Baze de Date - Anul 1

Gestiunea Unei Universități

Broscoteanu Daria Mihaela

Baze de date – Anul I
Seria 14

GESTIUNEA UNEI UNIVERSITĂȚI

Broscoțeanu Daria-Mihaela

Grupa 143

Studiu de caz – Ex. 1

Vom prezenta modelul de date, restricţiile pe care trebuie să le respecte şi vom încerca, într-o manieră didactică, să construim diagrama E/R corespunzătoare. Vom considera, în abordarea iniţială, anumite situaţii care nu sunt optime, în sensul că pot genera redundanţă, anomalii la reactualizări sau nu permit rezolvarea anumitor interogări asupra modelului. Vom încerca să arătăm care sunt deficienţele modelului, situaţiile care le-au generat şi cum pot fi corectate (parţial sau total) anomaliile respective.

Baza de date conține informații cu privire la facultățile unei universități, secțiile si grupele acestora, cât și profesorii care predau în cadrul universității, ce cursuri și căror studenți le predau aceștia, dacă sunt voluntari in cadrul unei asociaţii sau nu, cât și notele obținute de elevi în cadrul unui curs care se va desfăşura într-o anume sală.

Scopul creării acestei baze de date este de a putea ține evidența studenților din cadrul universității, a situației lor școlare, cât și cursurilor predate în cadrul fiecărei facultăți, cât și a profesorului care susține cursul.

Fiecare student este repartizat la o secție, într-o serie și grupă, participând la anumite cursuri predate de profesori care au posibilitatea de a avea contracte cu mai multe facultăți din cadrul universității. Studenții pot lua parte la acțiuni de voluntariat în cadrul unei asociații studențești.

Cursurile la care iau parte studenții sunt susținute la nivel de grupă, în diferite săli ale facultății, de către profesori diferiți.

Un profesor are contracte cu facultățile la care predă, acesta putând să predea diferite cursuri în cadrul unei facultăți.

Un examen poate lua mai multe forme: proiect, examen scris sau interviu. Forma acestuia diferă în funcție de materia la care este susținut. Un student promovează un examen în cazul în care obține o notă mai mare sau egală cu 5.

Restricţii de funcţionare – Ex. 2

Modelul de date respectă anumite restricţii de funcţionare.

- Un profesor este angajat la cel puțin o facultate, iar o facultate angajează cel puțin un profesor.
- O facultate are cel puțin o secție.
- O secție are cel puțin o serie.
- O serie este alcătuită din cel puțin o grupă.
- O grupă este formată din mai mulți studenți, iar un student poate să fie înscris la mai multe grupe.
- Un student poate să fie membru intr-o asociație de voluntariat.
- O asociație de voluntariat aparține doar unei facultăți.
- Un student promovează unul sau mai multe examene, un examen este promovat de unul sau mai mulți studenți.
- Un examen aparține unui curs, un curs poate avea mai multe examene.
- Un curs este predat de unul sau mai mulți profesori. Un curs este predat unei sau mai multor grupe. Un curs se poate desfășura în una sau mai multe săli.
- Un curs poate fi seminar, laborator sau curs.
- Un examen poate fi proiect, lucrare scrisă sau interviu.
- O asociație are cel puțin un membru.
- O grupă este formată din cel puțin un student.

Entităţi – Ex. 3

Pentru modelul de date referitor la gestiunea unei universităţi, structurile FACULTATE, SECTIE, SERIE, GRUPA, STUDENT, ASOCIATIE, SALA, PROFESOR, CURS, EXAMEN reprezintă entităţi.

Vom prezenta entităţile modelului de date, dând o descriere completă a fiecăreia. De asemenea, pentru fiecare entitate se va preciza cheia primară.

Toate entităţile care vor fi prezentate sunt independente, cu excepţia entităţilor dependente INSCRIERE, PROMOVEAZA, ORAR şi CONTRACT.

FACULTATE = instituţia în care se desfăşoară procesul de informare al studenţilor. Modelul de date consideră doar facultăţile semnificative din cadrul unei universităţi. Cheia primară este id_facultate.

SECTIE = modalitate de clasificare a studenţilor în cadrul facultăţii. In funcţie de aceasta, între doi studenţi aparţinând a două serii diferite pot să difere profesorii, cursurile, grupele şi sălile. Cheia primară este id_sectie.

SERIE = modalitate de clasificare a studenţilor în cadrul unei secţii. Studenţii aflaţi în aceeaşi serie au aceeaşi profesori în cadrul cursurilor. Cheia primară este id_serie.

GRUPA = modalitate de clasificare a studenţilor în cadrul unei serii. Studenţii aflaţi în aceeaşi grupa au aceeaşi profesori atat in cadrul laboratoarelor, al cursurilor, cât şi al seminariilor. Un student poate face parte din mai multe grupe în cazul în care este înscris la mai multe domenii. Cheia primară este id_grupa.

STUDENT = persoana fizică, aparţinând unei facultăţi, care studiază în cadrul universităţii prezentate. Stundentul poate lua parte la activităţi de voluntariat în cadrul asociaţiei de la facultatea la care studiază. Cheia primară este id_student.

ASOCIATIE = organizaţie de voluntariat la care pot să ia parte stundenţii din cadrul unei facultăţi. Cheia primară este id_asociatie.

EXAMEN = formă de evaluare scrisă sau prin prezentare de proiect prin care un student işi primeşte nota de la o materie. Promovarea examenului implica obţinerea notei 5. Cheia primară este id_examen.

CURS = materia pe care o studiază un student pe parcursul unui semestru. Un student poate avea cursuri, seminarii şi laboratoare în cadrul unei materii. Cheia primară este id_curs.

SALA = locatia de desfasurare a unui curs. Intr-o sală se pot desfăşura mai multe cursuri, la intervale orare diferite. Salile pot sa fie amfiteatre, sali de clasa si laboratoare. Cheia primară este id_sala.

PROFESOR = persoana fizică, angajat al unei facultăţi, care predă în cadrul universităţii prezentate. Profesorul poate preda atât cursuri diferite, cât şi la mai multe facultăţi. Profesorul poate preda atât cursuri, cât şi laboratoare şi seminarii. Cheia primară este id_profesor.

INSCRIERE =cuprinde informatii despre grupele din care face parte un student (în cazul în care este înscris la mai multe secţii sau facultăţi în acelaşi timp). Cheia primară compusă este formată din id_grupa si id_student.

ORAR = programul unei facultăţi, cuprinzând cursurile, laboratoarele, seminariile, programarea lor în săli, cât şi grupele la care participă la o acesta, alături de profesorul care predă. Cheia primară compusă este formată din id_grupa, id_profesor, id_sala și id_curs.

CONTRACT = act care cuprinde informaţii despre durata angajării şi salariul unui profesor în cadrul unei facultăţi. Cum un profesor poate preda la mai multe facultăţi, salariul poate să fie diferit în funcţie de facultate. Cheia primară compusă este alcătuită din id_profesor și id_facultate.

PROMOVEAZA = identifică nota la un examen si data la care această notă a fost acordată studentului în cadrul unui curs. Cheia primară compusă este alcătuită din id_examen și id_student.

RELAȚII – Ex. 4

Vom prezenta relaţiile modelului de date, dând o descriere completă a fiecăreia. De fapt, denumirile acestor legături sunt sugestive, reflectând conţinutul acestora şi entităţile pe care le leagă. Pentru fiecare relaţie se va preciza cardinalitatea minimă şi maximă.

cardinalitatea minimă şi maximă.

PROFESOR_este_angajat_la_FACULTATE = relaţie de tip many-to-many care leagă entităţile PROFESOR şi FACULTATE, reflectând legătura dintre acestea (la ce facultate este angajat un anumit profesor). Ea are cardinalitatea minimă 1:0 (o facultate trebuie să aibă cel puțin un profesor angajat şi un profesor poate să fie angajat la cel puțin o facultate) şi cardinalitatea maximă m:*n* (o facultate poate avea mai mulți profesori angajați – de regulă, aceasta este situația – și un profesor poate să predea la mai multe facultăți).

FACULTATE_are_ASOCIATIE = relaţie care leagă entităţile ASOCIATIE şi FACULTATE, reflectând legătura dintre acestea (pentru o facultate, există o asociație). Relaţia are cardinalitatea minimă și maximă 1:1.

FACULTATE_are_SECTIE = relaţie care leagă entităţile FACULTATE şi SECTIE, reflectând legătura dintre acestea (pentru o facultate, exista una sau mai multe secții). Relaţia are cardinalitatea minimă 1:1 şi cardinalitatea maximă 1:*n*.

SECTIE_are_SERIE = relaţie dintre SECTIE şi SERIE, reflectând legătura dintre acestea (ce serii din facultate sunt la o anumită secție). Ea are cardinalitatea minimă 1:1 şi cardinalitatea maximă 1:*n*.

SERIE_are_GRUPE = relaţie dintre entităţile SERIE şi GRUPE, reflectând legătura dintre acestea (ce grupe alcătuiesc o anumită serie în cadrul unei secții). Ea are cardinalitatea minimă 1:1 şi cardinalitatea maximă 1:*n*.

STUDENT_este_membru_in_ASOCIATIE = relaţie dintre entitatea STUDENT şi ASOCIATIE, reflectând legătura dintre acestea (ce studenți iau parte la activități de voluntariat în cadrul asociației de la facultate). Ea are cardinalitatea minimă 1:1 şi cardinalitatea maximă 1:*n*.

STUDENT_promovează_EXAMEN = relaţie de tip *many-to-many* dintre entităţile STUDENT şi EXAMEN (ce stundeți promovează ce examene). Ea arată și notele pe care stundeții le obțin, cât și data primirii notei. Relaţia are cardinalitatea minimă 1:1 şi cardinalitatea maximă *m*:*n*.

PROFESOR_preda_CURS_la_GRUPA_in_SALA = relaţie de tip 3 ce leagă entităţile PROFESOR, CURS, GRUPA si SALA, reflectând cine a predate un curs, in ce sala era susținut acest curs, ce grupă participa la acesta. Denumirea acestei relaţii va fi *ORAR*.

Atribute – Ex. 5

- Entitatea independentă *FACULTATE* are ca atribute:

cod_facultate = variabilă de tip întreg, de lungime maximă 5, care reprezintă codul unei facultăți.
denumire = variabilă de tip caracter, de lungime maximă 100, care reprezintă denumirea facultății.
adresă = variabilă de tip caracter, de lungime maximă 100, care reprezintă adresa facultății.
telefon = variabilă de tip caracter, de lungime 10, care reprezintă numărul de telefon al facultății.
mail = variabilă de tip caracter, de lungime maximă 50, care reprezintă adresa de mail a unei facultăți.
fax = variabilă de tip caracter, de lungime maximă 50, care reprezintă fax-ul unei facultăți.
cod_postal = variabilă de tip caracter, de lungime maximă 6, care reprezintă codul postal al unei facultăți.

Cod_facultate	denumire	adresa	telefon	mail	fax	Cod_postal
PK	NOT NULL	NOT NULL	NOT NULL	UNIQUE	-	-

- Entitatea *SECTIE* are ca atribute:

cod_sectie = variabilă de tip întreg, de lungime maximă 5, care reprezintă codul sectiei.
denumire = variabilă de tip caracter, de lungime maximă 100, care reprezintă denumirea sectiei.
cod_facultate= variabilă de tip întreg, de lungime maximă 5, care reprezintă codul facultății din care face parte secția. Atributul trebuie să corespundă la o valoare a cheii primare din tabelul FACULTATE.

Cod_Sectie	Cod_facultate	denumire
PK	FK	NOT NULL

- Entitatea *SERIE* are ca atribute:

cod_serie= variabilă de tip întreg, de lungime maximă 5, care reprezintă codul seriei.
denumire = variabilă de tip caracter, de lungime maximă 100, care reprezintă denumirea seriei.
cod_sectie= variabilă de tip întreg, de lungime maximă 5, care reprezintă codul secției din care face parte seria. Atributul trebuie să corespundă la o valoare a cheii primare din tabelul SECTIE.

Cod_Serie	Cod_sectie	denumire
PK	FK	NOT NULL

- Entitatea *GRUPA* are ca atribute:

cod_grupa= variabilă de tip întreg, de lungime maximă 5, care reprezintă codul seriei.

denumire = variabilă de tip caracter, de lungime maximă 100, care reprezintă denumirea grupei.

cod_serie= variabilă de tip întreg, de lungime maximă 5, care reprezintă codul secției din care face parte seria. Atributul trebuie să corespundă la o valoare a cheii primare din tabelul SERIE.

Cod_grupa	Cod_serie	denumire
PK	FK	NOT NULL

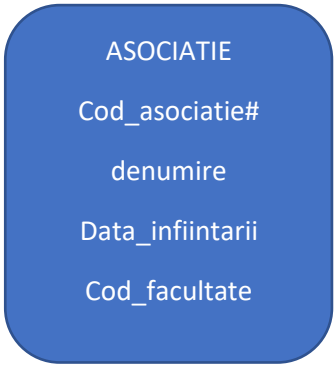
- Entitatea *STUDENT* are ca atribute:

cod_student = variabilă de tip întreg, de lungime maximă 5, care reprezintă codul unui student.
nume = variabilă de tip caracter, de lungime maximă 100, care reprezintă numele studentului.
prenume = variabilă de tip caracter, de lungime maximă 100, care reprezintă prenumele studentului.
data_nastere = variabilă de tip dată calendaristică, care reprezintă data naşterii studentului respectiv.
sex = variabilă de tip caracter, luând valorile *masculin* sau *feminin*, de lungime 10, care reprezintă sexul studentului.
nationalitate = variabilă de tip caracter, de lungime maximă 30, care reprezintă naţionalitatea unui student.
mail = variabilă de tip caracter, de lungime maximă 50, care reprezintă emailul
cod_asociatie = variabilă de tip intreg, de lungime maximă 5, care reprezintă codul asociației din care face parte studentul. Atributul trebuie să corespundă la o valoare a cheii primare din tabelul ASOCIATIE.

Cod_student	nume	prenume	Data_nasterii	mail	Sex	Nationalitate	Cod_asociatie
PK	NOT NULL	NOT NULL	NOT NULL	UNIQUE	NOT NULL	NOT NULL	FK

Baze de date – Anul I
Seria 14

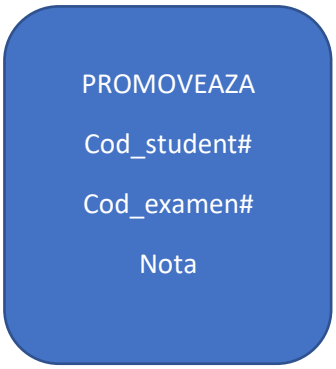
- Entitatea *ASOCIATIE* are ca atribute:



cod_asociatie = variabilă de tip întreg, de lungime maximă 5, care reprezintă codul unui asociatiei.
denumire = variabilă de tip caracter, de lungime maximă 100, care reprezintă numele studentului.
data_infiintarii = variabilă de tip dată calendaristică, care reprezintă data naşterii studentului respectiv.
cod_facultate = variabilă de tip intreg, de lungime maximă 5, care reprezintă codul facultatii de care parține asociația.
Atributul trebuie să corespundă la o valoare a cheii primare din tabelul FACULTATE.

Cod_asociatie	denumire	Data_infiintarii	Cod_facultate
PK	NOT NULL	NOT NULL	FK

- Relația STUDENT_promovează_EXAMEN (Entitatea *PROMOVEAZA)* are ca atribute:



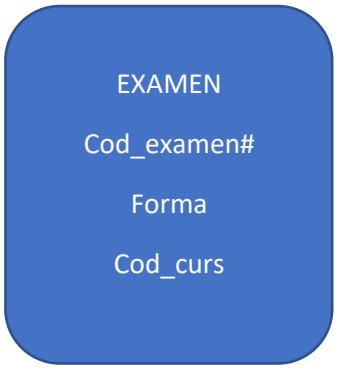
nota = variabilă de tip real, de lungime maximă 2, care reprezintă nota studentului.
cod_student = variabilă de tip intreg, de lungime maximă 5, care reprezintă codul studentului căruia îi aparține nota.
Atributul trebuie să corespundă la o valoare a cheii primare din tabelul STUDENT.
cod_examen = variabilă de tip intreg, de lungime maximă 5, care reprezintă codul examenului la care a fost acordată nota.
Atributul trebuie să corespundă la o valoare a cheii primare din tabelul EXAMEN.

Cod_student	Cod_facultate	Nota
PK		NOT NULL

Baze de date – Anul I
Seria 14

- Entitatea *EXAMEN* are ca atribute:

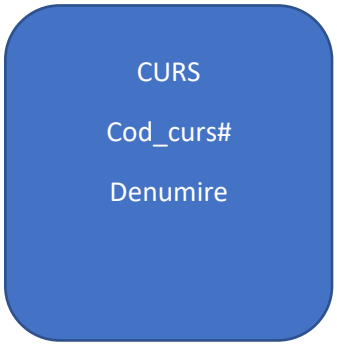
cod_examen = variabilă de tip întreg, de lungime maximă 5, care reprezintă codul unui examen.
forma = variabilă de tip caracter, luând valorile *”Examen Scris”, ”Interviu” sau ”Proiect”*, care reprezintă modalitatea de evaluare.
cod_curs = variabilă de tip intreg, de lungime maximă 5, care reprezintă codul cursului la care este susținut examenul. Atributul trebuie să corespundă la o valoare a cheii primare din tabelul CURS.



Cod_examen	Cod_curs	Forma
PK	FK	NOT NULL

- Entitatea *CURS* are ca atribute:

cod_curs = variabilă de tip întreg, de lungime maximă 5, care reprezintă codul unui curs.
denumire = variabilă de tip caracter, de lungime maximă 50, care reprezintă denumirea cursului.



Cod_curs	Denumire
PK	NOT NULL

Baze de date – Anul I
Seria 14

- Entitatea *SALA* are ca atribute:

cod_sala = variabilă de tip întreg, de lungime maximă 5, care reprezintă codul salii.
denumire = variabilă de tip caracter, de lungime maximă 30, care reprezintă denumirea salii.
locatie = variabilă de tip caracter, de lungime maximă 50, care reprezintă locatia salii.

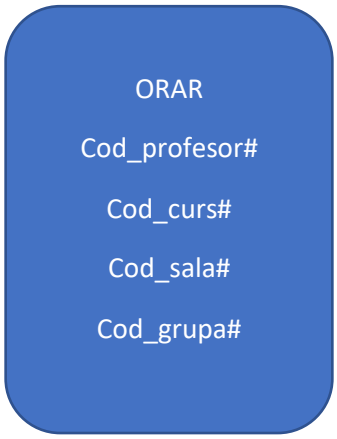
Cod_sala#	Locatie	Denumire
PK	NOT NULL	NOT NULL

- Entitatea *PROFESOR* are ca atribute:

cod_profesor = variabilă de tip întreg, de lungime maximă 5, care reprezintă codul unui profesor.
nume = variabilă de tip caracter, de lungime maximă 100, care reprezintă numele profesor.
prenume = variabilă de tip caracter, de lungime maximă 100, care reprezintă prenumele profesor.
telefon = variabilă de tip caracter, de lungime maximă 20, care reprezintă numarul de telefon.
mail = variabilă de tip caracter, de lungime maximă 50, care reprezintă emailul.

Cod_profesor#	Nume	Prenume	Telefon	Mail
PK	NOT NULL	NOT NULL	NOT NULL	UNIQUE

- Relația *PROFESOR_preda_CURS_la_GRUPA_in_SALA (*Entitatea *ORAR)* are ca atribute:



cod_curs = variabilă de tip intreg, de lungime maximă 5, care reprezintă codul cursului la care este susținut examenul. Atributul trebuie să corespundă la o valoare a cheii primare din tabelul CURS.
cod_grupa = variabilă de tip intreg, de lungime maximă 5, care reprezintă grupei care participa la curs. Atributul trebuie să corespundă la o valoare a cheii primare din tabelul GRUPA.
cod_profesor = variabilă de tip intreg, de lungime maximă 5, care reprezintă profesorului care predă cursul. Atributul trebuie să corespundă la o valoare a cheii primare din tabelul PROFESOR.
cod_sala = variabilă de tip intreg, de lungime maximă 5, care reprezintă codul sălii în care se desfășoară cursul. Atributul trebuie să corespundă la o valoare a cheii primare din tabelul SALA.)

Cod_profesor	Cod_curs	Cod_sala	Cod_grupa
PK			

- Relația *PROFESOR_este_angajat_la_FACULTATE (*Entitatea *CONTRACT)* are ca atribute:

salariu = variabilă de tip întreg, care reprezintă salariul
cod_profesor = variabilă de tip intreg, de lungime maximă 5, care reprezintă codul profesorului care preda la facultatea respectivă. Atributul trebuie să corespundă la o valoare a cheii primare din tabelul PROFESOR.
data_inceput = variabilă de tip dată calendaristică, care reprezintă data angajarii unui profesor la o facultate.
cod_facultate = variabilă de tip intreg, de lungime maximă 5, care reprezintă facultății la care predă profesorul. Atributul trebuie să corespundă la o valoare a cheii primare din tabelul FACULTATE.

Cod_profesor#	Cod_facultate#	Salariu	Data_inceput
PK		NOT NULL	NOT NULL

- Relația *GRUPA_are_STUDENȚI* (Entitatea *INSCRIERE)* are ca atribute:

data_inscriere = variabilă de tip dată calendaristică, care reprezintă data înscrierii unui stundent în cadrul unei grupe.
cod_grupa= variabilă de tip întreg, de lungime maximă 5, care reprezintă codul grupei din care face parte studentul. Atributul trebuie să corespundă la o valoare a cheii primare din tabelul GRUPA.
cod_student= variabilă de tip întreg, de lungime maximă 5, care reprezintă codul studentul. Atributul trebuie să corespundă la o valoare a cheii primare din tabelul STUDENT.

Cod_grupa#	Cod_student#	Data_inscrierii
PK		NOT NULL

Baze de date – Anul I
Seria 14

Diagrama Conceptuala – Ex. 7

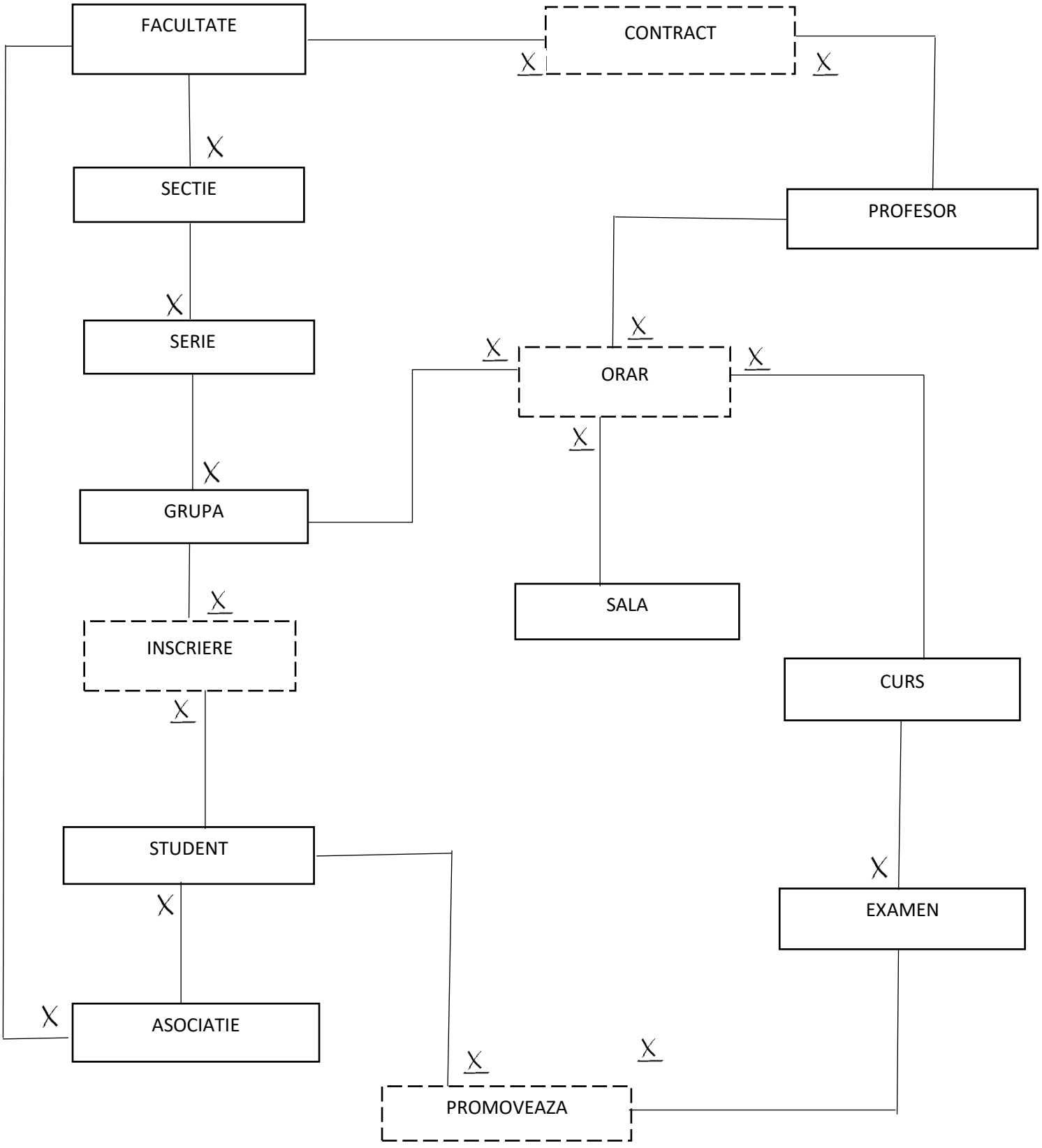
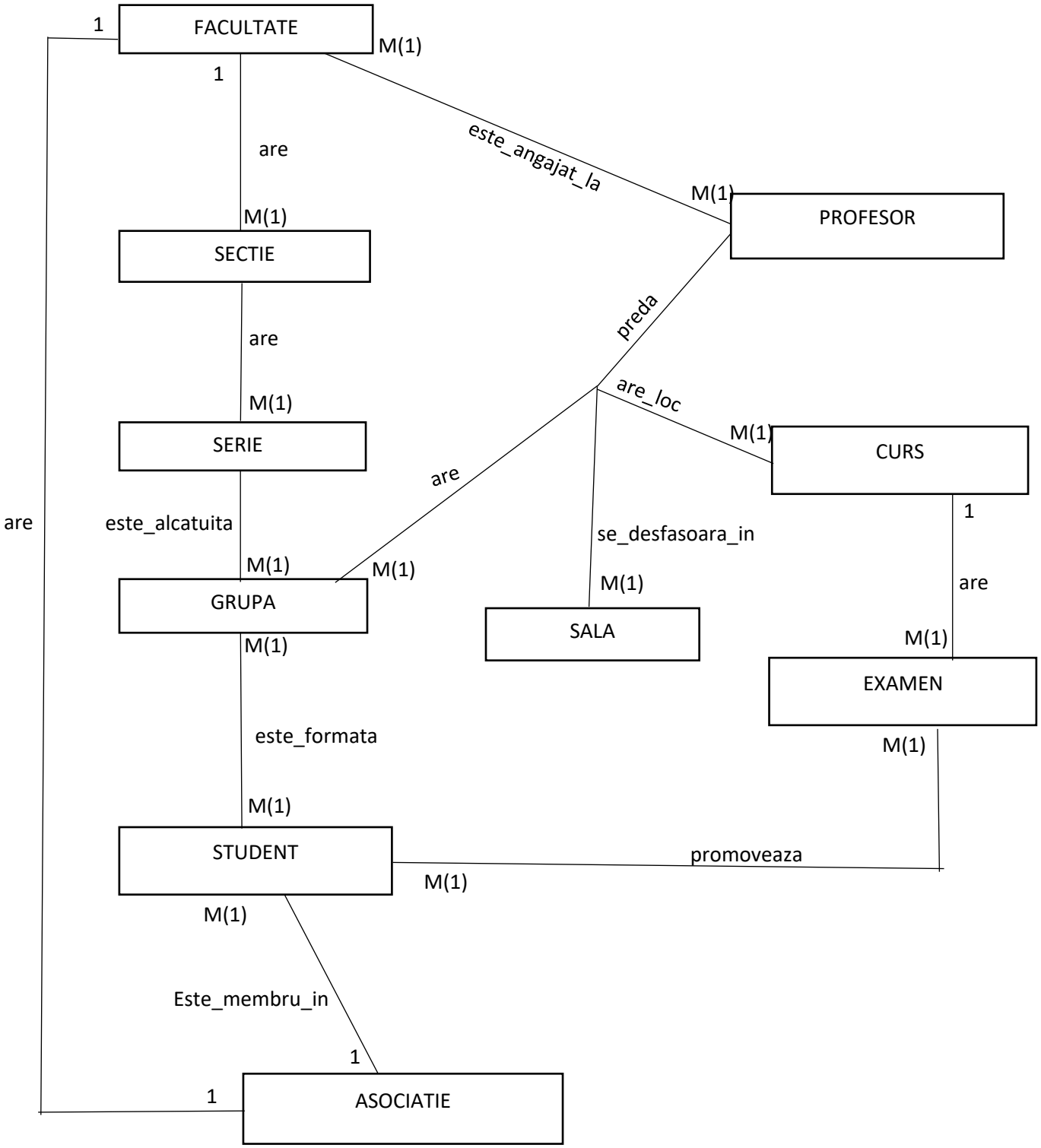


Diagrama E/R. – Ex. 6



Schemele relaţionale - Ex. 8

Schemele relaţionale corespunzătoare diagramei conceptuale sunt următoarele:

ASOCIATIE(cod_asociatie#, denumire, data_infiintarii,cod_facultate#)

CONTRACT(cod_profesor#, cod_facultate#,salariu)

CURS(cod_curs#, denumire)

EXAMEN(cod_examen#, forma,cod_curs#)

FACULTATE(cod_facultate#, denumire, adresă, telefon, mail, fax, cod_postal)

GRUPA(cod_grupa#, denumire, cod_serie#)

INSCRIERE(data_inscriere, cod_grupa#, cod_student#)

ORAR(cod_curs#, cod_grupa#, cod_profesor#, cod_sala#)

PROFESOR(cod_profesor#, nume, prenume, data_nastere, sex)

PROMOVEAZA(nota, data, cod_student#, cod_examen#)

SALA(cod_sala#, denumire, locatie)

SECTIE (cod_sectie#, denumire, cod_facultate#)

SERIE (cod_serie#,denumire, cod_sectie#)

STUDENT(cod_student#, nume, prenume, data_nastere, sex, nationalitate,telefon,email, cod_asociatie)

Normalizare și Denormalizare – Ex. 9 și Ex. 18

- **Forma normală 1 (FN1)**

O relație se află în FN1 dacă fiecărui atribut care o compune îi corespunde o valoare indivizabilă.

Forma normală 1 este și cea care impune și faptul că fiecare înregistrare să fie definită astfel încât să fie identificată unic prin intermediul unei chei primare.

În cadrul bazei de date pe care o construiesc, voi utiliza un exemplu pentru a prezenta aflarea acesteia in forma normal 1.

Voi lua entitățile FACULTATE și SECTIE.

În cadrul unei facultăți, denumirea secțiilor nu se repetă, dar în cadrul unei universități acest lucru nu este garantat, de multe ori chiar întâmplându-se acest lucru.

1. Mai multe valori seminifică același câmp

FACULTATE	SECTIE
Biologie	Anatomie, Biologie Animala
Chimie	Anatomie, Chimie Organica
Medicina	Anatomie, Biologie Animala, Chimie Organica

A selecta interegistrări pe baza câmpurilor care pot conține valori semnificative sunt foarte dificile si greu de implementat. Spre exemplu, o interogare care ar selecta acele facultătți care au si secție de Anatomie si secție de Chimie Organică, ar trebui să parcurgem fiecare șir ”Secție”, să identificăm subșirurile Anatomie și Chimie Organică și să selectăm numai acele înregistrări în care apar ambele subșiruri.

FACULTATE	SECTIE
Biologie	Anatomie, Biologie Animala
Chimie	Anatomie, Chimie Organica
Medicina	Anatomie, Biologie Animala, Chimie Organica

FACULTATE	SECTIE
Biologie	Anatomie
Biologie	Biologie Animala
Chimie	Chimie Organica
Chimie	Anatomie
Medicina	Anatomie
Medicina	Biologie Animala
Medicina	Chimie Organica

ID	FACULTATE	SECTIE
1	Biologie	Anatomie
2	Biologie	Biologie Animala
3	Chimie	Chimie Organica
4	Chimie	Anatomie
5	Medicina	Anatomie
6	Medicina	Biologie Animala
7	Medicina	Chimie Organica

2. Mai multe coloane reprezintă același tip de date.

FACULTATE	SECTIE	SECTIE(1)	SECTIE(2)
Biologie	Anatomie	Biologie Animala	
Chimie	Anatomie	Chimie Organica	
Medicina	Anatomie	Biologie Animala	Chimie Organica

FACULTATE	SECTIE
Biologie	Anatomie
Chimie	Anatomie
Medicina	Anatomie
Biologie	Biologie Animala
Chimie	Chimie Organica
Medicina	Biologie Animala
Medicina	Chimie Organica

ID	FACULTATE	SECTIE
1	Biologie	Anatomie
2	Biologie	Biologie Animala
3	Chimie	Chimie Organica
4	Chimie	Anatomie
5	Medicina	Anatomie
6	Medicina	Biologie Animala
7	Medicina	Chimie Organica

Pentru a asigura unicitatea unei inregistrari, se va utiliza cheia primara. In exemplul de mai sus, prin introducerea unei coloane aditionale de tip intreg se asigura unicitatea fiecarei inregistrari.

- **Forma Normală 2 (FN2)**

  O relație se află în a doua formă normal dacă și numai dacă această relație este deja in FN1 si fiecare atribut care nu este cheie primară este dependent de întreaga cheie primară.

FN2 interzice existența dependențelor funcționale parțiale în cadrul relației.

Daca unul sau mai multe elemente sunt dependente functional numai de o parte a cheii primare, atunci ele trebuie sa fie separate in tabele diferite. Daca tabela are o cheie primara formata din numai un atribut, atunci ea este automat in FN2.

Pentru baza mea de date, voi exemplifica pentru cazul diagramei CONTRACT.

Cod_profesor#	Nume Profesor	Cod_facultate#	Data_Inceput	Salariu
S1	N1	G1	12-04-2020	5400
S1	N1	G2	01-10-2019	5300
S2	N2	G2	03-02-2018	4600
S2	N2	G3	05-06-2017	4900
S2	N2	G4	01-10-2020	4500

Un profesor poate să fie predea la mai multe facultăți și o facultate are mai mulți profesori angajați.

Relația este în FN1 – avem identificator unic pentru toate intrările din table.

Fiecare atribut care nu este cheie (nu participă la cheia primară) este dependent de întreaga cheie primară – in cazul nostru atributul Nume_profesor, nu este cheie si trebuie sa depinda direct de intreaga cheie primara cod_profesor# si cod_facultate# -> aceste atribute nu depind direct de intreaga cheie primara deoarece se observa dependenta directa dintre Nume_profesor si cod_profesor,

insemnand ca Nume_profesor depinde direct doar de o parte a cheii primare, si anume doar de cod_profesor -> relatia nu se afla in FN2.

Astfel avem ca:

- {cod_profesor#} -> {nume_profesor} – cod_profesor determina functional cod_profesor
- {cod_profesor #, cod_facultate#} -> {data_inceput, salariu}

Se aplică regula Casey Delobel și va rezulta faptul că, pentru a avea relația în FN2, numele profesorului trebuie să fie doar în entitatea profesor.

Cod_profesor#	Nume Profesor	Cod_facultate#	Data_Inceput	Salariu
S1	N1	G1	12-04-2020	5400
S1	N1	G2	01-10-2019	5300
S2	N2	G2	03-02-2018	4600
S2	N2	G3	05-06-2017	4900
S2	N2	G4	01-10-2020	4500

Cod_profesor#	Cod_facultate#	Data_Inceput	Salariu
S1	G1	12-04-2020	5400
S1	G2	01-10-2019	5300
S2	G2	03-02-2018	4600
S2	G3	05-06-2017	4900
S2	G4	01-10-2020	4500

- **<u>Forma Normală 3 (FN3)</u>**

O relație este în a treia formă normală dacă și numai dacă este in FN2 și fiecare atribut care nu este cheie depinde direct de cheia primară.

Iau tabela STUDENT. Inițial, aceasta ar fi fost de forma:

Cod_student#	nume	prenume	Data_nasterii	sex	Nationalitate	mail	Denumire_asociatie	Data_infiintarii
P1	N1	Pr1	01-02-2000	m	Roman	M1	G1	D1
P2	N2	Pr2	17-01-2000	f	Roman	M2	G2	D2
P3	N3	Pr3	22-08-2001	f	Roman	M3	G3	D3

Se poate observa că atributelul Data_infiiintarii depinde de atributul Denumire_asociatie care depinde la randul sau de cheia primară cod_student, astfel nefiind in NF3.

Pentru a aduce in FN3, separ atributele despre asocitatie din STUDENT, apărând astfel tabela ASOCIATIE.

Astfel, inlocuiesc aceste atribute cu cheia străină cod_asociatie (apărând astfel o relație one-to-many între asociație si student) pentru a determina mai ușor din ce asociație face un student parte sau dacă nu face voluntariat (cheia străină e nula).

Cod_student#	nume	prenume	Data_nasterii	sex	Nationalitate	Cod_asociatie	mail
P1	N1	Pr1	01-02-2000	m	Roman	Null	M1
P2	N2	Pr2	17-01-2000	f	Roman	A1	M2
P3	N3	Pr3	22-08-2001	f	Roman	A2	M3

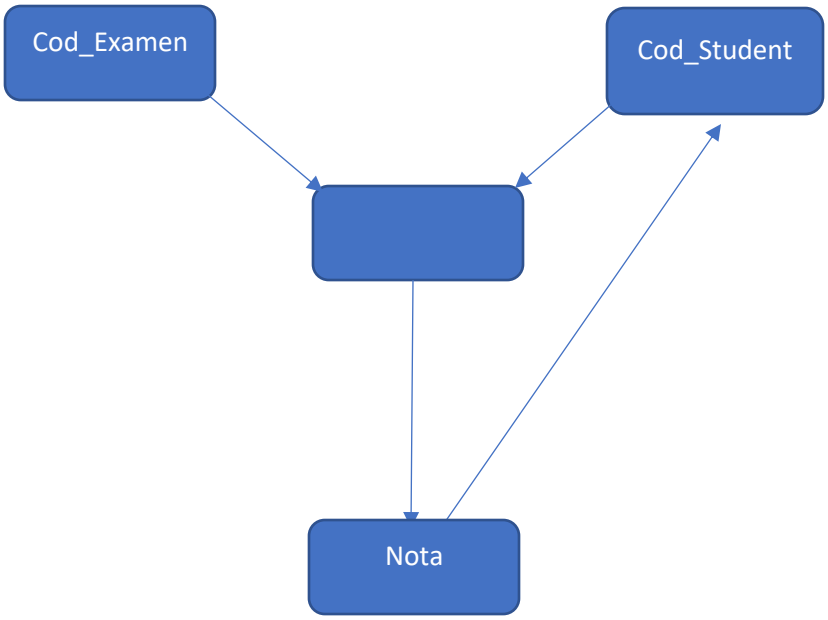
Cod_asoctiate	Denumire_asociatie	Data_infiintarii	cod_facultate
1	Asoc1	D1	F1
2	Asoc2	D2	F2
3	Asoc3	D3	F3

- **<u>Forma normală Boyce-Codd (BCNF)</u>**

Forma normală Boyce-Codd se bazează pe dependenţele funcţionale care iau în consideraţie toate cheile candidat dintr-o relaţie.

Pentru relatiile cu o singura cheie candidat, formele FN3 si BCNF sunt echivalente (Facultate, Curs etc.)

Să luăm relația PROMOVEAZA(cod_examen#, cod_student#, nota)



Regula Casey Delobel pentru PROMOVEAZA(cod_examen#, cod_student#, nota) având faptul că nota->cod_student

➡ PROMOVEAZA_1(cod_examen#,nota)

➡ PROMOVEAZA_2(nota,cod_student)

- **<u>Forma nomală 4 (FN4)</u>**

FN4 elimină redundanțele datorate relațiilor m:n, adică datorate dependenței multiple.

O relație este în a patra formă nomală dacă și numai dacă este în BCNF și nu conține relații m:n independente.

Iau relația SALA(cod_sala#, denumire, locatie) li presupun că o sală poate avea mai multe denumiri și mai multe locații – maă gândesc ca la un amfiteatru în care se poate intra prin mai multe locuri si fiecare intrare e numele salii + Intrarea (A, B, C, etc.) si locatiile sunt date de holurile din care se intra în acestea.

Cod_sala# ➡➡ denumire;

Cod_sala# ➡➡ locatie;

Relația Sala este in BCNF. Pentru a aduce relația în FN4 o vom descompune prin proiecție în două relații:

SALA1(cod_sala#, denumire)

SALA2(cod_sala#, locatie)

➡ SALA = JOIN(SALA1, SALA2)

- **<u>Forma normală 5 (FN5)</u>**

O relaţie R este în FN5 (numită şi forma normală proiecţie-uniune) dacă şi numai dacă orice dependenţă de uniune a lui R este o consecinţă a unei chei candidat a lui R.

Orice relaţie care este în FN5 este şi în FN4, deoarece fiecare dependenţă multivaloare poate fi privită ca un caz particular de dependenţă de uniune. Orice relaţie poate fi descompusă fără pierderi la uniune într-o mulţime de relaţii care sunt în FN5.

Pentru a preciza dacă o relaţie este în FN5, este suficient să cunoaştem cheile candidate şi toate dependenţele de uniune din R.

Aducerea în FN5 presupune eliminarea join dependențelor.

Sa luăm relația FACULTATE și să presupunem că am avea o join dependență în aceasta – iau mulțimea (fax, adresa, cod_postal) prespunem ca există multiple dependențe între fiecare dintre perechile din mulțimi (fax,adresa), (adresa,cod_postal) și (fax, cod_postal) – se pierde faptul că în acest caz cod_postal se duce in fax - este un exemplu pur ipotetic.

- **<u>Denormalizare</u>**

Denormalizarea este procesul invers al procesului de normalizare. Denormalizarea funcționează adăugând date redundante sau grupând date pentru a optimiza performanța.

Denormalizare are rolul de a realiza executarea mai rapidă a interpgărilor prin introducerea redundanței, punând accentul pe rapiditatea analizei și scăzând numărul de tabele.

Motivul efectuării denormalizării este costurile unui produs în procesor de interogare printr-o structură supra-normalizată.

Denormalizarea poate fi definită și ca metoda de stocare a îmbinării relațiilor de formă normală superioară ca relație de bază, care se află într-o formă normală inferioară. Reduce numărul de tabele și îmbinările de tabel complicate, deoarece un număr mai mare de îmbinări poate încetini procesul.

Aici abordarea denormalizării, subliniază conceptul că, plasând toate datele într-un singur loc, ar putea elimina necesitatea căutării acelor fișiere multiple pentru a colecta aceste date.

În cadrul bazei mele de date, luând spre exemplu relația PROMOVEAZA și descompunerea ei de la FN4, este inutil și mult mai costisitor din punct de vedere al timpului de executare să parcurgem datele si din PROMOVEAZA1 și PROMOVEAZA2.

Algebră Relațională – Ex. 17

- Sa se afiseze numele, prenumele si salariul profesorilor care lucreaza la facultatea cu codul 1, au salariu mai mare de 5000 de lei si au fost angajati dupa 1 august 2017.

SQL:

```
select p.nume, p.prenume, c.salariu

from profesor p join contract c on (p.cod_profesor = c.cod_profesor)

where c.cod_facultate in (select cod_facultate

                          from contract t

                          where t.cod_facultate = c.cod_facultate and t.cod_facultate = 1)

                    and  c.salariu in (select salariu

                                       from contract ct

                                       where ct.salariu >= 5000 and c.cod_profesor = ct.cod_profesor)

                                       and data_inceput in (select data_inceput

                                                            from contract cc

                                                            where cc.data_inceput > to_date('01-08-2017','dd-mm-yyyy') and  c.cod_profesor = cc.cod_profesor)

                                                            ;
```

Expresie Algebrică:

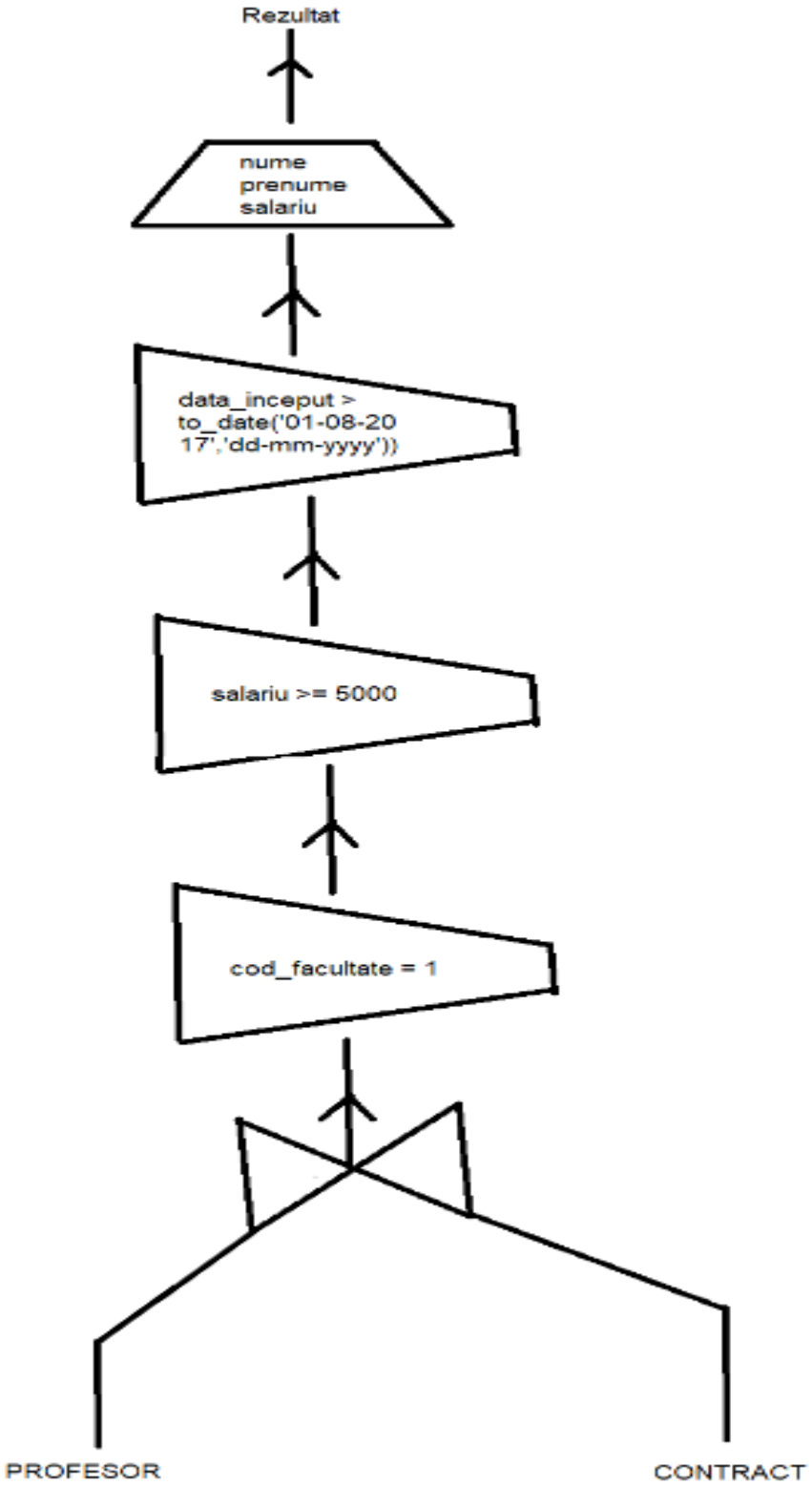
R1 = JOIN(profesor, contract)

R2 = SELECT(R1, cod_facultate = 1)

R3 = SELECT(R2, salariu >= 5000)

R4 = SELECT(R3, data_inceput > to_date('01-08-2017','dd-mm-yyyy'))

REZULTAT = R5 = PROJECT(R4, nume, prenume,salariu)


Rezultat
nume
prenume
salariu
data_inceput >
to_date('01-08-20
17','dd-mm-yyyy'))
salariu >= 5000
cod_facultate = 1
PROFESOR
CONTRACT

Etapa Intermediara:

SQL:

```
select p.nume, p.prenume, c.salariu
from profesor p join contract c on (p.cod_profesor = c.cod_profesor)
where c.cod_facultate = 1 and c.salariu >= 5000 and c.data_inceput > to_date('01-08-2017','dd-mm-yyyy');
```

Expresie Algebrică:

R1 = SELECT(contract, cod_facultate = 1 and salariu >= 5000 and data_inceput > to_date('01-08-2017','dd-mm-yyyy'))

R2 = JOIN(R1, profesor)

R3 = PROJECT(R2, nume, prenume, salariu)

Rezultat final:

SQL:

```
select p.nume, p.prenume, c.salariu
from (select salariu, cod_profesor
      from contract
      where cod_facultate = 1 and salariu >= 5000 and data_inceput > to_date('01-08-2017','dd-mm-yyyy') ) c
join(select nume,prenume, cod_profesor
      from profesor ) p on (p.cod_profesor = c.cod_profesor);
```

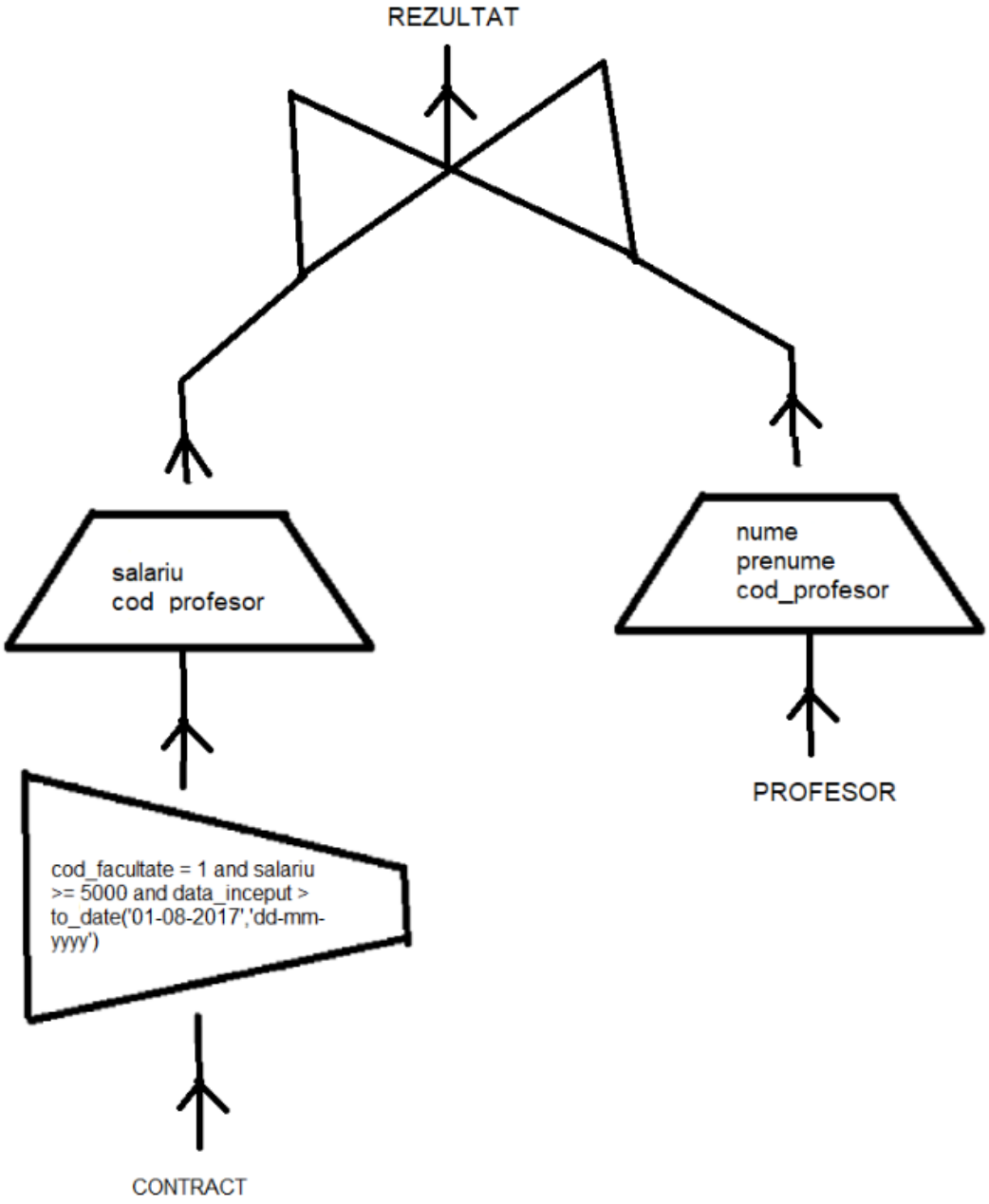
Expresie Algebrică:

R1 = SELECT(contract, cod_facultate = 1 and salariu >= 5000 and data_inceput > to_date('01-08-2017','dd-mm-yyyy'))

R2 = PROJECT(R1, salariu, cod_profesor)

R3 = PROJECT(profesor, nume, prenume, cod_profesor)

REZULTAT = R4 = JOIN(R2, R3)


REZULTAT
salariu
cod_profesor
nume
prenume
cod_profesor
PROFESOR
cod_facultate = 1 and salariu >= 5000 and data_inceput > to_date('01-08-2017','dd-mm-yyyy')
CONTRACT

SQL

Ex. 10

```
CREATE TABLE FACULTATE
                        (cod_facultate NUMBER(5) CONSTRAINT PKEY_FACULTATE PRIMARY KEY,
                         denumire VARCHAR(100) CONSTRAINT denumire_facultate NOT NULL,
                         adresa VARCHAR(100) CONSTRAINT adresa_facultate NOT NULL,
                         telefon VARCHAR(20) CONSTRAINT telefon_facultate NOT NULL,
                         mail VARCHAR(50) UNIQUE,
                         fax VARCHAR(50),
                         cod_postal VARCHAR(6)
                        );
```

```
INSERT INTO FACULTATE
VALUES(1,'Facultatea de Matematica si Informatica','Str. Academiei nr. 14', '+4021-314.28.63', 'secretariat@fmi.unibuc.ro','+4021-314.28.63','010014');

INSERT INTO FACULTATE
VALUES(2,'Facultatea de Drept','Bd. M. Kog?lniceanu, nr. 36-46', '+4021-312.49.48', 'informatii.drept@drept.unibuc.ro','+4021-312.49.48','050107');

INSERT INTO FACULTATE
VALUES(3,'Facultatea de Geografie','Bd. Nicolae B?lcescu Nr. 1', '+0421-305.38.10', 'secretariat@geo.unibuc.ro','+0421-305.38.10','010041');

INSERT INTO FACULTATE
VALUES(4,'Facultatea de Istorie','Bd. Regina Elisabeta nr. 4-12', '+4021-314.35.89', 'secretariat@istorie.unibuc.ro','+4021-314.35.89','030018');

INSERT INTO FACULTATE
VALUES(5,'Facultatea de Litere','Str. Edgar Quinet, nr. 5-7', '+4021-313.43.36', 'gabriela.dena@litere.unibuc.ro','+4021-313.43.36','010017');

COMMIT;
```

	COD_FACULTATE	DENUMIRE	ADRESA	TELEFON	MAIL	FAX	COD_POSTAL
1	1	Facultatea de Matematica si Informatica	Str. Academiei nr. 14	+4021-314.28.63	secretariat@fmi.unibuc.ro	+4021-314.28.63	010014
2	2	Facultatea de Drept	Bd. M. Kogalniceanu, nr. 36-46	+4021-312.49.48	informatii.drept@drept.unibuc.ro	+4021-312.49.48	050107
3	3	Facultatea de Geografie	Bd. Nicolae Balcescu Nr. 1	+0421-305.38.10	secretariat@geo.unibuc.ro	+0421-305.38.10	010041
4	4	Facultatea de Istorie	Bd. Regina Elisabeta nr. 4-12	+4021-314.35.89	secretariat@istorie.unibuc.ro	+4021-314.35.89	030018
5	5	Facultatea de Litere	Str. Edgar Quinet, nr. 5-7	+4021-313.43.36	gabriela.dena@litere.unibuc.ro	+4021-313.43.36	010017

```
CREATE TABLE ASOCIATIE(cod_asociatie NUMBER(5) CONSTRAINT PKEY_ASOCIATIE PRIMARY KEY,
                       denumire VARCHAR(100) CONSTRAINT denumire_asociatie NOT NULL,
                       data_infiintarii DATE CONSTRAINT data_infiintarii_const NOT NULL,
                      cod_facultate NUMBER(5),
                      CONSTRAINT fk_asoc FOREIGN KEY(cod_facultate) REFERENCES FACULTATE(cod_facultate)
                      );
```

```
INSERT INTO ASOCIATIE
VALUES(10,'ASMI', TO_DATE('12-04-2009','DD-MM-YYYY'), 1);

INSERT INTO ASOCIATIE
VALUES(11,'ASD', TO_DATE('17-06-2011','DD-MM-YYYY'), 2);

INSERT INTO ASOCIATIE
VALUES(12,'ASG', TO_DATE('02-08-2012','DD-MM-YYYY'), 3);

INSERT INTO ASOCIATIE
VALUES(13,'ASID', TO_DATE('18-05-2008','DD-MM-YYYY'), 4);

INSERT INTO ASOCIATIE
VALUES(14,'ASL', TO_DATE('14-07-2010','DD-MM-YYYY'), 5);

COMMIT;
```

	COD_ASO...	DENUMIRE	DATA_IN...	COD_FAC...
1	10	ASMI	12-APR-09	1
2	11	ASD	17-JUN-11	2
3	12	ASG	02-AUG-12	3
4	13	ASID	18-MAY-08	4
5	14	ASL	14-JUL-10	5

```
CREATE TABLE SECTIE(cod_sectie NUMBER(5) CONSTRAINT PKEY_SECTIE PRIMARY KEY,
                    denumire VARCHAR(100) CONSTRAINT denumire_serie NOT NULL,
                    cod_facultate NUMBER(5),
                     CONSTRAINT fk_sect FOREIGN KEY(cod_facultate) REFERENCES FACULTATE(cod_facultate)

                    );
```

```
INSERT INTO SECTIE
VALUES(20,'Informatica', 1);

INSERT INTO SECTIE
VALUES(21,'Matematica', 1);

INSERT INTO SECTIE
VALUES(22,'Calculatoare si Tehnologia Informatiei', 1);

INSERT INTO SECTIE
VALUES(23,'Drept Privat', 2);

INSERT INTO SECTIE
VALUES (24,'Drept Public',2);

INSERT INTO SECTIE
VALUES (25,' Drept Penal',2);
```

	COD_SECTIE	DENUMIRE	COD_FACULTATE
1	20	Informatica	1
2	21	Matematica	1
3	22	Calculatoare si Tehnologia Informatiei	1
4	23	Drept Privat	2
5	24	Drept Public	2
6	25	Drept Penal	2

```
CREATE TABLE SERIE(cod_serie NUMBER(5) CONSTRAINT PKEY_SERIE PRIMARY KEY,
                   denumire VARCHAR(100) CONSTRAINT denumire_sectie NOT NULL,
                   cod_sectie NUMBER(5),
                   CONSTRAINT fk_ser FOREIGN KEY( cod_sectie) REFERENCES SECTIE(cod_sectie)

                   );
```

```
INSERT INTO SERIE
VALUES(30,'13', 20);

INSERT INTO SERIE
VALUES(31,'14', 20);

INSERT INTO SERIE
VALUES(32,'15', 20);

INSERT INTO SERIE
VALUES(33,'16', 21);

INSERT INTO SERIE
VALUES (34,'11',22);

INSERT INTO SERIE
VALUES (35,'12',22);
```

	COD_SERIE	DENUMIRE	COD_SECTIE
1	30	13	20
2	31	14	20
3	32	15	20
4	33	16	21
5	34	11	22
6	35	12	22

```
CREATE TABLE GRUPA(cod_grupa NUMBER(5) CONSTRAINT PKEY_GRUPA PRIMARY KEY,
                   denumire VARCHAR(100) CONSTRAINT denumire_grupa NOT NULL,
                   cod_serie NUMBER(5),
                   CONSTRAINT fk_grupa FOREIGN KEY( cod_serie) REFERENCES SERIE(cod_serie)

                   );
```

```
INSERT INTO GRUPA
VALUES(40,'131', 30);

INSERT INTO GRUPA
VALUES(41,'132', 30);

INSERT INTO GRUPA
VALUES(42,'133', 30);

INSERT INTO GRUPA
VALUES(43,'141', 31);

INSERT INTO GRUPA
VALUES(44,'142', 31);

INSERT INTO GRUPA
VALUES(45,'143', 31);

INSERT INTO GRUPA
VALUES(46,'144', 31);

INSERT INTO GRUPA
VALUES(47,'151', 32);

INSERT INTO GRUPA
VALUES(48,'152', 32);

COMMIT;
```

	COD_GRUPA	DENUMIRE	COD_SERIE
1	40	131	30
2	41	132	30
3	42	133	30
4	43	141	31
5	44	142	31
6	45	143	31
7	46	144	31
8	47	151	32
9	48	152	32

```
CREATE TABLE STUDENT(
                        cod_student NUMBER(5) CONSTRAINT PKEY_STUDENT PRIMARY KEY,
                        nume VARCHAR(100) CONSTRAINT nume_student NOT NULL,
                        prenume VARCHAR(100) CONSTRAINT prenume_student NOT NULL,
                        data_nasterii DATE CONSTRAINT data_nasterii_const NOT NULL,
                        sex VARCHAR(10) CONSTRAINT sex_const NOT NULL,
                        nationalitate VARCHAR(30) CONSTRAINT nat_const NOT NULL,
                        telefon VARCHAR(20) CONSTRAINT telefon_student NOT NULL,
                        mail VARCHAR(50) UNIQUE,
                        cod_asociatie NUMBER(5),
                         CONSTRAINT fk_student FOREIGN KEY( cod_asociatie) REFERENCES ASOCIATIE(cod_asociatie)
                       );
```

```
INSERT INTO STUDENT
VALUES(51,'Nimara', 'Dan', TO_DATE('12-04-2000','dd-mm-yyyy'),'masculin','roman','0743234789', 'dnimara@gmail.com', 10);

INSERT INTO STUDENT
VALUES(52,'Dima', 'Oana', TO_DATE('26-01-2000','dd-mm-yyyy'),'feminin','roman','0757674789', 'dima.oana26@gmail.com', 10);

INSERT INTO STUDENT
VALUES(53,'Miu', 'Adania', TO_DATE('14-01-2001','dd-mm-yyyy'),'feminin','roman','0756789901', 'adania.miu@gmail.com', null);

INSERT INTO STUDENT
VALUES(54,'Gherghescu', 'Andreea', TO_DATE('27-09-2001','dd-mm-yyyy'),'feminin','roman','0778901456', 'gh.andreea@gmail.com', 10);

INSERT INTO STUDENT
VALUES(55,'Pascu', 'Adrian', TO_DATE('15-08-2001','dd-mm-yyyy'),'masculin','roman','0767891056', 'pascu.adi@gmail.com', 10);

INSERT INTO STUDENT
VALUES(56,'Baciu', 'Daniel', TO_DATE('24-06-2001','dd-mm-yyyy'),'masculin','roman','0748913234', 'dani.baciu@gmail.com', 10);

INSERT INTO STUDENT
VALUES(57,'Guleama', 'Dan', TO_DATE('17-07-2001','dd-mm-yyyy'),'masculin','roman','0767458910', 'dan.guleama@gmail.com', null);

INSERT INTO STUDENT
VALUES(58,'Marton', 'Sergiu', TO_DATE('15-02-2001','dd-mm-yyyy'),'masculin','roman','0742561340', 'sergiu.marton@gmail.com', null);

INSERT INTO STUDENT
VALUES(59,'Vultur', 'Sofia', TO_DATE('10-12-2001','dd-mm-yyyy'),'feminin','roman','0755678923', 'sofi.vultur@gmail.com', null);

INSERT INTO STUDENT
VALUES(60,'Fritz', 'Raluca', TO_DATE('11-10-2001','dd-mm-yyyy'),'feminin','roman','0765678432', 'fritz.ralu@gmail.com', 10);
```

	COD_STUDENT	NUME	PRENUME	DATA_NASTERII	SEX	NATIONALITATE	TELEFON	MAIL	COD_ASOCIATIE
1	51	Nimara	Dan	12-APR-00	masculin	roman	0743234789	dnimara@gmail.com	10
2	52	Dima	Oana	26-JAN-00	feminin	roman	0757674789	dima.oana26@gmail.com	10
3	53	Miu	Adania	14-JAN-01	feminin	roman	0756789901	adania.miu@gmail.com	(null)
4	54	Gherghescu	Andreea	27-SEP-01	feminin	roman	0778901456	gh.andreea@gmail.com	10
5	55	Pascu	Adrian	15-AUG-01	masculin	roman	0767891056	pascu.adi@gmail.com	10
6	56	Baciu	Daniel	24-JUN-01	masculin	roman	0748913234	dani.baciu@gmail.com	10
7	57	Guleama	Dan	17-JUL-01	masculin	roman	0767458910	dan.guleama@gmail.com	(null)
8	58	Marton	Sergiu	15-FEB-01	masculin	roman	0742561340	sergiu.marton@gmail.com	(null)
9	59	Vultur	Sofia	10-DEC-01	feminin	roman	0755678923	sofi.vultur@gmail.com	(null)
10	60	Fritz	Raluca	11-OCT-01	feminin	roman	0765678432	fritz.ralu@gmail.com	10

```
CREATE TABLE INSCRIERE(cod_student NUMBER(5)  CONSTRAINT pk_c_stundent REFERENCES STUDENT(cod_student),
                    data_inscrierii DATE CONSTRAINT data_inscr NOT NULL,
                    cod_grupa NUMBER(5)CONSTRAINT pk_c_grupa REFERENCES GRUPA(cod_grupa) ,
                    CONSTRAINT pk_compus_inscr primary key(cod_grupa,cod_student)
                    );
```

```
INSERT INTO INSCRIERE
VALUES(51,  TO_DATE('01-10-2020','dd-mm-yyyy'), 40);

INSERT INTO INSCRIERE
VALUES(52,  TO_DATE('01-10-2020','dd-mm-yyyy'), 41);

INSERT INTO INSCRIERE
VALUES(53,  TO_DATE('01-10-2020','dd-mm-yyyy'), 40);

INSERT INTO INSCRIERE
VALUES(54,  TO_DATE('01-10-2019','dd-mm-yyyy'), 43);

INSERT INTO INSCRIERE
VALUES(60,  TO_DATE('01-10-2019','dd-mm-yyyy'), 43);

INSERT INTO INSCRIERE
VALUES(59,  TO_DATE('01-10-2019','dd-mm-yyyy'), 44);

INSERT INTO INSCRIERE
VALUES(58,  TO_DATE('01-10-2019','dd-mm-yyyy'), 43);

INSERT INTO INSCRIERE
VALUES(57,  TO_DATE('01-10-2019','dd-mm-yyyy'), 42);

INSERT INTO INSCRIERE
VALUES(56,  TO_DATE('01-10-2018','dd-mm-yyyy'), 45);

INSERT INTO INSCRIERE
VALUES(55,  TO_DATE('01-10-2018','dd-mm-yyyy'), 45);
```

	COD_STUDENT	DATA_INSC...	COD_GRUPA
1	51	01-OCT-20	40
2	52	01-OCT-20	41
3	53	01-OCT-20	40
4	54	01-OCT-19	43
5	60	01-OCT-19	43
6	59	01-OCT-19	44
7	58	01-OCT-19	43
8	57	01-OCT-19	42
9	56	01-OCT-18	45
10	55	01-OCT-18	45

```
CREATE TABLE PROFESOR(
                        cod_profesor NUMBER(5) CONSTRAINT PKEY_profesor PRIMARY KEY,
                        nume VARCHAR(100) CONSTRAINT nume_prof NOT NULL,
                        prenume VARCHAR(100) CONSTRAINT prenume_prof NOT NULL,
                        telefon VARCHAR(20) CONSTRAINT telefon_prof NOT NULL,
                        mail VARCHAR(50) UNIQUE

                       );
```

```
INSERT INTO PROFESOR
VALUES(100,  'Popescu','Ion', '0756789546','ion.popescu@gmail.com');

INSERT INTO PROFESOR
VALUES(101,  'Ionescu','Irina', '0745678923','irina.ionescu@gmail.com');

INSERT INTO PROFESOR
VALUES(102,  'Avram','Bianca', '0745678213','bianca.avram@gmail.com');

INSERT INTO PROFESOR
VALUES(103,  'Branescu','Robert', '0789456234','robbranescu@gmail.com');

INSERT INTO PROFESOR
VALUES(104,  'Enache','Teodora', '0723563781','enache.teo@gmail.com');

INSERT INTO PROFESOR
VALUES(105,  'Boboc','Stefania', '0723157368','boboc.stefania@gmail.com');

COMMIT;
```

	COD_PROFESOR	NUME	PRENUME	TELEFON	MAIL
1	100	Popescu	Ion	0756789546	ion.popescu@gmail.com
2	101	Ionescu	Irina	0745678923	irina.ionescu@gmail.com
3	102	Avram	Bianca	0745678213	bianca.avram@gmail.com
4	103	Branescu	Robert	0789456234	robbranescu@gmail.com
5	104	Enache	Teodora	0723563781	enache.teo@gmail.com
6	105	Boboc	Stefania	0723157368	boboc.stefania@gmail.com

```
CREATE TABLE CONTRACT(cod_facultate NUMBER(5) CONSTRAINT pk_c_fac REFERENCES FACULTATE(cod_facultate),
                      data_inceput DATE CONSTRAINT data_inc NOT NULL,
                      cod_profesor NUMBER(5) CONSTRAINT pk_c_prof REFERENCES PROFESOR(cod_profesor),
                      salariu NUMBER CONSTRAINT sal_const NOT NULL,
                      CONSTRAINT pk_compus_cont primary key(cod_facultate,cod_profesor)
                      );
```

```
INSERT INTO CONTRACT
VALUES(1, TO_DATE('01-08-2018','dd-mm-yyyy'), 100, 5400);

INSERT INTO CONTRACT
VALUES(2, TO_DATE('01-07-2018','dd-mm-yyyy'), 101, 5200);

INSERT INTO CONTRACT
VALUES(3, TO_DATE('01-06-2017','dd-mm-yyyy'), 100, 5400);

INSERT INTO CONTRACT
VALUES(1, TO_DATE('01-06-2017','dd-mm-yyyy'), 102, 4500);

INSERT INTO CONTRACT
VALUES (1, TO_DATE('01-06-2017','dd-mm-yyyy'), 103, 4200);

INSERT INTO CONTRACT
VALUES (1,TO_DATE('01-06-2017','dd-mm-yyyy'), 104, 4700);

INSERT INTO CONTRACT
VALUES (1,TO_DATE('01-06-2019','dd-mm-yyyy'), 105, 4700);
```

	COD_FACULTATE	DATA_INCEPUT	COD_PROFESOR	SALARIU
1	1	01-AUG-18	100	5400
2	2	01-JUL-18	101	5200
3	3	01-JUN-17	100	5400
4	1	01-JUN-17	102	4500
5	1	01-JUN-17	103	4200
6	1	01-JUN-17	104	4700
7	1	01-JUN-19	105	4700

```
CREATE TABLE SALA(cod_sala NUMBER(5) CONSTRAINT PKEY_sala PRIMARY KEY,
                  denumire VARCHAR(30) CONSTRAINT denumire_sala NOT NULL,
                  locatie VARCHAR(50) CONSTRAINT locatie_sala NOT NULL
                  );

INSERT INTO SALA
VALUES (200,'Amfiteatrul Titulescu', 'Str. Alexandru Lapusneanu, nr.11');

INSERT INTO SALA
VALUES (201,'Amfiteatrul Haret', 'Str. Ion Minulescu, nr.12');

INSERT INTO SALA
VALUES (202,'Amfiteatrul Pompeiu', 'Str. Stefan cel Mare, nr.201');

INSERT INTO SALA
VALUES (203,'Amfiteatrul Moisil', 'Str. Lebedei, nr.304');

INSERT INTO SALA
VALUES (204,'Amfiteatrul Lalescu', 'Str. Herastrau, nr.215');

INSERT INTO SALA
VALUES (205,'Amfiteatrul Ghika', 'Splaiul Independentei, nr.5');

INSERT INTO SALA
VALUES (206,'Amfiteatrul Barbilian', 'Str. Unirii, nr.20');
```

	COD_SALA	DENUMIRE	LOCATIE
1	200	Amfiteatrul Titulescu	Str. Alexandru Lapusneanu, nr.11
2	201	Amfiteatrul Haret	Str. Ion Minulescu, nr.12
3	202	Amfiteatrul Pompeiu	Str. Stefan cel Mare, nr.201
4	203	Amfiteatrul Moisil	Str. Lebedei, nr.304
5	204	Amfiteatrul Lalescu	Str. Herastrau, nr.215
6	205	Amfiteatrul Ghika	Splaiul Independentei, nr.5
7	206	Amfiteatrul Barbilian	Str. Unirii, nr.20

```
CREATE TABLE CURS(cod_curs NUMBER(5) CONSTRAINT PKEY_curs PRIMARY KEY,
                  denumire VARCHAR(50) CONSTRAINT denumire_curs NOT NULL
                  );
```

```
INSERT INTO CURS
VALUES (300, 'Arhitectura sistemelor de calcul');

INSERT INTO CURS
VALUES (301, 'Gandire Critica si Etica Academica');

INSERT INTO CURS
VALUES (302, 'Limba si Literatura Engleza');

INSERT INTO CURS
VALUES (303, 'Programare Orientata pe Obiecte');

INSERT INTO CURS
VALUES (304, 'Educatie Fizica');

INSERT INTO CURS
VALUES (305, 'Baze de Date');

INSERT INTO CURS
VALUES (306, 'Istoria Religiilor');

INSERT INTO CURS
VALUES (307, 'Chimie organica');
```

	COD_CURS	DENUMIRE
1	300	Arhitectura sistemelor de calcul
2	301	Gandire Critica si Etica Academica
3	302	Limba si Literatura Engleza
4	303	Programare Orientata pe Obiecte
5	304	Educatie Fizica
6	305	Baze de Date
7	306	Istoria Religiilor
8	307	Chimie organica

```
CREATE TABLE EXAMEN(cod_examen NUMBER(5) CONSTRAINT PKEY_examen PRIMARY KEY,
                    forma VARCHAR(30) CONSTRAINT forma_const NOT NULL,
                    cod_curs NUMBER(5),
                    CONSTRAINT fk_examen FOREIGN KEY( cod_curs) REFERENCES CURS(cod_curs)
                    );

                    INSERT INTO EXAMEN
                    VALUES (SEQ_EXAM.NEXTVAL, 'Examen Scris', 300);

                    INSERT INTO EXAMEN
                    VALUES (SEQ_EXAM.NEXTVAL, 'Examen Scris', 301);

                    INSERT INTO EXAMEN
                    VALUES (SEQ_EXAM.NEXTVAL, 'Examen Scris', 302);

                    INSERT INTO EXAMEN
                    VALUES (SEQ_EXAM.NEXTVAL, 'Interviu', 303);

                    INSERT INTO EXAMEN
                    VALUES (SEQ_EXAM.NEXTVAL, 'Interviu', 304);

                    INSERT INTO EXAMEN
                    VALUES (SEQ_EXAM.NEXTVAL, 'Proiect', 305);

                    INSERT INTO EXAMEN
                    VALUES (SEQ_EXAM.NEXTVAL, 'Proiect', 306);

                    INSERT INTO EXAMEN
                    VALUES (SEQ_EXAM.NEXTVAL, 'Examen Scris', 307);
```

	COD_EXAMEN	FORMA	COD_CURS
1	410	Examen Scris	300
2	420	Examen Scris	301
3	430	Examen Scris	302
4	440	Interviu	303
5	450	Interviu	304
6	460	Proiect	305
7	470	Proiect	306
8	480	Examen Scris	307

Ex. 13

```
CREATE SEQUENCE SEQ_EXAM
INCREMENT by 10
START WITH 400
MAXVALUE 10000
NOCYCLE;
```

```
CREATE TABLE PROMOVEAZA(
                        nota NUMBER(5,2) CONSTRAINT nota_const NOT NULL,
                        cod_student  NUMBER(5) CONSTRAINT pk_c_s REFERENCES STUDENT(cod_student),
                        cod_examen NUMBER(5) CONSTRAINT pk_c_e REFERENCES EXAMEN(cod_examen),
                        CONSTRAINT pk_compus_prom primary key(cod_student,cod_examen)
                        );
```

```
INSERT INTO PROMOVEAZA
VALUES (10,51, 410);

INSERT INTO PROMOVEAZA
VALUES (10,52, 470);

INSERT INTO PROMOVEAZA
VALUES (8.5,53, 480);

INSERT INTO PROMOVEAZA
VALUES (7.5,54, 480);

INSERT INTO PROMOVEAZA
VALUES (9.5,55, 430);

INSERT INTO PROMOVEAZA
VALUES (8,56, 470);

INSERT INTO PROMOVEAZA
VALUES (7.5,53, 460);

INSERT INTO PROMOVEAZA
VALUES (9.3,55,460);

INSERT INTO PROMOVEAZA
VALUES (10,60, 470);

INSERT INTO PROMOVEAZA
VALUES (10,60, 420);

INSERT INTO PROMOVEAZA
VALUES (8.7, 51, 430);

INSERT INTO PROMOVEAZA
VALUES (5.6, 60, 460);

INSERT INTO PROMOVEAZA
VALUES (8.7, 59, 420);

INSERT INTO PROMOVEAZA
VALUES (7.7, 57, 470);
```

	NOTA	COD_STUDENT	COD_EXAMEN
1	10	51	410
2	10	52	470
3	8.5	53	480
4	7.5	54	480
5	9.5	55	430
6	8	56	470
7	7.5	53	460
8	9.3	55	460
9	10	60	470
10	10	60	420
11	8.7	51	430
12	5.6	60	460
13	8.7	59	420
14	7.7	57	470

```
CREATE TABLE ORAR(cod_grupa NUMBER(5) CONSTRAINT pk_c_gr REFERENCES GRUPA(cod_grupa),
                  cod_curs  NUMBER(5) CONSTRAINT pk_c_curs REFERENCES CURS(cod_curs),
                  cod_profesor NUMBER(5) CONSTRAINT pk_c_ REFERENCES PROFESOR(cod_profesor),
                  cod_sala NUMBER(5) CONSTRAINT pk_c_sala REFERENCES SALA(cod_sala),
                  CONSTRAINT pk_compus_orar primary key(cod_grupa, cod_curs,cod_profesor,cod_sala)
                  );
```

```
INSERT INTO ORAR
VALUES (40,300,100,200);

INSERT INTO ORAR
VALUES (41,303,102,201);

INSERT INTO ORAR
VALUES (42,305,105,205);

INSERT INTO ORAR
VALUES (43,300,103,202);

INSERT INTO ORAR
VALUES (44,302,102,203);

INSERT INTO ORAR
VALUES (45,303,101,202);

INSERT INTO ORAR
VALUES (46,305,101,201);
```

	COD_G...	COD_CURS	COD_PROFESOR	COD_SALA
1	40	300	100	200
2	41	303	102	201
3	42	305	105	205
4	43	300	103	202
5	44	302	102	203
6	45	303	101	202
7	46	305	101	201

Ex. 11

```
--afisati numele facultatilor care au grupe care desfasoara cursuri in Amfiteatrul Haret

SELECT UNIQUE CONCAT(CONCAT('Facultatea ',f.denumire),' are si cursuri in Amfiteatrul Haret') as "Rezultat"
FROM facultate f,sectie s, serie ser, grupa g, orar o, sala sal
where f.cod_facultate = s.cod_facultate and s.cod_sectie = ser.cod_sectie and ser.cod_serie = g.cod_serie
and g.cod_grupa = o.cod_grupa and o.cod_sala = sal.cod_sala and UPPER(sal.denumire) like UPPER('Amfiteatrul Haret');
```

	Rezultat
1	Facultatea Facultatea de Matematica si Informatica are si cursuri in Amfiteatrul Haret

```
-- afisati denumirea asociatiilor studentesti si numele facultatii de care apartin pentru cele care au
--aparut inainte de 01-04-2006 si care contin litera i in denumire

SELECT a.denumire as "Nume Asociatie"
FROM asociatie a JOIN facultate USING(cod_facultate)
WHERE a.data_infiintarii>TO_DATE('01-04-2006','dd-mm-yyyy') and a.denumire like '%I%';
```

	Nume Asociatie
1	ASMI
2	ASID

```
--afisati nume si emailul profesorilor care au salariu mai mare decat salariul mediu si
-- care lucreaza la facultati al caror cod postal incepe cu '010'

SELECT CONCAT(CONCAT(p.nume,' '),p.prenume) as "Profesor", p.mail as "Mail"
from profesor p
where p.cod_profesor in (select cod_profesor
                         from contract
                         where salariu > (select avg(salariu)
                                          from contract))
        and p.cod_profesor in(select cod_profesor
                              from contract
                              where cod_facultate in (select cod_facultate
                                                      from facultate
                                                      where cod_postal LIKE '010%'
                                                      )
                             );
```

	Profesor	Mail
1	Popescu Ion	ion.popescu@gmail.com

```
--pentru fiecare profesor sa se afiseze ce salariu obtin pentru fiecare dintre facultatile cu
-- codurile de la 1 pana la 5 si salariu total obtinut de un profesor, se eticheteaza coloanele corespunzator.


SELECT cod_profesor, SUM(DECODE(cod_facultate, 1, salariu)) Facultate1,
 SUM(DECODE(cod_facultate, 2, salariu)) Facultate2,
 SUM(DECODE(cod_facultate, 3, salariu)) Facultate3,
 SUM(DECODE(cod_facultate, 4, salariu)) Facultate4,
 SUM(DECODE(cod_facultate, 5, salariu)) Facultate5,
 SUM(salariu) as Total
FROM contract
GROUP BY cod_profesor;
```

	COD_PROFESOR	FACULTATE1	FACULTATE2	FACULTATE3	FACULTATE4	FACULTATE5	TOTAL
1	100	5400	(null)	5400	(null)	(null)	10800
2	102	4500	(null)	(null)	(null)	(null)	4500
3	101	(null)	5200	(null)	(null)	(null)	5200
4	104	4700	(null)	(null)	(null)	(null)	4700
5	105	4700	(null)	(null)	(null)	(null)	4700
6	103	4200	(null)	(null)	(null)	(null)	4200

```
-- afisati numele, prenumele data inscrierii,codul asociatiei la care esti voluntar sau un mesaj
-- daca nu este pentru stundetii care s-au inscris in ultimelr 15 luni la facultate

select s.nume as Nume, s.prenume as Prenume, i.data_inscrierii as Data, NVL2(to_char(s.cod_asociatie), to_char(s.cod_asociatie), 'Nu e voluntar') as "Voluntar"
from student s, inscriere i
where s.cod_student = i.cod_student and MONTHS_BETWEEN(sysdate,i.data_inscrierii)<15
Order by s.nume;
```

	NUME	PRENUME	DATA	Voluntar
1	Dima	Oana	01-OCT-20	10
2	Miu	Adania	01-OCT-20	Nu e voluntar
3	Nimara	Dan	01-OCT-20	10

```sql
-- afisati pentru fiecare student forma examenului si nota sub forma unui mesaj, utilizand case

SELECT s.nume, s.prenume, e.forma, c.denumire,
CASE
    WHEN p.nota >= 10 THEN 'Nota e maxima'
    WHEN p.nota >= 9 THEN 'Nota e peste 9'
    WHEN p.nota >= 8 THEN 'Nota e peste 8'
    WHEN p.nota >= 7 THEN 'Nota e peste 7'
    WHEN p.nota >= 6 THEN 'Nota e peste 6'
    WHEN p.nota >= 5 THEN 'Nota e peste 5'
    ELSE 'Este picat'
END AS Nota
FROM student s, promoveaza p, examen e, curs c
WHERE s.cod_student = p.cod_student and p.cod_examen = e.cod_examen and e.cod_curs = c.cod_curs
ORDER By s.nume, s.prenume, e.forma, c.denumire;
```

	NUME	PRENUME	FORMA	DENUMIRE	NOTA
1	Baciu	Daniel	Proiect	Istoria Religiilor	Nota e peste 8
2	Dima	Oana	Proiect	Istoria Religiilor	Nota e maxima
3	Fritz	Raluca	Examen Scris	Gandire Critica si Etica Academica	Nota e maxima
4	Fritz	Raluca	Proiect	Baze de Date	Nota e peste 5
5	Fritz	Raluca	Proiect	Istoria Religiilor	Nota e maxima
6	Gherghescu	Andreea	Examen Scris	Chimie organica	Nota e peste 7
7	Guleama	Dan	Proiect	Istoria Religiilor	Nota e peste 7
8	Miu	Adania	Examen Scris	Chimie organica	Nota e peste 8
9	Miu	Adania	Proiect	Baze de Date	Nota e peste 7
10	Nimara	Dan	Examen Scris	Arhitectura sistemelor de calcul	Nota e maxima
11	Nimara	Dan	Examen Scris	Limba si Literatura Engleza	Nota e peste 8
12	Pascu	Adrian	Examen Scris	Limba si Literatura Engleza	Nota e peste 9
13	Pascu	Adrian	Proiect	Baze de Date	Nota e peste 9
14	Vultur	Sofia	Examen Scris	Gandire Critica si Etica Academica	Nota e peste 8

Ex. 12

```
UPDATE CONTRACT
SET salariu = salariu + 0.1*salariu
WHERE cod_profesor in (select cod_profesor
                       from profesor
                       where telefon LIKE '074%');


UPDATE PROMOVEAZA
SET nota = nota + 0.5
WHERE nota < 8 and cod_student in (select cod_student
                       from student
                       where sex LIKE 'feminin');


UPDATE PROMOVEAZA
SET nota = nota - 0.5
WHERE nota >9 and cod_student in (select cod_student
                       from student
                       where nationalitate LIKE 'roman');

COMMIT;
```

--1. Sa se mareasca salariul cu 10% pentru profesorii care au numarul de telefon incepand cu ”074”

--2. Sa se mareasca notele cu 0.5 pentru studentii care au nota mai mica decat 8 si sunt fete - marire de 1 martie

--3. Sa se scada nota studentilor care au nota mai mare decat 9 si sunt romani

Ex. 11 - continuare

```
-- afisez pentru fiecare facultate care are salariu mediu mai mare decat 5200 de lei denumirea acesteia si
--codul profesorului care are salariul mai mare de 5200 de lei si salariul acestuia
with salariu as (select cod_facultate,avg(salariu) media, cod_profesor
                 from contract
                 group by cod_facultate, cod_profesor
                having cod_facultate in (select cod_facultate
                                         from facultate)
                    and cod_profesor in (select cod_profesor
                                         from profesor)
                 )
select f.denumire as "Denumire facultate", cod_profesor as "Cod Profesor", salariu.media as "Salariu"
from salariu join  facultate f on (f.cod_facultate = salariu.cod_facultate)
where salariu.media > 5200
order by f.denumire;
```

	Denumire facultate	Cod Profesor	Salariu
1	Facultatea de Drept	101	5720
2	Facultatea de Geografie	100	5400
3	Facultatea de Matematica si Informatica	100	5400

```
-- sa se afiseze informatiile despre profesorii al caror salariu depaseste in cadrul unei facultati media colegilor sai,
--media salariilor si numarul de angajati de la acea facultate

select *
from contract;

select p.nume, p.prenume, p.mail, p.telefon, c.salariu, (select avg(salariu)
                                                         from contract
                                                         where cod_facultate = c.cod_facultate) as "Salariu Mediu",
                                                         (Select count(*)
                                                         from contract
                                                         where f.cod_facultate = cod_facultate) as "Numar Angajati"

from profesor p, contract c,  facultate f
where f.cod_facultate = c.cod_facultate and p.cod_profesor = c.cod_profesor
and salariu > (select avg(salariu)
               from contract
               where cod_facultate = c.cod_facultate);
```

	NUME	PRENUME	MAIL	TELEFON	SALARIU	Salariu Mediu	Numar Angajati
1	Popescu	Ion	ion.popescu@gmail.com	0756789546	5400	4889	5
2	Avram	Bianca	bianca.avram@gmail.com	0745678213	5445	4889	5

Ex. 16

```
--OUTER JOIN

--returnati pentru fiecare student  numele si prenumele, forma examenului si numele cursului
select s.nume||' '||s.prenume as Student, e.forma as Forma, c.denumire as Curs
from student s right outer join promoveaza p on (p.cod_student = s.cod_student) left outer join examen e
on (p.cod_examen = e.cod_examen) full outer join curs c on (c.cod_curs = e.cod_curs)
order by s.nume;

--returnati numarul de cursuri desfasurate intr-o sala
select s.cod_sala,NVL(count(o.cod_sala),0) as "Numar cursuri"
from sala s left outer join orar o on (s.cod_sala=o.cod_sala)
group by s.cod_sala;

select s.cod_sala,NVL(numar,0) as "Numar cursuri"
from sala s left outer join (select count(*) numar ,cod_sala
from orar
group by cod_sala)o on (s.cod_sala=o.cod_sala);
```

	STUDENT	FORMA	CURS
1	Baciu Daniel	Proiect	Istoria Religiilor
2	Dima Oana	Proiect	Istoria Religiilor
3	Fritz Raluca	Proiect	Istoria Religiilor
4	Fritz Raluca	Examen Scris	Gandire Critica si Etica Academica
5	Fritz Raluca	Proiect	Baze de Date
6	Gherghescu Andreea	Examen Scris	Chimie organica
7	Guleama Dan	Proiect	Istoria Religiilor
8	Miu Adania	Examen Scris	Chimie organica
9	Miu Adania	Proiect	Baze de Date
10	Nimara Dan	Examen Scris	Limba si Literatura Engleza
11	Nimara Dan	Examen Scris	Arhitectura sistemelor de calcul
12	Pascu Adrian	Proiect	Baze de Date
13	Pascu Adrian	Examen Scris	Limba si Literatura Engleza
14	Vultur Sofia	Examen Scris	Gandire Critica si Etica Academica
15		(null)	Educatie Fizica
16		(null)	Programare Orientata pe Obiecte

	COD_SALA	Numar cursuri
1	205	1
2	201	2
3	206	0
4	200	1
5	202	2
6	203	1
7	204	0

Ex. 17 - cod

```
--pentru arbore
select p.nume, p.prenume, c.salariu
from profesor p join contract c on (p.cod_profesor = c.cod_profesor)
where c.cod_facultate = 1 and c.salariu >= 5000 and data_inceput > to_date('01-08-2017','dd-mm-yyyy');


select p.nume, p.prenume, c.salariu
from profesor p join contract c on (p.cod_profesor = c.cod_profesor)
where c.cod_facultate in (select cod_facultate
                          from contract t
                          where t.cod_facultate = c.cod_facultate and t.cod_facultate = 1)
                      and  c.salariu in (select salariu
                                         from contract ct
                                         where ct.salariu >= 5000 and c.cod_profesor = ct.cod_profesor)
                                         and data_inceput in (select data_inceput
                                                              from contract cc
                                                              where cc.data_inceput > to_date('01-08-2017','dd-mm-yyyy') and  c.cod_profesor = cc.cod_profesor)
                                                              ;


select p.nume, p.prenume, c.salariu
from (select salariu, cod_profesor
        from contract
        where cod_facultate = 1 and salariu >= 5000 and data_inceput > to_date('01-08-2017','dd-mm-yyyy') ) c
join(select nume,prenume, cod_profesor
        from profesor ) p on (p.cod_profesor = c.cod_profesor);
```

	NUME	PRENUME	SALARIU
1	Popescu	Ion	5400

Ex. 16 - division

```
--Sa se obtina profesorii care au ore cu grupe din seria 15.
SELECT *
FROM profesor  p
where not exists (
    select cod_grupa
    from orar
    where p.cod_profesor=cod_profesor
    minus
    select cod_grupa
    from grupa
    where denumire like '15%' );
```

	COD_PROFESOR	NUME	PRENUME	TELEFON	MAIL
1	104	Enache	Teodora	0723563781	enache.teo@gmail.com

```
--returneaza toti studentii care au promovat examenul de la cursul cu codul 303


select *
from student s
where not exists (
    (select cod_examen
    from promoveaza
    where s.cod_student=cod_student)
    minus
    (select cod_examen
    from examen
    where cod_curs= 303) );
```

	COD_STUDENT	NUME	PRENUME	DATA_NASTERII	SEX	NATIONALITATE	TELEFON	MAIL	COD_ASOCIATIE
1	58	Marton	Sergiu	15-FEB-01	masculin	roman	0742561340	sergiu.marton@gmail.com	(null)