

# PROIECT

## BAZE DE DATE

NUMELE PROIECTULUI: Gestionarea cinematografelor

STUDENT: David-Leonard Vilvoi

GRUPA: 211

Facultatea de Matematica si Informatica, domeniul Matematica-Informatica

## **1. Descrierea modelului real, a utilitatii acestuia si a regulilor de functionare**

Modelul de date gestioneaza informatii legate de mai multe cinematografe. Scopul modelului este de a stoca si a organiza informatii intr-un mod structurat si coerent.

Avem diferite cinematografe, aflate in mai multe locatii. In cadrul fiecarui cinematograf exista angajati care se ocupa de acesta. Filmele sunt redade in salile de vizionare, dupa un program stabilit, iar acestea pot primi recenzii din partea clientilor. Accesul in sala se face pe baza unui bilet, iar pentru acestea se poate efectua o rezervare.

## **2. Prezentarea constrangerilor impuse asupra modelului**

- Fiecare film este asociat cu minim un gen.
- Fiecare sala de cinema este asociata unui singur cinematograf.
- Un cinematograf se afla intr-o locatie unica.
- Un angajat lucreaza in cadrul unui singur cinematograf.
- Fiecare loc din sala este asociat unui bilet.
- Clientul poate da mai multe recenzii.
- Fiecare proiectie trebuie asociata unui film si cinematograf, ambele fiind valide.
- Fiecare bilet este asociat unei singure proiectii.
- Rezervarea este legata de un client si un bilet.

### 3. Descreirea entitatilor, incluzand precizarea cheii primare

Entitatile modelului sunt: FILME, CINEMATOGRAFE, SALI\_CINEMA, PROGRAM\_PROIECTIE, BILETE, CLIENTI, REZERVARI, ANGAJATI, RECENZII, JOBS, LOCATII si GENURI.

Toate entitatile prezentate sunt dependente, cu exceptia: CLIENTI, GENURI, LOCATII si JOBS.

1. FILME stocheaza informatii despre filmele care pot fi vizionate. Cheia primara este film\_id.
2. CINEMATOGRAFE contine informatii desore cinematografe: numele, locatia si capacitatea. Cheia primara a acestui tabel este cinema\_id.
3. SALI\_CINEMA pastreaza informatii despre salile de cinema din diferite cinematografe. Aceasta are o anumita capacitate, si un numar in cadrul cinematografului. Cheia primara este sala\_id.
4. PROGRAM\_PROIECTIE contine informatii despre programul filmelor ce ruleaza in cadrul unui cinematograf. Cheia primara este program\_id.
5. BILETE asociaza fiecare bilet unui film din program, si contine si alte informatii, cum ar fi pretul si numarul scaunului. Cheia primara a acestui tabel este bilet\_id.
6. CLIENTI stocheaza date de baza despre clienti, cum ar fi nume, prenume si adresa de email. Cheia primara este client\_id.
7. REZERVARI contine informatii despre rezervarile facute de catre clienti. Cheia primara a acestei entitati este rezervare\_id.
8. ANGAJATI pastreaza informatii despre angajatii cinematografelor, cum ar fi pozitia acestora si salariul. Cheia primara este angajat\_id.
9. RECENZII stocheaza informatii despre recenziile date de catre clienti filmelor vizionate. Cheia primara pentru acest tabel este recenzie\_id.
10. JOBS contine informatii despre job-uri: titlul acestuia si range-ul salarial. Cheia primara este job\_id.
11. LOCATII pastreaza informatii despre locatiile in care se afla cinematografele. Cheia primara este locatie\_id.
12. GENURI stocheaza informatii despre genurile de filme. Cheia primara a acestui tabel este gen\_id.

#### **4. Descrierea relatiilor, incluzand precizarea cardinalitatii acestora.**

FILME are GENURI = relatie care leaga entitatile FILME si GENURI, reflectand legatura dintre acestea. Cardinalitatea minima 1:0, deoarece un film poate avea zero genuri, iar un gen poate fi asociat cu zero filme. Cardinalitatea maxima este n:n, deoarece un film poate avea mai multe genuri, iar un gen poate fi asociat mai multor filme. Relatia este de tip "Many-to-Many".

CINEMATOGRAFUL are LOCATIE = relatie care leaga entitatile CINEMATOGRAFE si LOCATII, reflectand legatura dintre acestea. Cardinalitatea minima este 0:1, deoarece un cinematograf poate fi asociat cu o singura locatie, iar o locatie poate fi asociata cu zero cinematografe. Cardinalitatea maxima este n:1, deoarece o locatie poate fi asociata cu mai multe cinematografe. Relatia este de tip "Many-to-One".

SALI \_CINEMA are CINEMATOGRAFE = relatie care leaga entitatile CINEMATOGRAFE si SALI \_CINEMA, reflectand legatura dintre acestea. Cardinalitatea minima este 1:1, deoarece fiecare sala de cinema trebuie asociata unui cinematograf. Cardinalitatea maxima este n:1, deoarece un cinematograf poate avea mai multe sali de cinema. Relatia este de tip "Many-to-One".

PROGRAMUL de PROIECTIE reda FILM = relatie care leaga entitatea PROGRAM \_PROIECTIE si FILME, reflectand legatura dintre acestea. Cardinalitatea minima este 1:1, adica fiecare program de proiectie este asociat unui singur film. Cardinalitatea maxima este n:1, deoarece un film poate fi proiectat in mai multe programe. Relatia este de tip "One-to-Many".

PROGRAMUL de PROIECTIE are SALA = relatie care leaga entitatea SALA\_ID de PROGRAM \_PROIECTIE, reflectand legatura dintre acestea. Cardinalitatea minima este 1:1, deoarece un program de proiectie este asociat unei sali. Cardinalitatea maxima este n:1, deoarece o sala poate avea mai multe programe de proiectie. Relatia este de tip "Many-to-One".

BILETUL are PROGRAM de PROIECTIE = relatie care leaga entitatea BILETE de PROGRAM \_PROIECTIE, reflectand legatura dintre acestea (biletul este pentru vizionarea unui film din program). Cardinalitatea minima este 1:0, deoarece un program de proiectie poate sa nu aiba bilete asociate. Cardinalitatea maxima este n:1, deoarece un program de proiectie poate avea mai multe bilete. Relatia este de tip "Many-to-One".

REZERVARE are CLIENT = relatie care leaga entitatile CLIENTI si REZERVARI, reflectand legatura dintre acestea (rezervarea este asociata unui client). Cardinalitatea minima este 1:1, deoarece o rezervare trebuie sa fie facuta de un client. Cardinalitatea maxima este n:1, deoarece un client poate avea mai multe rezervari. Relatia este de tip "Many-to-One".

REZERVARE are BILET = relatie care leaga entitatea REZERVARI de BILETE, reflectand legatura dintre acestea (rezervarea este asociata unui bilet). Cardinalitatea minima este 1:1, deoarece o rezervare trebuie sa aibe un bilet. Cardinalitatea maxima este 1:1, deoarece un bilet poate avea o singura rezervare. Relatia este de tip "One-to-One".

ANGAJATUL are JOB = relatie care leaga entitatea JOBS de ANGAJATI, reflectand legatura dintre acestea. Cardinalitatea minima este 1:1, deoarece un angajat trebuie sa aibe un job, iar cardinalitatea maxima este n:1 , deoarece mai multi angajati pot avea acelasi job. Relatia este de tip "One-to-Many".

ANAGAJAT lucreaza CINEMA = relatie care leaga entitatile ANGAJATI de CINEMATOGRAFE, reflectand legatura dintre acestea (angajatul lucreaza in cadrul unui cinematograful). Cardinalitatea minima este 1:1, deoarece angajatul trebuie sa lucreze in cadrul unui cinematograful, iar cardinalitatea maxima este n:1 deoarece mai multi angajati pot lucra in cadrul unui singur cinematograful. Relatia este de tip "Many-to-One".

RECENZIA are FILM = relatie care leaga entitatea RECENZII de FILME, reflectand legatura dintre acestea. Cardinalitatea minima este 1:1, deoarece o recenzie este asociata unui singur film. Cardinalitatea maxima este n:1, deoarece un film poate avea mai multe recenzii. Relatia este de tip "Many-to-One".

RECENZIA are CLIENT = relatie care leaga entitatile RECENZII de CLIENTI, reflectand legatura dintre acestea. Cardinalitatea minima este 1:1, deoarece o recenzie este asociata unui singur client. Cardinalitatea maxima este n:1, deoarece un client poate lasa mai multe recenzii. Relatia este de tip "Many-to-One".

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

1. Entitatea dependenta FILME are ca attribute:

- film\_id = cheie primara, numar;
- titlu = titlul filmului, sir de caractere;
- durata = durata filmului, numar;
- data\_lansare = data de lansare a filmului, de tip data;

2. Entitatea dependenta CINEMATOGRAFE are ca attribute:

- cinema\_id = cheie primara, numar;
- nume = numele cinematografului, sir de caractere;
- capacitate = numarul de locuri disponibile in cinematograf, number;
- locatie\_id = locatia cinematografului, numar, trebuie sa corespunda unei chei primare din LOCATII (nu poate fi null).

3. Entitatea dependenta SALI\_CINEMA are ca attribute:

- sala\_id = cheie primara, numar;
- nr\_sala = numarul salii din cadrul cinematografului, numar;
- capacitate = numarul de locuri disponibile intr-o sala, numar;
- cinema\_id = numar, trebuie sa corespunda unei chei primare din CINEMATOGRAFE (nu poate fi null).

4. Entitatea dependenta PROGRAM\_PROIECTIE are ca attribute:

- program\_id = cheie primara, numar;
- start\_time = momentul in care incepe redarea unui film, timestamp;
- end\_time = momentul in care se sfarseste redarea unui film, timestamp;
- film\_id = filmul care trebuie redat, numar, trebuie sa corespunda unei chei primare din FILME;
- sala\_id = sala in care este redat filmul, numar, trebuie sa corespunda unei chei primare din SALI\_CINEMA.

5. Entitatea dependentă BILETE are ca atribute:

- bilet\_id = cheie primară, număr;
- nr\_scaun = locul de pe bilet, sir de caractere;
- pret = costul biletului, număr;
- status = starea biletului (valabil, vandut, rezervat);
- program\_id = programul de pe bilet, număr, trebuie să corespundă unei chei primare din PROGRAM\_PROIECTIE (nu poate fi null).

6. Entitatea independentă CLIENTI are următoarele atribute:

- client\_id = cheie primară, număr;
- nume = numele clientului, sir de caractere;
- prenume = prenumele clientului, sir de caractere;
- email = email-ul clientului, sir de caractere;
- nr\_telefon = numărul de telefon al clientului, sir de caractere.

7. Entitatea dependentă REZERVARI are ca atribute:

- rezervare\_id = cheie primară, număr;
- client\_id = clientul care a făcut rezervarea, număr, trebuie să corespundă unei chei primare din CLIENTI (nu poate fi null);
- bilet\_id = ciletul care este rezervat, număr, trebuie să corespundă unei chei primare din BILETE (nu poate fi null);
- data\_rezervare = data în care a fost făcută rezervarea, dată.

8. Entitatea dependentă ANGAJATI are ca atribute:

- angajat\_id = cheie primară, număr;
- nume = numele angajatului, sir de caractere;
- prenume = prenumele angajatului, sir de caractere;

- cinema\_id = cinematograful unde lucreaza, numar, trebuie sa corespunda unei chei primare din CINEMATOGRAFE (nu poate fi null);
- job\_id = jobul pe care il are, numar, trebuie sa corespunda unei chei primare din JOBS (nu poate fi null);
- salariu = remuneratia angajatului, numar;

9. Entitatea dependenta RECENZII are urmatoarele atribute:

- recenzie\_id = cheie primara, numar;
- film\_id = filmul caruia ii este data recenzia, numar, trebuie sa corespunda unei chei primare din FILME (nu poate fi null).
- client\_id = clientul care da recenzia, numar, trebuie sa corespunda unei chei primare din CLIENTI (nu poate fi null).
- rating = nota acordata filmului, numar;
- data\_recenzie = data in care a fost lasata recenzia, data.

10. Entitatea independenta JOBS are ca atribute:

- job\_id = cheia primara, numar;
- titlu\_job = titlul jobului, sir de caractere;
- range\_salariu = minimul si maximul pe care il poate castiga un angajat lucrind acest job, sir de caractere;

11. Entitatea independenta LOCATII are ca atribute:

- locatie\_id = cheie primara, numar;
- oras = orasul in care se poate afla cinematograful, sir de caractere;
- tara = tara in care se afla orasul, sir de caractere.

12. Entitatea independenta GENURI are urmatoarele atribute:

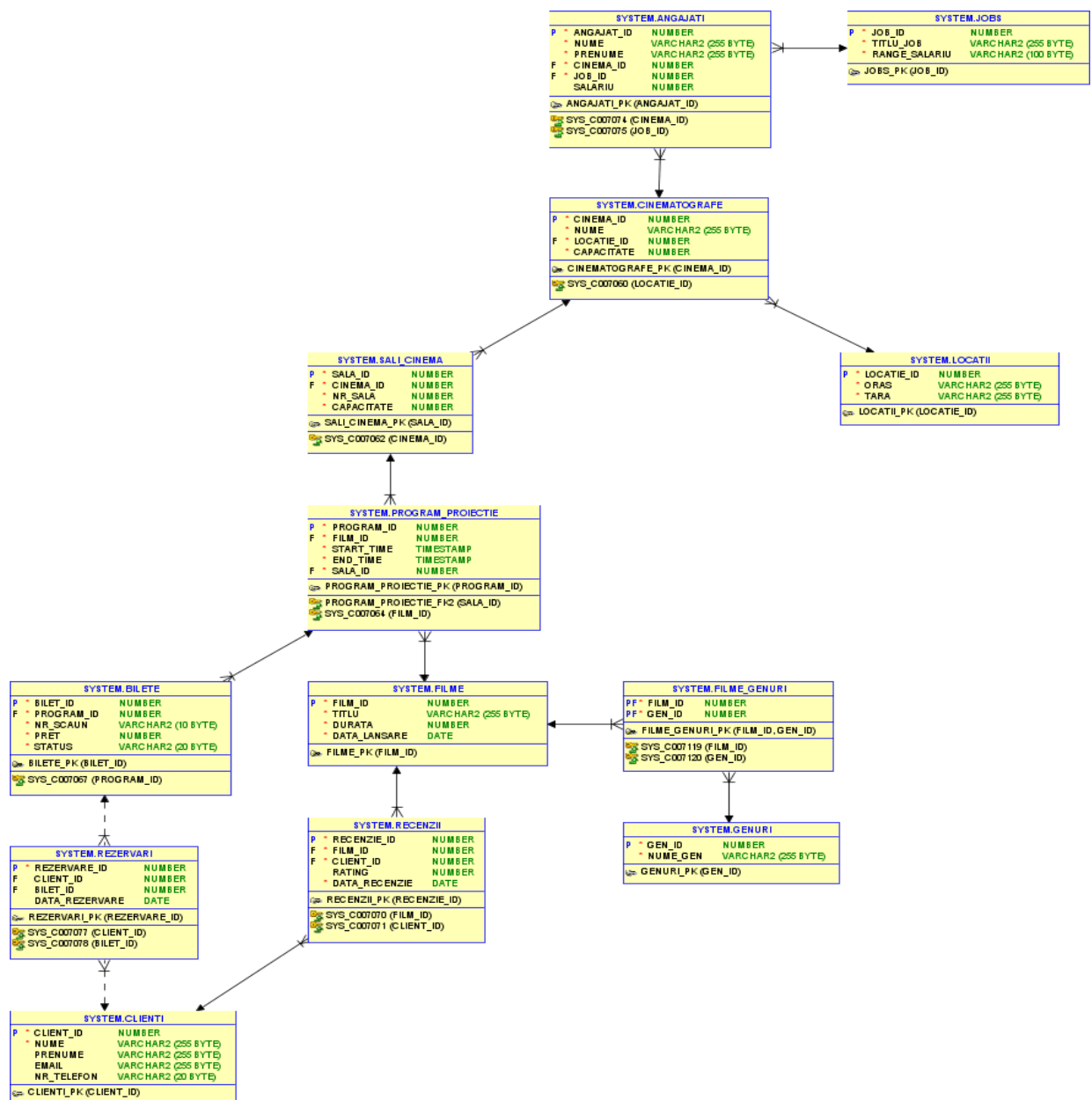
- gen\_id = cheie primara, numar;
- nume\_gen = denumirea genului, sir de caractere;

13. Entitatea dependenta FILME\_GENURI are urmatoarele atribute:

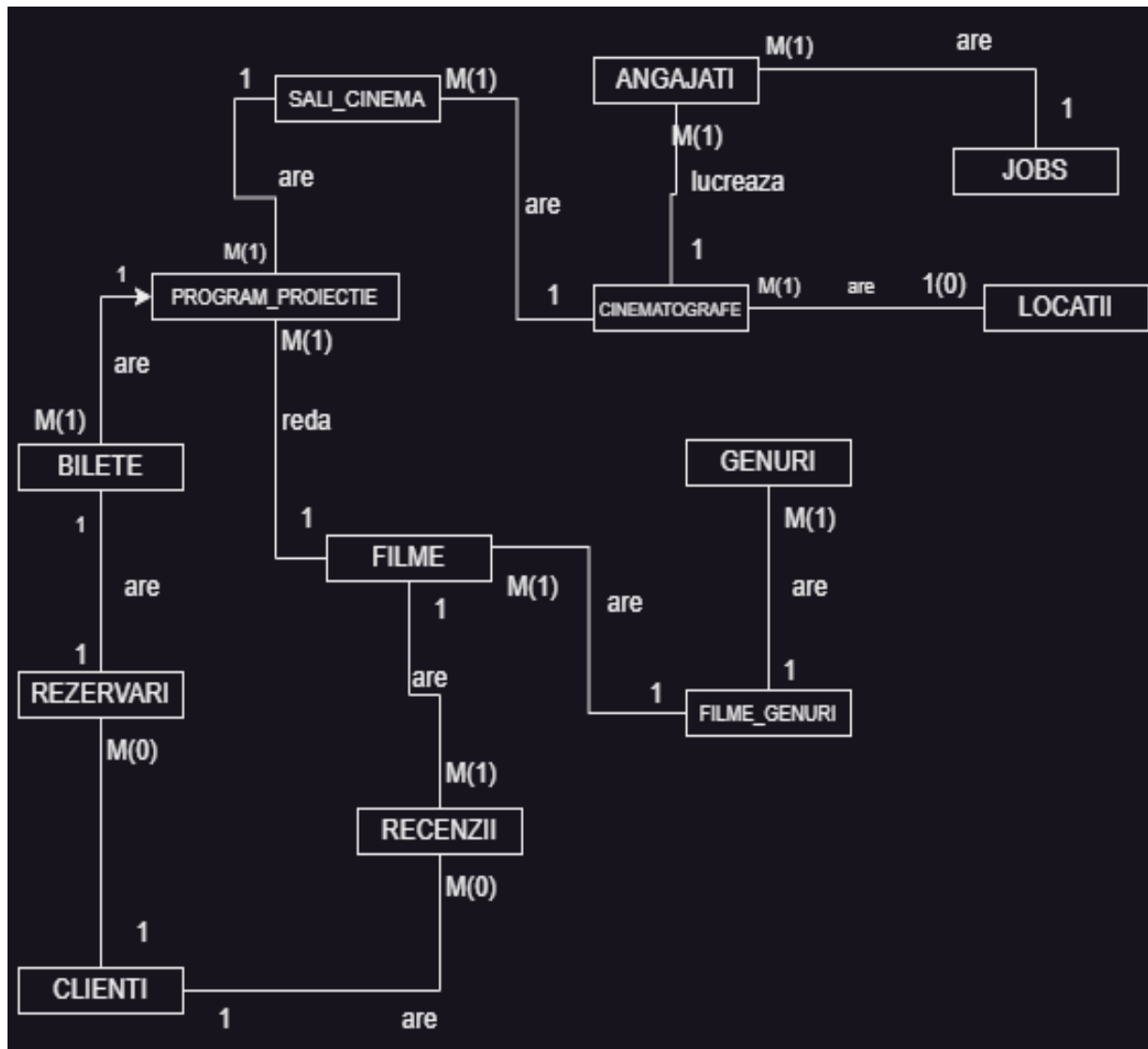
- Film\_id = numar, cheie primara, trebuie sa corespunda unei chei primare din FILME;
- Gen\_id = numar, cheie primara, trebuie sa corespunda unei chei primare din GENURI.



## 6. Realizarea diagramei entitate-relatie corespunzatoare descrierii



7. Realizarea diagramei conceptuale corespunzatoare diagramei entitate relatie proiectate la punctul 6. Diagrama conceptuala trebuie sa contina minim 6 tabele (fara considerarea subentitatilor), dintre care cel putin un tabel asociativ.



## **8. Enumerarea schemelor relationale corespunzatoare diagramei conceptuale proiectate la punctul 7.**

FILME(film\_id#, titlu, durata, data\_lansare)

CINEMATOGRAFE(cinema\_id#, nume, locatie\_id, capacitate)

SALI\_CINEMA(sala\_id#, cinema\_id, nr\_sala, capacitate)

PROGRAM\_PROIECTIE(program\_id#, film\_id, sala\_id, start\_time, end\_time)

BILETE(bilet\_id#, program\_id, nr\_scaun, pret, status)

CLIENTI(client\_id#, nume, prenume, email, nr\_telefon)

REZERVARI(rezervare\_id#, client\_id, bilet\_id, data\_rezervare)

ANGAJATI(angajat\_id#, nume, prenume, cinema\_id, job\_id, salariu)

RECENZII(recenzie\_id#, film\_id, client\_id, ratinf, data\_recenzie)

JOBS(job\_id#, titlu\_job, range\_salariu)

LOCATII(locatie\_id#, oras, tara)

GENURI(gen\_id#, nume\_gen)

FILME\_GENURI(film\_id#, gen\_id#)

## 9. Realizarea normalizarii pana la forma normala 3(FN1-FN3).

Un exemplu non-FN1 este acela daca in tabelul PROGRAM\_PROIECTIE am fi avut:

program_id	film_id
1	10001, 10002, 10003
2	10004, 10005, 10006
3	10007, 10008, 10009

Aceasta relatie nu este in FN1 deoarece exista attribute care sunt compuse.

Pentru ca relatia sa fie in FN1 trebuie ca fiecarui atribut sa ii corespunda o valoare indivizibila.

Un exemplu non-FN2 este daca in tabelul CINEMATOGRAFE am fi avut:

cinema_id	nume	locatie_id	capacitate	oras
501	Cinemagia	1	600	Bucuresti
502	Cineverse	2	500	Timisoara

Cum atributul oras depinde direct de locatie, el nu trebuie sa fie inclus in acest tabel, deoarece in FN2 fiecare atribut care nu este cheie este dependent de cheia primara.

Un exemplu non-FN3 este daca in tabelul PROGRAM\_PROIECTIE am fi avut:

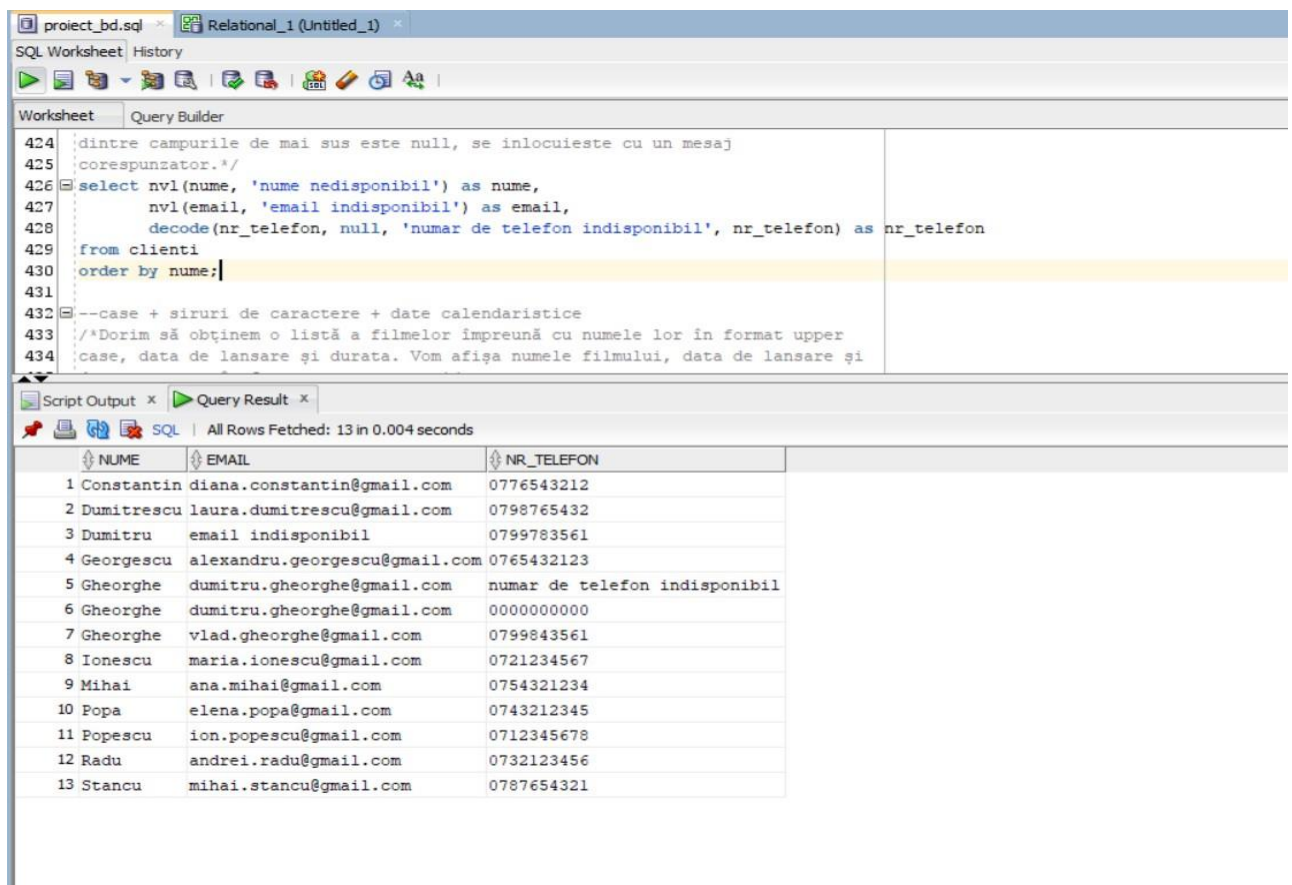
program_id	film_id	sala_id	start_time	end_time	film_titlu
1	10001	3002	18:30	20:00	Iron Man
2	10001	3004	17:00	18:30	Iron Man

Cum atributul film\_titlu depinde de film, el nu trebuie inclus in acest tabel.

## 12. Formulati in limbaj natural si implementati 5 cereri SQL complexe ce vor utiliza, in ansamblul lor, urmatoarele elemente:

- subcereri sincronizate în care intervin cel puțin 3 tabele
- subcereri nesincronizate în clauza FROM
- grupări de date cu subcereri nesincronizate în care intervin cel puțin 3 tabele, funcții grup, filtrare la nivel de grupuri (în cadrul aceleiași cereri)
- Ordonări și utilizarea funcțiilor NVL și DECODE (în cadrul aceleiași cereri)
- utilizarea a cel puțin 2 funcții pe șiruri de caractere, 2 funcții pe date calendaristice, a cel puțin unei expresii CASE
- utilizarea a cel puțin 1 bloc de cerere (clauza WITH)

- a) Afisati numele, emailul si nr de telefon. Rezultatele sunt ordonate alfabetic dupa nume. In cazul in care vreunul dintre campurile de mai sus este null, se inlocuieste cu un mesaj corespunzator.



The screenshot shows an SQL IDE with a query window and a results window. The query is as follows:

```

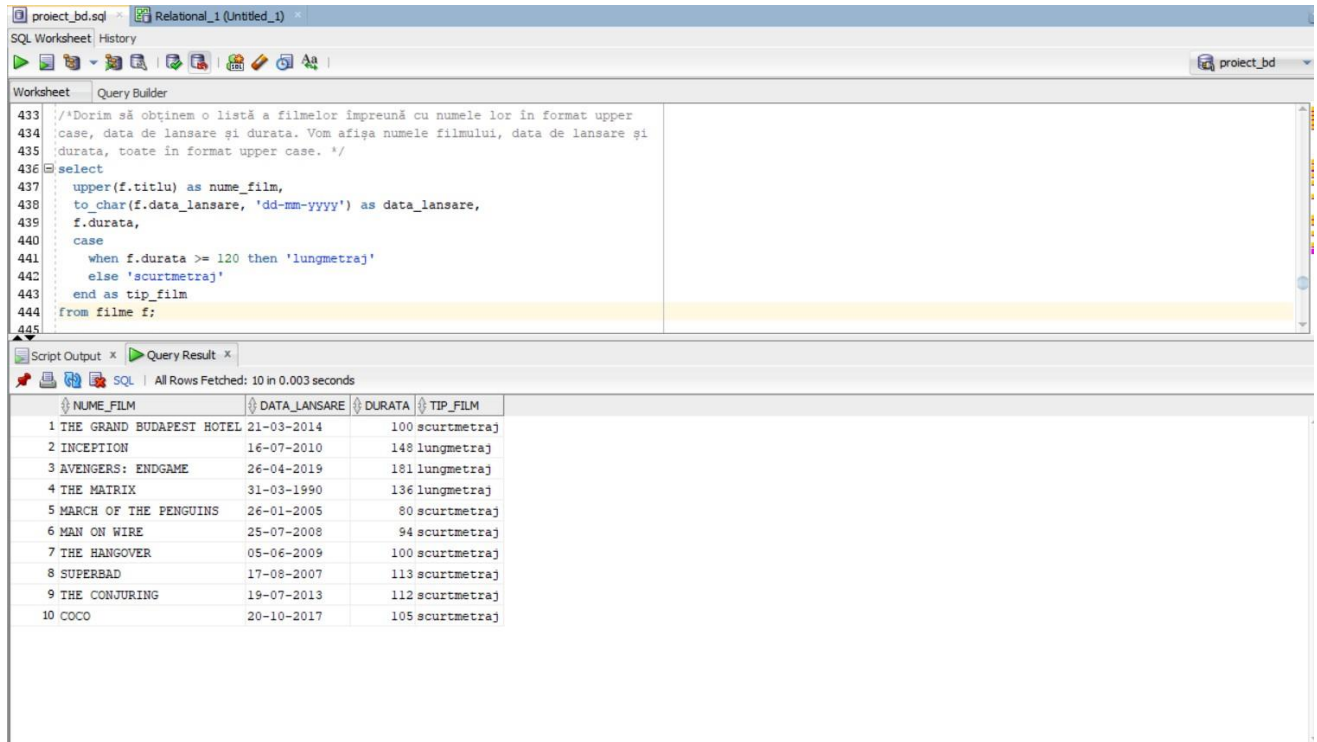
424 dintre campurile de mai sus este null, se inlocuieste cu un mesaj
425 corespunzator.*/
426 select nvl(nume, 'nume nedisponibil') as nume,
427        nvl(email, 'email indisponibil') as email,
428        decode(nr_telefon, null, 'numar de telefon indisponibil', nr_telefon) as nr_telefon
429 from clienti
430 order by nume;
431
432 --case + siruri de caractere + date calendaristice
433 /*Dorim să obținem o listă a filmelor împreună cu numele lor în format upper
434 case, data de lansare și durata. Vom afișa numele filmului, data de lansare și

```

The results window shows the following data:

	NUME	EMAIL	NR_TELEFON
1	Constantin	diana.constantin@gmail.com	0776543212
2	Dumitrescu	laura.dumitrescu@gmail.com	0798765432
3	Dumitru	email indisponibil	0799783561
4	Georgescu	alexandru.georgescu@gmail.com	0765432123
5	Gheorghe	dumitru.gheorghe@gmail.com	numar de telefon indisponibil
6	Gheorghe	dumitru.gheorghe@gmail.com	0000000000
7	Gheorghe	vlad.gheorghe@gmail.com	0799843561
8	Ionescu	maria.ionescu@gmail.com	0721234567
9	Mihai	ana.mihai@gmail.com	0754321234
10	Popa	elena.popa@gmail.com	0743212345
11	Popescu	ion.popescu@gmail.com	0712345678
12	Radu	andrei.radu@gmail.com	0732123456
13	Stancu	mihai.stancu@gmail.com	0787654321

- b) Dorim să obținem o listă a filmelor împreună cu numele lor în format upper case, data de lansare și durata. Vom afișa numele filmului, data de lansare și durata, toate în format upper case.



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

```

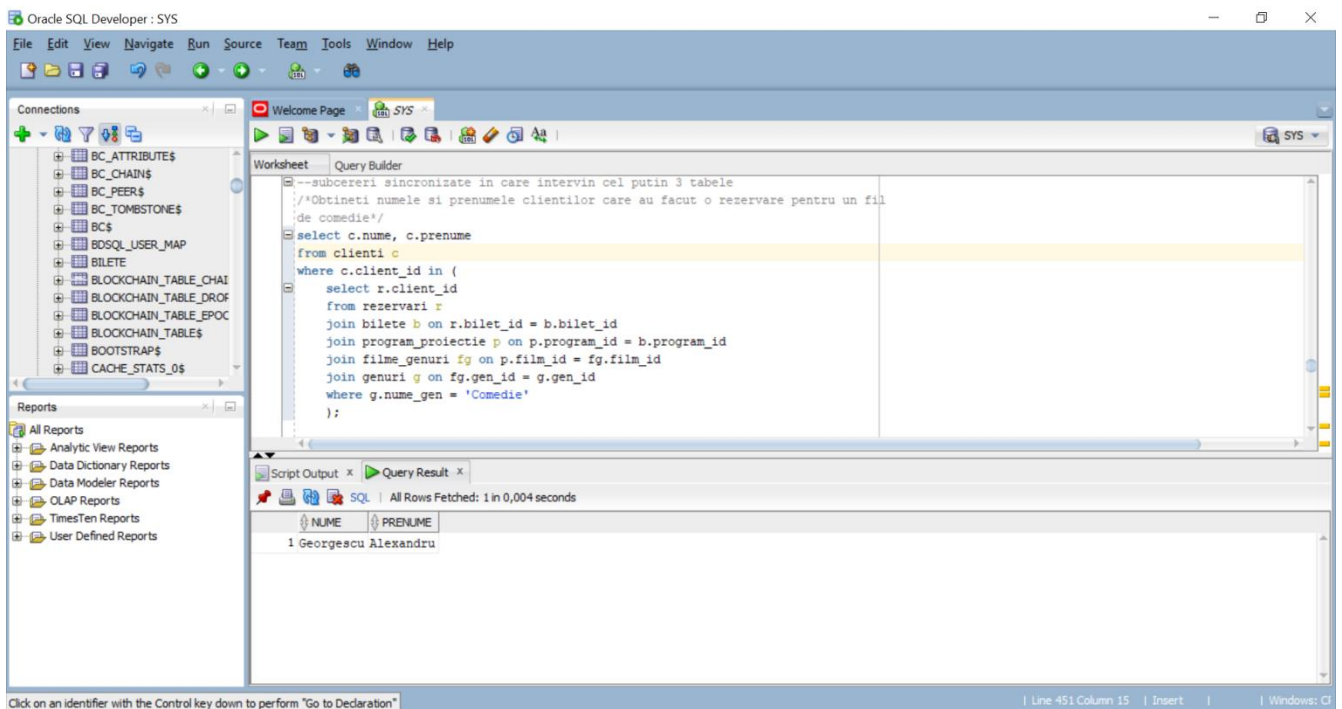
433 /*Dorim să obținem o listă a filmelor împreună cu numele lor în format upper
434 case, data de lansare și durata. Vom afișa numele filmului, data de lansare și
435 durata, toate în format upper case. */
436 select
437     upper(f.titlu) as nume_film,
438     to_char(f.data_lansare, 'dd-mm-yyyy') as data_lansare,
439     f.durata,
440     case
441         when f.durata >= 120 then 'lungmetraj'
442         else 'scurtmetraj'
443     end as tip_film
444 from filme f;
445

```

The results are displayed in a table with 10 rows and 4 columns: NUME\_FILM, DATA\_LANSARE, DURATA, and TIP\_FILM.

	NUME_FILM	DATA_LANSARE	DURATA	TIP_FILM
1	THE GRAND BUDAPEST HOTEL	21-03-2014	100	scurtmetraj
2	INCEPTION	16-07-2010	148	lungmetraj
3	AVENGERS: ENDGAME	26-04-2019	181	lungmetraj
4	THE MATRIX	31-03-1990	136	lungmetraj
5	MARCH OF THE PENGUINS	26-01-2005	80	scurtmetraj
6	MAN ON WIRE	25-07-2008	94	scurtmetraj
7	THE HANGOVER	05-06-2009	100	scurtmetraj
8	SUPERBAD	17-08-2007	113	scurtmetraj
9	THE CONJURING	19-07-2013	112	scurtmetraj
10	COCO	20-10-2017	105	scurtmetraj

- c) Obținem numele și prenumele clienților care au făcut o rezervare pentru un film de comedie.



The screenshot shows the Oracle SQL Developer interface with a query and its results. The query is as follows:

```

--subcereri sincronizate in care intervin cel puțin 3 tabele
/*Obținem numele și prenumele clienților care au făcut o rezervare pentru un film
de comedie*/
select c.num, c.prenume
from clienti c
where c.client_id in (
    select r.client_id
    from rezervari r
    join bilete b on r.bilet_id = b.bilet_id
    join program_proiectie p on p.program_id = b.program_id
    join filme_genuri fg on p.film_id = fg.film_id
    join genuri g on fg.gen_id = g.gen_id
    where g.num_gen = 'Comedie'
);

```

The results are displayed in a table with 2 columns: NUME and PRENUME.

NUME	PRENUME
1	Georgescu Alexandru

- d) Obțineți numărul total de angajați pentru fiecare cinema și oras, afișând rezultatele în ordine descrescătoare a nr de angajați.

The screenshot shows a SQL Worksheet with a query and its results. The query is as follows:

```
-- 3 tabele, funcții grup, filtrare la nivel de grupuri (în cadrul aceleiași cereri)
-- obțineți numărul total de angajați pentru fiecare cinema și oras, afișând
-- rezultatele în ordine descrescătoare a nr de angajați
select c.nume as nume_cinema, l.oras, count(a.angajat_id) as numar_angajati
from cinematografe c
join locatii l on c.locatie_id = l.locatie_id
join angajati a on c.cinema_id = a.cinema_id
group by c.nume, l.oras
order by numar_angajati desc;
```

The results are displayed in a table with the following columns: NUME\_CINEMA, ORAS, and NUMAR\_ANGAJATI. The data is sorted in descending order of the number of employees.

NUME_CINEMA	ORAS	NUMAR_ANGAJATI
1 Cinemagia	Pitești	2
2 Dream Cinema Brasov	Brasov	2
3 GalaxyGlobe	Cluj-Napoca	2
4 Cineverse Craiova	Craiova	2
5 Dream Cinema Bucuresti	Bucuresti	2
6 Cineverse Pitești	Pitești	2
7 Cineverse Brasov	Brasov	2
8 Cinema Enigma	Craiova	2
9 Cineverse Bucuresti	Bucuresti	2
10 Dream Cinema Pitești	Pitești	2

- e) Calculează numărul total de angajați pentru fiecare cinematograful și afișează rezultatele împreună cu numele cinematografului și orașul în care se află.

The screenshot shows a SQL Worksheet with a query and its results. The query is as follows:

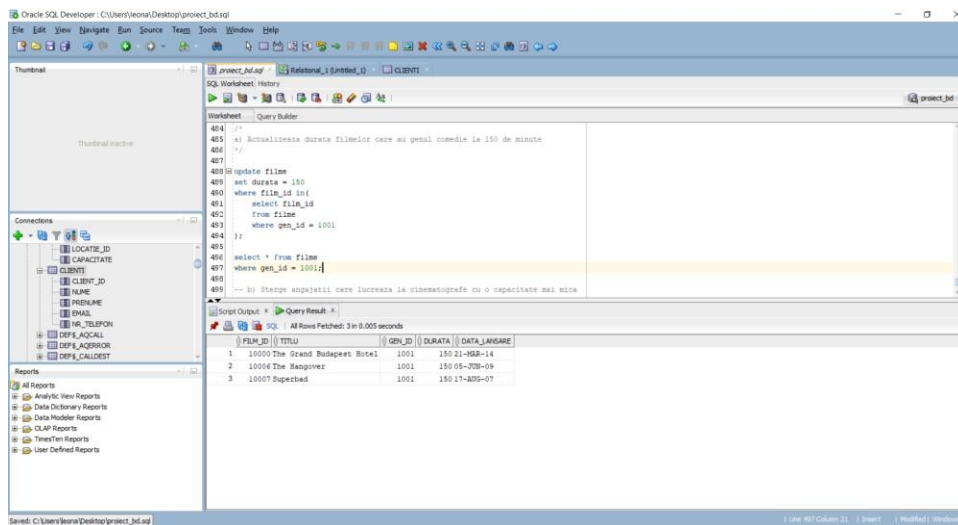
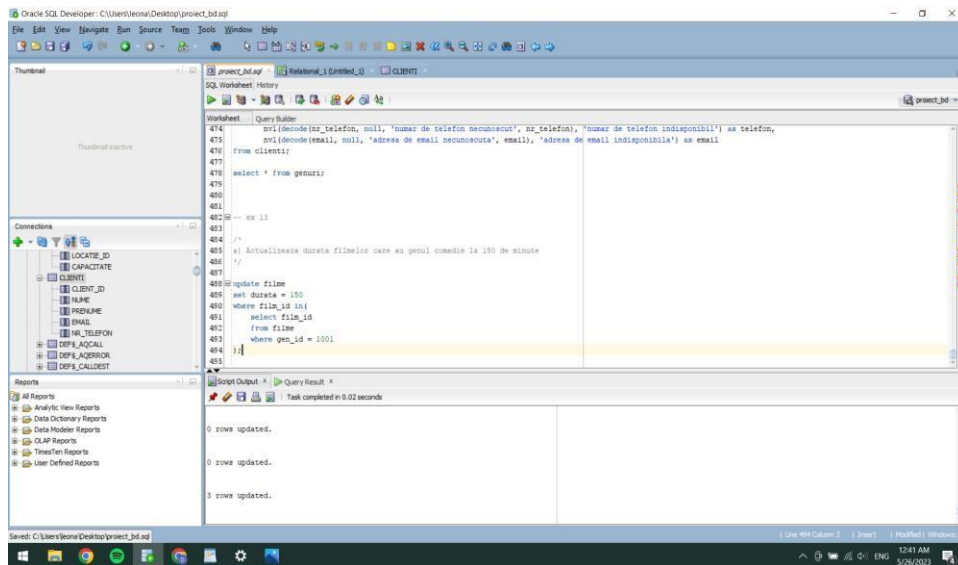
```
-- utilizarea a cel puțin 1 bloc de cerere (clauza WITH)
-- calculează numărul total de angajați pentru fiecare cinematograful și afișează
-- rezultatele împreună cu numele cinematografului și orașul în care se află.
with statistici_angajati as (
  select cinema_id, count(*) as numar_angajati
  from angajati
  group by cinema_id
)
select c.nume, l.oras, sa.numar_angajati
from cinematografe c
inner join locatii l on c.locatie_id = l.locatie_id
left join statistici_angajati sa on c.cinema_id = sa.cinema_id
order by sa.numar_angajati desc;
```

The results are displayed in a table with the following columns: NUME, ORAS, and NUMAR\_ANGAJATI. The data is sorted in descending order of the number of employees.

NUME	ORAS	NUMAR_ANGAJATI
1 Cinema Enigma	Craiova	2
2 Cineverse Bucuresti	Bucuresti	2
3 Dream Cinema Bucuresti	Bucuresti	2
4 Cineverse Pitești	Pitești	2
5 GalaxyGlobe	Cluj-Napoca	2
6 Cineverse Craiova	Craiova	2
7 Cineverse Brasov	Brasov	2
8 Dream Cinema Pitești	Pitești	2
9 Cinemagia	Pitești	2
10 Dream Cinema Brasov	Brasov	2

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

a) Actualizeaza durata filmelor care au genul comedie la 150 de minute.





b) Sterge angajatii care lucreaza la cinematografe cu o capacitate mai mica de 300 de locuri.

The screenshot shows the Oracle SQL Developer interface. The main window displays a SQL script with the following content:

```

485 -- b) Sterge angajatii care lucreaza la cinematografe cu o capacitate mai mica
486 -- de 300 de locuri.
487
488 select * from angajati;
489
490 delete from angajati
491 where cinema_id in (

```

The left sidebar shows the database schema with tables: LOCATIE\_ID, CAPACITATE, CLIENTI, CLIENT\_ID, Nume, PRENUME, EMAIL, NR\_TELEFON, DEPS\_AQCALL, DEPS\_AQBRIOR, and DEPS\_CALLDEST. The bottom right pane shows the query result with 20 rows fetched in 0.064 seconds.

ANGAJAT_ID	Nume	PRENUME	CINEMA_ID	JOB_ID	SALARIU
1	Pogoraru	Ion	500	110	8500
2	Popescu	Maria	501	110	8000
3	Popescu	Ana	502	110	4500
4	Popa	Mihai	503	110	5000
5	Constantin	Andrei	504	110	6000
6	Tranescu	Elena	505	110	6000
7	Dragomir	Cristina	506	110	4000
8	Popovici	Alexandru	507	110	5500
9	Tranescu	Adrian	508	110	4500
10	Tranacu	Mara	509	110	6000
11	Popescu	Andrei	500	140	5500
12	Capatana	Delia	501	140	4500
13	Dragomir	Elena	502	140	6000
14	Dragomirescu	Vasile	503	140	4500
15	Tranacu	Alexandru	504	140	5500
16	Tranescu	Ayren	505	140	6000
17	Tranacu	Delia	506	140	5000
18	Constantinescu	Vlad	507	140	5000
19	Martinovic	Ruxandra	508	140	4500
20	Corodiu	Maria	509	140	5500

The screenshot shows the Oracle SQL Developer interface with a SQL script that includes an update and a delete statement. The script is as follows:

```

487
488 -- b) Sterge angajatii care lucreaza la cinematografe cu o capacitate mai mica
489 -- de 300 de locuri.
490
491 select * from angajati;
492
493 delete from angajati
494 where cinema_id in (
495   select cinema_id
496   from cinematografe
497   where capacitate < 300
498 );

```

The bottom right pane shows the query result with 0 rows updated and 6 rows deleted.

Oracle SQL Developer - C:\Users\leona\Desktop\project\_bd.sql

SQL Worksheet: History

Worksheet: Query Builder

```

494 -- b) Sterge angajatii care inlocuiesc la cinematografe cu o capacitate mai mica
495 -- de 300 de locuri.
496
497 select * from angajati;
498
499 delete from angajati
500 where cinema_id in (
501   select cinema_id
502   from cinematografe
503   where capacitate < 300
504 );

```

Script Output: Query Result: 14 in 0.001 seconds

ANGAJAT_ID	NUME	PRENUME	CINEMA_ID	JOB_ID	SALARIU
1	Popeacu	Ion	500	110	8000
2	Ionescu	Maria	501	110	8000
3	Georgeacu	Ana	502	110	6500
4	Constantin	Andrei	504	110	6000
5	Vasileacu	Elena	505	110	8000
6	Dragomir	Cristina	506	110	6000
7	Popovici	Alexandru	507	110	5500
8	Popeacu	Andrei	500	140	5500
9	Capatana	Delia	501	140	4500
10	Origoire	Elena	502	140	6000
11	Istrate	Alexandru	504	140	5500
12	Barbolescu	Ayten	505	140	6000
13	Bocila	Delia	506	140	5000
14	Constantinescu	Vlad	507	140	5000

Connections: LOCATIE\_ID, CAPACITATE, CLIENT\_ID, NUME, PRENUME, EMAIL, NR\_TELEFON, DEF\_AQ\_CALC, DEF\_AQ\_ERROR, DEF\_AQ\_DELETE

Reports: All Reports, Analytic View Reports, Data Dictionary Reports, OLAP Reports, TimeTen Reports, User Defined Reports

Save: C:\Users\leona\Desktop\project\_bd.sql

c) Actualizeaza salariul managerilor cu 10%

Oracle SQL Developer - C:\Users\leona\Desktop\project\_bd.sql

SQL Worksheet: History

Worksheet: Query Builder

```

504 delete from angajati
505 where cinema_id in (
506   select cinema_id
507   from cinematografe
508   where capacitate < 300
509 );
510
511 -- c) Actualizeaza salariul managerilor cu 10%
512
513 select * from angajati;
514
515 update angajati
516 set salary = salary * 1.1
517 where job_id = 110;

```

Script Output: Query Result: 14 in 0.002 seconds

ANGAJAT_ID	NUME	PRENUME	CINEMA_ID	JOB_ID	SALARIU
1	Popeacu	Ion	500	110	8000
2	Ionescu	Maria	501	110	8000
3	Georgeacu	Ana	502	110	6500
4	Constantin	Andrei	504	110	6000
5	Vasileacu	Elena	505	110	8000
6	Dragomir	Cristina	506	110	6000
7	Popovici	Alexandru	507	110	5500
8	Popeacu	Andrei	500	140	5500
9	Capatana	Delia	501	140	4500
10	Origoire	Elena	502	140	6000
11	Istrate	Alexandru	504	140	5500
12	Barbolescu	Ayten	505	140	6000
13	Bocila	Delia	506	140	5000
14	Constantinescu	Vlad	507	140	5000

Connections: LOCATIE\_ID, CAPACITATE, CLIENT\_ID, NUME, PRENUME, EMAIL, NR\_TELEFON, DEF\_AQ\_CALC, DEF\_AQ\_ERROR, DEF\_AQ\_DELETE

Reports: All Reports, Analytic View Reports, Data Dictionary Reports, OLAP Reports, TimeTen Reports, User Defined Reports

Save: C:\Users\leona\Desktop\project\_bd.sql

Oracle SQL Developer: C:\Users\leonat\Desktop\project\_bd.sql

File Edit View Navigate Run Source Team Tools Window Help

Connections

- LOCATIE\_ID
- CAPACITATE
- CLIENTI
  - CLIENT\_ID
  - NUME
  - PRENUME
  - EMAIL
  - NR\_TELEFON
  - DEPS\_AQCALL
  - DEPS\_AQERROR
  - DEPS\_CALLDEST

Reports

- All Reports
- Analytic View Reports
- Data Dictionary Reports
- Data Modeler Reports
- OLAP Reports
- TimeTen Reports
- User Defined Reports

SQL Worksheet: History

Worksheet Query Builder

```

509 /
510
511 -- c) Actualizeaza salariul managerilor cu 10%
512 select * from angajati;
513
514 update angajati
515 set salarius = salarius * 1.1
516 where job_id = (select job_id from jobs where title_job = 'manager');
517
518
519
520
521

```

Script Output

Task completed in 0.041 seconds

0 rows updated.

Error starting at line : 515 in command -

update angajati

set salarius = salarius \* 1.1

where job\_id = (select job\_id from angajati where title\_job = 'casier')

Error at Command Line : 517 Column : 51

Error report -

SQL Error: ORA-00904: "TITLE\_JOB": invalid identifier

ORA-00904: 00904 - "kai": invalid identifier

\*Cause:

\*Action:

0 rows updated.

7 rows updated.

Click on an identifier with the Control key down to perform "Go to Declaration"

Line 517 Column 76 | Insert | Modified | Windows

Oracle SQL Developer: C:\Users\leonat\Desktop\project\_bd.sql

File Edit View Navigate Run Source Team Tools Window Help

Connections

- LOCATIE\_ID
- CAPACITATE
- CLIENTI
  - CLIENT\_ID
  - NUME
  - PRENUME
  - EMAIL
  - NR\_TELEFON
  - DEPS\_AQCALL
  - DEPS\_AQERROR
  - DEPS\_CALLDEST

Reports

- All Reports
- Analytic View Reports
- Data Dictionary Reports
- Data Modeler Reports
- OLAP Reports
- TimeTen Reports
- User Defined Reports

SQL Worksheet: History

Worksheet Query Builder

```

509 /
510
511 -- c) Actualizeaza salariul managerilor cu 10%
512 select * from angajati;
513
514 update angajati
515 set salarius = salarius * 1.1
516 where job_id = (select job_id from jobs where title_job = 'manager');
517
518
519
520
521

```

Script Output

Query Result

All Rows Fetched: 14 in 0.002 seconds

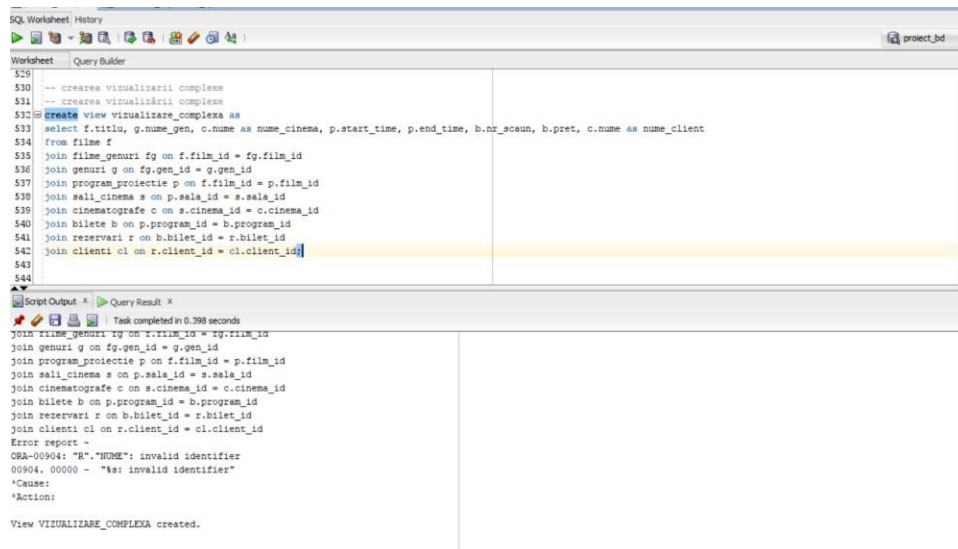
ANGAJ_ID	NUME	PRENUME	CINEMAJ_ID	JOB_ID	SALARIU
1	Popescu	Ion	500	110	8800
2	Popescu	Maria	501	110	8800
3	Popescu	Ana	502	110	7150
4	Constantin	Andrei	504	110	6600
5	Vasilescu	Elena	505	110	8800
6	Dragomir	Cristina	506	110	6600
7	Popescu	Alexandru	507	110	6600
8	Popescu	Andrei	500	140	5500
9	Capatana	Delia	501	140	4500
10	Orsiroze	Elena	502	140	6000
11	Latesate	Alexandru	504	140	5500
12	Badrudeanu	Ayten	505	140	6000
13	Bocila	Delia	506	140	5000
14	Constantinescu	Vlad	507	140	5000

Click on an identifier with the Control key down to perform "Go to Declaration"

Line 513 Column 24 | Insert | Modified | Windows

## 14. Crearea unei vizualizări complexe. Dați un exemplu de operație LMD permisă pe vizualizarea respectivă și un exemplu de operație LMD nepermisă.

Crearea vizualizării:



```
-- crearea vizualizării complexe
-- crearea vizualizării complexe
529 create view vizualizare_complexa as
530 select f.titlu, g.nume_gen, c.nume as nume_cinema, p.start_time, p.end_time, b.nr_scaun, b.pret, c.nume as nume_client
531 from filme f
532 join filme_genuri fg on f.film_id = fg.film_id
533 join genuri g on fg.gen_id = g.gen_id
534 join program_proiectie p on f.film_id = p.film_id
535 join sali_cinema s on p.sala_id = s.sala_id
536 join cinematografe c on s.cinema_id = c.cinema_id
537 join bilete b on p.program_id = b.program_id
538 join rezervari r on b.bilet_id = r.bilet_id
539 join clienti cl on r.client_id = cl.client_id
540
541
542
543
544
```

Script Output: Task completed in 0.398 seconds

Error report -

ORA-00904: "R"."NUME": invalid identifier

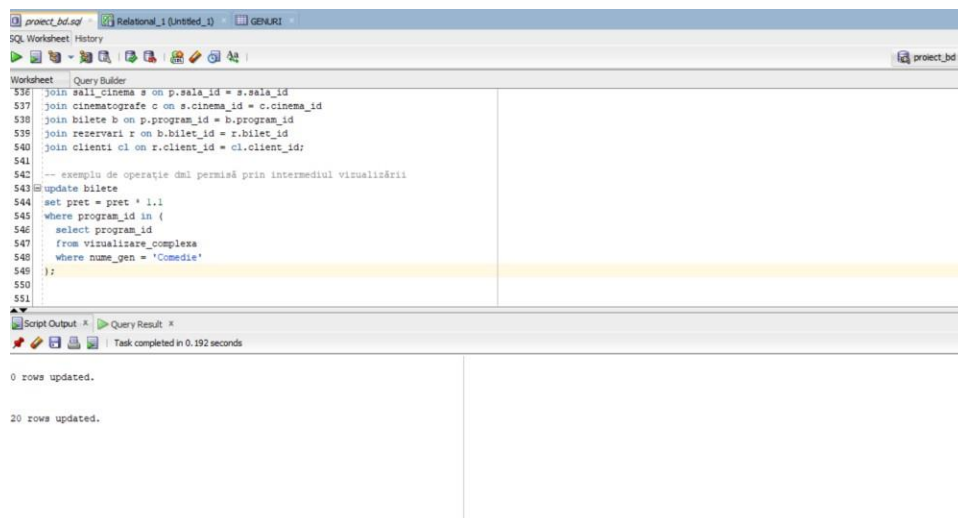
00904. 00000 - "R"."NUME": invalid identifier

\*Cause:

\*Action:

View VIZUALIZARE\_COMPLEXA created.

Un exemplu de operație LMD permisă pe aceasta vizualizare este o actualizare asupra tabelului BILETE pentru a crește prețul biletelor cu 10% pentru toate proiectiile de filme de genul "comedie", utilizând o interogare care se bazează pe vizualizarea "vizualizare\_complexa". Deși actualizarea se aplică direct asupra tabelului BILETE, filtrele și condițiile sunt derivate din vizualizare.



```
536 join sali_cinema s on p.sala_id = s.sala_id
537 join cinematografe c on s.cinema_id = c.cinema_id
538 join bilete b on p.program_id = b.program_id
539 join rezervari r on b.bilet_id = r.bilet_id
540 join clienti cl on r.client_id = cl.client_id
541
542 -- exemplu de operație LMD permisă prin intermediul vizualizării
543 update bilete
544 set pret = pret * 1.1
545 where program_id in (
546 select program_id
547 from vizualizare_complexa
548 where nume_gen = 'Comedie'
549 );
550
551
```

Script Output: Task completed in 0.192 seconds

0 rows updated.

20 rows updated.

Deoarece vizualizarile în bazele de date sunt rezultate ale unor interogări și nu au o structură fizică de stocare a datelor, operațiile LMD nu sunt permise direct pe vizualizări. Prin urmare, nu putem efectua astfel de operații direct pe vizualizarea "vizualizare\_complexa" prezentată anterior.

Un exemplu pentru a ilustra acest lucru este:

delete from vizualizare\_complexa

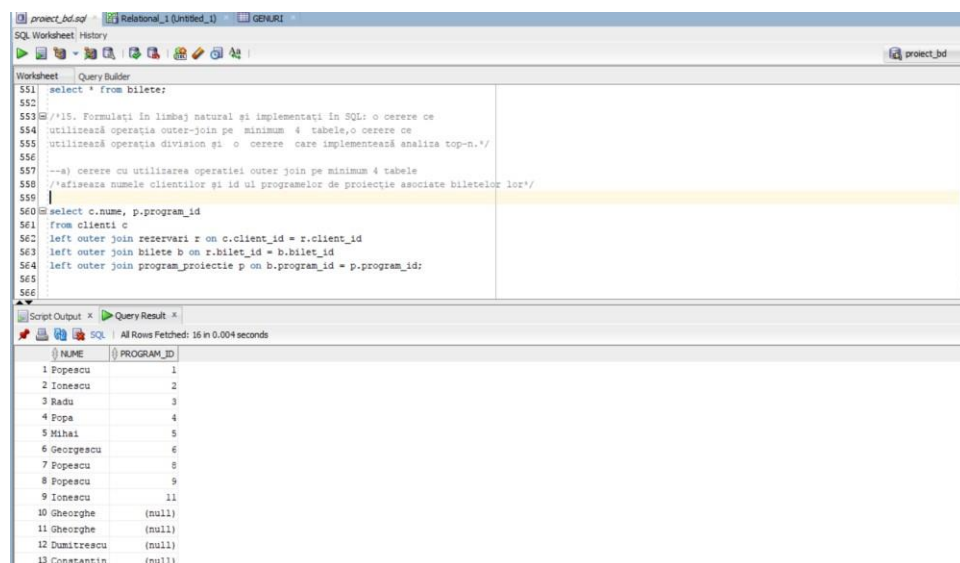
where nume\_client = 'Serban';

În acest exemplu, încercăm să ștergem înregistrările din vizualizare pe baza condiției “nume\_client = ‘Serban’”. Operația “delete” aplicată direct pe vizualizare nu este permisă, deoarece vizualizarea nu reprezintă o entitate fizică cu date stocate, ci este doar o reprezentare virtuală a rezultatelor unei interogări.

### 15. Formulați în limbaj natural și implementați în SQL: o cerere ce utilizează operația outer-join pe minimum 4 tabele, o cerere ce utilizează operația division și o cerere care implementează analiza top-n.

--a) cerere cu utilizarea operației outer join pe minimum 4 tabele

Afișează numele clienților și id-ul programelor de proiecție asociate biletelor lor



The screenshot shows the SQL Developer interface with a query in the 'Query Builder' tab. The query is as follows:

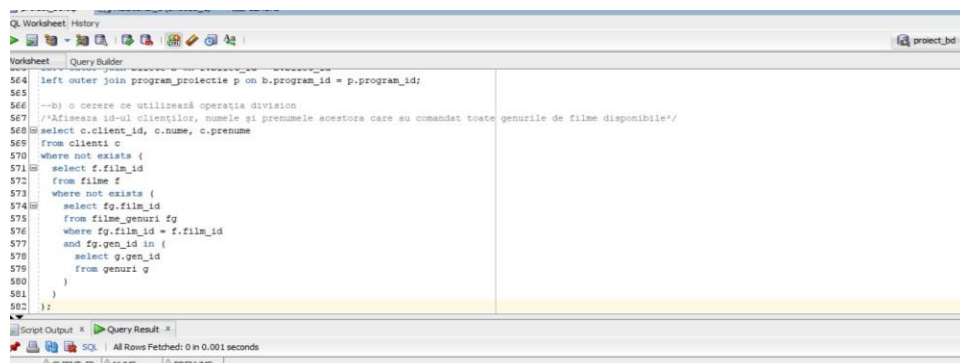
```
551 select * from Bilete;
552
553 /**15. Formulați în limbaj natural și implementați în SQL: o cerere ce
554 utilizează operația outer-join pe minimum 4 tabele, o cerere ce
555 utilizează operația division și o cerere care implementează analiza top-n.*/
556
557 --a) cerere cu utilizarea operației outer join pe minimum 4 tabele
558 /*afișează numele clienților și id-ul programelor de proiecție asociate biletelor lor*/
559
560 select c.nume, p.program_id
561 from clienti c
562 left outer join rezervari r on c.client_id = r.client_id
563 left outer join bilete b on r.bilet_id = b.bilet_id
564 left outer join program_proiectie p on b.program_id = p.program_id;
565
566
```

The 'Query Result' tab shows the following data:

NUME	PROGRAM_ID
1 Popescu	1
2 Ionescu	2
3 Radu	3
4 Popa	4
5 Mihai	5
6 Georgescu	6
7 Popescu	8
8 Popescu	9
9 Ionescu	11
10 Gheorghe	(null)
11 Gheorghe	(null)
12 Dumitrescu	(null)
13 Constantin	(null)

--b) o cerere ce utilizează operația division

Afișează id-ul clienților, numele și prenumele acestora care au comandat toate genurile de filme disponibile



The screenshot shows the SQL Developer interface with a query in the 'Query Builder' tab. The query is as follows:

```
564 left outer join program_proiectie p on b.program_id = p.program_id;
565
566 --b) o cerere ce utilizează operația division
567 /*afișează id-ul clienților, numele și prenumele acestora care au comandat toate genurile de filme disponibile*/
568
569 select c.client_id, c.nume, c.prenume
570 from clienti c
571 where not exists (
572   select f.film_id
573   from filme f
574   where not exists (
575     select fg.film_id
576     from filme_genuri fg
577     where fg.film_id = f.film_id
578     and fg.gen_id in (
579       select g.gen_id
580       from genuri g
581     )
582   )
583 )
584
```

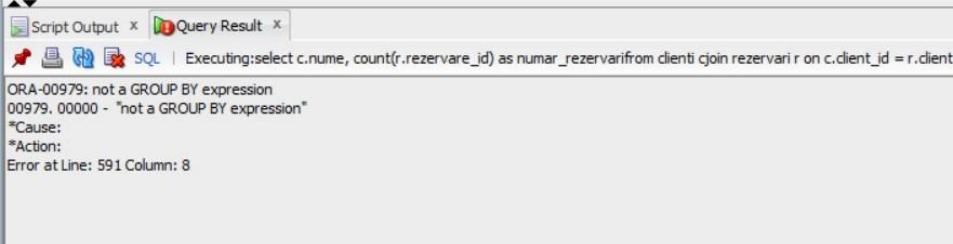
The 'Query Result' tab shows the following data:

CLIENT_ID	NUME	PRENUME
-----------	------	---------

--c) analiza top n

Afisează top 5 clienți cu cele mai multe rezervări.

```
583
584 --c) analiza top-n
585 /*Afisează top 5 clienți cu cele mai multe rezervări*/
586
587 select c.ume, count(r.rezervare_id) as numar_rezervari
588 from clienti c
589 join rezervari r on c.client_id = r.client_id
590 group by c.ume
591 having rownum <= 5
592 order by numar_rezervari desc;
593 --fetch first 5 rows only;
594
595
```



The screenshot shows an Oracle SQL Developer window with two tabs: 'Script Output' and 'Query Result'. The 'Query Result' tab is active, displaying an error message. The error message is: 'ORA-00979: not a GROUP BY expression' followed by '00979. 00000 - "not a GROUP BY expression"', '\*Cause:', '\*Action:', and 'Error at Line: 591 Column: 8'. The error is caused by the 'having rownum <= 5' clause in the query, which is not a valid GROUP BY expression.

Tot programul:

```
create table FILME(
    film_id number primary key,
    titlu varchar2(255),
    durata number,
    data_lansare date
);
```

```
select * from filme;
```

```
create table CINEMATOGRAFE(
    cinema_id number primary key,
    nume varchar(255),
    locatie_id number,
    capacitate number,
    foreign key(locatie_id) references LOCATII(locatie_id)
);
```

```
select * from cinematografe;
```

```
create table SALI_CINEMA(  
    sala_id number primary key,  
    cinema_id number,  
    nr_sala number,  
    capacitate number,  
    foreign key (cinema_id) references CINEMATOGRAFE(cinema_id)  
);
```

```
select * from sali_cinema;
```

```
create table PROGRAM_PROIECTIE(  
    program_id number primary key,  
    film_id number,  
    sala_id number,  
    start_time timestamp,  
    end_time timestamp,  
    foreign key (film_id) references FILME(film_id),  
    foreign key (sala_id) references SALI_CINEMA(sala_id)  
);
```

```
select * from program_proiectie;
```

```
create table BILETE(  
    bilet_id number primary key,  
    program_id number,  
    nr_scaun varchar2(10),  
    pret number,  
    status varchar(20),  
    foreign key (program_id) references program_proiectie(program_id)  
);
```

```
select * from bilete;
```

```
create table clienti(  
    client_id number primary key,  
    nume varchar2(255),  
    prenume varchar2(255),  
    email varchar2(255),  
    nr_telefon varchar2(20)  
);
```

```
create table rezervari(  
    rezervare_id number primary key,  
    client_id number,  
    bilet_id number,  
    data_rezervare date,  
    foreign key (client_id) references clienti(client_id),  
    foreign key (bilet_id) references bilete(bilet_id)  
);
```

```
create table angajati(  
    angajat_id number primary key,  
    nume varchar2(255),  
    prenume varchar2(255),  
    cinema_id number,  
    job_id number,  
    salariu number,  
    foreign key (cinema_id) references cinematografe(cinema_id),  
    foreign key (job_id) references jobs(job_id)  
);
```



```
create table recenzii(  
    recenzie_id number primary key,  
    film_id number,  
    client_id number,  
    rating number,  
    data_recenzie date,  
    foreign key (film_id) references filme(film_id),  
    foreign key (client_id) references clienti(client_id)  
);
```

```
create table jobs(  
    job_id number primary key,  
    titlu_job varchar2(255),  
    range_salariu varchar2(100)  
);
```

```
create table locatii(  
    locatie_id number primary key,  
    oras varchar2(255),  
    tara varchar2(255)  
);
```

```
create table genuri(  
    gen_id number primary key,  
    nume_gen varchar2(255)  
);
```

```
create table filme_genuri(  
    film_id number,  
    gen_id number,  
    primary key (film_id, gen_id),
```

```
foreign key (film_id) references FILME(film_id),
foreign key (gen_id) references GENURI(gen_id)
);
```

--- popularea tabelelor

-- 1. genuri

insert all

```
into genuri(gen_id, nume_gen) values (1000,'Actiune')
into genuri(gen_id, nume_gen) values (1001,'Comedie')
into genuri(gen_id, nume_gen) values (1002,'Drama')
into genuri(gen_id, nume_gen) values (1003,'Horror')
into genuri(gen_id, nume_gen) values (1004,'SF')
into genuri(gen_id, nume_gen) values (1005,'Romantic')
into genuri(gen_id, nume_gen) values (1006,'Thriller')
into genuri(gen_id, nume_gen) values (1007,'Documentar')
into genuri(gen_id, nume_gen) values (1008,'Animatie')
into genuri(gen_id, nume_gen) values (1009,'Mister')
select 1 from dual;
```

```
select * from genuri;
```

```
commit;
```

-- 2. filme

INSERT ALL

```
INTO FILME (film_id, titlu, durata, data_lansare) VALUES (10000, 'The Grand Budapest Hotel', 100,
TO_DATE('2014-03-21', 'YYYY-MM-DD'))
```

```
INTO FILME (film_id, titlu, durata, data_lansare) VALUES (10001, 'Inception', 148, TO_DATE('2010-
07-16', 'YYYY-MM-DD'))
```

```
INTO FILME (film_id, titlu, durata, data_lansare) VALUES (10002, 'Avengers: Endgame', 181,
TO_DATE('2019-04-26', 'YYYY-MM-DD'))
```

```
INTO FILME (film_id, titlu, durata, data_lansare) VALUES (10003, 'The Matrix', 136,  
TO_DATE('1990-03-31', 'YYYY-MM-DD'))
```

```
INTO FILME (film_id, titlu, durata, data_lansare) VALUES (10004, 'March of The Penguins', 80,  
TO_DATE('2005-01-26', 'YYYY-MM-DD'))
```

```
INTO FILME (film_id, titlu, durata, data_lansare) VALUES (10005, 'Man on Wire', 94,  
TO_DATE('2008-07-25', 'YYYY-MM-DD'))
```

```
INTO FILME (film_id, titlu, durata, data_lansare) VALUES (10006, 'The Hangover', 100,  
TO_DATE('2009-06-05', 'YYYY-MM-DD'))
```

```
INTO FILME (film_id, titlu, durata, data_lansare) VALUES (10007, 'Superbad', 113, TO_DATE('2007-  
08-17', 'YYYY-MM-DD'))
```

```
INTO FILME (film_id, titlu, durata, data_lansare) VALUES (10008, 'The Conjuring', 112,  
TO_DATE('2013-07-19', 'YYYY-MM-DD'))
```

```
INTO FILME (film_id, titlu, durata, data_lansare) VALUES (10009, 'Coco', 105, TO_DATE('2017-10-  
20', 'YYYY-MM-DD'))
```

```
SELECT * FROM dual;
```

```
select * from filme;
```

```
commit;
```

```
-- 3. locatii
```

```
insert all
```

```
into locatii (locatie_id, oras, tara) values (1, 'Bucuresti', 'Romania')
```

```
into locatii (locatie_id, oras, tara) values (2, 'Cluj-Napoca', 'Romania')
```

```
into locatii (locatie_id, oras, tara) values (3, 'Pitesti', 'Romania')
```

```
into locatii (locatie_id, oras, tara) values (4, 'Craiova', 'Romania')
```

```
into locatii (locatie_id, oras, tara) values (5, 'Brasov', 'Romania')
```

```
select 1 from dual;
```

```
select * from locatii;
```

```
commit;
```

-- 4. cinematografe

insert all

into cinematografe (cinema\_id, nume, locatie\_id, capacitate) values (500, 'Cineverse Bucuresti', 1, 600)

into cinematografe (cinema\_id, nume, locatie\_id, capacitate) values (501, 'GalaxyGlobe', 2, 350)

into cinematografe (cinema\_id, nume, locatie\_id, capacitate) values (502, 'Dream Cinema Bucuresti', 1, 300)

into cinematografe (cinema\_id, nume, locatie\_id, capacitate) values (503, 'Cinemagia', 3, 250)

into cinematografe (cinema\_id, nume, locatie\_id, capacitate) values (504, 'Cinema Enigma', 4, 300)

into cinematografe (cinema\_id, nume, locatie\_id, capacitate) values (505, 'Cineverse Pitesti', 3, 300)

into cinematografe (cinema\_id, nume, locatie\_id, capacitate) values (506, 'Cineverse Craiova', 4, 300)

into cinematografe (cinema\_id, nume, locatie\_id, capacitate) values (507, 'Cineverse Brasov', 5, 300)

into cinematografe (cinema\_id, nume, locatie\_id, capacitate) values (508, 'Dream Cinema Pitesti', 3, 200)

into cinematografe (cinema\_id, nume, locatie\_id, capacitate) values (509, 'Dream Cinema Brasov', 5, 200)

select 1 from dual;

select \* from cinematografe;

commit;

-- 5. sali\_cinema

insert all

into sali\_cinema (sala\_id, cinema\_id, nr\_sala, capacitate) values (3000, 500, 1, 150)

into sali\_cinema (sala\_id, cinema\_id, nr\_sala, capacitate) values (3001, 500, 2, 150)

into sali\_cinema (sala\_id, cinema\_id, nr\_sala, capacitate) values (3002, 500, 3, 150)

into sali\_cinema (sala\_id, cinema\_id, nr\_sala, capacitate) values (3003, 500, 4, 150)

```
into sali_cinema (sala_id, cinema_id, nr_sala, capacitate) values (3004, 501, 1, 150)
into sali_cinema (sala_id, cinema_id, nr_sala, capacitate) values (3005, 501, 2, 200)
into sali_cinema (sala_id, cinema_id, nr_sala, capacitate) values (3006, 502, 1, 150)
into sali_cinema (sala_id, cinema_id, nr_sala, capacitate) values (3007, 502, 2, 150)
into sali_cinema (sala_id, cinema_id, nr_sala, capacitate) values (3008, 503, 1, 100)
into sali_cinema (sala_id, cinema_id, nr_sala, capacitate) values (3009, 503, 2, 150)
into sali_cinema (sala_id, cinema_id, nr_sala, capacitate) values (3010, 504, 1, 150)
into sali_cinema (sala_id, cinema_id, nr_sala, capacitate) values (3011, 504, 2, 150)
into sali_cinema (sala_id, cinema_id, nr_sala, capacitate) values (3012, 505, 1, 150)
into sali_cinema (sala_id, cinema_id, nr_sala, capacitate) values (3013, 505, 2, 150)
into sali_cinema (sala_id, cinema_id, nr_sala, capacitate) values (3014, 506, 1, 150)
into sali_cinema (sala_id, cinema_id, nr_sala, capacitate) values (3015, 506, 2, 150)
into sali_cinema (sala_id, cinema_id, nr_sala, capacitate) values (3016, 507, 1, 150)
into sali_cinema (sala_id, cinema_id, nr_sala, capacitate) values (3017, 507, 2, 150)
into sali_cinema (sala_id, cinema_id, nr_sala, capacitate) values (3018, 508, 1, 100)
into sali_cinema (sala_id, cinema_id, nr_sala, capacitate) values (3019, 508, 2, 100)
into sali_cinema (sala_id, cinema_id, nr_sala, capacitate) values (3020, 509, 1, 100)
into sali_cinema (sala_id, cinema_id, nr_sala, capacitate) values (3021, 509, 2, 100)
select 1 from dual;
```

```
select * from sali_cinema;
```

```
commit;
```

```
-- 6. programe_proiectie
```

```
insert all
```

```
into program_proiectie (program_id, film_id, sala_id, start_time, end_time)
values (1, 10000, 3000, to_timestamp('2023-05-29 10:00:00', 'yyyy-mm-dd hh24:mi:ss'),
to_timestamp('2023-05-29 11:40:00', 'yyyy-mm-dd hh24:mi:ss'))
```

```
into program_proiectie (program_id, film_id, sala_id, start_time, end_time)

values (2, 10000, 3005, to_timestamp('2023-05-29 10:00:00', 'yyyy-mm-dd hh24:mi:ss'),
to_timestamp('2023-05-29 11:40:00', 'yyyy-mm-dd hh24:mi:ss'))
```

```
into program_proiectie (program_id, film_id, sala_id, start_time, end_time)

values (3, 10000, 3010, to_timestamp('2023-05-29 10:00:00', 'yyyy-mm-dd hh24:mi:ss'),
to_timestamp('2023-05-29 11:40:00', 'yyyy-mm-dd hh24:mi:ss'))
```

```
into program_proiectie (program_id, film_id, sala_id, start_time, end_time)

values (4, 10000, 3015, to_timestamp('2023-05-29 10:00:00', 'yyyy-mm-dd hh24:mi:ss'),
to_timestamp('2023-05-29 11:40:00', 'yyyy-mm-dd hh24:mi:ss'))
```

```
into program_proiectie (program_id, film_id, sala_id, start_time, end_time)

values (5, 10000, 3020, to_timestamp('2023-05-29 10:00:00', 'yyyy-mm-dd hh24:mi:ss'),
to_timestamp('2023-05-29 11:40:00', 'yyyy-mm-dd hh24:mi:ss'))
```

```
into program_proiectie (program_id, film_id, sala_id, start_time, end_time)

values (6, 10001, 3001, to_timestamp('2023-05-29 10:00:00', 'yyyy-mm-dd hh24:mi:ss'),
to_timestamp('2023-05-29 11:40:00', 'yyyy-mm-dd hh24:mi:ss'))
```

```
into program_proiectie (program_id, film_id, sala_id, start_time, end_time)

values (7, 10001, 3006, to_timestamp('2023-05-29 10:00:00', 'yyyy-mm-dd hh24:mi:ss'),
to_timestamp('2023-05-29 11:40:00', 'yyyy-mm-dd hh24:mi:ss'))
```

```
into program_proiectie (program_id, film_id, sala_id, start_time, end_time)

values (8, 10001, 3011, to_timestamp('2023-05-29 10:00:00', 'yyyy-mm-dd hh24:mi:ss'),
to_timestamp('2023-05-29 11:40:00', 'yyyy-mm-dd hh24:mi:ss'))
```

```
into program_proiectie (program_id, film_id, sala_id, start_time, end_time)

values (9, 10001, 3016, to_timestamp('2023-05-29 10:00:00', 'yyyy-mm-dd hh24:mi:ss'),
to_timestamp('2023-05-29 11:40:00', 'yyyy-mm-dd hh24:mi:ss'))
```

```
into program_proiectie (program_id, film_id, sala_id, start_time, end_time)
```

```
values (10, 10001, 3021, to_timestamp('2023-05-29 10:00:00', 'yyyy-mm-dd hh24:mi:ss'),
to_timestamp('2023-05-29 11:40:00', 'yyyy-mm-dd hh24:mi:ss'))
```

```
into program_proiectie (program_id, film_id, sala_id, start_time, end_time)
```

```
values (11, 10002, 3002, to_timestamp('2023-05-29 10:00:00', 'yyyy-mm-dd hh24:mi:ss'),
to_timestamp('2023-05-29 11:40:00', 'yyyy-mm-dd hh24:mi:ss'))
```

```
into program_proiectie (program_id, film_id, sala_id, start_time, end_time)
```

```
values (12, 10002, 3007, to_timestamp('2023-05-29 10:00:00', 'yyyy-mm-dd hh24:mi:ss'),
to_timestamp('2023-05-29 11:40:00', 'yyyy-mm-dd hh24:mi:ss'))
```

```
into program_proiectie (program_id, film_id, sala_id, start_time, end_time)
```

```
values (13, 10002, 3012, to_timestamp('2023-05-29 10:00:00', 'yyyy-mm-dd hh24:mi:ss'),
to_timestamp('2023-05-29 11:40:00', 'yyyy-mm-dd hh24:mi:ss'))
```

```
into program_proiectie (program_id, film_id, sala_id, start_time, end_time)
```

```
values (14, 10002, 3017, to_timestamp('2023-05-29 10:00:00', 'yyyy-mm-dd hh24:mi:ss'),
to_timestamp('2023-05-29 11:40:00', 'yyyy-mm-dd hh24:mi:ss'))
```

```
select 1 from dual;
```

```
select * from program_proiectie; -- trebuie modificat modul de afisare start_time si end_time
```

```
commit;
```

```
-- 7. jobs
```

```
insert all
```

```
into jobs (job_id, titlu_job, range_salariu) values (110, 'manager', '3500-8000')
```

```
into jobs (job_id, titlu_job, range_salariu) values (120, 'casier', '2500-4500')
```

```
into jobs (job_id, titlu_job, range_salariu) values (130, 'usher', '2500-4000')
```

```
into jobs (job_id, titlu_job, range_salariu) values (140, 'tehnician de proiectie', '3000-6000')
```

```
into jobs (job_id, titlu_job, range_salariu) values (150, 'personal de intretinere', '2200-3500')
```

```
select 1 from dual;
```

```
select * from jobs;
```

```
commit;
```

```
-- 8. clienti
```

```
insert all
```

```
    into clienti (client_id, nume, prenume, email, nr_telefon) values (1, 'Popescu', 'Ion',  
'ion.popescu@gmail.com', '0712345678')
```

```
    into clienti (client_id, nume, prenume, email, nr_telefon) values (2, 'Ionescu', 'Maria',  
'maria.ionescu@gmail.com', '0721234567')
```

```
    into clienti (client_id, nume, prenume, email, nr_telefon) values (3, 'Radu', 'Andrei',  
'andrei.radu@gmail.com', '0732123456')
```

```
    into clienti (client_id, nume, prenume, email, nr_telefon) values (4, 'Popa', 'Elena',  
'elena.popa@gmail.com', '0743212345')
```

```
    into clienti (client_id, nume, prenume, email, nr_telefon) values (5, 'Mihai', 'Ana',  
'ana.mihai@gmail.com', '0754321234')
```

```
    into clienti (client_id, nume, prenume, email, nr_telefon) values (6, 'Georgescu', 'Alexandru',  
'alexandru.georgescu@gmail.com', '0765432123')
```

```
    into clienti (client_id, nume, prenume, email, nr_telefon) values (7, 'Constantin', 'Diana',  
'diana.constantin@gmail.com', '0776543212')
```

```
    into clienti (client_id, nume, prenume, email, nr_telefon) values (8, 'Stancu', 'Mihai',  
'mihai.stancu@gmail.com', '0787654321')
```

```
    into clienti (client_id, nume, prenume, email, nr_telefon) values (9, 'Dumitrescu', 'Laura',  
'laura.dumitrescu@gmail.com', '0798765432')
```

```
    into clienti (client_id, nume, prenume, email, nr_telefon) values (10, 'Gheorghe', 'Vlad',  
'vlad.gheorghe@gmail.com', '0799843561')
```

```
select 1 from dual;
```

```
insert into clienti (client_id, nume, prenume, email, nr_telefon) values (11, 'Gheorghe', 'Dumitru',  
'dumitru.gheorghe@gmail.com', '0000000000');
```

```
insert into clienti (client_id, nume, prenume, email, nr_telefon) values (12, 'Dumitru', 'Alex', null,  
'0799783561');
```

```
insert into clienti (client_id, nume, prenume, email, nr_telefon) values (13, 'Gheorghe', 'Dumitru',  
'dumitru.gheorghe@gmail.com', null);
```



```
select * from clienti;
```

```
commit;
```

```
-- 9. bilete
```

```
insert all
```

```
into bilete (bilet_id, program_id, nr_scaun, pret, status) values (20000, 1, 'A1', 20, 'disponibil')
```

```
into bilete (bilet_id, program_id, nr_scaun, pret, status) values (20001, 2, 'A2', 20, 'disponibil')
```

```
into bilete (bilet_id, program_id, nr_scaun, pret, status) values (20002, 3, 'A3', 20, 'disponibil')
```

```
into bilete (bilet_id, program_id, nr_scaun, pret, status) values (20003, 4, 'A4', 20, 'disponibil')
```

```
into bilete (bilet_id, program_id, nr_scaun, pret, status) values (20004, 5, 'A5', 20, 'disponibil')
```

```
into bilete (bilet_id, program_id, nr_scaun, pret, status) values (20005, 6, 'A6', 20, 'disponibil')
```

```
into bilete (bilet_id, program_id, nr_scaun, pret, status) values (20006, 7, 'A7', 20, 'disponibil')
```

```
into bilete (bilet_id, program_id, nr_scaun, pret, status) values (20007, 8, 'A8', 20, 'disponibil')
```

```
into bilete (bilet_id, program_id, nr_scaun, pret, status) values (20008, 9, 'A9', 20, 'disponibil')
```

```
into bilete (bilet_id, program_id, nr_scaun, pret, status) values (20009, 10, 'A10', 20, 'disponibil')
```

```
into bilete (bilet_id, program_id, nr_scaun, pret, status) values (20010, 11, 'B1', 20, 'disponibil')
```

```
into bilete (bilet_id, program_id, nr_scaun, pret, status) values (20011, 12, 'B2', 20, 'disponibil')
```

```
into bilete (bilet_id, program_id, nr_scaun, pret, status) values (20012, 13, 'B3', 20, 'disponibil')
```

```
into bilete (bilet_id, program_id, nr_scaun, pret, status) values (20013, 14, 'B4', 20, 'disponibil')
```

```
into bilete (bilet_id, program_id, nr_scaun, pret, status) values (20014, 1, 'B5', 20, 'disponibil')
```

```
into bilete (bilet_id, program_id, nr_scaun, pret, status) values (20015, 2, 'B6', 20, 'disponibil')
```

```
into bilete (bilet_id, program_id, nr_scaun, pret, status) values (20016, 3, 'B7', 20, 'disponibil')
```

```
into bilete (bilet_id, program_id, nr_scaun, pret, status) values (20017, 4, 'B8', 20, 'disponibil')
```

```
into bilete (bilet_id, program_id, nr_scaun, pret, status) values (20018, 5, 'B9', 20, 'disponibil')
```

```
into bilete (bilet_id, program_id, nr_scaun, pret, status) values (20019, 6, 'B10', 20, 'disponibil')
```

```
select 1 from dual;
```

```
select * from bilete;
```

```
commit;
```

```
-- 10. rezervari
```

```
insert all/*
```

```
into rezervari (rezervare_id, client_id, bilet_id, data_rezervare) values (1, 1, 20000, to_date('2023-05-24', 'yyyy-mm-dd'))
```

```
into rezervari (rezervare_id, client_id, bilet_id, data_rezervare) values (2, 2, 20001, to_date('2023-05-25', 'yyyy-mm-dd'))
```

```
into rezervari (rezervare_id, client_id, bilet_id, data_rezervare) values (3, 3, 20002, to_date('2023-05-26', 'yyyy-mm-dd'))
```

```
into rezervari (rezervare_id, client_id, bilet_id, data_rezervare) values (4, 4, 20003, to_date('2023-05-27', 'yyyy-mm-dd'))
```

```
into rezervari (rezervare_id, client_id, bilet_id, data_rezervare) values (5, 5, 20004, to_date('2023-05-28', 'yyyy-mm-dd'))
```

```
into rezervari (rezervare_id, client_id, bilet_id, data_rezervare) values (6, 6, 20005, to_date('2023-05-29', 'yyyy-mm-dd'))
```

```
*/
```

```
select 1 from dual;
```

```
select * from rezervari;
```

```
commit;
```

```
-- 11. angajati
```

```
insert all
```

```
into angajati(angajat_id, nume, prenume, cinema_id, job_id, salariu) values (1, 'Popescu', 'Ion', 500, 110, 8000)
```

```
into angajati(angajat_id, nume, prenume, cinema_id, job_id, salariu) values (2, 'Ionescu', 'Maria', 501, 110, 8000)
```

```
into angajati(angajat_id, nume, prenume, cinema_id, job_id, salariu) values (3, 'Georgescu', 'Ana',
502, 110, 6500)

into angajati(angajat_id, nume, prenume, cinema_id, job_id, salariu) values (4, 'Popa', 'Mihai', 503,
110, 8000)

into angajati(angajat_id, nume, prenume, cinema_id, job_id, salariu) values (5, 'Constantin', 'Andrei',
504, 110, 6000)

into angajati(angajat_id, nume, prenume, cinema_id, job_id, salariu) values (6, 'Vasilescu', 'Elena',
505, 110, 8000)

into angajati(angajat_id, nume, prenume, cinema_id, job_id, salariu) values (7, 'Dragomir', 'Cristina',
506, 110, 6000)

into angajati(angajat_id, nume, prenume, cinema_id, job_id, salariu) values (8, 'Popovici',
'Alexandru', 507, 110, 5500)

into angajati(angajat_id, nume, prenume, cinema_id, job_id, salariu) values (9, 'Iorgulescu', 'Adrian',
508, 110, 6500)

into angajati(angajat_id, nume, prenume, cinema_id, job_id, salariu) values (10, 'Stancu', 'Mara',
509, 110, 6000)

into angajati(angajat_id, nume, prenume, cinema_id, job_id, salariu) values (11, 'Popescu', 'Andrei',
500, 140, 5500)

into angajati(angajat_id, nume, prenume, cinema_id, job_id, salariu) values (12, 'Capatana', 'Delia',
501, 140, 4500)

into angajati(angajat_id, nume, prenume, cinema_id, job_id, salariu) values (13, 'Grigore', 'Elena',
502, 140, 6000)

into angajati(angajat_id, nume, prenume, cinema_id, job_id, salariu) values (14, 'Dragomirescu',
'Valentin', 503, 140, 4500)

into angajati(angajat_id, nume, prenume, cinema_id, job_id, salariu) values (15, 'Istrate', 'Alexandru',
504, 140, 5500)

into angajati(angajat_id, nume, prenume, cinema_id, job_id, salariu) values (16, 'Barbulescu',
'Ayten', 505, 140, 6000)

into angajati(angajat_id, nume, prenume, cinema_id, job_id, salariu) values (17, 'Bocila', 'Delia', 506,
140, 5000)

into angajati(angajat_id, nume, prenume, cinema_id, job_id, salariu) values (18, 'Constantinescu',
'Vlad', 507, 140, 5000)

into angajati(angajat_id, nume, prenume, cinema_id, job_id, salariu) values (19, 'Martinovic',
'Ruxandra', 508, 140, 4500)

into angajati(angajat_id, nume, prenume, cinema_id, job_id, salariu) values (20, 'Corodi', 'Maria',
509, 140, 5500)

select 1 from dual;
```

```
select * from angajati;
```

```
commit;
```

```
-- 12. recenzii
```

```
insert all
```

```
into recenzii(recenzie_id, film_id, client_id, rating, data_recenzie) values (1, 10001, 1, 8,  
to_date('30.05.2023', 'dd.mm.yyyy'))
```

```
into recenzii(recenzie_id, film_id, client_id, rating, data_recenzie) values (2, 10002, 2, 7,  
to_date('31.05.2023', 'dd.mm.yyyy'))
```

```
into recenzii(recenzie_id, film_id, client_id, rating, data_recenzie) values (3, 10003, 3, 9,  
to_date('01.06.2023', 'dd.mm.yyyy'))
```

```
into recenzii(recenzie_id, film_id, client_id, rating, data_recenzie) values (4, 10004, 4, 6,  
to_date('02.06.2023', 'dd.mm.yyyy'))
```

```
into recenzii(recenzie_id, film_id, client_id, rating, data_recenzie) values (5, 10005, 5, 8,  
to_date('03.06.2023', 'dd.mm.yyyy'))
```

```
into recenzii(recenzie_id, film_id, client_id, rating, data_recenzie) values (6, 10006, 6, 7,  
to_date('04.06.2023', 'dd.mm.yyyy'))
```

```
into recenzii(recenzie_id, film_id, client_id, rating, data_recenzie) values (10, 10007, 7, 9,  
to_date('05.06.2023', 'dd.mm.yyyy'))
```

```
into recenzii(recenzie_id, film_id, client_id, rating, data_recenzie) values (11, 10008, 8, 6,  
to_date('06.06.2023', 'dd.mm.yyyy'))
```

```
into recenzii(recenzie_id, film_id, client_id, rating, data_recenzie) values (12, 10009, 9, 8,  
to_date('07.06.2023', 'dd.mm.yyyy'))
```

```
into recenzii(recenzie_id, film_id, client_id, rating, data_recenzie) values (13, 10009, 10, 7,  
to_date('08.06.2023', 'dd.mm.yyyy'))
```

```
select 1 from dual;
```

```
select * from recenzii;
```

```
commit;
```

```

select * from genuri;

select * from filme;

-- 13. filme_genuri

insert all

into filme_genuri(film_id, gen_id) values (10001,1001)
into filme_genuri(film_id, gen_id) values (10001,1002)
into filme_genuri(film_id, gen_id) values (10002,1000)
into filme_genuri(film_id, gen_id) values (10002,1006)
into filme_genuri(film_id, gen_id) values (10002,1002)
into filme_genuri(film_id, gen_id) values (10003,1000)
into filme_genuri(film_id, gen_id) values (10003,1004)
into filme_genuri(film_id, gen_id) values (10009,1008)
into filme_genuri(film_id, gen_id) values (10009,1009)
select 1 from dual;

```

```

commit;

```

```

select * from filme_genuri;

select * from filme;

```

```

--ex12

```

```

--NVL și DECODE în cadrul aceleiași cereri

```

```

/* Afisati numele, emailul si nr de telefon. Rezulatele
sunt ordonate alfabetic dupa nume. In cazul incare vreunul
dintre campurile de mai sus este null, se inlocuieste cu un mesaj
corespunzator.*/

```

```

select nvl(nume, 'nume nedisponibil') as nume,
       nvl(email, 'email indisponibil') as email,
       decode(nr_telefon, null, 'numar de telefon indisponibil', nr_telefon) as nr_telefon

```

```
from clienti
order by nume;
```

--case + siruri de caractere + date calendaristice

/\*Dorim să obținem o listă a filmelor împreună cu numele lor în format upper case, data de lansare și durata. Vom afișa numele filmului, data de lansare și durata, toate în format upper case. \*/

```
select
    upper(f.titlu) as nume_film,
    to_char(f.data_lansare, 'dd-mm-yyyy') as data_lansare,
    f.durata,
    case
        when f.durata >= 120 then 'lungmetraj'
        else 'scurtmetraj'
    end as tip_film
from filme f;
```

--subcereri sincronizate in care intervin cel puțin 3 tabele

/\*Dorim sa afisam ID-ul și numele clienților care au făcut cel puțin 3 rezervări pentru proiecții care au început după 1 ianuarie 2023. \*/

```
select c.client_id, c.nume, r.numar_rezervari
from clienti c
inner join (
    select r.client_id, count(*) as numar_rezervari
    from rezervari r
    inner join bilete b on r.bilet_id = b.bilet_id
    inner join program_proiectie pp on b.program_id = pp.program_id
    where pp.start_time >= to_timestamp('2023-01-01 00:00:00', 'yyyy-mm-dd hh24:mi:ss')
    group by r.client_id
    having count(*) >= 3
) r on c.client_id = r.client_id;
```

```
-- grupări de date cu subcereri nesincronizate in care intervin cel puțin
-- 3 tabele, funcții grup, filtrare la nivel degrupuri(in cadrul aceleiasi cereri)
/*obtineti numarul total de angajati pentru fiecare cinema si oras, afisand
rezultatele in ordine descrescatoare a nr de angajati*/
select c.numa as nume_cinema, l.oras, count(a.angajat_id) as numar_angajati
from cinematografe c
join locatii l on c.locatie_id = l.locatie_id
join angajati a on c.cinema_id = a.cinema_id
group by c.numa, l.oras
order by numar_angajati desc;
```

```
--utilizarea a cel puțin 1 bloc de cerere(clauza WITH)
/*calculează numărul total de angajați pentru fiecare cinematograful și afișează
rezultatele împreună cu numele cinematografului și orașul în care se află.*/
```

```
with statistici_angajati as (
    select cinema_id, count(*) as numar_angajati
    from angajati
    group by cinema_id
)
select c.numa, l.oras, sa.numar_angajati
from cinematografe c
inner join locatii l on c.locatie_id = l.locatie_id
left join statistici_angajati sa on c.cinema_id = sa.cinema_id
order by sa.numar_angajati desc;
```

```
-- ex 13
```

```
/*
```

```
a) Actualizeaza durata filmelor care au genul comedie la 150 de minute
```

\*/

```
update filme
set durata = 150
where film_id in(
    select film_id
    from filme
    where gen_id = 1001
);
```

```
select * from filme
where gen_id = 1001;
```

-- b) Sterge angajatii care lucreaza la cinematografe cu o capacitate mai mica  
-- de 300 de locuri.

```
select * from angajati;
```

```
delete from angajati
where cinema_id in(
    select cinema_id
    from cinematografe
    where capacitate < 300
);
```

-- c) Actualizeaza slariul managerilor cu 10%

```
select * from angajati;
```

```
update angajati
set salariu = salariu * 1.1
```



```
where job_id = (select job_id from jobs where titlu_job = 'manager');
```

```
select * from jobs;
```

-- 14. Crearea unei vizualizări complexe. Dați un exemplu de operație LMD

-- permisă pe vizualizarea respectivă și un exemplu de operație LMD nepermisă.

```
create view vizualizare_complexa as
```

```
select f.titlu, g.num_e_gen, c.num_e as num_e_cinema, p.start_time, p.end_time, b.nr_scaun, b.pret,  
c.num_e as num_e_client
```

```
from filme f
```

```
join filme_genuri fg on f.film_id = fg.film_id
```

```
join genuri g on fg.gen_id = g.gen_id
```

```
join program_proiectie p on f.film_id = p.film_id
```

```
join sali_cinema s on p.sala_id = s.sala_id
```

```
join cinematografe c on s.cinema_id = c.cinema_id
```

```
join bilete b on p.program_id = b.program_id
```

```
join rezervari r on b.bilet_id = r.bilet_id
```

```
join clienti cl on r.client_id = cl.client_id;
```

-- exemplu de operație dml permisă prin intermediul vizualizării

```
update bilete
```

```
set pret = pret * 1.1
```

```
where program_id in (
```

```
select program_id
```

```
from vizualizare_complexa
```

```
where num_e_gen = 'Comedie'
```

```
);
```

```
select * from bilete;
```

/\*15. Formulați în limbaj natural și implementați în SQL: o cerere ce utilizează operația outer-join pe minimum 4 tabele,o cerere ce utilizează operația division și o cerere care implementează analiza top-n.\*/

--a) cerere cu utilizarea operației outer join pe minimum 4 tabele

/\*afiseaza numele clientilor și id ul programelor de proiecție asociate biletelor lor\*/

```
select c.nume, p.program_id
from clienti c
left outer join rezervari r on c.client_id = r.client_id
left outer join bilete b on r.bilet_id = b.bilet_id
left outer join program_proiectie p on b.program_id = p.program_id;
```

--b) o cerere ce utilizează operația division

/\*Afiseaza id-ul clienților, numele și prenumele acestora care au comandat toate genurile de filme disponibile\*/

```
select c.client_id, c.nume, c.prenume
from clienti c
where not exists (
  select f.film_id
  from filme f
  where not exists (
    select fg.film_id
    from filme_genuri fg
    where fg.film_id = f.film_id
    and fg.gen_id in (
      select g.gen_id
      from genuri g
    )
  )
)
```

```
);
```

```
--c) analiza top-n
```

```
/*Afisează top 5 clienti cu cele mai multe rezervari*/
```

```
select c.nume, count(r.rezervare_id) as numar_rezervari
```

```
from clienti c
```

```
join rezervari r on c.client_id = r.client_id
```

```
group by c.nume
```

```
having rownum <= 5
```

```
order by numar_rezervari desc;
```

```
--fetch first 5 rows only;
```