**Gestionarea unei săli de dans, a angajaților, dansatorilor și a rezultatelor acestora**

1. **Descrierea modelului real, a utlității acestuia și a regulilor de funcționare**

Această bază de date are ca scop ținerea evidenței rezultatelor unor dansatori. Modelul de date ne oferă informații despre dansatori, cum ar fi: sala de dans de care aparține, profesorul-dansator ce îl îndrumă, abonamentul său achiziționat la acea sală de dans, concursurile la care a participat (care fac parte din diferite categorii; astfel pot fi competiții naționale, locale, internaționale), susținătorii, cât și dosarul său ca dansator, unde sunt consemnate notele sale primite la competitiile de dans.

Sălile de dans angajează profesori-dansatori pentru a antrena copiii. Dansatorii cei buni vor putea fi aleși într-o grupă specială, unde ca bonus vor primi o sumă de bani, drept recompensă, în fiecare lună. Ei pot participa la ce competiții și probe doresc, depinzând de dorința fiecărui copil.

Pentru fiecare probă la care a participat dansatorul, se va reține nota obținută. În dosarul acestuia vor fi consemnate notele obținute la fiecare probă din fiecare concurs.

Ca reguli de funcționare:

-o sală de dans angajează cel puțin un profesor de dans, profesor ce nu poate fi angajat la mai multe săli de dans

-dansatorii trebuie să aibă doar un singur profesor, pe când acesta poate avea mai mulți elevi

-dansatorii pot să facă parte din grupa specială

-un dansator poate avea privilegiul de a avea susținători care îl sponsorizează, un susținător poate sponsoriza unul sau mai mulți copii

-dansatorii nu sunt obligați să participe la concursuri, ei aleg dacă vor participa și la ce probă

-fiecare dansator are un tip de abonament, tip ce îl pot avea mai mulți copii

-la competiții diferite, dansatorul poate obține rezultate diferite pentru aceeași probă

-în dosarul dansatorului sunt afișate toate notele acestuia până în ziua curentă, dosarul fiind unic, fiecare copil are dosarul său propriu

-dosarul se întemeiază după ce elevul a participat la cel puțin o competiție, adică atunci când se obține un rezultat

-un concurs poate avea loc doar într-o singură sală de concurs, dar într-o sală de concurs pot avea loc mai multe competiții

-o sală de concurs se află într-un oraș, iar într-un oraș există mai multe săli de concurs

1. **Prezentarea constrângerilor impuse asupra modelului**

-unconcurs are loc doar într-o anumită sală de concurs

-profesorul este angajat doar la o singură sală de dans

-un dansator e antrenat doar de un profesor

-un dansator are un abonament

1. **Descrierea entităților, incluzând precizarea cheii primare**

Entități: SALĂ\_DE\_DANS, PROFESOR, DANSATOR, ABONAMENT, SUSȚINĂTOR, DOSAR\_DANSATOR, PROBĂ\_DANS, CONCURS, CATEGORIE, NOTĂ, SALĂ\_CONCURS, ORAȘ.

Subentitate: GRUPA\_SPECIALĂ. Entitatea dependentă: DOSAR\_DANSATOR.

SALĂ\_DE\_DANS= sală menită să învețe copiii să danseze și să performeze în acest domeniu, având dansatori (elevi) și profesori care să îi antreneze. Cheia primară: id\_salădans.

PROFESOR= persoană calificată, cu studii în domeniul dansului, ce îi antrenează pe elevi. Cheia primară: id\_profesor.

DANSATOR= persoană înscrisă la o sală de dans, ce plătește un anumit abonament, pentru a dispune de lecții de dans din partea profesoului său. Acesta poate fi selectat în grupa specială, dacă se consideră că performează în domeniul dansului. Dansatorul poate fi susținut de sponsorii ce oferă o numită sumă de bani. Totodată, dacă dorește, elevul poate participa la anumite competiții și la anumite probe, ce îi sunt pe plac. Cheia primară id\_dansator.

ABONAMENT= fiecare elev are un tip de abonament, achiziționat de la sala de dans, cu numărul de ședințe. Cheia primară id\_abonament.

SUSȚINĂTOR= persoană ce dorește să susțină dansatorul, sponsorizându-l cu o anumită sumă de bani, pentru a-l ajuta și a aprecia munca sa. Cheia primară id\_susținător.

DOSAR\_DANSATOR= este o entitate, dependenta de DANSATOR și NOTĂ, ce reține data obținerii notei la concurs de către dansator. Cheia primară este compusă din id\_dansator, id\_notă, id\_dosar\_dansator.

CONCURS= o competiție destinată mai multor dansatori, cu scopul de a fi premiați cei mai buni dintre cei care participă. Cheia primară id\_concurs.

CATEGORIE= un tip ce se referă la felul concursului, ce poate fi național, internațional etc. Cheia primară id\_categorie.

PROBA\_DANS= un anumit dans pe care îl alege dansatorul să îl presteze în cadrul concursului. Cheia primară id\_proba\_dans.

NOTA= nota obținută în urma deciziei juriului cu privire la prestația fiecărui dansator (notele sunt de la 1 la 10, 10 fiind cea mai bună). Cheia primară id\_nota.

SALA\_CONCURS= sala în care se desfășoară concursul de dans la care participă dansatorul. Cheia primară id\_sala\_concurs.

ORAS= orașul în care are loc concursul de dans. Cheia primară id\_oras.

GRUPA\_SPECIALA= este o subentitate care are ca entitate DANSATOR, având informații legate de elevii ce fac parte din grupa specială, a dansatorilor de performanță. Cheia primară id\_dansator.

1. **Descrierea relațiilor, incluzând precizarea cardinalității acestora**

SALA \_DE\_DANS angajează PROFESOR = relație ce unește entitățile SALA\_DE\_DANS și PROFESOR; legătura dintre acestea este: care sunt profesorii angajați de sala de dans. Aceasta are cardinalitatea minimă 1:1, întrucât o sală de dans trebuie obligatoriu să angajeze cel puțin un profesor și un profesor trebuie să fie angajat de cel puțin o sală de dans. Cardinalitatea maximă este 1:M, deoarece o sală de dans poate angaja mai mulți profesori, dar un profesor poate fi angajat doar de o singură sală.

PROFESOR antreneaza/ învață DANSATOR= relație ce unește cele două entități PROFESOR și DANSATOR; legătura dintre acestea este: care sunt dansatorii învățați de profesor. Aceasta are cardinalitatea minimă 1:1, întrucât un profesor trebuie obligatoriu să învețe un dansator, iar un dansator trebuie să fie învățat de cel puțin un profesor. Cardinalitatea maximă este 1:M, deoarece un profesor poate învăța mai mulți elevi, dar un elev nu poate fi învățat de mai mulți profesori.

DANSATOR are în posesie DOSAR\_DANSATOR= relație ce unește cele două entități DANSATOR și DOSAR\_DANSATOR; legătura dintre acestea este: care sunt notele obținute de dansator la concursuri de la înscrierea la această sală de dans până în prezent. Aceasta are cardinalitatea minimă 1:0, întrucât un dansator nu trebuie să aibă neapărat un dosar de dansator, pentru că poate nu a avut vreo ocazie să participe la un concurs, iar un dosar de dansator trebuie să fie în posesia a celui puțin un dansator. Cardinalitatea maximă este 1:1, deoarece un dansator poate să aibă în posesie un singur dosar de dansator și un dosar de dansator poate fi în posesia unui singur dansator.

SUSȚINĂTOR sponsorizează DANSATOR= relație ce unește cele două entități SUSȚINĂTOR și DANSATOR; legătura dintre acestea este: care sunt susținătorii ce sponsorizează dansatorul. Aceasta are cardinalitatea minimă 0:0, întrucât un susținător nu trebuie obligatoriu să susțină(sponsorizeze) dansatorul, iar un dansator nu trebuie neapărat să aibă un susținător care să-l sponsorizeze cu o anumită sumă de bani. Cardinalitatea maximă este M:M, pentru că un dansator poate avea mai mulți susținători care să-l sponsorizeze, iar un susținător poate ajuta ca anumite sume de bani mai mulți dansatori.

DANSATOR \_participă\_la\_PROBA\_DANS\_de\_la\_CONCURS= este o relație de tipul trei, care unește cele trei entități: DANSATOR, CONCURS, PROBA\_DANS; legătura dintre ele este: dansatorul care a participat la o anumită probă dintr-un concurs. Aceasta are cardinalitatea minimă 0:0:0, deoarece atunci când dansatorul ajunge la sala de dans să nu fie nevoit să își aleagă un concurs și o probă de dans la care să fie concurent. Cardinalitatea maximă este M:M:M.

CONCURS \_are loc\_SALA\_CONCURS= relație ce unește cele două entități CONCURS și SALA\_CONCURS; legătura dintre acestea este: în ce săli de dans au loc concursurile. Aceasta are cardinalitatea minimă 1:1, întrucât concursul trebuie să fie în cel puțin o sală de dans, iar ăntr-o sală de dans trebuie să fie minim un concurs. Cardinalitatea maximă este M:1, pentru că un concurs se poate desfășura într-o singură sală de dans și o sală de dans poate găzdui mai multe concursuri.

SALA\_CONCURS \_se afla\_ORAS= relație ce unește cele două entități SALA\_CONCURS și ORAS; legătura dintre acestea este: în ce oraș se află sala de concurs. Aceasta are cardinalitatea minimă 1:1, întrucât o sală de dans trebuie să se afle neapărat într-un oraș, iar într-un oraș trebuie să se afle cel puțin o sală de concurs. Cardinalitatea maximă este M:1, deoarece o sală de dans poate fi într-un singur oraș și un oraș poate găzdui mai multe săli de dans.

CONCURS\_este\_CATEGORIA= relație ce unește cele două entități CONCURS și CATEGORIA; legătura dintre acestea este: precizează de ce fel este concursul. Aceasta are cardinalitatea minimă 0:1, întrucât un concurs trebuie sa fie de minim un tip, iar un fel nu trebuie să aibă neapărat un concurs. Cardinalitatea maximă este M:1, deoarece un concurs poate avea doar o singura categorie, pe când o categorie poate să fie la mai multe concursuri.

PROBA\_DANS\_obține\_NOTA= relație ce unește cele două entități PROBA\_DANS și NOTA; legătura dintre acestea este: precizează ce note primește dansatorul la proba din cadrul concursului. Aceasta are cardinalitatea minimă 1:1, întrucât la o probă de dans trebuie neapărat să se obțină cel puțin o notă , iar o notă se obține la cel puțin un concurs. Cardinalitatea maximă este 1:M, deoarece la un concurs se pot obține mai multe note, iar o notă poate fi primită la o probă.

NOTA\_apare la\_DOSAR\_DANSATOR= relație ce unește cele două entități NOTA și DOSAR\_DANSATOR; legătura dintre ele este: notele primite de dansator la o probă din competiție sunt afișate in dosarul dansatorului. Aceasta are cardinalitatea minimă 0:1, întrucât o notă trebuie neapărat să se regăsească într-un dosar, pe când un dosar nu trebuie să conțină obligatoriu o notă. Cardinalitatea maximă este M:1, deoarece o notă este pusă într-un singur dosar al dansatorului în cauză și un dosar de dansator poate conține mai multe note obținute.

DANSATOR\_are\_ABONAMENT= relație ce unește cele două entități DANSATOR și ABONAMENT; legătura dintre ele este: precizează ce fel de abonament la sala de dans are dansatorul. Cardinalitatea sa este 1:1, deoarece fiecare dansator trebuie sa dispuna de un abonament pentru a face parte din cadrul dansatorilor salii de dans.

1. **Descrierea atributelor, inlcuzând tipul de date și eventualele constrângeri, valori implicite, valori posibile ale atributelor**

Entitatea SALA\_DE\_DANS are următoarele atribute:

•id\_saladans= codul sălii de dans; este o variabilă de tip întreg, lungime maximă 6

•nume\_saladans= denumirea sălii de dans, variabilă tip caracter, lungime maxima 20

Entitatea PROFESOR are următoarele atribute:

•id\_profesor= codul profesorului, variabilă de tip întreg, lungime maximă 6

•id\_saladans= codul sălii de dans; este o variabilă de tip întreg, lungime maximă 6; acesta corespunde la o valoare a cheii primare din tabela SALA\_DE\_DANS (FK)

•nume= numele profesoului, variabilă de tip caracter, de lungime maximă 20

•prenume= prenumele profesorului, variabilă de tip caracter, de lungime maximă 20

•salariu= salariul pe care îl câștigă lunar profesorul, variabilă de tip numeric, minim 1000 de lei pe lună

•an\_nastere= anul in care s-a născut antrenorul, variabilă de tip întreg, lungime maximă 4

Entitatea DANSATOR are următoarele atribute:

•id\_dansator= codul dansatorului, variabilă de tip întreg, lungime maximă 6

•id\_profesor= codul profesorului; este o variabilă de tip întreg, lungime maximă 6; acesta corespunde la o valoare a cheii primare din tabela PROFESOR (FK)

•id\_abonament= codul abonamentului; este o variabilă de tip întreg, lungime maximă 6; acesta corespunde la o valoare a cheii primare din tabela ABONAMENT (FK)

•nume= numele dansatorului, variabilă de tip caracter, de lungime maximă 20

•prenume= prenumele dansatorului, variabilă de tip caracter, de lungime maximă 20

•an\_nastere= anul in care s-a născut dansatorului, variabilă de tip întreg, lungime maximă 4

•experienta= anii de când copilul practică acest sport, pentru a cunoaște experiența sa în domeniu, variabilă de tip numeric ,lungime maxima 2

Subentitatea GRUPA\_SPECIALĂ are următoarele atribute:

•id\_dansator= codul dansatorului, variabilă de tip întreg, lungime maximă 6

•bonus= banii pe care îi câștigă lunar dansatorul ales să facă parte din grupa specială, datorită performanței sale, variabilă de tip numeric, lungime maxima 3

Entitatea ABOANAMENT are următoarele atribute:

•id\_abonament= codul abonamentului, variabilă de tip întreg, lungime maximă 6

•numar\_sedințe= orele la care va participa dansatorul pe lună, variabilă de tip numeric maxim 2

•tip\_abonament= felul abonamentului(de elev, student, copil), variabilă de tip caracter, de lungime maximă 15

•pret= pretul abonamentului, variabilă de tip numeric, lungime maxima 3

Entitatea SUSȚINĂTOR are următoarele atribute:

•id\_sustinator= codul susținătorului, variabilă de tip întreg, lungime maximă 6

•nume= numele susținătorului, variabilă de tip caracter, de lungime maximă 20

•prenume= prenumele susținătorului, variabilă de tip caracter, de lungime maximă 20

•suma= suma oferită dansatorului drept sprijin, variabilă de tip numeric

Entitatea DOSAR\_DANSATOR are următoarele atribute:

•id\_dosar\_dansator= codul dosarului de dansator, variabilă de tip întreg, lungime maximă 6

•id\_nota= codul notei care este afișat în dosar, variabilă de tip întreg, lungime maximă 6; acesta corespunde la o valoare a cheii primare din tabela NOTA (FK)

•id\_dansator= codul dansatorului a cărui este dosarul de dansator; lungime maximă 6; acesta corespunde la o valoare a cheii primare din tabela DANSATOR (FK)

•data\_rez= reprezintă data la care dansatorul primește nota la proba de dans din cadrul concursului; variabilă de tip dată calendaristică

Entitate CONCURS are următoarele atribute:

•id\_concurs= codul concursului, variabilă de tip întreg, lungime maximă 6

•nume= numele concurs, variabilă de tip caracter, de lungime maximă 20

•ziua\_concurs=reprezinta ziua din săptămână în care are loc concusrul, variabilă de tip caracter lungime maximă 10

•id\_salaconcurs= codul sălii de dans, variabilă de tip întreg, lungime maximă 6; acesta corespunde la o valoare a cheii primare din tabela SALĂ\_CONCURS(FK)

•id\_categorie= codul categoriei concursului, variabilă de tip întreg, lungime maximă 6; acesta corespunde la o valoare a cheii primare din tabela CATEGORIE(FK)

Entitatea CATEGORIE are următoarele atribute:

•id\_categorie= codul categoriei, variabilă de tip întreg, lungime maximă 6

•nume\_categorie= denumirea categoriei, variabilă de tip caracter, de lungime maximă 20, fiind unic și nenul

Entitatea SALA\_CONCURS are ca atribute:

•id\_salaconcurs= codul sălii de dans, variabilă de tip întreg, lungime maximă 6

•id\_oras= variabilă de tip caracter, reprezentând cele 2 litere reprezentative ale orașului, prefixul, de lungime maxima 2; acesta corespunde la o valoare a cheii primare din tabela ORAS(FK)

•nume\_salaconcurs= denumirea sălii de dans, variabilă de tip caracter, lungime maximă 20

Entitatea ORAS are ca atribute:

•id\_oras= variabilă de tip caracter, reprezentând cele 2 litere reprezentative ale orașului, prefixul, de lungime maxima 2

•denumire\_oras= numele orașului în care se află sala de dans în care se desfășoară competiția, variabilă de tip caracter lungime maximă 20

Entitatea PROBA\_DANS are ca atribute:

•id\_proba\_dans= codul probei de dans, variabilă de tip întreg, lungime maximă 6

•nume\_dans= numele dansului/ probei, variabilă de tip caracter lungime maximă 20

Entitatea NOTA are ca atribute:

•id\_nota= codul notei obținute în cadrul probei de dans din competiție, variabilă de tip întreg, lungime maximă 6

•id\_proba\_dans= codul probei de dans, variabilă de tip întreg, lungime maximă 6; acesta corespunde la o valoare a cheii primare din tabela PROBA\_DANS(FK)

•nota\_obtinuta=variabilă de tip întreg ce reprezintă nota obținută de dansator la proba de dans

•id\_dansator= codul dansatorului care a obținut nota, variabilă de tip întreg, lungime maximă 6; acesta corespunde la o valoare a cheii primare din tabela DANSATOR(FK)

•id\_dosar\_dansator= codul dosarului de dansator al celui care a obținut nota la proba din cadrul concursului, variabilă de tip întreg, lungime maximă 6; acesta corespunde la o valoare a cheii primare din tabela DOSAR\_DANSATOR(FK)

Relația SUSTINATOR \_sponsorizează\_ DANSATOR are ca atribute:

•id\_sustinator= codul sustinatorului, variabilă de tip întreg, lungime maximă 6; acesta corespunde la o valoare a cheii primare din tabela SUSTINATOR

•id\_dansator= codul dansatorului susțnut, variabilă de tip întreg, lungime maximă 6; acesta corespunde la o valoare a cheii primare din tabela DANSATOR

•început\_perioada= precizează data la care a început susținătorul să-l sponsorizeze pe dansator cu o sumă de bani; variabilă de tip calendaristic

•sfarsit\_perioada= precizează data la care se sfârșete colaborarea susținătorulului cu dansatorul; variabilă de tip calendaristic

Relația DANSATOR \_participă la\_PROBA\_DANS\_ din\_CONCURS are ca atribute:

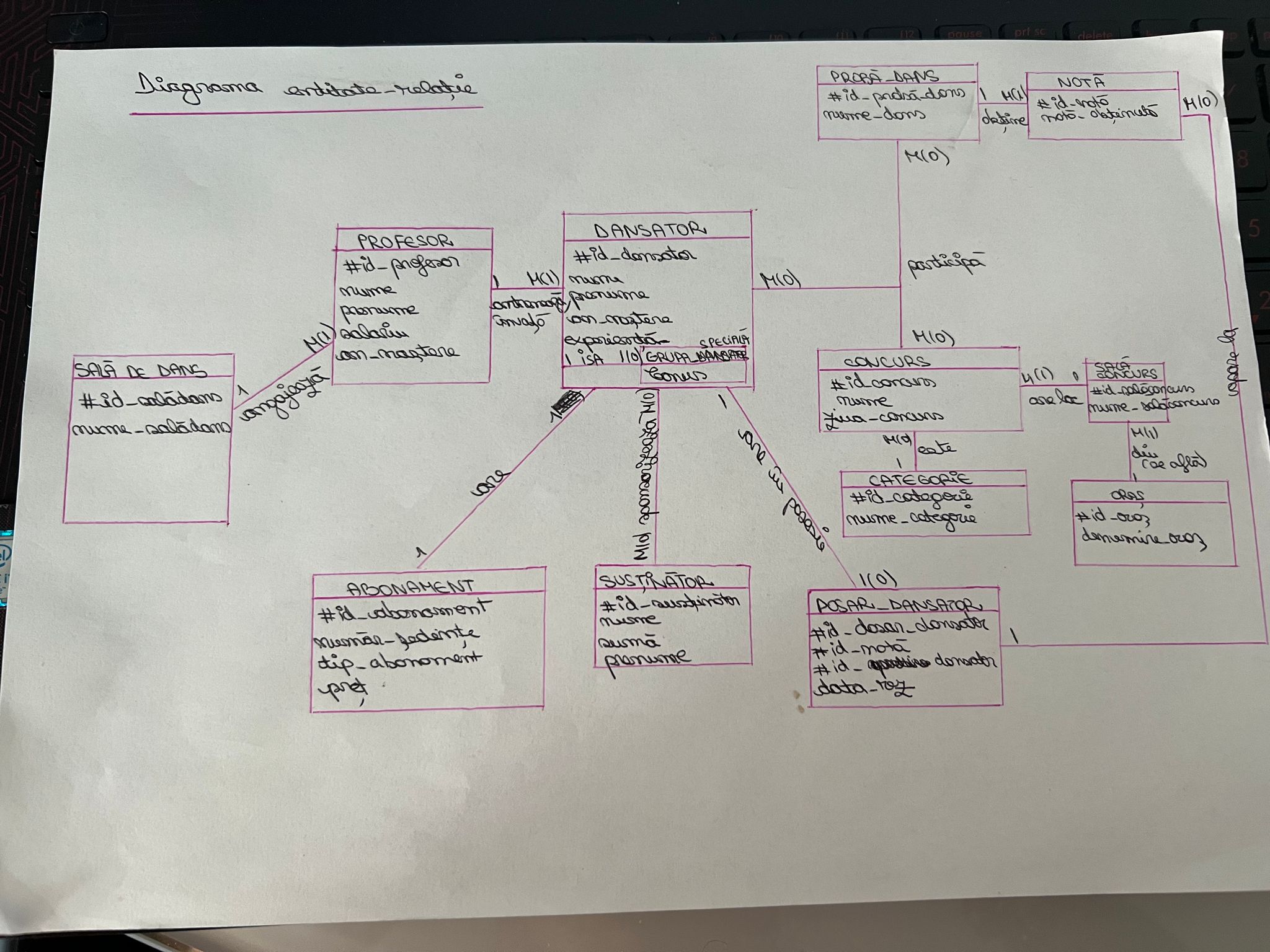
•id\_dansator= codul dansatorului susțnut, variabilă de tip întreg, lungime maximă 6; acesta corespunde la o valoare a cheii primare din tabela DANSATOR

•id\_proba\_dans= codul probei de dans, variabilă de tip întreg, lungime maximă 6; acesta corespunde la o valoare a cheii primare din tabela PROBA\_DANS

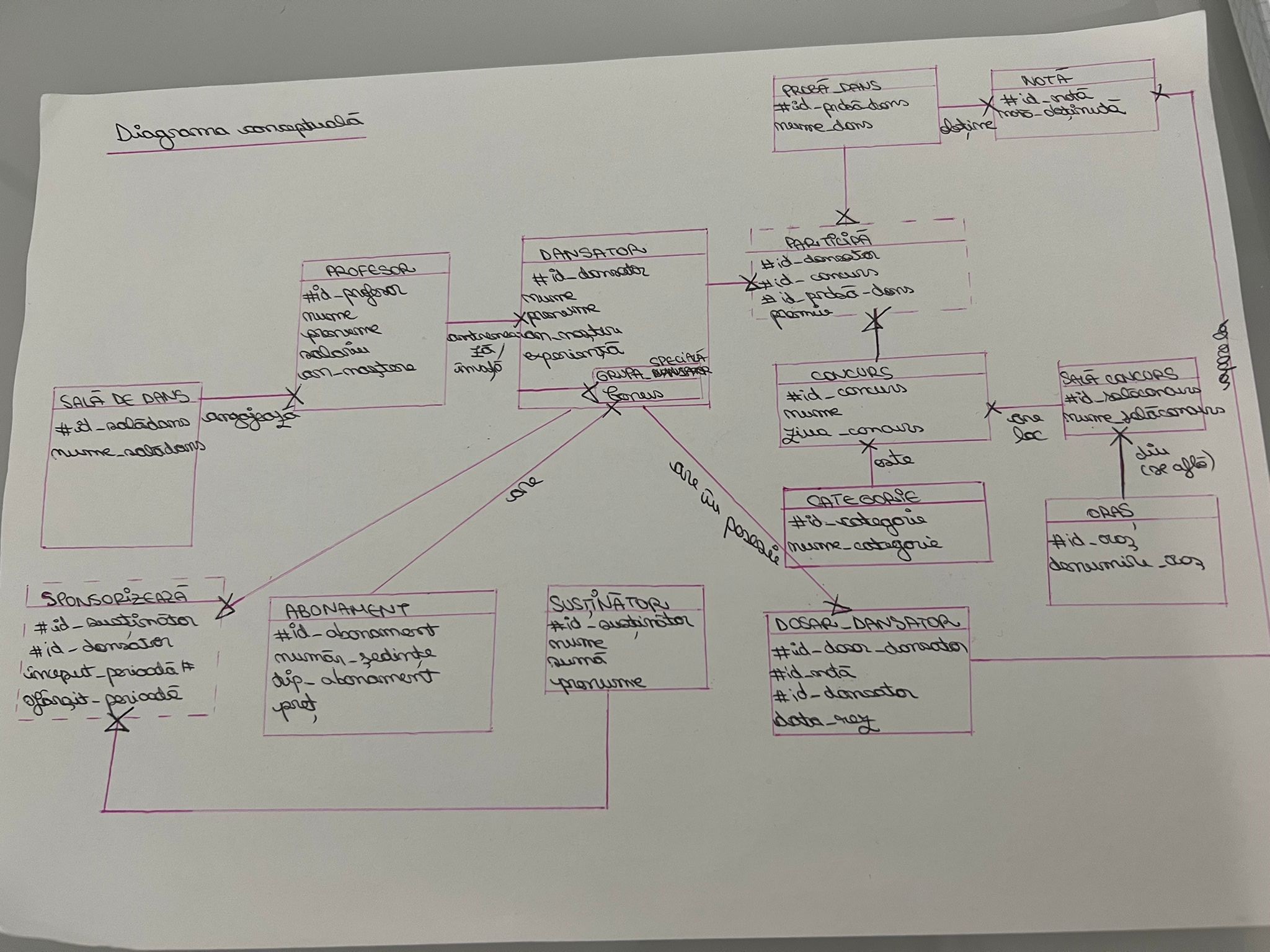
•id\_concurs= codul concursului, variabilă de tip întreg, lungime maximă 6, acesta corespunde la o valoare a cheii primare din tabela CONCURS

•premiu= variabilă booleană care ne informează dacă dansatorul a obținut unul dintre premiile 1, 2 sau 3

1. **Diagrama entitate - relație**



1. **Diagrama conceptuală**



1. **Enumerarea schemelor relaționale corespunzătoare diagramei conceptuale proiectate la punctul 7**

SALA\_DE\_DANS (id\_saladans#, nume\_saladans)

PROFESOR (id\_profesor#, nume, prenume, salariu, an\_nastere, id\_saladans)

DANSATOR (id\_dansator#, nume, prenume, an\_nastere, experienta, id\_profesor, id\_abonament)

GRUPA\_SPECIALA (id\_dansator#, bonus)

ABONAMENT (id\_abonament#, numar\_sedinte, tip\_abonament, pret)

SPONSORIZEAZA (id\_sustinator#, id\_dansator#, inceput\_perioada#, sfarsit\_perioada)

SUSTINATOR (id\_sustinator#, nume, prenume, suma)

DOSAR\_DANSATOR (id\_dosar\_dansator#, id\_nota#, id\_dansator#, data\_rez)

PARTICIPA (id\_dansator#, id\_concurs#, id\_proba\_dans#, premiu)

PROBA (id\_proba\_dans#, nume\_dans)

NOTA (id\_nota#, id\_proba\_dans, id\_dosar\_dansator, id\_dansator, nota\_obtinuta)

CONCURS (id\_concurs#, nume, ziua\_concurs, id\_categorie, id\_salaconcurs)

CATEGORIE (id\_categorie#, nume\_categorie)

SALA\_CONCURS (id\_salaconcurs#, nume\_salaconcurs, id\_oras)

ORAS (id\_oras#, denumire\_oras)

1. **Realizarea normalizării până la forma nominală 3 (FN1-FN3)**

**FORMA NORMALA 1 (FN1)**

Relațiile din diagramă sunt în forma normală 1 (FN1) și voi demonstra acst lucru mai jos, cu exemple. O relație care se află în forma normală 1 este o relație în care există un identificator unic; fiecărui atribut care îi aparține relației, îi corespunde o valoare indivizibilă.

Ca prim exemplu, în relația SALA\_DE\_DANS, cheia primară este id\_saladans și este un identificator unic. Astfel, relația este în forma normală 1. Fiecare sală de dans are un id propriu.

Al doilea exemplu este relația DOSAR\_DANSATOR, unde cheia primară este formată din: id\_dosar\_dansator, id\_dansator, id\_nota. Este identificator unic. Atributele id\_dosar\_dansator, id\_dansator, id\_nota formează împreună o valoare indivizibilă. Așadar, relația este în forma normală 1 (FN1).

Acum vom lua un exemplu fictiv pentru a arăta cum se face această normalizare. Spunem că pentru fiecare concurs pot exista mai multe zile de desfășurare a acestuia (ziua\_concurs).

Relația concurs\_nonF1 este următoarea:

|  |  |
| --- | --- |
| id\_concurs# | ziua\_concurs |
| 1 | luni, marti, miercuri |
| 2 | marti |
| 3 | joi, vineri, sambata |

Atributul ziua\_concurs nu are o valoare indivizibilă, astfel relația nu se află în forma normală 1 (FN1). Dar, când îi este aplicată forman normală 1 (FN1), fiecărui atribut îi corespunde o valoare indivizibilă; relația va fi așa:

|  |  |
| --- | --- |
| id\_concurs# | ziua\_concurs# |
| 1 | luni |
| 1 | marti |
| 1 | miercuri |
| 2 | marti |
| 3 | joi, vineri |
| 3 | sambata |

**FORMA NORMALA 2 (FN2)**

O relație se află în forma normală 2, doar dacă este în prima formă normală 2 și dacă fiecare atribut care nu face parte din cheie este dependent de toată cheia primară.

Exemplificăm trecerea de la o relație nonFN2 la o relație FN2:

Inițial:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| id\_dosar\_dasator# | id\_nota# | id\_dansator# | data\_rez | nota\_obtinuta |
| 1 | 1 | 1 | 01-08-2020 | 8 |
| 1 | 2 | 1 | 17-02-2021 | 9 |
| 2 | 3 | 2 | 27-03-2020 | 10 |
| 2 | 4 | 2 | 02-03-2021 | 7 |
| 7 | 5 | 7 | 25-07-2022 | 8 |

Relația se află în FN1, pentru că există identificator unic pentru toate intrările. Dar, pentru a fi în FN2, fiecare atribut (care nu face parte in cheie) trebuie să fie dependent de toată cheia primară. Așadar, data\_rez și nota\_obtinuta trebuie să fie dependente de toată cheia primară compusă din: id\_dosar\_dansator#, id\_nota#, id\_dansator#, lucru ce nu se întâmplă. Există o dependență directă între id\_nota# și nota\_obtinuta, deci nota\_obtinuta depinde doar de o parte a cheii primare, și anume id\_nota#. Relația de mai sus nu este în FN2.

nota\_obtinuta depinde de id\_nota#, iar data\_rez depinde de id\_nota, id\_dansator și id\_dosar\_dansator.

Transformarea în FN2:

DOSAR\_DANSATOR

|  |  |  |  |
| --- | --- | --- | --- |
| id\_dosar\_dansator# | id\_nota# | id\_dansator# | data\_rez |
| 1 | 1 | 1 | 01-08-2020 |
| 1 | 2 | 1 | 17-02-2021 |
| 2 | 3 | 2 | 27-03-2020 |
| 2 | 4 | 2 | 02-03-2021 |
| 7 | 5 | 7 | 25-07-2022 |

NOTA

|  |  |
| --- | --- |
| id\_nota | nota\_obtinuta |
| 1 | 8 |
| 2 | 9 |
| 3 | 10 |
| 4 | 7 |
| 5 | 8 |

O relație care se află în FN2 este DOSAR\_DANSATOR. Este în FN1, existând identificator unic pentru fiecare intrare în tabel. data\_rez este dependent de întreaga cheie primara care este compusă din id\_dosar\_dansator#, id\_nota#, id\_dansator#.

**FORMA NORMALA 3 (FN3)**

O relație se află în FN3 doar dacă se află în FN2 și dacă fiecare atribut care nu este cheie este dependent direct de cheia primară și numai de aceasta.

Realizăm un exemplu fictiv PROFESOR, unde salariul depinde de anul nasterii, adică cu cât este mai în vârstă profesorul, se presupune că are experiență și va câștiga mai mulți bani, lucru fals în diagramă.

profesor\_nonFN3

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| id\_profesor# | nume | prenume | salariu | an\_nastere | id\_saladans(FK) |
| 10 | Popescu | Călin | 700 | 1987 | 80 |
| 20 | Adam | Dan | 600 | 1999 | 90 |
| 30 | Ionescu | Ioana | 2000 | 1970 | 100 |
| 40 | Gherman | David | 700 | 1987 | 110 |
| 50 | Aioanei | Maria | 800 | 1985 | 120 |

Această relație este în FN1, pentru că există un identificator unic id\_profesor# pentru fiecare intrare. Este totodată și în FN2, pentru că atributele care nu fac parte din cheia primară: nume, prenume, salariu și an\_nastere sunt dependente de cheia primară id\_profesor#. Aici, salariu depinde de cheia primară id\_profesor# prin atributul an\_nastere. Nu se află în FN3, pentru că nume, prenume, an\_nastere depind de id\_profesor#. salariu depinde tranzitiv de id\_profesor# (cheie primară) prin atributul an\_nastere. Transformarea în FN3 se face cu ajutorul regulii Casey-Delobel, prin eliminarea dependenței tranzitive. Astfel, relația devine:

PROFESOR\_FN3(id\_profesor#, nume, prenume, an\_nastere)

AN\_NASTERE(an\_nastere, salariu)

Transformare:

PROFESOR\_FN3

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| id\_profesor# | nume | prenume | an\_nastere | id\_saladans(FK) |
| 10 | Popescu | Călin | 1987 | 80 |
| 20 | Adam | Dan | 1999 | 90 |
| 30 | Ionescu | Ioana | 1970 | 100 |
| 40 | Gherman | David | 1987 | 110 |
| 50 | Aioanei | Maria | 1985 | 120 |

AN\_NASTERE\_SALARIU

|  |  |
| --- | --- |
| an\_nastere | salariu |
| 1987 | 80 |
| 1999 | 90 |
| 1970 | 100 |
| 1987 | 110 |
| 1985 | 120 |

1. **Crearea unei secvențe ce va fi utilizată în inserarea înregistrărilor în tabele + 11. Crearea tabelelor în SQLși inserarea de date coerente în fiecare dintre acestea (minimum 5 înregistrări în fiecare tabel neasociativ; minimum 10 înregistrări în tabelele asociative).**

create table SALA\_DE\_DANS

(id\_saladans NUMBER(6),

nume\_saladans VARCHAR2(20) constraint

nume\_saladans\_nn not null,

unique(nume\_saladans)

);

alter table SALA\_DE\_DANS

add constraint id\_saladans\_pk primary key (id\_saladans);

create table PROFESOR (

id\_profesor number(6),

nume varchar2(20) constraint nume\_profesor\_nn not null,

prenume varchar2(20) constraint prenume\_profesor\_nn not null,

an\_nastere number(4) constraint an\_nastere\_profesor\_nn not null,

salariu number(8, 2) constraint salariu\_min check (salariu > 1000),

id\_saladans number(6)

);

alter table PROFESOR

add constraint id\_profesor\_pk primary key (id\_profesor);

alter table PROFESOR

add constraint id\_saladans\_kf foreign key (id\_saladans) references SALA\_DE\_DANS (id\_saladans);

create table ABONAMENT(

id\_abonament number(6),

numar\_sedinte number(2) constraint numar\_sedinte\_nn not null,

tip\_abonament varchar2(15) constraint tip\_abonament\_nn not null,

pret number(3) constraint pret\_nn not null

);

alter table ABONAMENT

add constraint id\_abonament\_pk primary key (id\_abonament);

create table DANSATOR(

id\_dansator number(6),

nume varchar2(20) constraint nume\_dansator\_nn not null,

prenume varchar2(20) constraint prenume\_dansator\_nn not null,

an\_nastere number(4) constraint an\_nastere\_dansator\_nn not null,

experienta number(2) constraint experienta\_dansator\_nn not null,

id\_abonament number(6) constraint abonament\_dansator\_nn not null,

bonus number(3),

id\_profesor number(6),

constraint id\_dansator\_pk primary key (id\_dansator),

constraint id\_profesor\_fk foreign key (id\_profesor) references PROFESOR(id\_profesor),

constraint id\_abonament\_fk foreign key (id\_abonament) references ABONAMENT (id\_abonament)

);

create table SUSTINATOR (

id\_sustinator number(6),

nume varchar2(20) constraint nume\_sustinator\_nn not null,

prenume varchar2(20) constraint prenume\_sustinator\_nn not null,

suma number(5, 1) constraint suma\_nn not null,

unique (nume),

constraint id\_sustinator\_pk primary key (id\_sustinator)

);

create table SPONSORIZEAZA (

id\_sustinator number(6),

id\_dansator number(6),

inceput\_perioada date constraint inceput\_perioada\_nn not null,

sfarsit\_perioada date constraint sfarsit\_perioada\_nn not null,

constraint perioade check (inceput\_perioada < sfarsit\_perioada),

constraint id\_sustinator\_id\_dansator\_pk primary key (id\_sustinator, id\_dansator, inceput\_perioada),

constraint id\_sustinator\_fk foreign key (id\_sustinator) references SUSTINATOR (id\_sustinator),

constraint id\_dansator\_fk foreign key (id\_dansator) references DANSATOR (id\_dansator)

);

create table CATEGORIE (

id\_categorie number(6),

nume\_categorie varchar(20) constraint nume\_categorie\_nn not null,

unique(nume\_categorie),

constraint id\_categorie\_pk primary key (id\_categorie)

);

create table ORAS (

id\_oras char(2),

denumire\_oras varchar2(20) constraint denumire\_oras\_nn not null,

constraint id\_oras\_pk primary key (id\_oras)

);

create table SALA\_CONCURS (

id\_salaconcurs number(6),

nume\_salaconcurs varchar2(20) constraint nume\_salaconcurs\_nn not null,

id\_oras char(2),

constraint id\_salaconcurs\_pk primary key (id\_salaconcurs),

constraint id\_oras\_fk foreign key (id\_oras) references ORAS (id\_oras),

unique(nume\_salaconcurs)

);

create table CONCURS (

id\_concurs number(6),

nume varchar2(20) constraint nume\_nn not null,

ziua\_concurs varchar2(10) constraint ziua\_concurs\_nn not null,

id\_salaconcurs number(6),

id\_categorie number(6),

constraint id\_concurs\_pk primary key (id\_concurs),

constraint id\_categorie\_fk foreign key (id\_categorie) references CATEGORIE (id\_categorie),

constraint id\_salaconcurs\_fk foreign key (id\_salaconcurs) references SALA\_CONCURS(id\_salaconcurs),

unique(nume)

);

create table DOSAR\_DANSATOR (

id\_dosar\_dansator number(6),

data\_rez date constraint data\_rez\_nn not null,

id\_nota number(6),

id\_dansator number(6),

constraint dosar\_nota\_dansator\_pk primary key (id\_dosar\_dansator, id\_nota, id\_dansator),

constraint id\_dansatorul\_fk foreign key (id\_dansator) references DANSATOR (id\_dansator)

);

create table PROBA\_DANS (

id\_proba\_dans number(6),

nume\_dans varchar2(20) constraint nume\_dans\_nn not null,

constraint id\_proba\_dans\_pk primary key (id\_proba\_dans),

unique(nume\_dans)

);

create table NOTA (

id\_nota number(6),

id\_proba\_dans number(6),

nota\_obtinuta number(2) constraint nota\_obtinuta\_nn not null,

id\_dansator number(6),

id\_dosar\_dansator number(6),

constraint id\_nota\_pk primary key (id\_nota),

constraint dosar\_nota\_dansator\_fk foreign key (id\_dosar\_dansator, id\_nota, id\_dansator) references DOSAR\_DANSATOR (id\_dosar\_dansator, id\_nota, id\_dansator),

constraint proba\_fk foreign key (id\_proba\_dans) references PROBA\_DANS (id\_proba\_dans)

);

create table PARTICIPA (

id\_dansator number(6),

id\_concurs number(6),

id\_proba\_dans number (6),

premiu number(1) constraint premiu\_nn not null,

constraint are\_sau\_nu\_premiu check (premiu in (1, 0)),

constraint dansator\_concurs\_proba\_pk primary key (id\_dansator, id\_concurs, id\_proba\_dans),

constraint probaa\_fk foreign key (id\_proba\_dans) references PROBA\_DANS (id\_proba\_dans),

constraint dansatorul\_fk foreign key (id\_dansator) references DANSATOR (id\_dansator),

constraint concurs\_fk foreign key (id\_concurs) references CONCURS (id\_concurs)

);

**Crearea unei secvențe utilizată pentru inserarea înregistrărilor în tabele (exercițiul 13)**

select \* from SALA\_DE\_DANS;

create sequence seq\_sala

increment by 10

start with 10

maxvalue 10000

nocycle;

insert into SALA\_DE\_DANS

values(seq\_sala.nextval, 'Bucovina Dance');

insert into SALA\_DE\_DANS

values(seq\_sala.nextval, 'Best Dance');

insert into SALA\_DE\_DANS

values(seq\_sala.nextval, 'Phoenix Dance');

insert into SALA\_DE\_DANS

values(seq\_sala.nextval, 'Dancing on ice');

insert into SALA\_DE\_DANS

values(seq\_sala.nextval, 'Dance and Fun');

commit;

select \* from SALA\_DE\_DANS;

create sequence seq\_PROFESOR

increment by 10

start with 10

maxvalue 10000

nocycle;

select \* from PROFESOR;

insert into PROFESOR (id\_profesor, nume, prenume, an\_nastere, salariu, id\_saladans)

values (seq\_PROFESOR.nextval, 'Cojocaru', 'Anca', 1989, 1100, 20);

insert into PROFESOR (id\_profesor, nume, prenume, an\_nastere, salariu, id\_saladans)

values (seq\_PROFESOR.nextval, 'Popescu', 'Lavinia', 1979, 1152, 10);

insert into PROFESOR (id\_profesor, nume, prenume, an\_nastere, salariu, id\_saladans)

values (seq\_PROFESOR.nextval, 'Matei', 'Ioana', 1977, 1200, 40);

insert into PROFESOR (id\_profesor, nume, prenume, an\_nastere, salariu, id\_saladans)

values (seq\_PROFESOR.nextval, 'Aioanei', 'Catalina', 1998, 1023, 30);

insert into PROFESOR (id\_profesor, nume, prenume, an\_nastere, salariu, id\_saladans)

values (seq\_PROFESOR.nextval, 'Ion', 'Bogdan', 1989, 14000, 20);

insert into PROFESOR (id\_profesor, nume, prenume, an\_nastere, salariu, id\_saladans)

values (seq\_PROFESOR.nextval, 'Serban', 'Denisa', 1994, 1325, 10);

insert into PROFESOR (id\_profesor, nume, prenume, an\_nastere, salariu, id\_saladans)

values (seq\_PROFESOR.nextval, 'Ardelean', 'Loredana', 1976, 1900, 50);

insert into PROFESOR (id\_profesor, nume, prenume, an\_nastere, salariu, id\_saladans)

values (seq\_PROFESOR.nextval, 'Raicu', 'Sabina', 1999, 1702, 20);

insert into PROFESOR (id\_profesor, nume, prenume, an\_nastere, salariu, id\_saladans)

values (seq\_PROFESOR.nextval, 'Coroc', 'Ioana', 1979, 1400, 30);

insert into PROFESOR (id\_profesor, nume, prenume, an\_nastere, salariu, id\_saladans)

values (seq\_PROFESOR.nextval, 'Salajeanu', 'Nicole', 1984, 1850, 10);

insert into PROFESOR (id\_profesor, nume, prenume, an\_nastere, salariu, id\_saladans)

values (seq\_PROFESOR.nextval, 'Zancu', 'Alexandru', 1997, 1300, 50);

commit;

select \* from PROFESOR;

**Continuare inserari:**

select \* from ABONAMENT;

insert into

ABONAMENT(id\_abonament, numar\_sedinte, tip\_abonament, pret)

values (1, 8, 'elev', 160);

insert into

ABONAMENT(id\_abonament, numar\_sedinte, tip\_abonament, pret)

values (2, 4, 'elev', 80);

insert into

ABONAMENT(id\_abonament, numar\_sedinte, tip\_abonament, pret)

values (3, 12, 'elev', 240);

insert into

ABONAMENT(id\_abonament, numar\_sedinte, tip\_abonament, pret)

values (4, 4, 'copil', 75);

insert into

ABONAMENT(id\_abonament, numar\_sedinte, tip\_abonament, pret)

values (5, 8, 'copil', 155);

insert into

ABONAMENT(id\_abonament, numar\_sedinte, tip\_abonament, pret)

values (6, 4, 'student', 80);

insert into

ABONAMENT(id\_abonament, numar\_sedinte, tip\_abonament, pret)

values (7, 16, 'elev', 320);

insert into

ABONAMENT(id\_abonament, numar\_sedinte, tip\_abonament, pret)

values (8, 48, 'elev', 960);

insert into

ABONAMENT(id\_abonament, numar\_sedinte, tip\_abonament, pret)

values (9, 4, 'copil', 75);

insert into

ABONAMENT(id\_abonament, numar\_sedinte, tip\_abonament, pret)

values (10, 20, 'student', 400);

commit;

select \* from ABONAMENT;

select \* from dansator;

alter table dansator

modify id\_profesor number(6) not null;

insert into

dansator(id\_dansator, nume, prenume, an\_nastere, experienta, id\_abonament, bonus, id\_profesor)

values (1, 'Popescu', 'Daria', 2004, 0, 1, null, 10);

select \* from dansator;

alter table dansator

modify id\_profesor number(6) not null;

insert into

dansator(id\_dansator, nume, prenume, an\_nastere, experienta, id\_abonament, bonus, id\_profesor)

values (1, 'Popescu', 'Daria', 2004, 0, 1, null, 10);

insert into

dansator(id\_dansator, nume, prenume, an\_nastere, experienta, id\_abonament, bonus, id\_profesor)

values (2, 'Careche', 'Bianca', 2006, 1, 2, null, 10);

insert into

dansator(id\_dansator, nume, prenume, an\_nastere, experienta, id\_abonament, bonus, id\_profesor)

values (3, 'Stefnache', 'Roxana', 2010, 1, 3, 100, 10);

insert into

dansator(id\_dansator, nume, prenume, an\_nastere, experienta, id\_abonament, bonus, id\_profesor)

values (4, 'Tinche', 'Iustina', 2006, 0, 4, 50, 30);

insert into

dansator(id\_dansator, nume, prenume, an\_nastere, experienta, id\_abonament, bonus, id\_profesor)

values (5, 'Resic', 'Carolina', 2002, 5, 5, 200, 30);

insert into

dansator(id\_dansator, nume, prenume, an\_nastere, experienta, id\_abonament, bonus, id\_profesor)

values (7, 'Bilius', 'Ana', 2009, 2, 7, null, 40);

insert into

dansator(id\_dansator, nume, prenume, an\_nastere, experienta, id\_abonament, bonus, id\_profesor)

values (8, 'Popescu', 'Catinca', 2003, 0, 8, null, 10);

insert into

dansator(id\_dansator, nume, prenume, an\_nastere, experienta, id\_abonament, bonus, id\_profesor)

values (9, 'Salachi', 'Carmen', 2001, 5, 9, 300, 40);

insert into

dansator(id\_dansator, nume, prenume, an\_nastere, experienta, id\_abonament, bonus, id\_profesor)

values (10, 'Pop', 'Stefania', 2004, 0, 10, null, 30);

insert into

dansator(id\_dansator, nume, prenume, an\_nastere, experienta, id\_abonament, bonus, id\_profesor)

values (6, 'Tirnovan', 'Sabrina', 2007, 0, 6, null, 10);

commit;

select \* from dansator;

insert into

sustinator(id\_sustinator, nume, prenume, suma)

values (1, 'Iordache', 'Cristina', 1200);

insert into

sustinator(id\_sustinator, nume, prenume, suma)

values (2, 'Erisam', 'Alex', 500);

insert into

sustinator(id\_sustinator, nume, prenume, suma)

values (3, 'Ticalache', 'Nicoleta', 600);

insert into

sustinator(id\_sustinator, nume, prenume, suma)

values (4, 'Chirita', 'Rodica', 300);

insert into

sustinator(id\_sustinator, nume, prenume, suma)

values (5, 'Ionescu', 'Maria', 200);

commit;

select \* from sustinator;

insert into

sponsorizeaza (id\_sustinator, id\_dansator, inceput\_perioada, sfarsit\_perioada)

values (1, 1, to\_date('01-05-2020', 'dd-mm-yyyy'), to\_date('14-07-2020'));

insert into

sponsorizeaza (id\_sustinator, id\_dansator, inceput\_perioada, sfarsit\_perioada)

values (1, 2, to\_date('07-05-2020', 'dd-mm-yyyy'), to\_date('21-07-2020'));

insert into

sponsorizeaza (id\_sustinator, id\_dansator, inceput\_perioada, sfarsit\_perioada)

values (2, 1, to\_date('13-01-2020', 'dd-mm-yyyy'), to\_date('13-09-2020'));

insert into

sponsorizeaza (id\_sustinator, id\_dansator, inceput\_perioada, sfarsit\_perioada)

values (2, 10, to\_date('01-05-2021', 'dd-mm-yyyy'), to\_date('01-05-2022'));

insert into

sponsorizeaza (id\_sustinator, id\_dansator, inceput\_perioada, sfarsit\_perioada)

values (3, 4, to\_date('11-01-2022', 'dd-mm-yyyy'), to\_date('14-07-2022'));

insert into

sponsorizeaza (id\_sustinator, id\_dansator, inceput\_perioada, sfarsit\_perioada)

values (3, 5, to\_date('21-11-2021', 'dd-mm-yyyy'), to\_date('21-12-2021'));

insert into

sponsorizeaza (id\_sustinator, id\_dansator, inceput\_perioada, sfarsit\_perioada)

values (4, 8, to\_date('27-03-2019', 'dd-mm-yyyy'), to\_date('13-08-2020'));

insert into

sponsorizeaza (id\_sustinator, id\_dansator, inceput\_perioada, sfarsit\_perioada)

values (4, 6, to\_date('07-07-2020', 'dd-mm-yyyy'), to\_date('31-07-2020'));

insert into

sponsorizeaza (id\_sustinator, id\_dansator, inceput\_perioada, sfarsit\_perioada)

values (5, 9, to\_date('07-04-2022', 'dd-mm-yyyy'), to\_date('04-09-2022'));

insert into

sponsorizeaza (id\_sustinator, id\_dansator, inceput\_perioada, sfarsit\_perioada)

values (5, 4, to\_date('11-05-2020', 'dd-mm-yyyy'), to\_date('14-07-2020'));

commit;

select \* from sponsorizeaza;

select \* from categorie;

insert into

categorie (id\_categorie, nume\_categorie)

values (1, 'national');

insert into

categorie (id\_categorie, nume\_categorie)

values (2, 'regional');

insert into

categorie (id\_categorie, nume\_categorie)

values (3, 'local');

insert into

categorie (id\_categorie, nume\_categorie)

values (4, 'judetean');

insert into

categorie (id\_categorie, nume\_categorie)

values (5, 'mondial');

insert into

categorie (id\_categorie, nume\_categorie)

values (6, 'american');

commit;

select \* from categorie;

select \* from oras;

insert into

oras (id\_oras, denumire\_oras)

values ('B', 'Bucuresti');

insert into

oras (id\_oras, denumire\_oras)

values ('SV', 'Suceava');

insert into

oras (id\_oras, denumire\_oras)

values ('PH', 'Prahova');

insert into

oras (id\_oras, denumire\_oras)

values ('IS', 'Iasi');

insert into

oras (id\_oras, denumire\_oras)

values ('CJ', 'Cluj');

insert into

oras (id\_oras, denumire\_oras)

values ('GL', 'Galati');

commit;

select \* from sala\_concurs;

insert into

sala\_concurs (id\_salaconcurs, nume\_salaconcurs, id\_oras)

values (1, 'Dance sport', 'B');

insert into

sala\_concurs (id\_salaconcurs, nume\_salaconcurs, id\_oras)

values (2, 'Dancing in rain', 'CJ');

insert into

sala\_concurs (id\_salaconcurs, nume\_salaconcurs, id\_oras)

values (3, 'Butterfly sport', 'IS');

insert into

sala\_concurs (id\_salaconcurs, nume\_salaconcurs, id\_oras)

values (4, 'Funny dance', 'SV');

insert into

sala\_concurs (id\_salaconcurs, nume\_salaconcurs, id\_oras)

values (5, 'Summer dances', 'PH');

insert into

sala\_concurs (id\_salaconcurs, nume\_salaconcurs, id\_oras)

values (6, 'Dance kids show', 'GL');

commit;

select \* from sala\_concurs;

alter table profesor

modify id\_saladans number(6) not null;

select \* from profesor;

alter table concurs

modify id\_salaconcurs number(6) not null;

alter table concurs

modify id\_categorie number(6) not null;

alter table sala\_concurs

modify id\_oras char(2) not null;

insert into

concurs (id\_concurs, nume, ziua\_concurs, id\_categorie, id\_salaconcurs)

values (1, 'Latina Dance', 'Sambata', 1, 1);

insert into

concurs (id\_concurs, nume, ziua\_concurs, id\_categorie, id\_salaconcurs)

values (2, 'Humans dance', 'Duminica', 3, 2);

insert into

concurs (id\_concurs, nume, ziua\_concurs, id\_categorie, id\_salaconcurs)

values (3, 'Cha-Cha Dance', 'Vineri', 2, 4);

insert into

concurs (id\_concurs, nume, ziua\_concurs, id\_categorie, id\_salaconcurs)

values (4, 'Dance in pair', 'Joi', 5, 3);

insert into

concurs (id\_concurs, nume, ziua\_concurs, id\_categorie, id\_salaconcurs)

values (5, 'Spring dance time', 'Marti', 4, 6);

insert into

concurs (id\_concurs, nume, ziua\_concurs, id\_categorie, id\_salaconcurs)

values (6, 'newDance', 'Joi', 6, 5);

insert into

concurs (id\_concurs, nume, ziua\_concurs, id\_categorie, id\_salaconcurs)

values (7, 'Plus Dance', 'Luni', 6, 6);

insert into

concurs (id\_concurs, nume, ziua\_concurs, id\_categorie, id\_salaconcurs)

values (8, 'American Dance', 'Vineri', 3, 2);

commit;

select \* from concurs;

insert into

proba\_dans(id\_proba\_dans, nume\_dans)

values (1, 'vals');

insert into

proba\_dans(id\_proba\_dans, nume\_dans)

values (2, 'tango');

insert into

proba\_dans(id\_proba\_dans, nume\_dans)

values (3, 'bachata');

insert into

proba\_dans(id\_proba\_dans, nume\_dans)

values (4, 'latin');

insert into

proba\_dans(id\_proba\_dans, nume\_dans)

values (5, 'samba');

insert into

proba\_dans(id\_proba\_dans, nume\_dans)

values (6, 'sportiv');

select \* from proba\_dans;

commit;

select \* from participa;

insert into

participa (id\_dansator, id\_concurs, id\_proba\_dans, premiu)

values (1, 1, 1, 1);

insert into

participa (id\_dansator, id\_concurs, id\_proba\_dans, premiu)

values (2, 3, 4, 1);

insert into

participa (id\_dansator, id\_concurs, id\_proba\_dans, premiu)

values (3, 2, 1, 0);

insert into

participa (id\_dansator, id\_concurs, id\_proba\_dans, premiu)

values (5, 6, 4, 0);

insert into

participa (id\_dansator, id\_concurs, id\_proba\_dans, premiu)

values (9, 3, 5, 0);

insert into

participa (id\_dansator, id\_concurs, id\_proba\_dans, premiu)

values (9, 1, 3, 1);

insert into

participa (id\_dansator, id\_concurs, id\_proba\_dans, premiu)

values (8, 1, 6, 0);

insert into

participa (id\_dansator, id\_concurs, id\_proba\_dans, premiu)

values (8, 3, 2, 1);

insert into

participa (id\_dansator, id\_concurs, id\_proba\_dans, premiu)

values (1, 3, 4, 1);

insert into

participa (id\_dansator, id\_concurs, id\_proba\_dans, premiu)

values (2, 4, 4, 0);

select \* from participa;

commit;

select \* from dosar\_dansator;

insert into

dosar\_dansator(data\_rez, id\_dosar\_dansator, id\_dansator, id\_nota)

values (to\_date('16-05-2021', 'dd-mm-yyyy'), 1, 1, 1);

insert into

dosar\_dansator(data\_rez, id\_dosar\_dansator, id\_dansator, id\_nota)

values (to\_date('17-05-2021', 'dd-mm-yyyy'), 1, 1, 2);

insert into

dosar\_dansator(data\_rez, id\_dosar\_dansator, id\_dansator, id\_nota)

values (to\_date('21-06-2021', 'dd-mm-yyyy'), 2, 2, 3);

insert into

dosar\_dansator(data\_rez, id\_dosar\_dansator, id\_dansator, id\_nota)

values (to\_date('30-08-2021', 'dd-mm-yyyy'), 2, 2, 4);

insert into

dosar\_dansator(data\_rez, id\_dosar\_dansator, id\_dansator, id\_nota)

values (to\_date('26-01-2022', 'dd-mm-yyyy'), 3, 3, 5);

insert into

dosar\_dansator(data\_rez, id\_dosar\_dansator, id\_dansator, id\_nota)

values (to\_date('18-05-2022', 'dd-mm-yyyy'), 4, 5, 6);

insert into

dosar\_dansator(data\_rez, id\_dosar\_dansator, id\_dansator, id\_nota)

values (to\_date('28-02-2021', 'dd-mm-yyyy'), 5, 8, 7);

insert into

dosar\_dansator(data\_rez, id\_dosar\_dansator, id\_dansator, id\_nota)

values (to\_date('01-05-2022', 'dd-mm-yyyy'), 5, 8, 8);

insert into

dosar\_dansator(data\_rez, id\_dosar\_dansator, id\_dansator, id\_nota)

values (to\_date('11-11-2021', 'dd-mm-yyyy'), 6, 9, 9);

insert into

dosar\_dansator(data\_rez, id\_dosar\_dansator, id\_dansator, id\_nota)

values (to\_date('16-05-2021', 'dd-mm-yyyy'), 6, 9, 10);

select \* from dosar\_dansator;

commit;

select \* from nota;

insert into

nota (nota\_obtinuta, id\_nota, id\_proba\_dans, id\_dansator, id\_dosar\_dansator)

values (9, 1, 1, 1, 1);

insert into

nota (nota\_obtinuta, id\_nota, id\_proba\_dans, id\_dansator, id\_dosar\_dansator)

values (9, 2, 4, 1, 1);

insert into

nota (nota\_obtinuta, id\_nota, id\_proba\_dans, id\_dansator, id\_dosar\_dansator)

values (8, 3, 4, 2, 2);

insert into

nota (nota\_obtinuta, id\_nota, id\_proba\_dans, id\_dansator, id\_dosar\_dansator)

values (7, 4, 4, 2, 2);

insert into

nota (nota\_obtinuta, id\_nota, id\_proba\_dans, id\_dansator, id\_dosar\_dansator)

values (6, 5, 1, 3, 3);

insert into

nota (nota\_obtinuta, id\_nota, id\_proba\_dans, id\_dansator, id\_dosar\_dansator)

values (5, 6, 4, 5, 4);

insert into

nota (nota\_obtinuta, id\_nota, id\_proba\_dans, id\_dansator, id\_dosar\_dansator)

values (9, 7, 6, 8, 5);

insert into

nota (nota\_obtinuta, id\_nota, id\_proba\_dans, id\_dansator, id\_dosar\_dansator)

values (8, 8, 2, 8, 5);

insert into

nota (nota\_obtinuta, id\_nota, id\_proba\_dans, id\_dansator, id\_dosar\_dansator)

values (7, 9, 3, 9, 6);

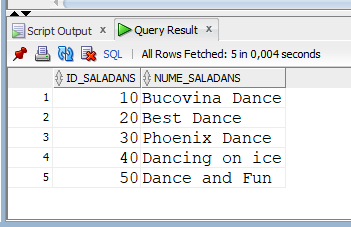
insert into

nota (nota\_obtinuta, id\_nota, id\_proba\_dans, id\_dansator, id\_dosar\_dansator)

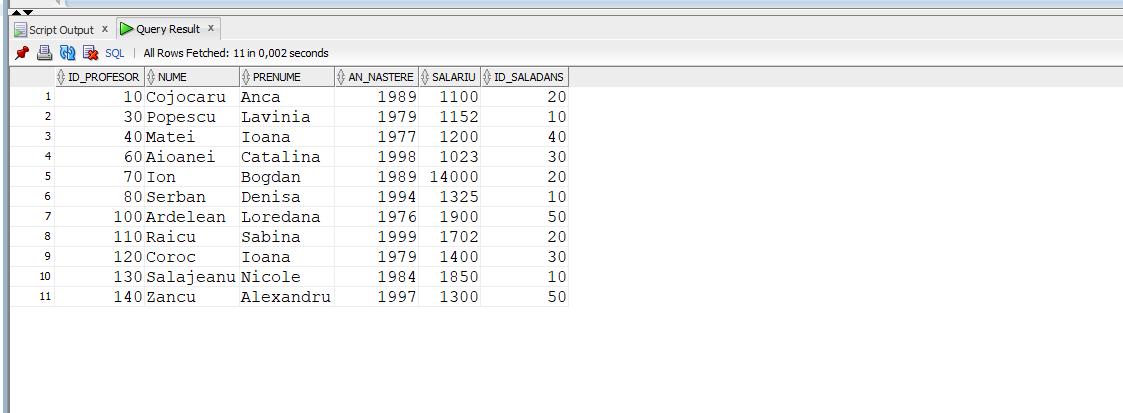
values (6, 10, 5, 9, 6);

commit;

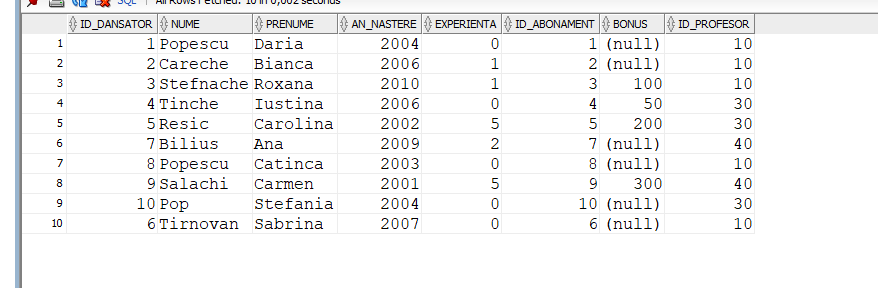
select \* from sala\_de\_dans;



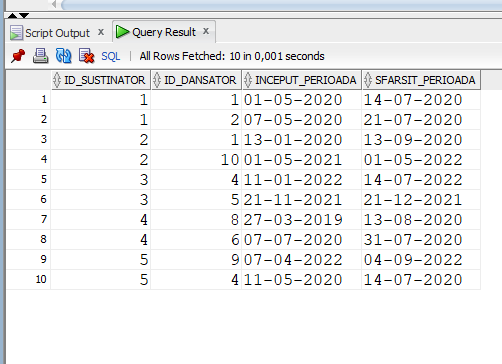
select \* from profesor;



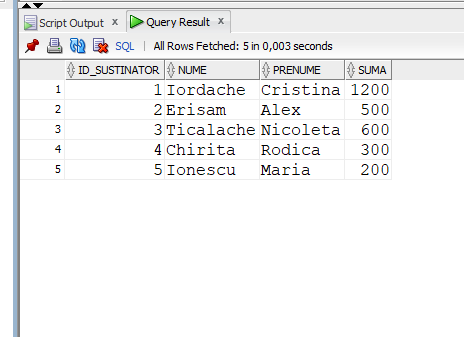
select \* from dansator;



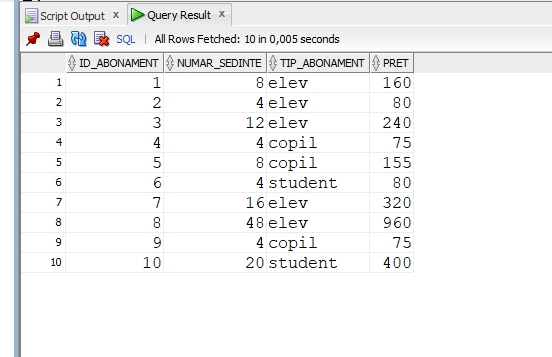
select \* from sponsorizeaza;



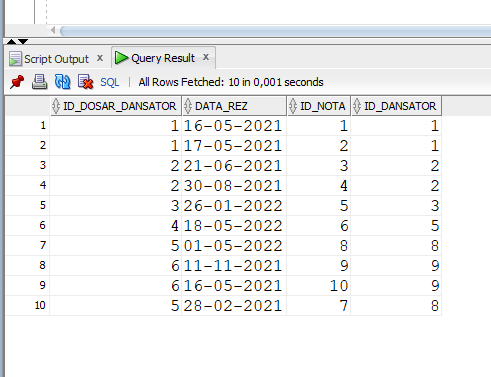
select \* from sustinator;



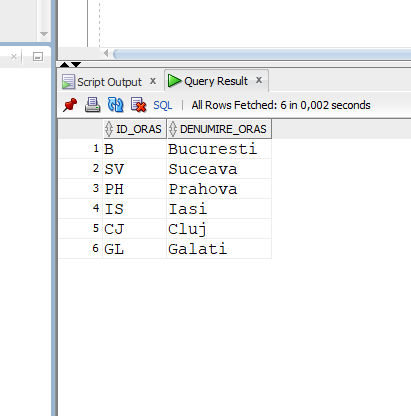
select \* from abonament;



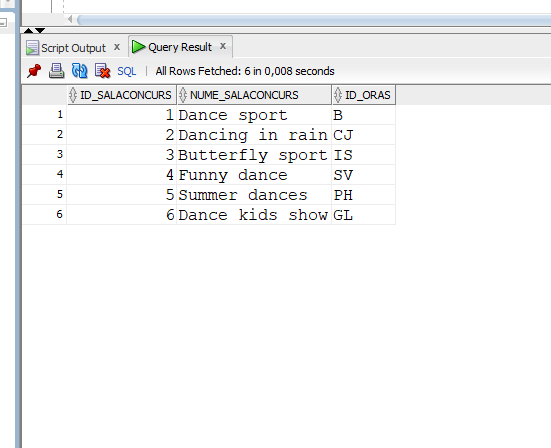
select \* from dosar\_dansator;



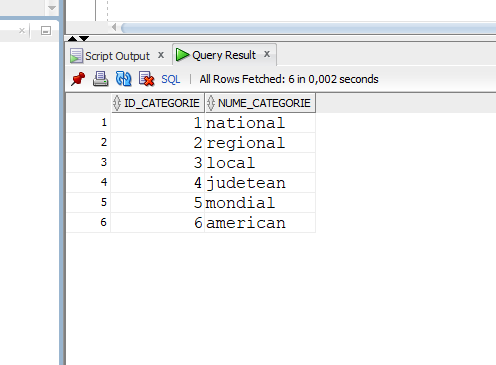
select \* from oras;



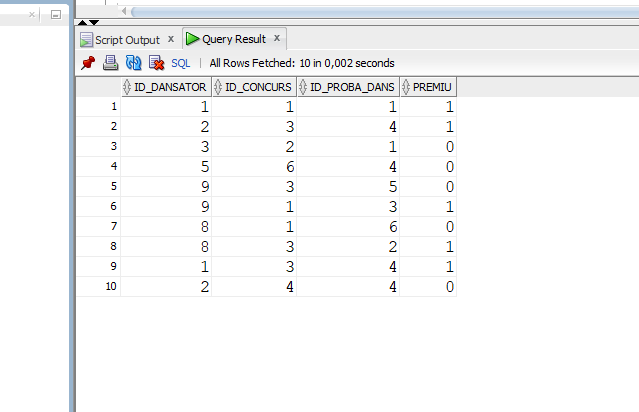
select \* from sala\_concurs;



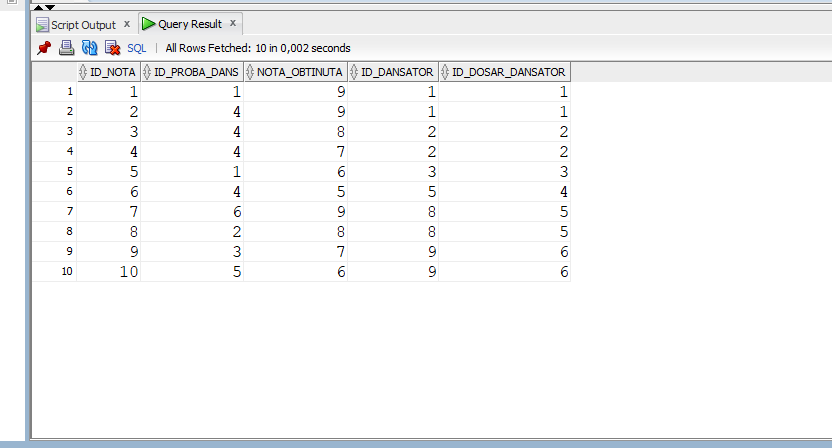
select \* from categorie;



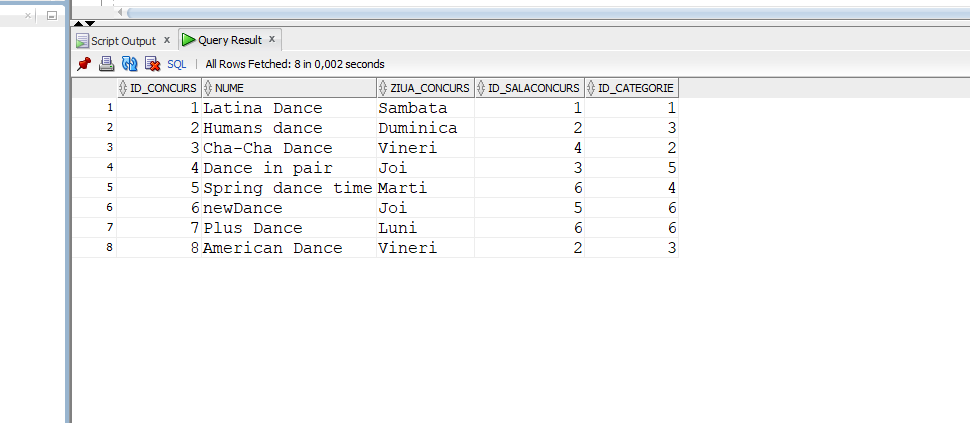
select \* from participa;



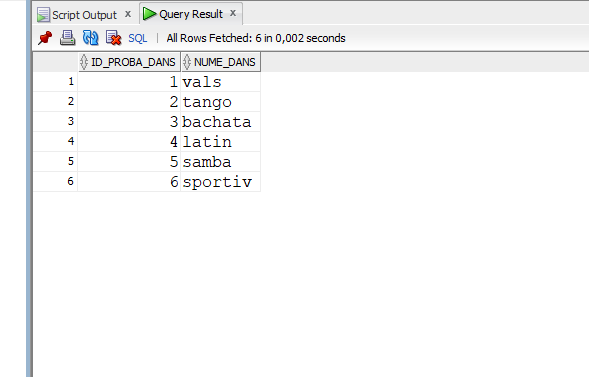
select \* from nota;



select \* from concurs;



select \* from proba\_dans;



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

**--exercitiul 1**: sa afisam id-ul dansatorului, numele, prenumele, bonusul sau, daca au primit premii (ordonam dupa numele dansatorului)

--ziua concursului la care au participat

--pentru cei ce sunt in grupa speciala, adica primesc bonus ( nu este 0) / cei care participa la

--concursuri in care denumirea acestuia cuvantul dance

--amandoua cazuri trebuie sa fi participat la concursuri ce au fost in ziua de vineri

--in aceasta problema pentru rezolvarea ei vom folosi NVL, concat pentru ca numele si prenumele sa fie concatenate

--in coloana, upper, ordonare, join pe 4 tabele, filtrare pe linii, substr

select d.id\_dansator as "id dansator", concat (d.nume, ' ') || d.prenume as "Numele dansatorului",

nvl (d.bonus, 0) "bonusul dansatorului", p.premiu, c.nume as "Nume concurs", c.ziua\_concurs, cat.nume\_categorie

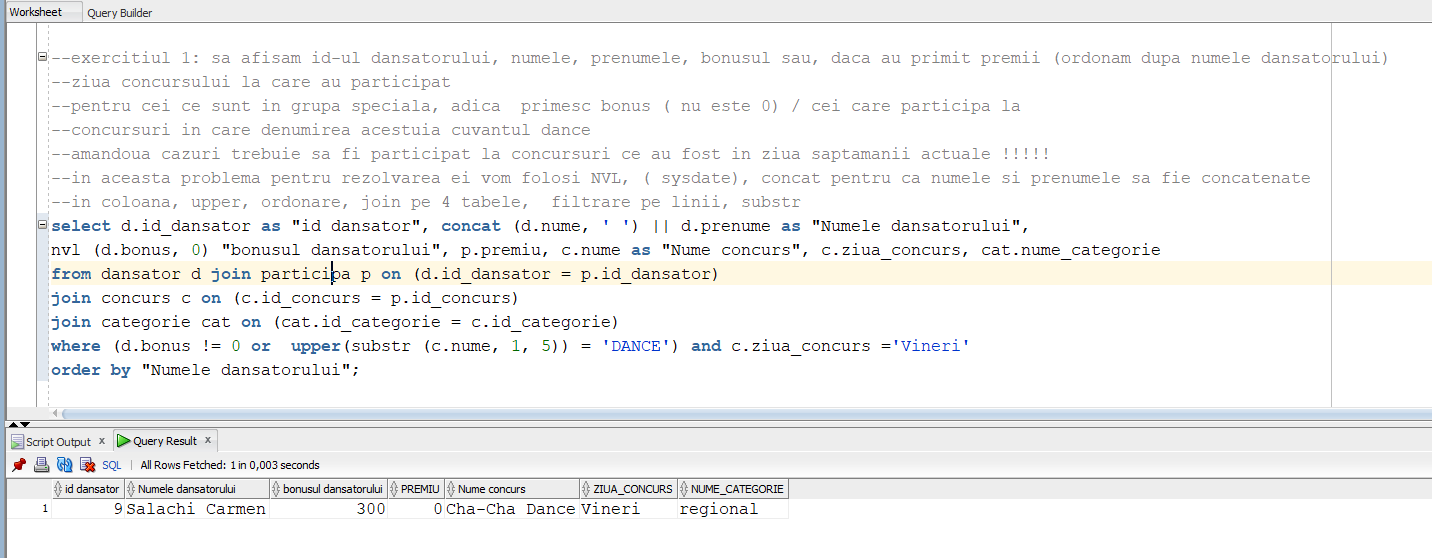
from dansator d join participa p on (d.id\_dansator = p.id\_dansator)

join concurs c on (c.id\_concurs = p.id\_concurs)

join categorie cat on (cat.id\_categorie = c.id\_categorie)

where (d.bonus != 0 or upper(substr (c.nume, 1, 5)) = 'DANCE') and c.ziua\_concurs ='Vineri'

order by "Numele dansatorului";



**---exercitiul 2**: sa se afiseze id-ul, numele salilor de dans si bonusul maxim si minim pe sala de dans (fara null), pentru cei

--salile de dans unde bonsul maxim este mai mare de 150 si bonusul minim mai mic de 60.

--folosim filtrare pe grupuri si functiile min, max

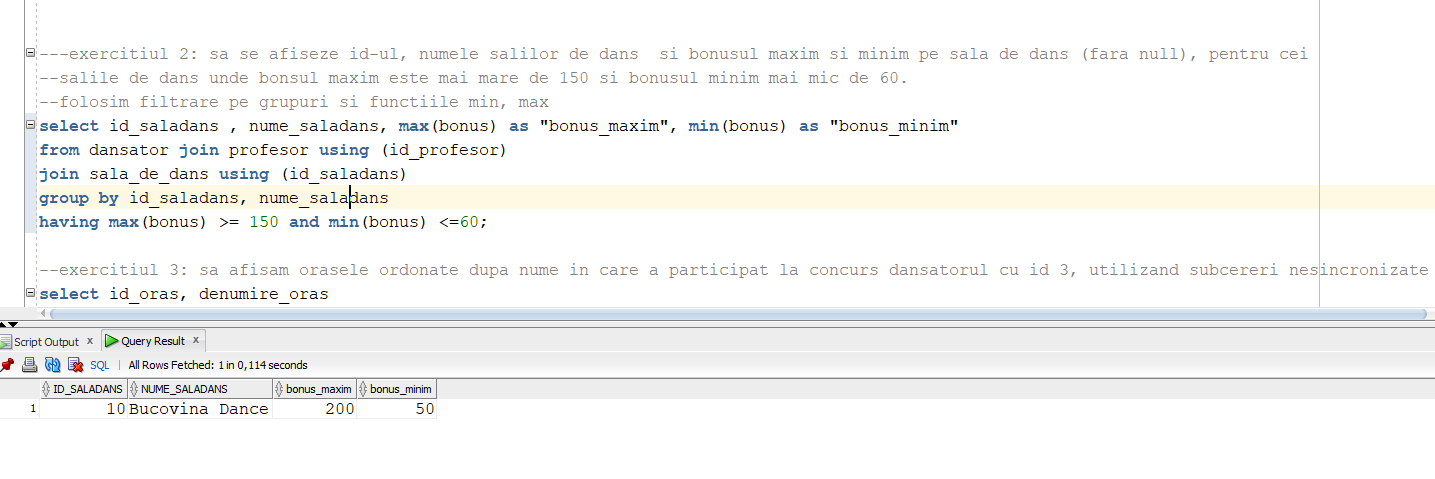
select id\_saladans , nume\_saladans, max(bonus) as "bonus\_maxim", min(bonus) as "bonus\_minim"

from dansator join profesor using (id\_profesor)

join sala\_de\_dans using (id\_saladans)

group by id\_saladans, nume\_saladans

having max(bonus) >= 150 and min(bonus) <=60;



**--exercitiul 3**: sa afisam orasele ordonate dupa nume in care a participat la concurs dansatorul cu id 3, utilizand subcereri nesincronizate

select id\_oras, denumire\_oras

from oras

where id\_oras in (select id\_oras

from sala\_concurs

where id\_salaconcurs in (select id\_salaconcurs

from concurs

where id\_concurs in (select id\_concurs

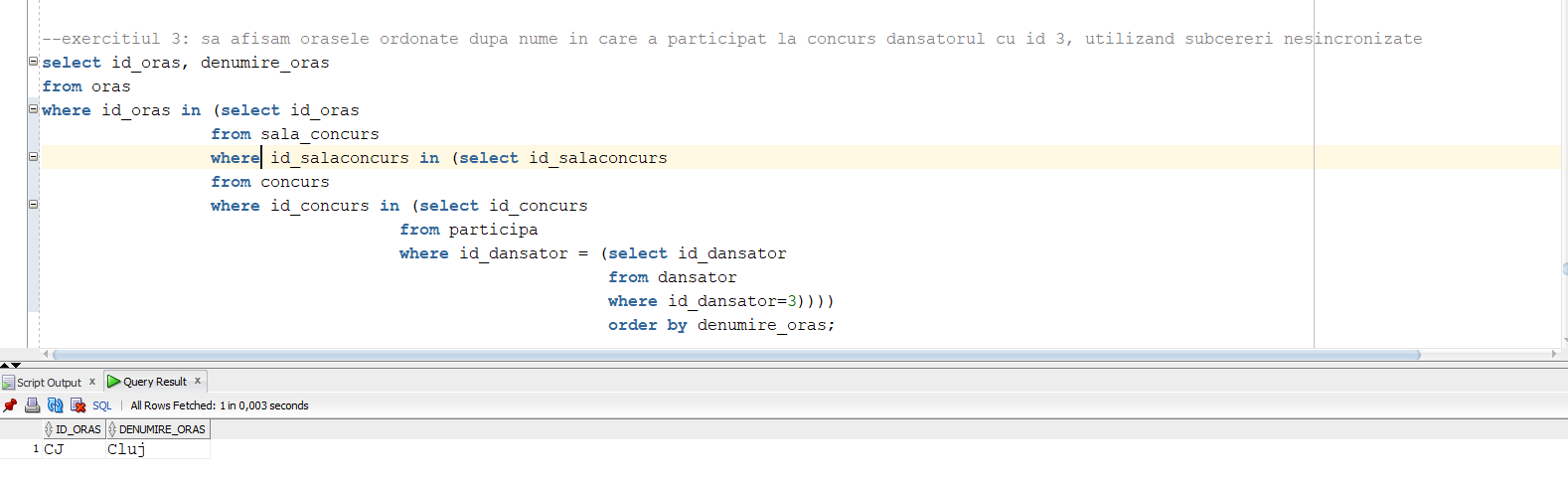
from participa

where id\_dansator = (select id\_dansator

from dansator

where id\_dansator=3))))

order by denumire\_oras;



**--exercitiul 4**: sa afisam id sala de dans, suma salariilor oferite profesorilor si o coloana separata in care vom spune daca

--suma\_salariu este mare (peste 1500)

--suma\_salariu este buna (intre 1190- 1200)

--restul uma este mica

--ne folosim de with, case

with sume as (select id\_saladans, sum(salariu) suma\_salariu,

from profesor

group by id\_saladans

order by id\_saladans )

select id\_saladans, suma\_salariu,

case when suma\_salariu > 1500 then 'suma salariu este mare'

when suma\_salariu < 1200 and suma\_salariu > 1190 then 'suma salariu este buna'

else 'suma salariu este mica'

end as 'cum este suma salariilor?'

from sume;

!!!e corect, dar nu imi merge aplicatia sa fac print screen

**--exercitiul 5**: sa inseram in tabela dansator dansatorul cu id 11, nume Resic, prenume Ioana, an nastere 1979,

--id abonament 7, bonus null, id profesor 10

--sa afisam numele si prenumele dansatorului nascut in acelasi an cu profesorul

--dupa rulare stegem inserarea

--formam/afisam o coloana separata in care daca dansatorul are bonusul 0 atunci scriem dansatorul nu face parte

--din grupa speciala

--daca are bonus de 300lei atunci scriem dansator performant

--daca are alta suma diferita de 0 scriem dansator din grupa speicala

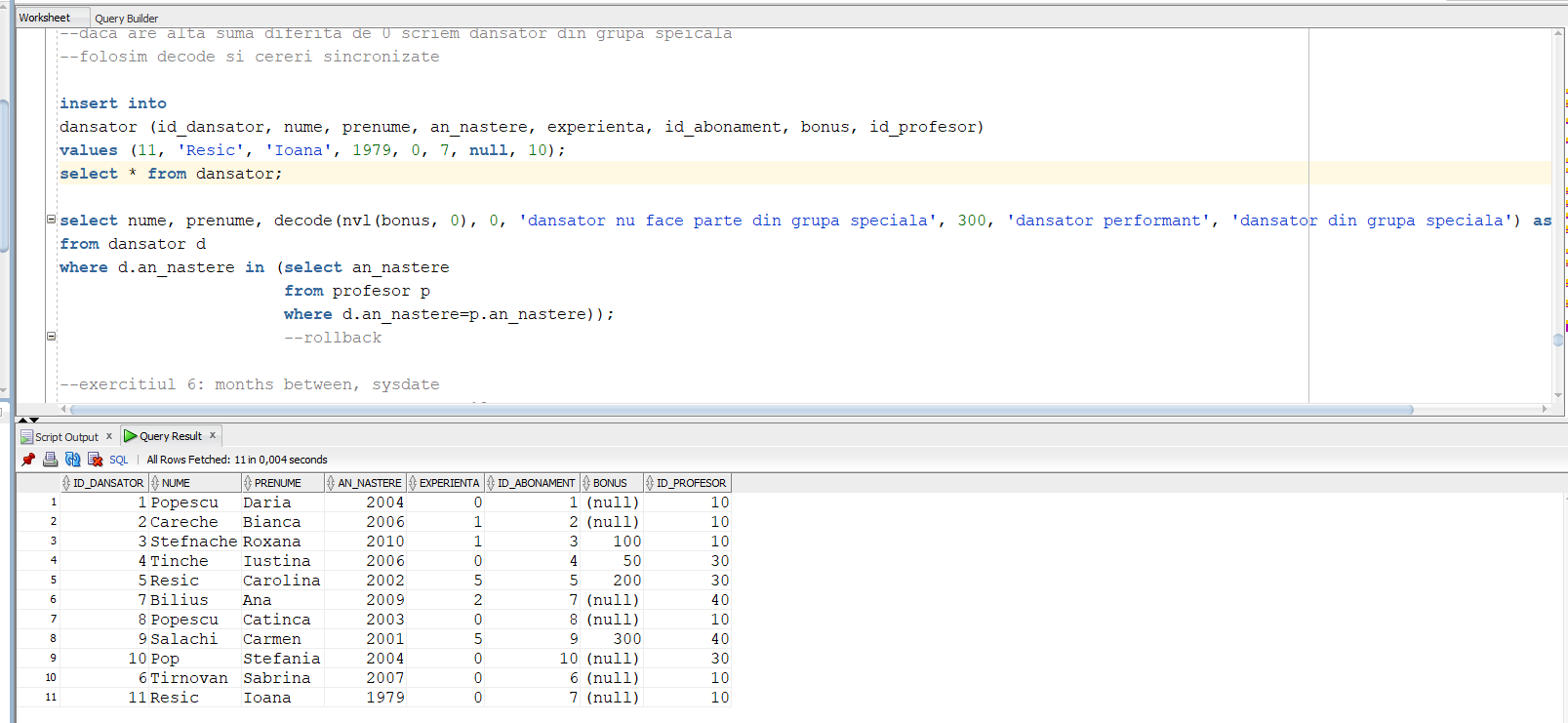
--folosim decode si cereri sincronizate

insert into

dansator (id\_dansator, nume, prenume, an\_nastere, experienta, id\_abonament, bonus, id\_profesor)

values (11, 'Resic', 'Ioana', 1979, 0, 7, null, 10);

--dupa inserare, tabela va arata asa:



select nume, prenume, decode(nvl(bonus, 0), 0, 'dansator nu face parte din grupa speciala', 300, 'dansator performant', 'dansator din grupa speciala') as "Performanti"

from dansator d

where d.an\_nastere in (select an\_nastere

from profesor p

where d.an\_nastere=p.an\_nastere));

rollback;

**--exercitiul 6:**

**13.Implementarea a 3 operații de actualizare și de suprimare a datelor utilizând subcereri**

**--ex1:** crestem cu 30% salariul profesorilor de la bucovina dance, dupa anulam modificarile

update profesor

set salariu=salariu\*1.3

where salariu in (select salariu

from profesor

where id\_saladans=(select id\_saladans

from sala\_de\_dans

where nume\_saladans=upper('DANCING ON ICE')));

rollback;

**--ex2**: eliminam categoriile de concursuri care nu au niciun concurs cu acea categorie

select \* from concurs;

select \* from categorie;

delete

from categorie

where id\_categorie not in (select unique id\_categorie

from concurs);

rollback;

--nu se sterge nimic, deoarece toate categoriile, sunt folosite, adica exista concursuri de acea categorie, de aceea nu ne ajuta cu nimic daca dam rollback

**--ex3**: celor care au obtinut premiu, bonusul le creste cu 20%

update dansator

set bonus = bonus\*1.2

where id\_dansator in (select unique id\_dansator

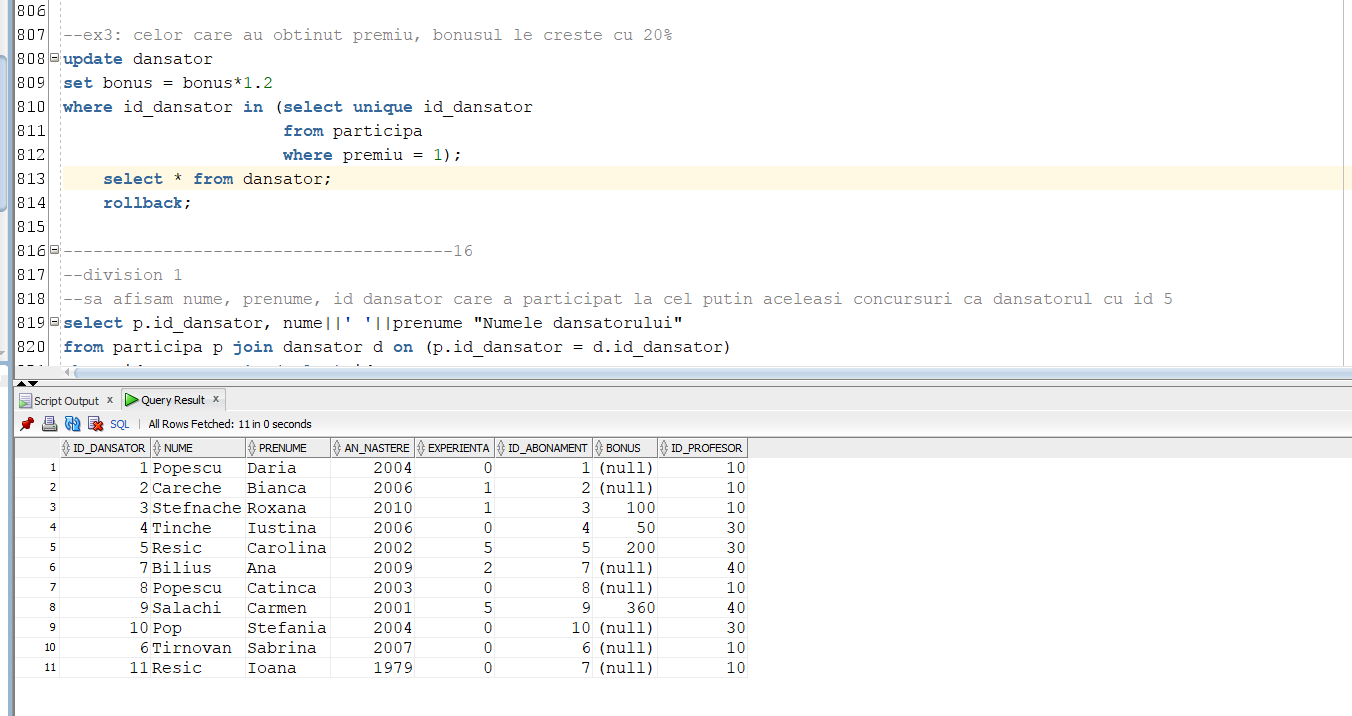
from participa

where premiu = 1);

select \* from dansator;

rollback;

--inainte de rollback arata asa:



**16.Formulați în limbaj natural și implementați în SQL: o cerere ce utilizează operația outer-join pe minimum 4 tabele și două cereri ce utilizează operația division.**

**--division 1**

--sa afisam nume, prenume, id dansator care a participat la cel putin aceleasi concursuri ca dansatorul cu id 5

select p.id\_dansator, nume||' '||prenume "Numele dansatorului"

from participa p join dansator d on (p.id\_dansator = d.id\_dansator)

where id\_concurs in (select id\_concurs

from participa

where id\_dansator = 5

)

and p.id\_dansator != 5

group by p.id\_dansator, nume, prenume

having count(\*) = (select count(id\_concurs)

from participa

where id\_dansator = 5

);

--nu exista niciun rezultat

**--division 2**

--afisam numele, prenumele dansatoruilor care merg la aceleasi concursuri cu cel cu id ul 1

select p.id\_dansator, nume||' '||prenume "Numele dansatorului"

from participa p join dansator d on (p.id\_dansator = d.id\_dansator)

where id\_concurs in (select id\_concurs

from participa

where id\_dansator= 1

)

and p.id\_dansator != 1

group by p.id\_dansator, nume, prenume

having count(\*) = (select count(id\_concurs)

from participa

where id\_dansator= 1

)

minus

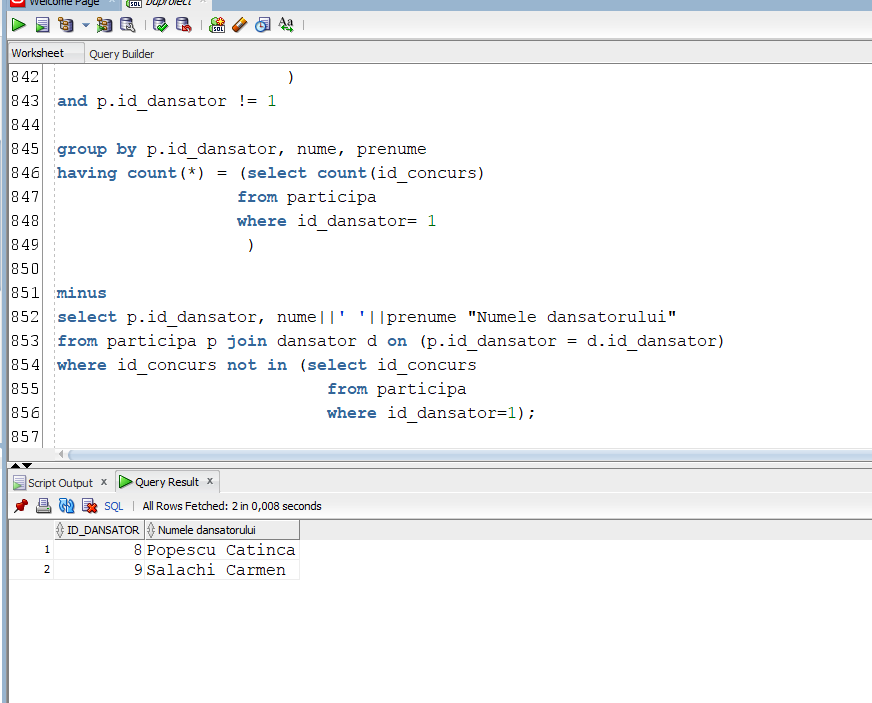
select p.id\_dansator, nume||' '||prenume "Numele dansatorului"

from participa p join dansator d on (p.id\_dansator = d.id\_dansator)

where id\_concurs not in (select id\_concurs

from participa

where id\_dansator=1);



**--outer join pe 4 tabele**

--afisam nume prenume pentru dansatori, si sponsorii sai (numele), numele probelor la care au fost participanti si fiecare sponsor al probei

select d.nume as "numele dansatorului", d.prenume as "prenumele dansatorului", sust.nume as "nume sustinator", sust.prenume as "prenume sustinator",

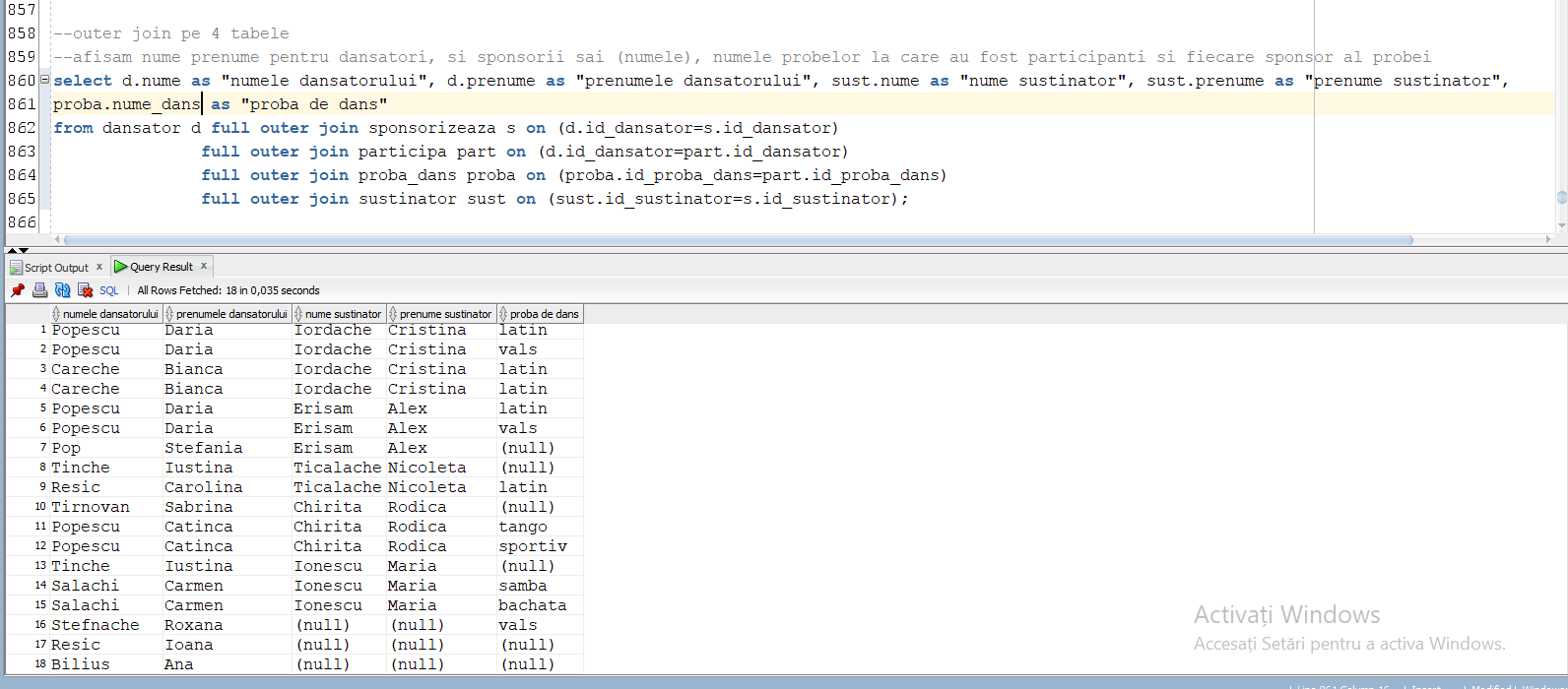
proba.nume\_dans as "proba de dans"

from dansator d full outer join sponsorizeaza s on (d.id\_dansator=s.id\_dansator)

full outer join participa part on (d.id\_dansator=part.id\_dansator)

full outer join proba\_dans proba on (proba.id\_proba\_dans=part.id\_proba\_dans)

full outer join sustinator sust on (sust.id\_sustinator=s.id\_sustinator);



**17.Optimizarea unei cereri, aplicând regulile de optimizare ce derivă din proprietățile operatorilor algebrei relaționale. Cererea va fi exprimatăprin expresie algebrică, arbore algebric și limbaj (SQL), atât anterior cât și ulterior optimizării.**

Alegem cererea de la exercitiul 12 (ex1).

**--exercitiul 1**: sa afisam id-ul dansatorului, numele, prenumele, bonusul sau, daca au primit premii (ordonam dupa numele dansatorului)

--ziua concursului la care au participat

--pentru cei ce sunt in grupa speciala, adica primesc bonus ( nu este 0) / cei care participa la

--concursuri in care denumirea acestuia cuvantul dance

--amandoua cazuri trebuie sa fi participat la concursuri ce au fost in ziua de vineri

--in aceasta problema pentru rezolvarea ei vom folosi NVL, concat pentru ca numele si prenumele sa fie concatenate

--in coloana, upper, ordonare, join pe 4 tabele, filtrare pe linii, substr

**CODUL SQL:**

select d.id\_dansator as "id dansator", concat (d.nume, ' ') || d.prenume as "Numele dansatorului",

nvl (d.bonus, 0) "bonusul dansatorului", p.premiu, c.nume as "Nume concurs", c.ziua\_concurs, cat.nume\_categorie

from dansator d join participa p on (d.id\_dansator = p.id\_dansator)

join concurs c on (c.id\_concurs = p.id\_concurs)

join categorie cat on (cat.id\_categorie = c.id\_categorie)

where (d.bonus != 0 or upper(substr (c.nume, 1, 5)) = 'DANCE') and c.ziua\_concurs ='Vineri'

order by "Numele dansatorului";

**CERERE EXPRIMATA PRIN EXPRESII ALGEBRICE:**

R1=select(dansator ,bonus!=0)

R2=project(dansator, id\_dansator, nume, prenume, bonus)

R3=project(participa, premiu, id\_dansator, id\_concurs)

R4=join(R2, R3, id\_dansator)

R5=select(concurs, upper(substr(c.nume, 1, 5)) = 'DANCE') and c.ziua\_concurs ='Vineri'

R6=project(R5, id\_categorie, nume, ziua\_concurs, id\_concurs)

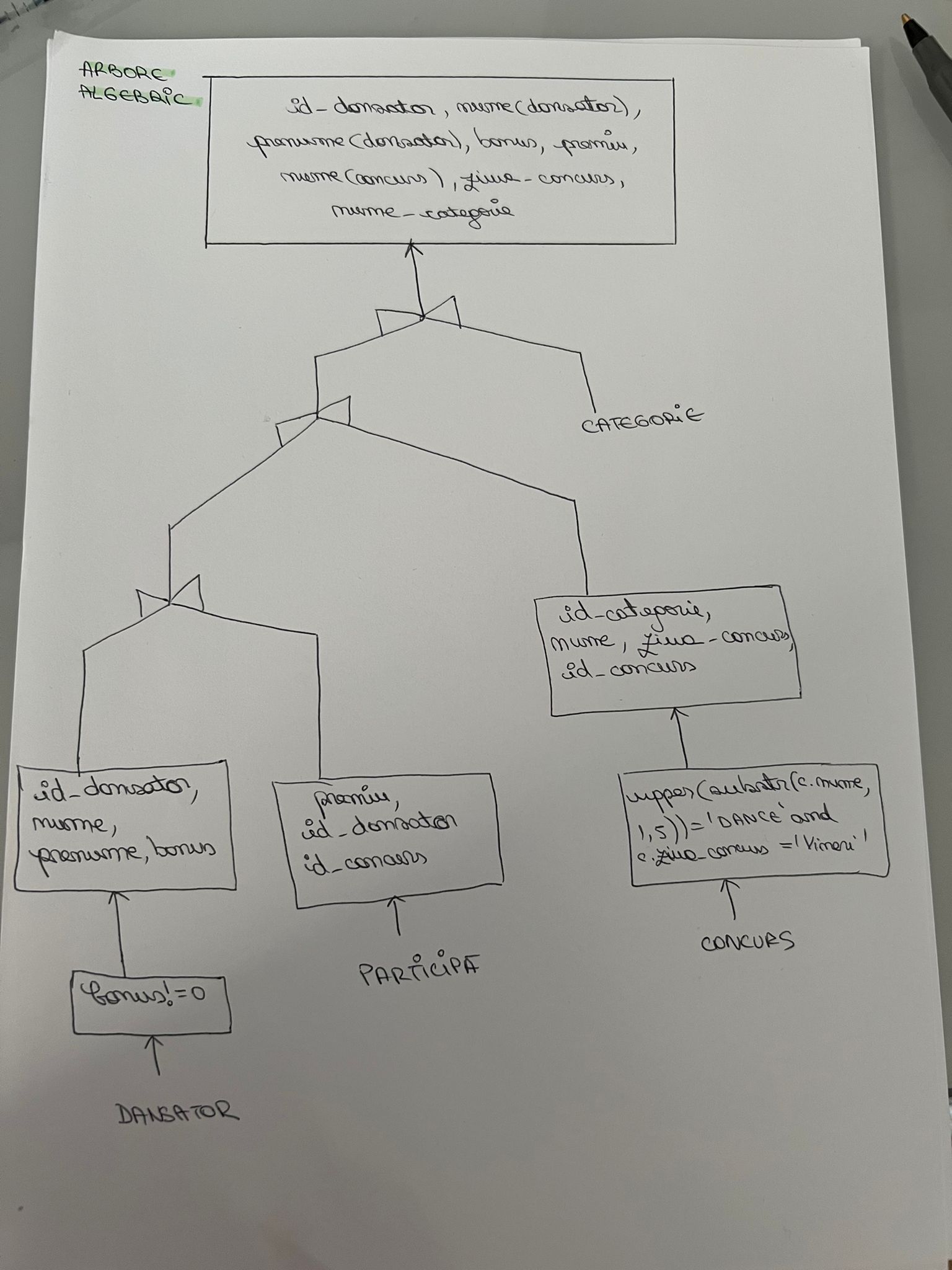
R7=join(R4,R6)

R8=join(R7, categorie)

rezultatafisat=R9=project(R8, id\_dansator, nume(dansator), prenume, bonus, premiu, nume(concurs), ziua\_concurs, nume\_categorie)

Cererea este una optima, pentru ca atunci cand s-a creat, selectiile s-au executat devreme, la fel si proiectiile, produsele carteziene s-au inlocuit cu join-uri. Asadar, s-a realizat comutarea selectiei cu proiectia si comutarea proiectiei cu operatia de join (intai se realizeaza proiectia, apoi join pentru a nu se efectua join pe toate coloanele tabelei).

**ARBORE ALGEBRC**



**18. a. Realizarea normalizării BCNF, FN4, FN5.**

**b. Aplicarea denormalizării, justificând necesitatea acesteia.**

**☺BCNF**

O relatie este in forma normala BCNF doar daca fiecare determinant reprezinta o cheie candidat. Cream un exemplu fictiv, nu este valabil pentru diagrama noastra. Sa spunem ca data\_rez din DOSAR\_DANSATOR este unic si determina atributul id\_nota# (atribut care e in componenta cheii primare).

DOSAR\_DANSATOR\_FICTIV

|  |  |  |  |
| --- | --- | --- | --- |
| data\_rez | id\_dosar\_dansator# | id\_nota# | id\_dansator# |
| 04-02-2020 | 2 | 1 | 2 |
| 17-02-2022 | 2 | 2 | 2 |
| 21-04-2022 | 3 | 3 | 3 |
| 03-02-2020 | 3 | 4 | 3 |

In relatia dintre atributele din DOSAR\_DANSATOR\_FICTIV exista urmatoarele dependente:

(id\_dosar\_dansator#, id\_nota#, id\_dansator#)-data\_rez

(data\_rez)-id\_nota#

data\_rez=cheie candidat; id\_nota depinde de data\_rez

Aplicam regula Casey-Delobel pentru a forma relatia in BCNF:

-DOSAR\_DANSATOR\_F (id\_nota, data\_rez#)

-DOSAR\_DANSATOR\_FF( id\_dosar\_dansator#, id\_dansator#, data\_rez#); data\_rez este in componenta cheii primare compuse, singur e cheie externa

DOSAR\_DANSATOR\_F:

|  |  |
| --- | --- |
| data\_rez# | id\_nota |
| 04-02-2020 | 1 |
| 17-02-2022 | 2 |
| 21-04-2022 | 3 |
| 03-02-2020 | 4 |

DOSAR\_DANSATOR\_FF:

|  |  |  |
| --- | --- | --- |
| data\_rez# | id\_dosar\_dansator# | id\_dansator# |
| 04-02-2020 | 2 | 2 |
| 17-02-2022 | 2 | 2 |
| 21-04-2022 | 3 | 3 |
| 03-02-2020 | 3 | 3 |

**☺FN4**

O relatie este in FN4 doar daca se afla in BCNF si nu are relatii x:y independente.

Luam un exemplu fictiv, un dansator are mai multi sustinatori si dosare de dansator. Daca inseram un dansator, atunci va trebui sa inseram si sponsorul si dosarul.

|  |  |  |
| --- | --- | --- |
| id\_dansator# | id\_dosar\_dansator# | id\_sustinator# |
| 2 | 2 | 1 |
| 2 | 2 | 3 |
| 5 | 6 | 2 |
| 5 | 3 | 2 |

Relatia dansator\_nonfn4 (id\_dansator#, id\_abonament#, id\_sustinator#) se afla in BCNF.

Dependente:

Id\_dansator-id\_dosar\_dansator

Id\_dansator-id\_sustinator

Aplicam FN4 si rezulta:

Dansator\_1(id\_dansator#, id\_dosar\_dansator#)

Dansator \_2 (id\_dansator#, id\_sustinator)

Dansator\_1:

|  |  |
| --- | --- |
| Id\_dansator# | Id\_dosar\_dansator# |
| 2 | 2 |
| 5 | 6 |
| 5 | 3 |

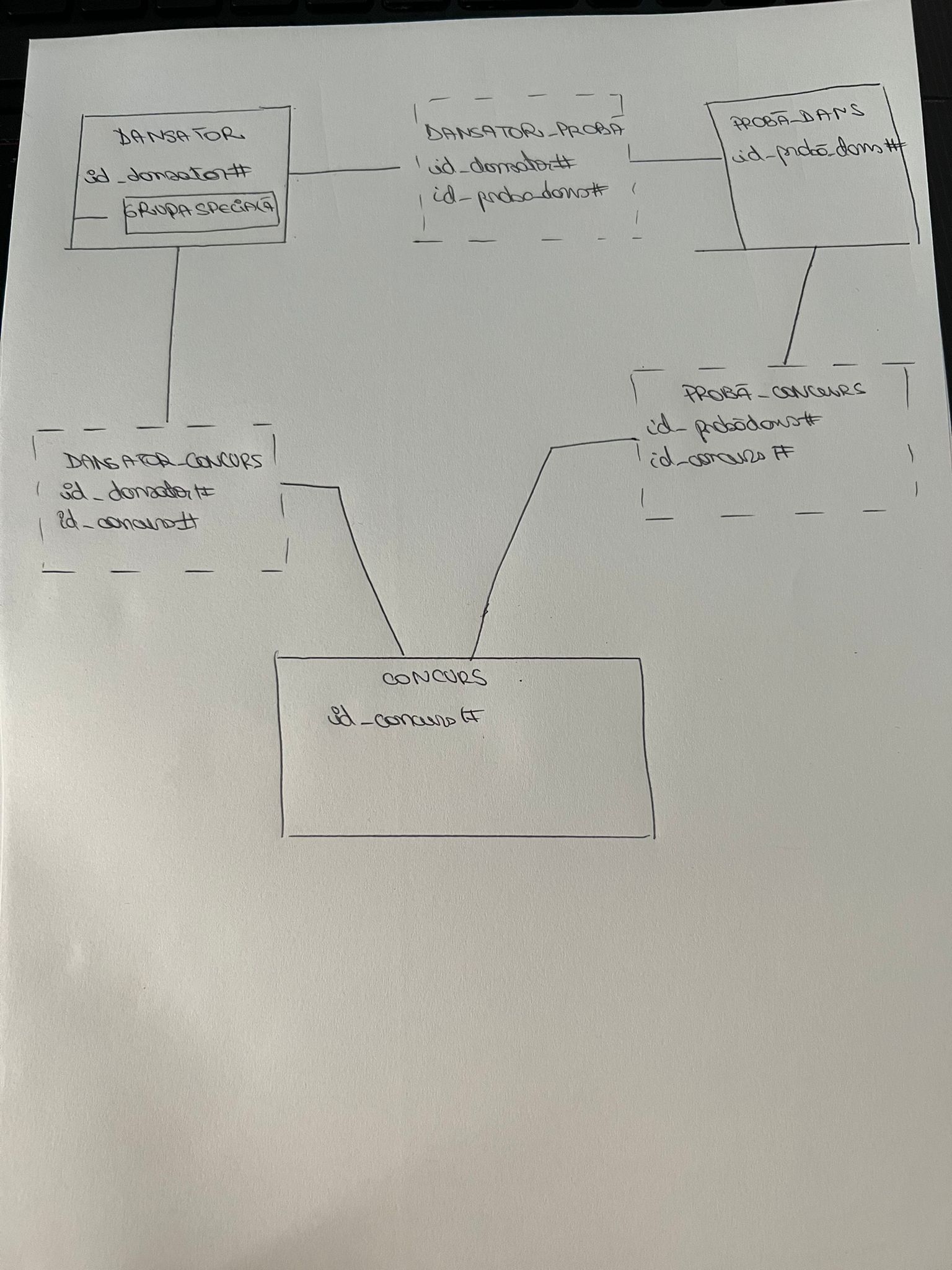
Dansator\_2:

|  |  |
| --- | --- |
| Id\_dansator# | Id\_dosar\_dansator# |
| 2 | 1 |
| 2 | 3 |
| 5 | 2 |

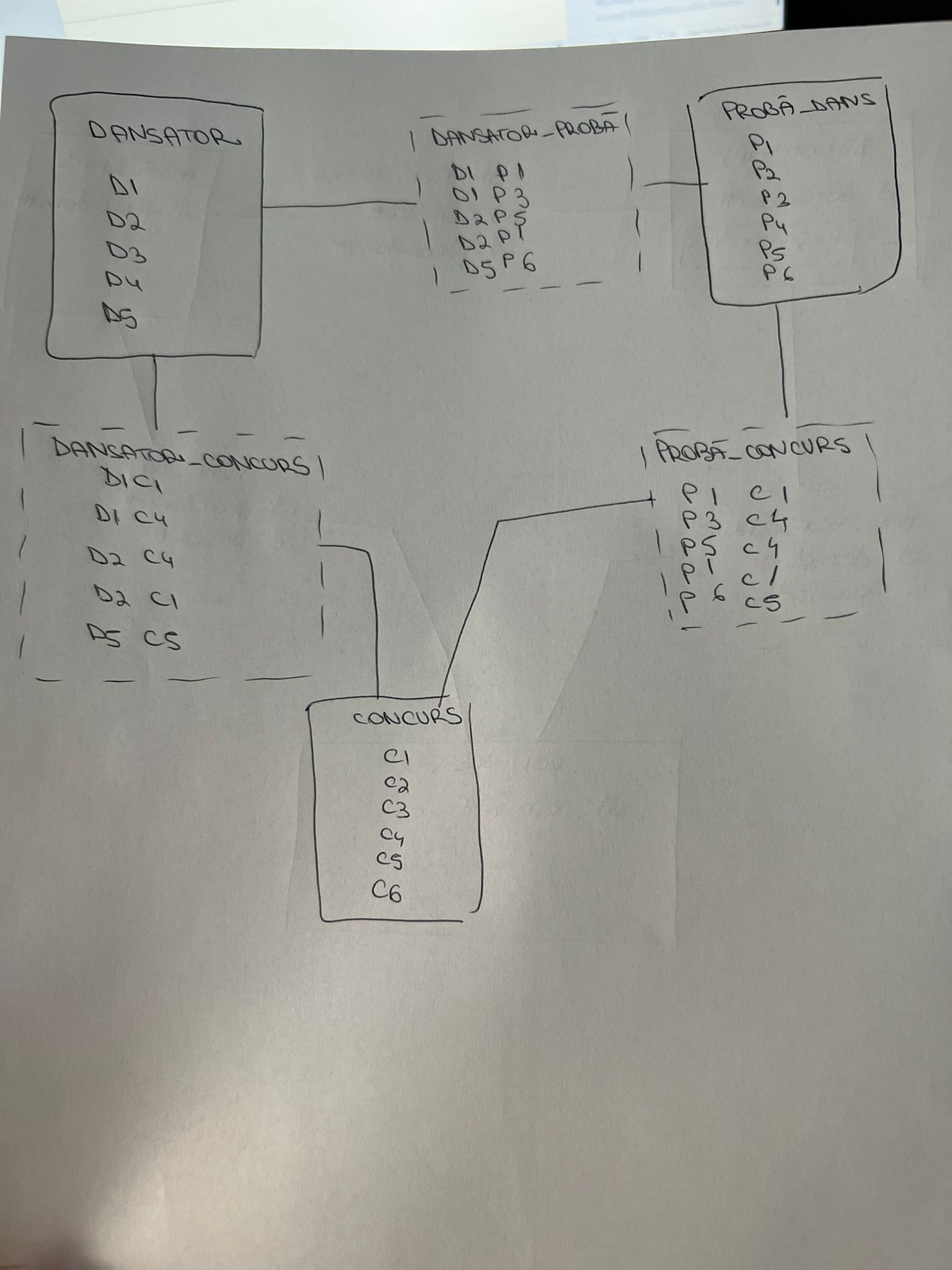
**☺FN5**

O relatie este in FN5 doar daca este in FN4 si nu are dependente ciclice.

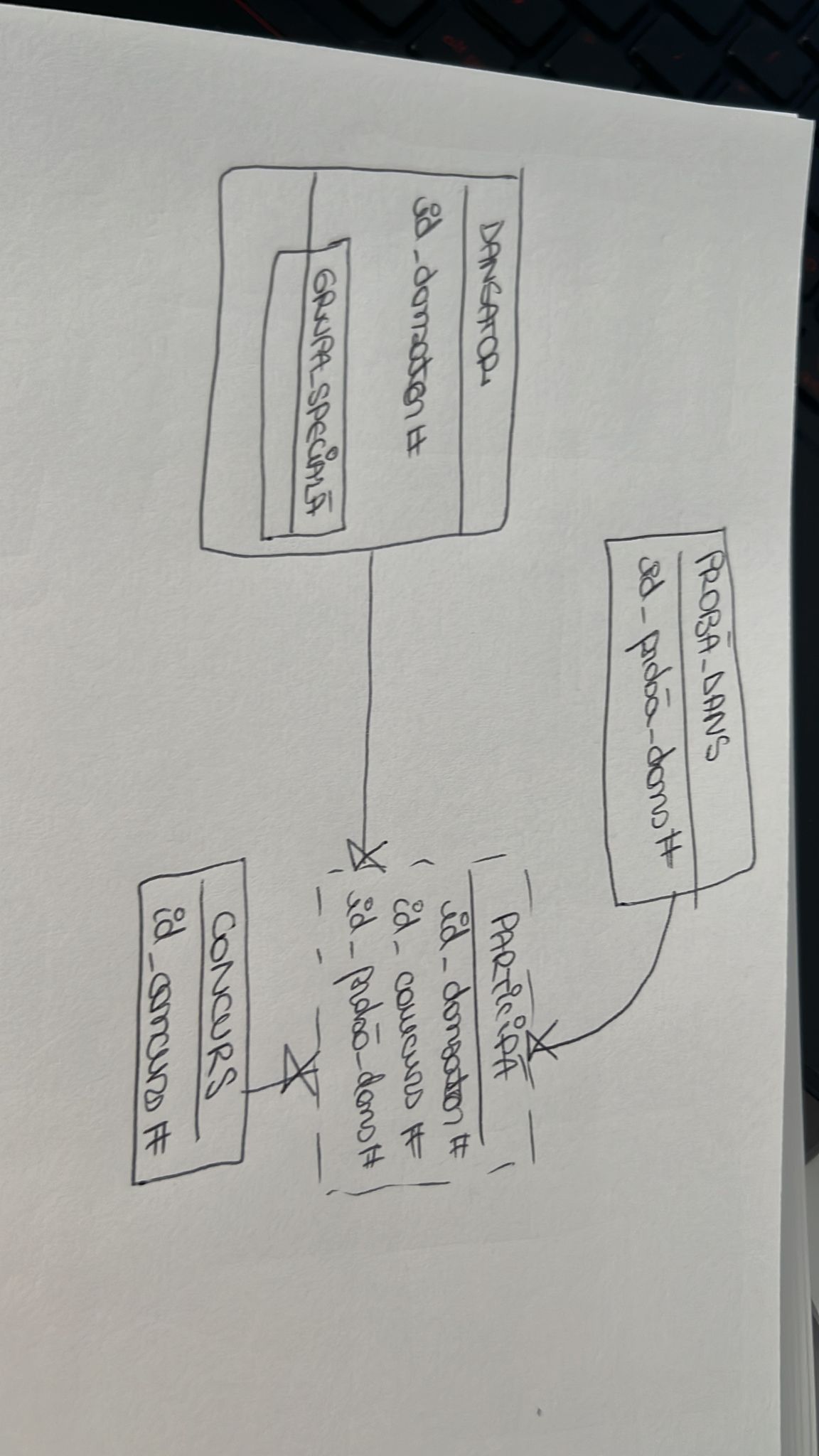
Vom crea un exemplu pentru aceasta. Pornim de la felul in care arata o parte din diagrama inainte de normalizare, stabilim numai cheile tabelelor.



Prezentam relatia dintre chei. Indicii aici vor avea o litera specifica (D-dansator, C-concurs, P-proba de dans).



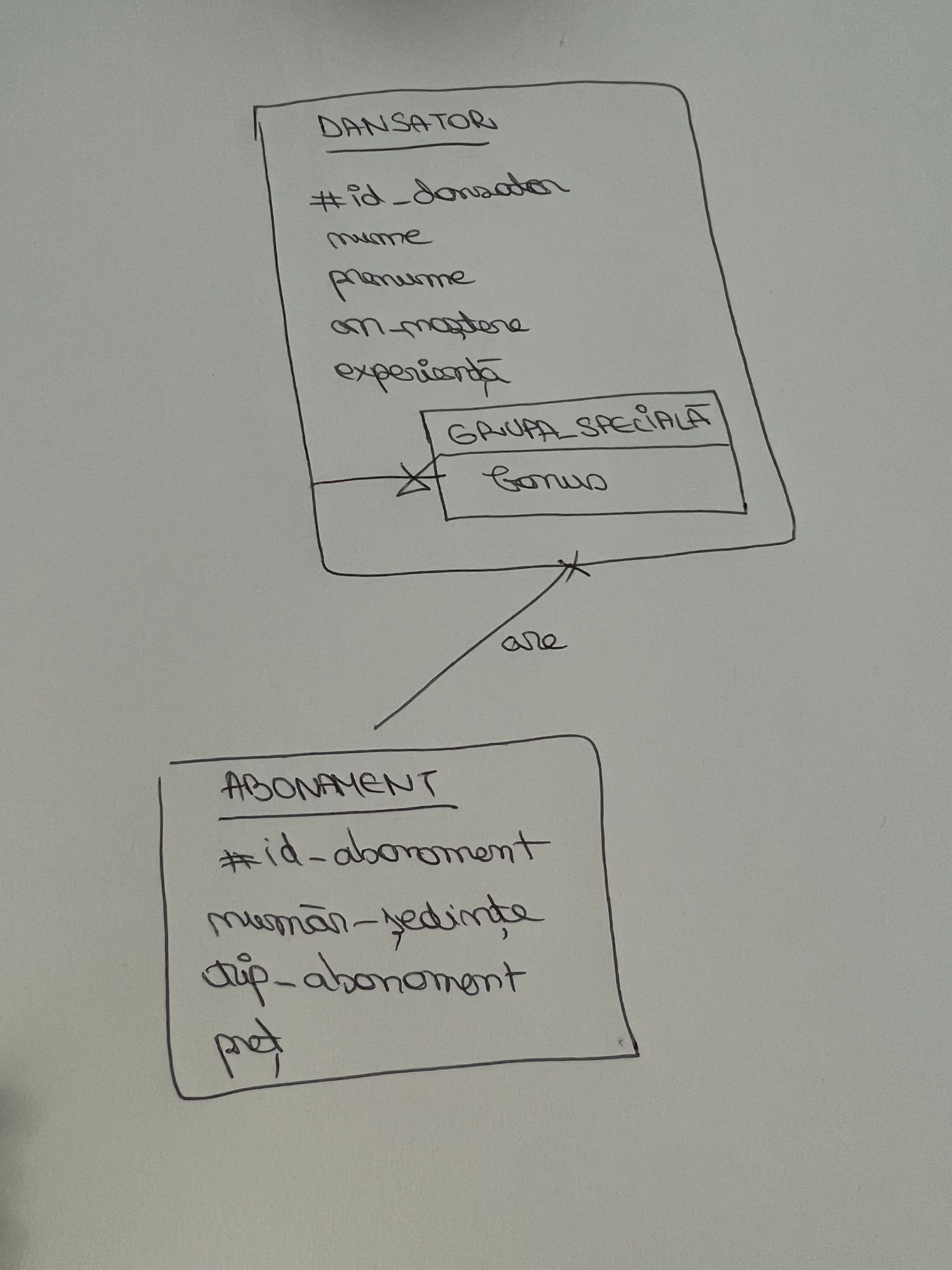
Cele 3 relatii de tip 2 au compus o diagrama cu dependente ciclice, adica nu se afla in FN5. Relatia fiind ciclica, rezulta ca are o relatie de tip 3 echivalenta. Totodata, relatia se afla in FN4, pentru ca dansatorul are multidependenta la concurs si proba. Dupa eliminarea ciclului, diagrama va fi o relatie de tip 3 aratand asa:

**Denormalizarea**

-se realizeaza pentru a reduce numarul de join-uri ce trebuie efectuate pentru rezolvarea interogarii

Exemplificam denormalizarea cu ajutorul celor 2 tabele DANSATOR si ABONAMENT. In baza de date create de mine, exista 2 tabele separate. Relatia dintre cele doua este 1:1.

Cele 2 tabele inainte de denormalizare:



Dupa denormalizare, atributul abonament va fi pus in tabelul DANSATOR, astfel eliminandu-se o operatie de jon, diagrama devenind mai eficeinta. Denormalizarea trebuia efectuata, deoarece nu era eficient ca atributele din abonament sa se afle intr-un tabel diferit (asa, implica un join in plus pentru a accesa ABONAMENT).

Dupa denormalizare:

