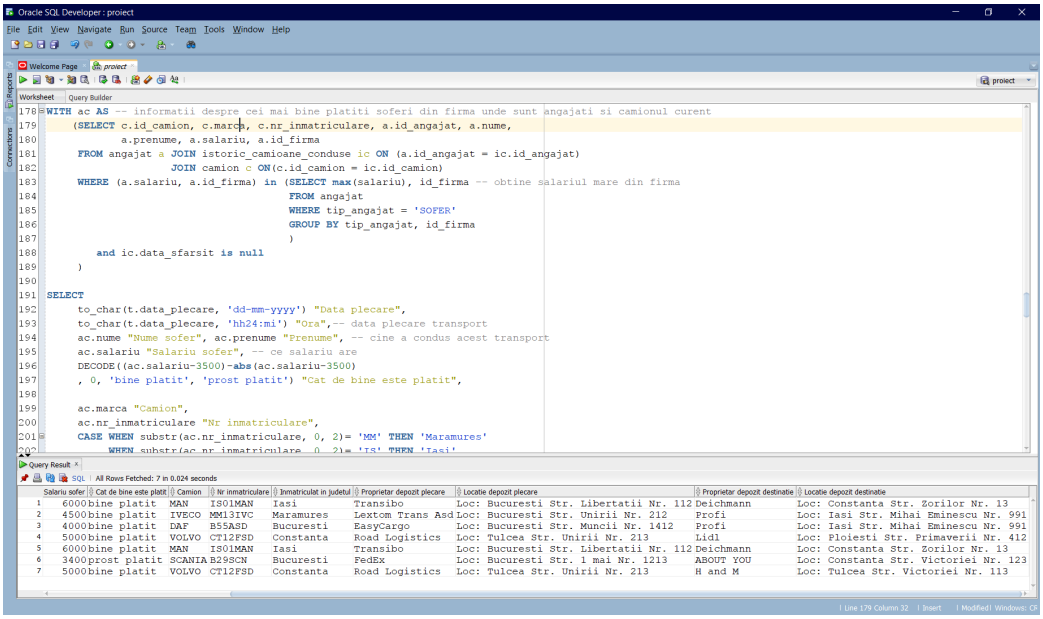
Ex. 11 - 1



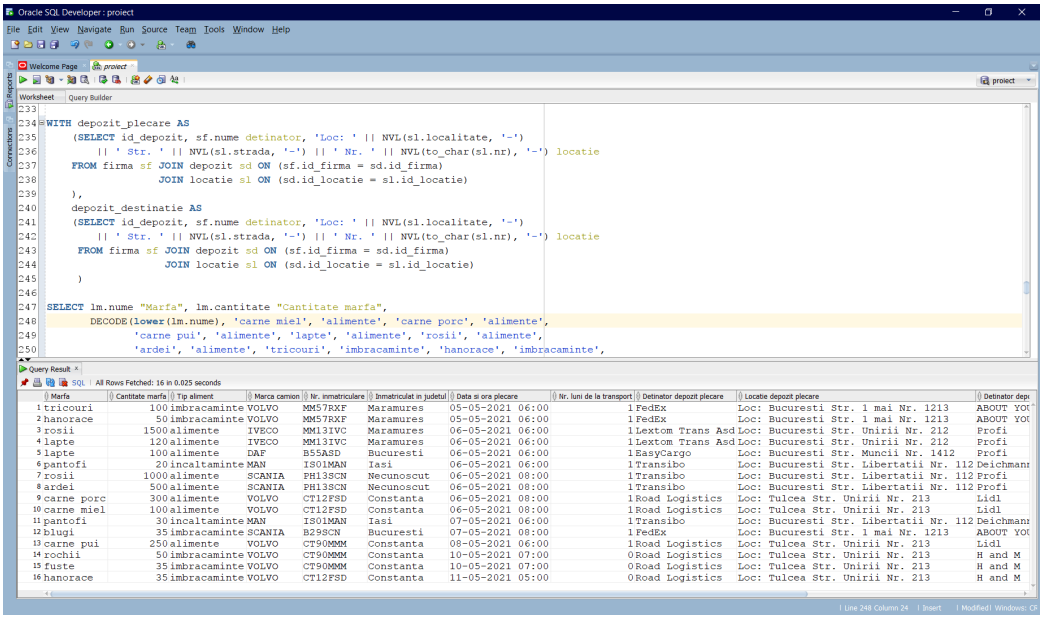
Ex. 11 - 2



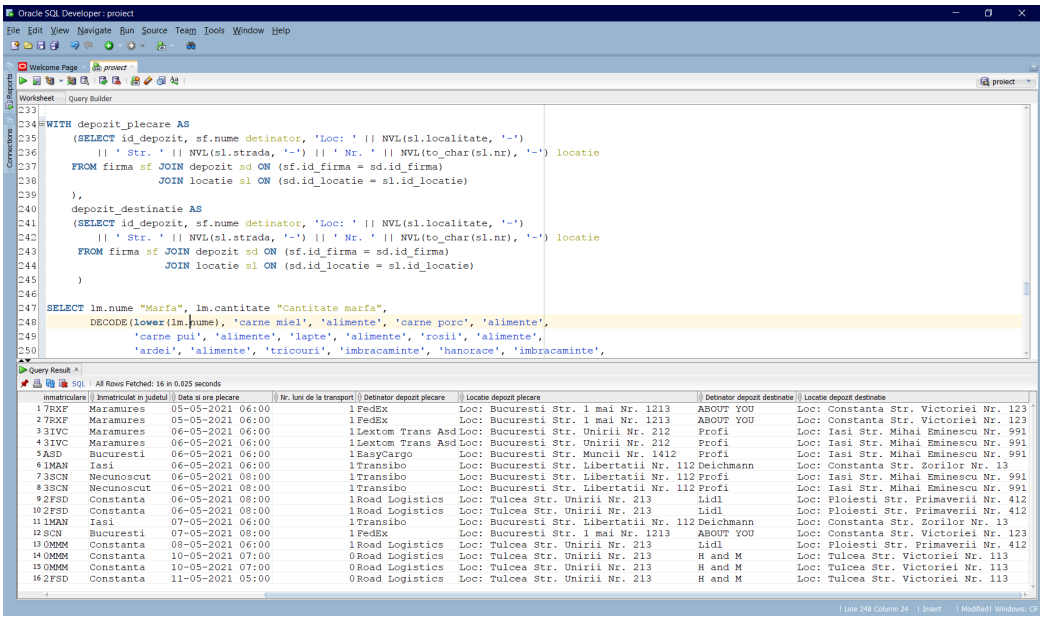
Ex. 11 - 3.1



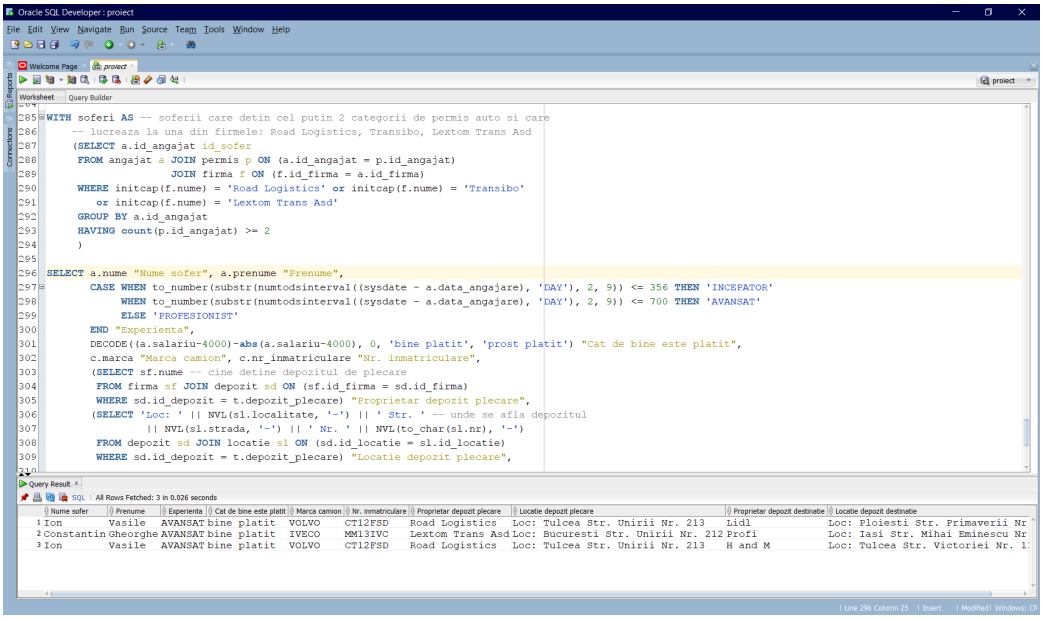
Ex. 11 - 3.2



Ex. 11 - 4.1



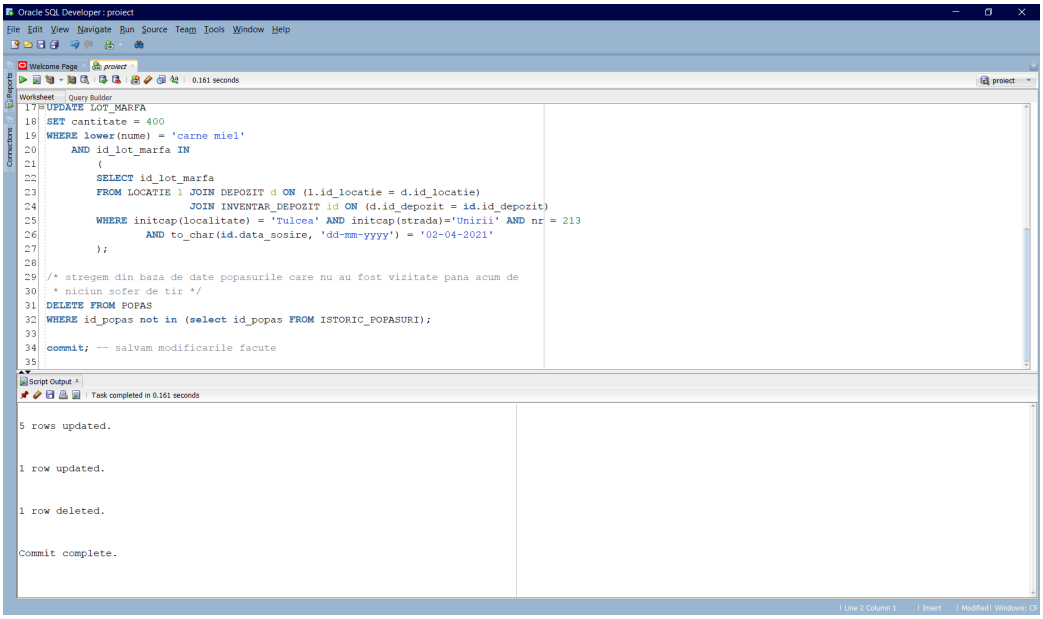
Ex. 11 - 4.2



```
WITH soferi AS -- soferii care detin cel putin 2 categorii de permis auto si care
     -- lucreaza la una din firmele: Road Logistics, Transibo, Lextom Trans Asd
    (SELECT a.id_angajat id_sofer
     FROM angajat a JOIN permis p ON (a.id_angajat = p.id_angajat)
                    JOIN firma f ON (f.id_firma = a.id_firma)
     WHERE initcap(f.nume) = 'Road Logistics' or initcap(f.nume) = 'Transibo'
        or initcap(f.nume) = 'Lextom Trans Asd'
     GROUP BY a.id_angajat
     HAVING count(p.id_angajat) >= 2
     )

SELECT a.nume "Nume sofer", a.prenume "Prenume",
       CASE WHEN to_number(substr(numtodsinterval((sysdate - a.data_angajare), 'DAY'), 2, 9)) <= 356 THEN 'INCEPATOR'
            WHEN to_number(substr(numtodsinterval((sysdate - a.data_angajare), 'DAY'), 2, 9)) <= 700 THEN 'AVANSAT'
            ELSE 'PROFESIONIST'
       END "Experienta",
       DECODE((a.salariu-4000)-abs(a.salariu-4000)), 0, 'bine platit', 'prost platit') "Cat de bine este platit",
       c.marca "Marca camion", c.nr_inmatriculare "Nr. inmatriculare",
       (SELECT sf.nume -- cine detine depozitul de plecare
        FROM firma sf JOIN depozit sd ON (sf.id_firma = sd.id_firma)
        WHERE sd.id_depozit = t.depozit_plecare) "Proprietar depozit plecare",
       (SELECT 'Loc: ' || NVL(sl.localitate, '-') || ' Str. ' -- unde se afla depozitul
                || NVL(sl.strada, '-') || ' Nr. ' || NVL(to_char(sl.nr), '-')
        FROM depozit sd JOIN locatie sl ON (sd.id_locatie = sl.id_locatie)
        WHERE sd.id_depozit = t.depozit_plecare) "Locatie depozit plecare",
```

	Nume sofer	Prenume	Experienta	Cat de bine este platit	Marca camion	Nr. inmatriculare	Proprietar depozit plecare	Locatie depozit plecare	Proprietar depozit destinatie	Locatie depozit destinatie
1	Ion	Vasile	AVANSAT	bine platit	VOLVO	CT12FSD	Road Logistics	Loc: Tulcea Str. Unirii Nr. 213	Lidl	Loc: Ploiesti Str. Primaverii Nr
2	Constantin	Gheorghe	AVANSAT	bine platit	IVECO	MM13IVC	Lextom Trans Asd	Loc: Bucuresti Str. Unirii Nr. 212	Profi	Loc: Iasi Str. Mihai Eminescu Nr
3	Ion	Vasile	AVANSAT	bine platit	VOLVO	CT12FSD	Road Logistics	Loc: Tulcea Str. Unirii Nr. 213	H and M	Loc: Tulcea Str. Victoriei Nr. 1

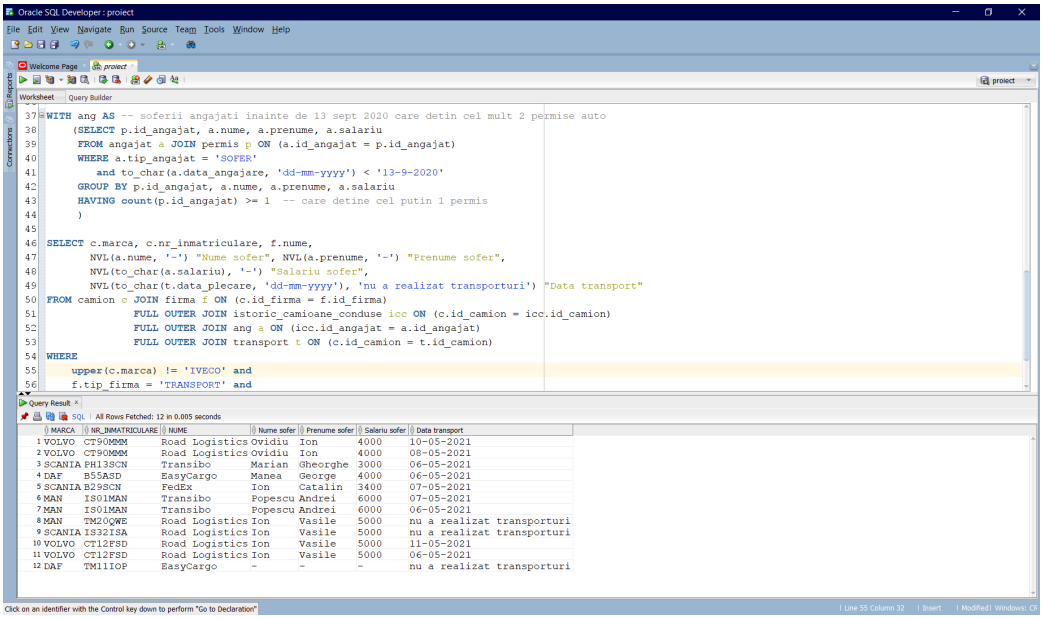
Ex. 11 - 5



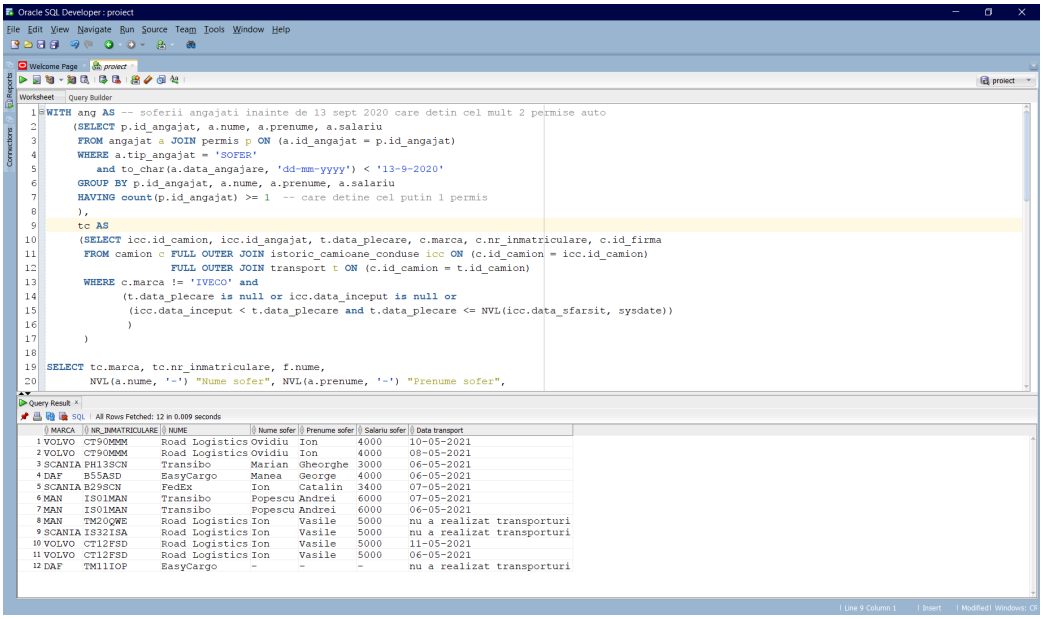
```
UPDATE LOT_MARFA
SET cantitate = 400
WHERE lower(nume) = 'carne miel'
    AND id_lot_marfa IN
        (
        SELECT id_lot_marfa
        FROM LOCATIE l JOIN DEPOZIT d ON (l.id_locatie = d.id_locatie)
                       JOIN INVENTAR_DEPOZIT id ON (d.id_depozit = id.id_depozit)
        WHERE initcap(localitate) = 'Tulcea' AND initcap(strada)='Unirii' AND nr = 213
              AND to_char(id.data_sosire, 'dd-mm-yyyy') = '02-04-2021'
        );

/* stregem din baza de date popasurile care nu au fost vizitate pana acum de
 * niciun sofer de tir */
DELETE FROM POPAS
WHERE id_popas not in (select id_popas FROM ISTORIC_POPASURI);

commit; -- salvam modificarile facute
```

```
5 rows updated.


1 row updated.


1 row deleted.


Commit complete.
```

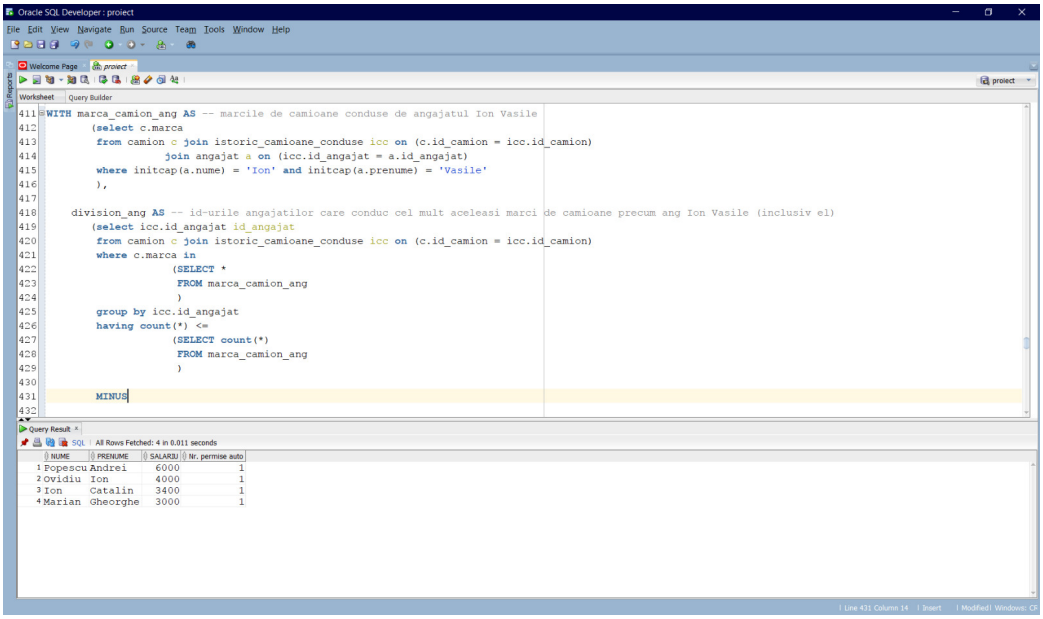
Ex. 12 - 2 updateuri + 1 delete urmate de commit



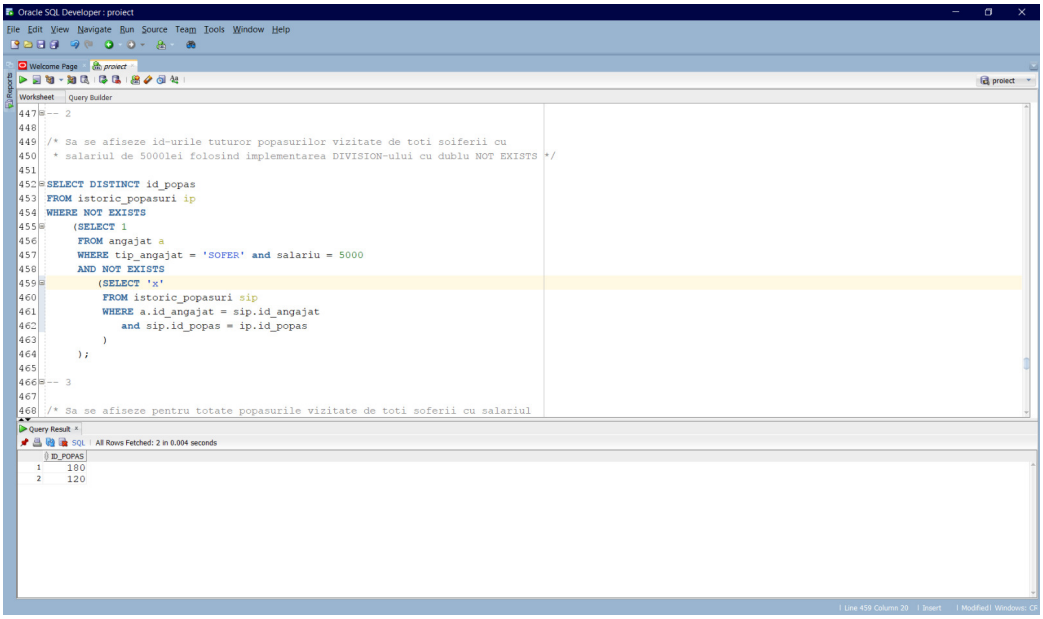
Ex. 16 - 1v1 outer join



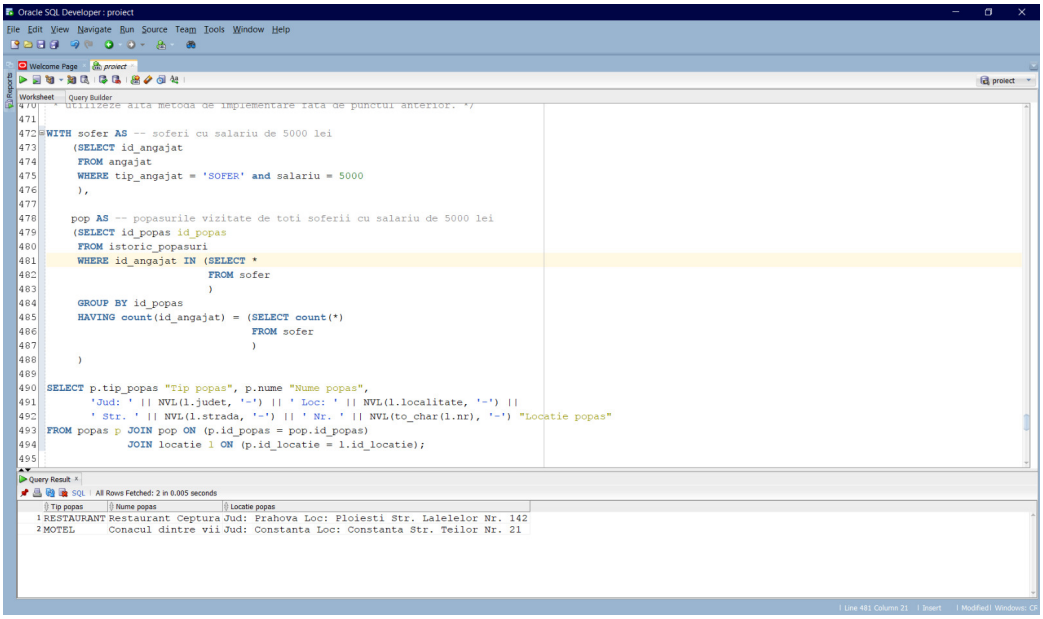
Ex. 17 - 1v2 outer join versiune îmbunătățită



Ex. 16 - 2-1 division



Ex. 16 - 2-2 division



Ex. 16 - 2-3 division