Baza de date pentru Netflix

Prof. Laborator: Natalia Gabriela Ozunu

Prof. Curs: Letiția Ana Marin

Mincu Adrian-Lucian

grupa 132

Cuprins

Contents

[Descriere model 3](#_Toc144110902)

[Constrângeri 3](#_Toc144110903)

[Descriere entități 4](#_Toc144110904)

[Descriere relații 5](#_Toc144110905)

[Atribute (\* ⬄ obligatoriu) 7](#_Toc144110906)

[Diagrama entitate-relație 11](#_Toc144110907)

[Diagrama conceptuală 12](#_Toc144110908)

[Schemele relaționale 13](#_Toc144110909)

[Normalizări 14](#_Toc144110910)

[FN1 14](#_Toc144110911)

[FN2 15](#_Toc144110912)

[FN3 15](#_Toc144110913)

[Secvență 15](#_Toc144110914)

[Creare și inserare 16](#_Toc144110915)

[creare de tabele: 16](#_Toc144110916)

[inserari: 20](#_Toc144110917)

[Query-uri 32](#_Toc144110918)

[Grupări de date cu subcereri nesincronizate în care intervin cel puțin 3 tabele, funcții grup, filtrare la nivel de grupuri 32](#_Toc144110919)

[Bloc de cerere with 33](#_Toc144110920)

[nvl, decode, case 33](#_Toc144110921)

[Subcereri sincronizate cu 3 tabele, subcere nesincronizata în clauza from 2 funcții pe string uri, 2 pe date 35](#_Toc144110922)

[Pentru fiecare subscripție să se afișeze ce serial și cât durează ordonate după durată și numele subscripției descrescător 36](#_Toc144110923)

[Operații 37](#_Toc144110924)

[ștergerea directorului cu cea mai mică notă 37](#_Toc144110925)

[ștergearea ultimului episod care aparține serialului cu cea mai mare durată 37](#_Toc144110926)

[schimbare filmelor cu nota null in nota 10 38](#_Toc144110927)

[Cereri 38](#_Toc144110928)

[Division 38](#_Toc144110929)

[Top-n 39](#_Toc144110930)

[Outer Join 40](#_Toc144110931)

[Optimizarea unei cereri 41](#_Toc144110932)

[Cererea neoptimizată 41](#_Toc144110933)

[Expresie algebrică: 42](#_Toc144110934)

[Arbore algebric: 42](#_Toc144110935)

[Cererea optimizată: 43](#_Toc144110936)

[Expresie algebrică: 44](#_Toc144110937)

[Arbore algebric: 44](#_Toc144110938)

[A) Normalizări 45](#_Toc144110939)

[BCNF: 45](#_Toc144110940)

[FN4: 46](#_Toc144110941)

[FN5: 46](#_Toc144110942)

[B) Denormalizări 47](#_Toc144110943)

# Descriere model

Un utilizator iși poate alege o singură subscripție si e obligatoriu să aleagă una, altfel nu va putea folosi serviciile oferite, la care se pot utiliza mai multe metode de plată. Filmele sau serialele aparțin unei subscripții, iar utilizatorul se poate uita la ele doar daca este abonat la acea subscripție care le conține.

Fiecare film are un director și trebuie să aibă unul, iar actorii aparțin și filmelor și serialelor, dar pot și să nu fie. Fiecare actor are un rol in film/serial, iar fiecare film/serial are propriile sale roluri.

# Constrângeri

* Un utilizator trebuie să iși aleagă o singură subscripție
* O subscripție poată sa aiba un singur tip de plată, card
* O subscripție are mai multe filme și seriale, dar filmele și serialele pot aparține tuturor subscripțiilor, sau doar la câteva
* Un serial poate să aibă mai multe episoade, dar un episod are un singur serial
* Un film trebuie să aibă un director, dar un director poate să regizeze mai multe filme
* Un actor poate să nu aibă niciun rol, sau mai multe, și poate juca in mai multe filme/seriale, sau niciunul.

# Descriere entități

* **UTILIZATOR**
* persoana fizică care iși poate crea cont pe platforma Netflix
* entitate dependentă de **SUBSCRIPȚIE** și **PLATĂ**
* cheie primară: Utilizator\_ID
* **SUBSCRIPȚIE**
* un pachet care contine filme/seriale cu un pret specific
* entitate independentă
* cheie primară: Subscripție\_ID
* **PLATĂ**
* informațiile cardului cu care plătește utilizatorul
* entitate independentă
* cheie primară: Plată\_ID
* **FILM**
* film, la care se poate uita un utilizator
* entitate dependentă de **DIRECTOR**
* cheie primară: Film\_ID
* **SERIAL**
* serial, la care se poate uita un utilizator
* entitate independentă
* cheie primară: Serial\_ID
* **EPISOD**
* episod, care aparține unui serial
* entitate dependentă de **SERIAL**
* cheieprimară: Episod\_ID
* **ACTOR**
* persoană care joacă într-un film/serial
* entitate independentă
* cheie primară: Actor\_ID
* **ROL**
* caracter dintr-un film care poate fi jucat de un/mai mulți actor/i
* entitate independentă
* cheie primară: Rol\_ID
* **DIRECTOR**
* persoană care regizează un film
* entitate independentă
* cheie primară: Director\_ID

# Descriere relații

UTILIZATOR\_alege\_o\_SUBSCRIPȚIE = relație între UTILIZATOR și SUBSCRIPȚIE

Un UTILIZATOR poate să aleagă decât o singură SUBSCRIPȚIE

O SUBSCRIPȚIE poate să fie aleasă de zero sau mai mulți UTILIZATORI

* Cardinalitate maximă: M:1
* Cardinalitate minimă: 0:1

UTILIZATOR\_adaugă \_o\_PLATĂ = relație între UTILIZATOR și PLATĂ

Un UTILIZATOR poate să adauge decât o singură PLATĂ

O PLATĂ poate să fie adăugată de 1 sau mai mulți UTILIZATORI

* Cardinalitate maximă: M:1
* Cardinalitate minimă: 1:1

SUBSCRIȚIE\_are \_un\_SERIAL = relație între SUBSCRIPȚIE și SERIAL

O SUBSCRIPȚIE poate să aibă zero sau mai multe SERIALE

Un SERIAL poate să aparțină de zero sau mai multe SUBSCRIPȚII

* Cardinalitate maximă: M:M
* Cardinalitate minimă: 0:0

SUBSCRIȚIE\_are \_un\_FILM = relație între SUBSCRIPȚIE și FILM

O SUBSCRIPȚIE poate să aibă zero sau mai multe FILME

Un FILM poate să aparțină de zero sau mai multe SUBSCRIPȚII

* Cardinalitate maximă: M:M
* Cardinalitate minimă: 0:0

ACTOR\_joacă\_în \_FILM = relație între ACTOR și FILM

Un ACTOR poate să joace în zero sau mai multe FILME

Într-un FILM poate să joace zero sau mai mulți ACTORI

* Cardinalitate maximă: M:M
* Cardinalitate minimă: 0:0

ACTOR\_joacă\_în \_SERIAL = relație între ACTOR și SERIAL

Un ACTOR poate să joace în zero sau mai multe SERIALE

Într-un SERIAL poate să joace zero sau mai mulți ACTORI

* Cardinalitate maximă: M:M
* Cardinalitate minimă: 0:0

ACTOR\_are \_ROL = relație între ACTOR și ROL

Un ACTOR poate să aibă zero sau mai multe ROLURI

Un ROL poate să aibă zero sau mai mulți ACTORI

* Cardinalitate maximă: M:M
* Cardinalitate minimă: 0:0

FILM\_are \_ROL = relație între FILM și ROL

Un FILM poate să aibă zero sau mai multe ROLURI

Un ROL poate să aibă zero sau mai multe FILME

* Cardinalitate maximă: M:M
* Cardinalitate minimă: 0:0

DIRECTOR\_regizează\_un\_FILM = relație între DIRECTOR și FILM

Un DIRECTOR poate să regizeze în unul sau mai multe FILME

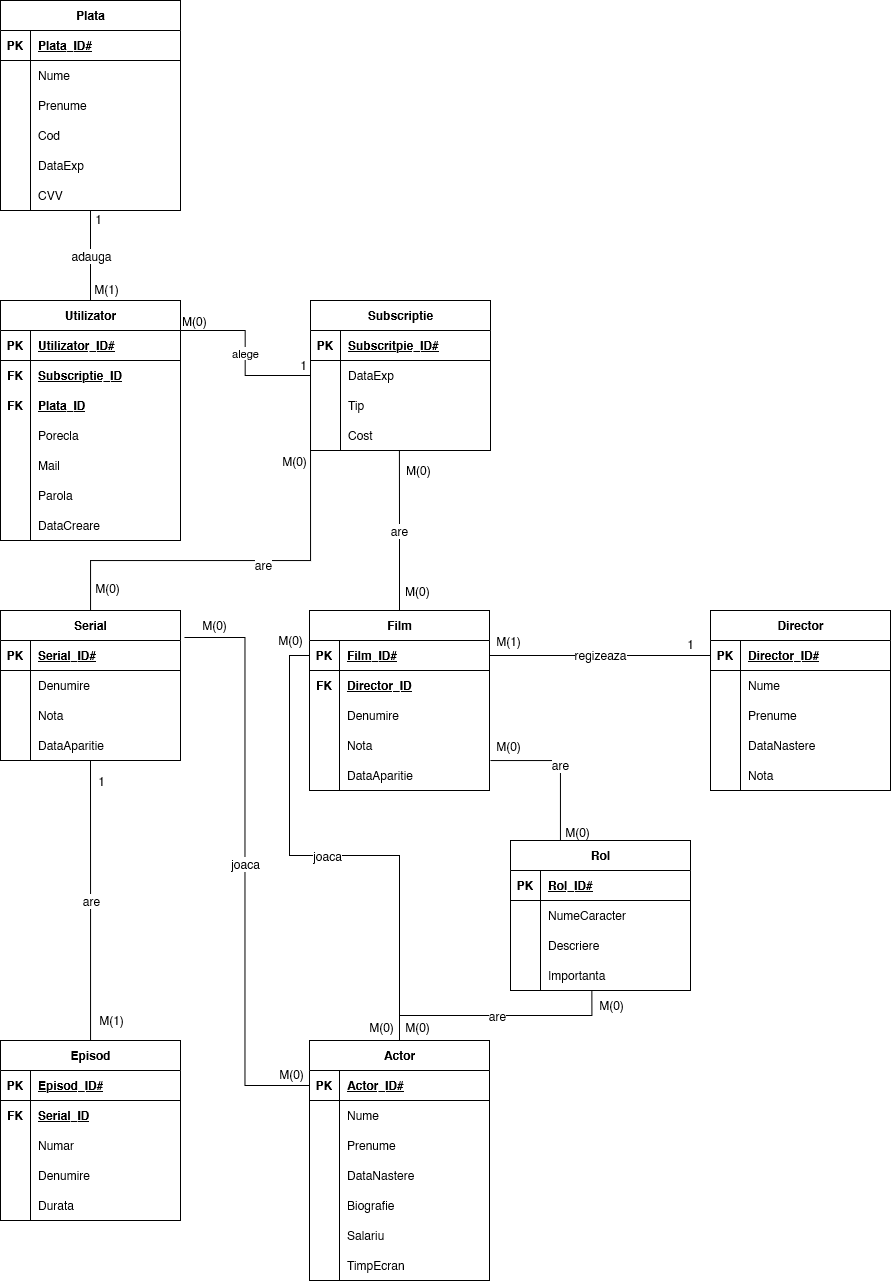
FILM trebuie să fie regizat de un singur DIRECTOR

* Cardinalitate maximă: 1:M
* Cardinalitate minimă: 1:1

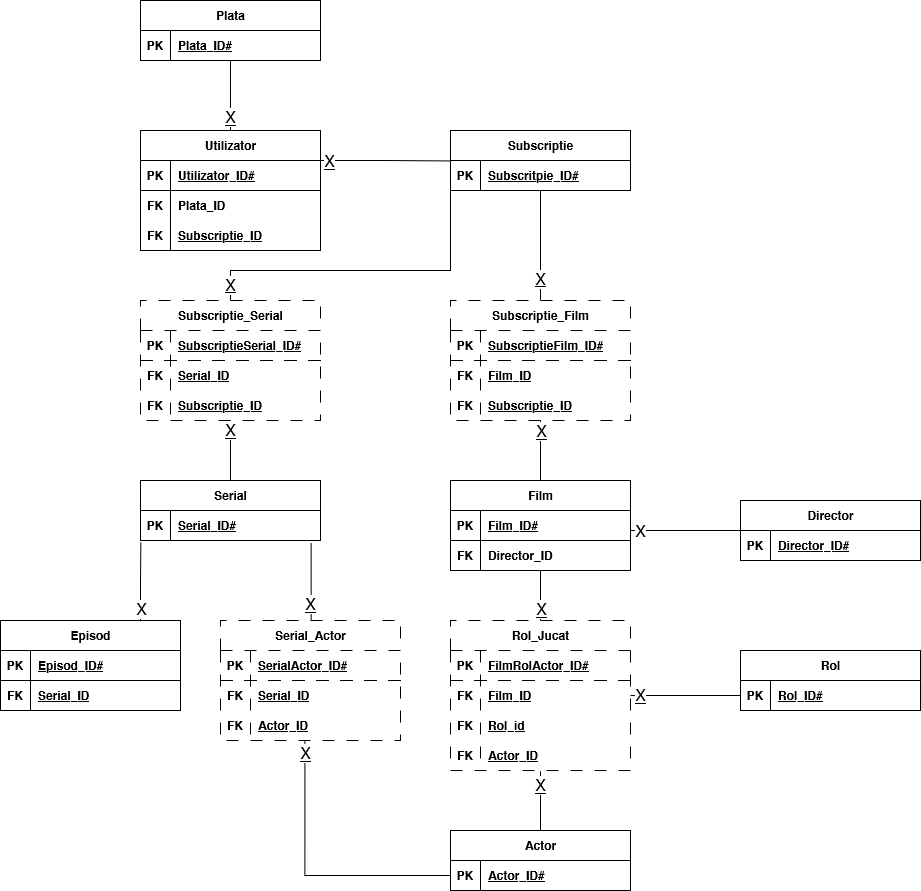
# Atribute (\* ⬄ obligatoriu)

* **UTILIZATOR**
* Utilizator\_ID\* = cod numeric unic, de tip number, de 6 cifre pentru întregistrarea contului unui utilizator
* Plată\_ID\* = cheie străină din tabela Plată
* Subscripție \_ID\* = cheie străină din tabela Subscripție
* Poreclă\* = variabilă de tip varchar2, de lungime 50
* Mail\* = variabilă de tip varchar2 de lungime 255
* Parolă\* = variabilă de tip varchar2 de lungime 255
* DatăCreare = variabilă de tip date (valoare implicită = dată curentă)
* Data\_Exp\_Sub\* = date, cand utilizatorul trebuie sa plătească iar
* **SUBSCRIPȚIE**
* Subscriptie\_ID\* = cod numeric unic, de tip number, de 6 cifre pentru pachetele cu filme/seriale
* Tip\* = varchar2(20) dintre valorile: basic, simplu, normal, premium, ultimate, basic e implicită
* Cost = variabilă de tip number, de 2 cifre
* **PLATĂ**
* Plată\_ID\* = cod numeric unic, de tip number, de 6 cifre pentru evidența plăților
* Nume\* = varchar2(50)
* Prenume\* = varchar2(50)
* Cod\* = number(16)
* DatăExp\* = date
* CVV\* = number(3), obligatoriu de 3 cifre
* **FILM**
* Film\_ID\* = cod numeric unic, de tip number, de 6 cifre pentru evidența filmelor
* Director\_ID\* = cheie străină din tabela Director
* Denumire\* = varchar2(50)
* Notă = number(4,2), număr între 1-10
* DatăApariție = date
* **SERIAL**
* Serial\_ID\* = cod numeric unic, de tip number, de 6 cifre pentru evidența serialelor
* Denumire\* = varchar2(50)
* Notă = number(4,2), număr între 1-10
* DatăApariție = date
* **EPISOD**
* Episod\_ID\* = cod numeric unic, de tip number, de 6 cifre pentru evidența episoadelor
* Serial\_ID\* = cheie străină din tabela Serial
* Număr = number(2), numărul episodului în serial
* Denumire\* = varchar2(50)
* Durată\* = number(3), unitate de măsură = minut
* **ACTOR**
* Actor\_ID\* = cod numeric unic, de tip number, de 6 cifre pentru fiecare actor
* Nume\* = varchar2(50)
* Prenume\* = varchar2(50)
* DatăNaștere = date
* Salariu = number(7), valore implicită = 3000
* **ROL**
* Rol\_ID\* = cod numeric unic, de tip number, de 6 cifre pentru evidența rolurilor
* NumeCaracter\* = varchar2(50)
* Descriere = varchar2(255)
* Importanță = number(4,2), număr între 1-10
* **DIRECTOR**
* Director\_ID\* = cod numeric unic, de tip number, de 6 cifre pentru fiecare director
* Nume\* = varchar2(50)
* Prenume = varchar2(50)
* DatăNaștere = date
* Notă = number(4,2), număr între 1-10
* **Subscripție\_Serial**
* SubscripțieSerial\_ID = cod numeric unic, de tip number, de 6 cifre
* Serial\_ID = cheie străină din tabela Serial
* Subscripție\_ID = cheie străină din tabela Subsripție
* **Subscripție\_Film**
* SubscripțieFilm\_ID = cod numeric unic, de tip number, de 6 cifre
* Film\_ID = cheie străină din tabela Film
* Subscripție\_ID = cheie străină din tabela Subsripție
* **Serial\_Actor**
* SerialActor\_ID = cod numeric unic, de tip number, de 6 cifre
* Serial\_ID = cheie străină din tabela Serial
* Actor\_ID = cheie străină din tabela Actor
* **Rol\_Jucat**
* FilmRolActor\_ID = cod numeric unic, de tip number, de 6 cifre
* Film\_ID = cheie străină din tabela FILM
* Rol\_ID = cheie străină din tabela ROL
* Actor\_ID = cheie străină din tabela ACTOR
* Timp\_Ecran = number(3), unitate de măsură = minut

# Diagrama entitate-relație

****

# Diagrama conceptuală



# Schemele relaționale

PLATĂ(#Plată\_id, Nume, Prenume, Cod, DataExp, CVV)

UTILIZATOR(#Utilizator\_id, #Subscripție\_id, #Plată\_id, Poreclă, Mail,

Parolă, DatăCreare, Dată\_Exp\_Sub)

SUBSCRIPȚIE(#Subscipție\_id, Tip, Cost)

SubscripțieSerial(#SubscripțieSerial\_id, #Serial\_id, #Subscripție\_id)

SubscripțieFilm(#SubscripțieFilm\_id, #Film \_id, #Subscripție\_id)

SERIAL(#Serial\_id, Denumire, Notă, DatăApariție)

EPISOD(#Episod\_id, #Serial\_id, Număr, Denumire, Durată)

SerialActor(#SerialActor\_id, #Serial\_id, #Actor\_id)

FILM(#Film\_id, #Director\_id, Denumire, Notă, DatăApariție)

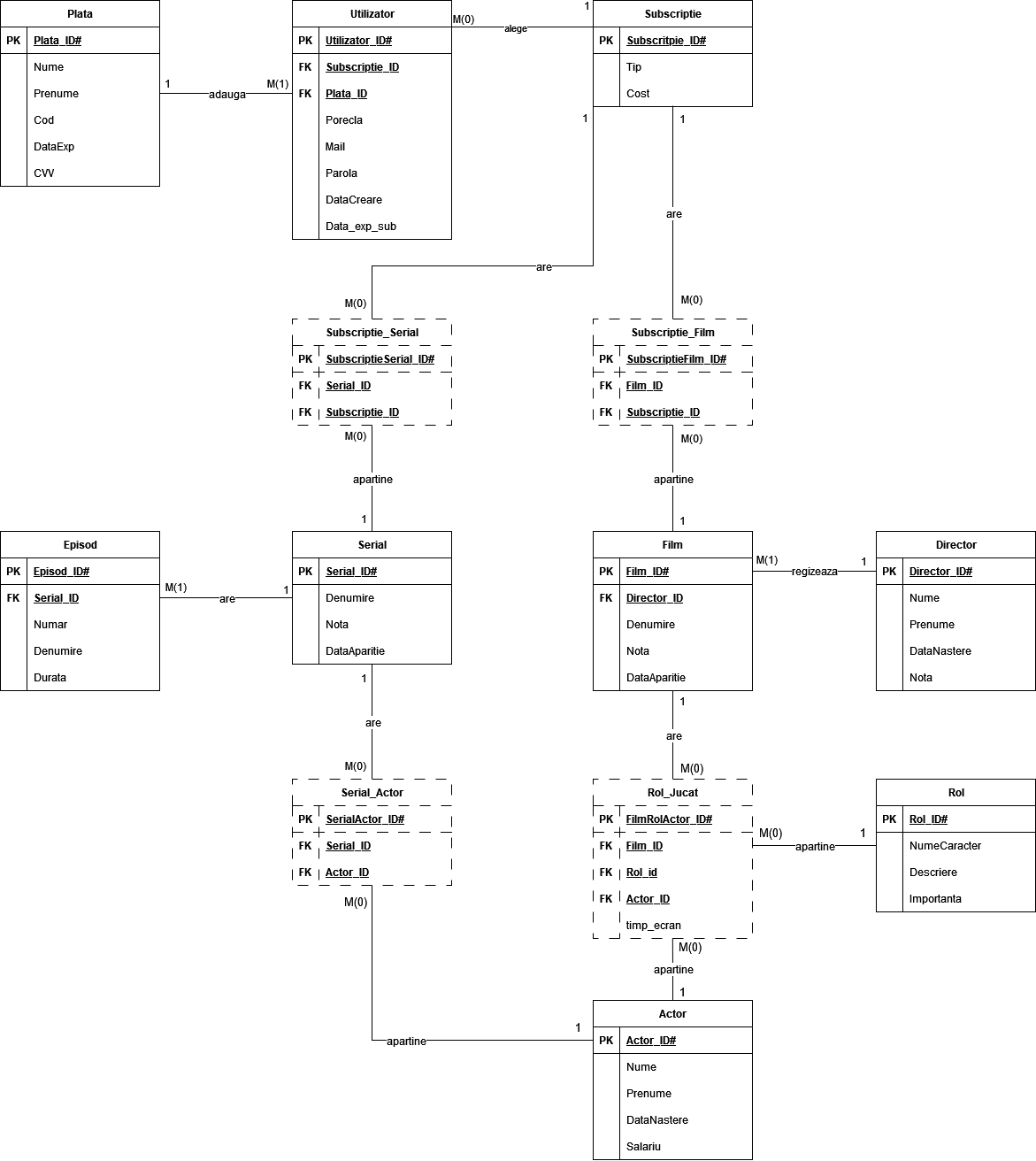
DIRECTOR(#Director\_id, Nume, Prenume, DatăNaștere, Notă)

ROL(#Rol\_id, NumeCaracter, Descriere, Importanță)

ACTOR(#Actor\_id, Nume, Prenume, DatăNaștere, Salariu)

Rol\_Jucat(#FilmRolActor\_id, #Film\_id, #Rol\_id, #Actor\_id, Timp\_Ecran)

# Normalizări

****

## FN1

Original, entitatea UTILIZATOR conținea toate informațiile necesare unei plăți. Din acest motiv, am creat o nouă entitate PLATĂ, care să rețină toate aceste informații, iar entitatea UTILIZATOR va conține doar id-ul către entitatea PLATĂ.

## FN2

Pentru a obține FN2 am rezolvat toate relațiile many-to-many cu ajutorul tabelelor asociative astfel: Între tabela SUBSCRIPȚIE si tabela SERIAL am mai adaugat Subscripție\_Serial care are cheile străine celor două și transformă relația many-to-many în one-to-many. Analog am făcut și pentru tabelele FILM, SUBSCRIPȚIE, SERIAL si ACTOR. Între FILM, ROL și ACTOR, se găsește o relație de tip superior, rezolvată prin tabela asociativă Rol\_Jucat.

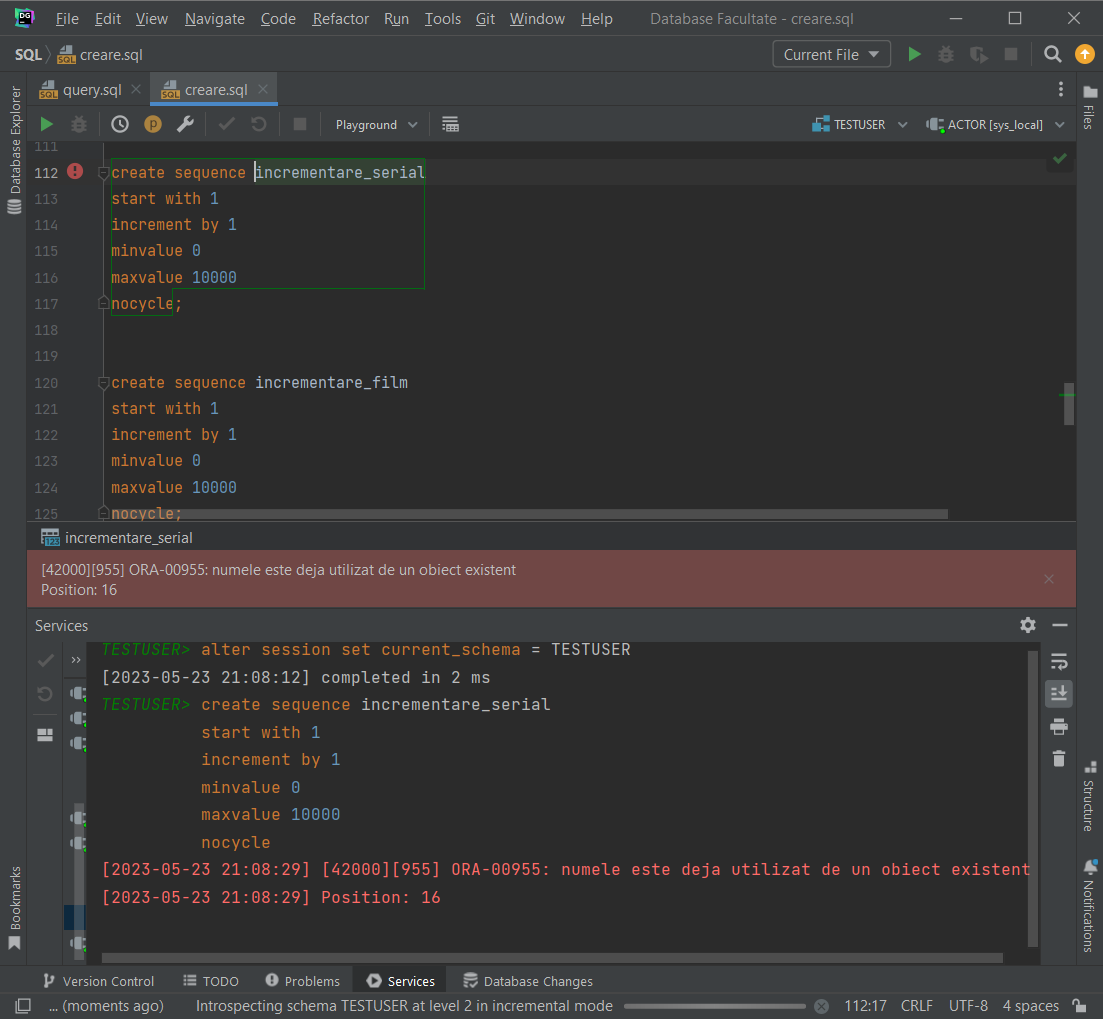
## FN3

Am eliminat dependețele tranzitive, astfel separând entitățile și atribulete în tabele separate acolo unde a fost nevoie. Acest fapt se poate observa prin mutarea atributului Timp\_Ecran din tabela actor in tabela Rol\_Jucat, deoarece acest atribut depindea si de film si de rol, nu numai de actor.

# Secvență

create sequence incrementare\_serial  
start with 1  
increment by 1  
minvalue 0  
maxvalue 10000  
nocycle;  
  
  
create sequence incrementare\_film  
start with 1  
increment by 1  
minvalue 0  
maxvalue 10000  
nocycle;  
  
  
create sequence incrementare\_actor  
start with 1  
increment by 1  
minvalue 0  
maxvalue 10000  
nocycle;

create sequence incrementare\_rol\_jucat  
start with 1  
increment by 1  
minvalue 0  
maxvalue 10000  
nocycle;

****

# Creare și inserare

## creare de tabele:

create table PLATA (  
 plata\_id number(6) constraint pk\_plata primary key,  
 nume varchar2(50) constraint nume\_null not null ,  
 prenume varchar2(50) constraint prenume\_null not null ,  
 cod number(16) constraint cod\_null not null ,  
 data\_exp date constraint date\_null not null ,  
 cvv number(3) constraint cvv\_null not null  
);

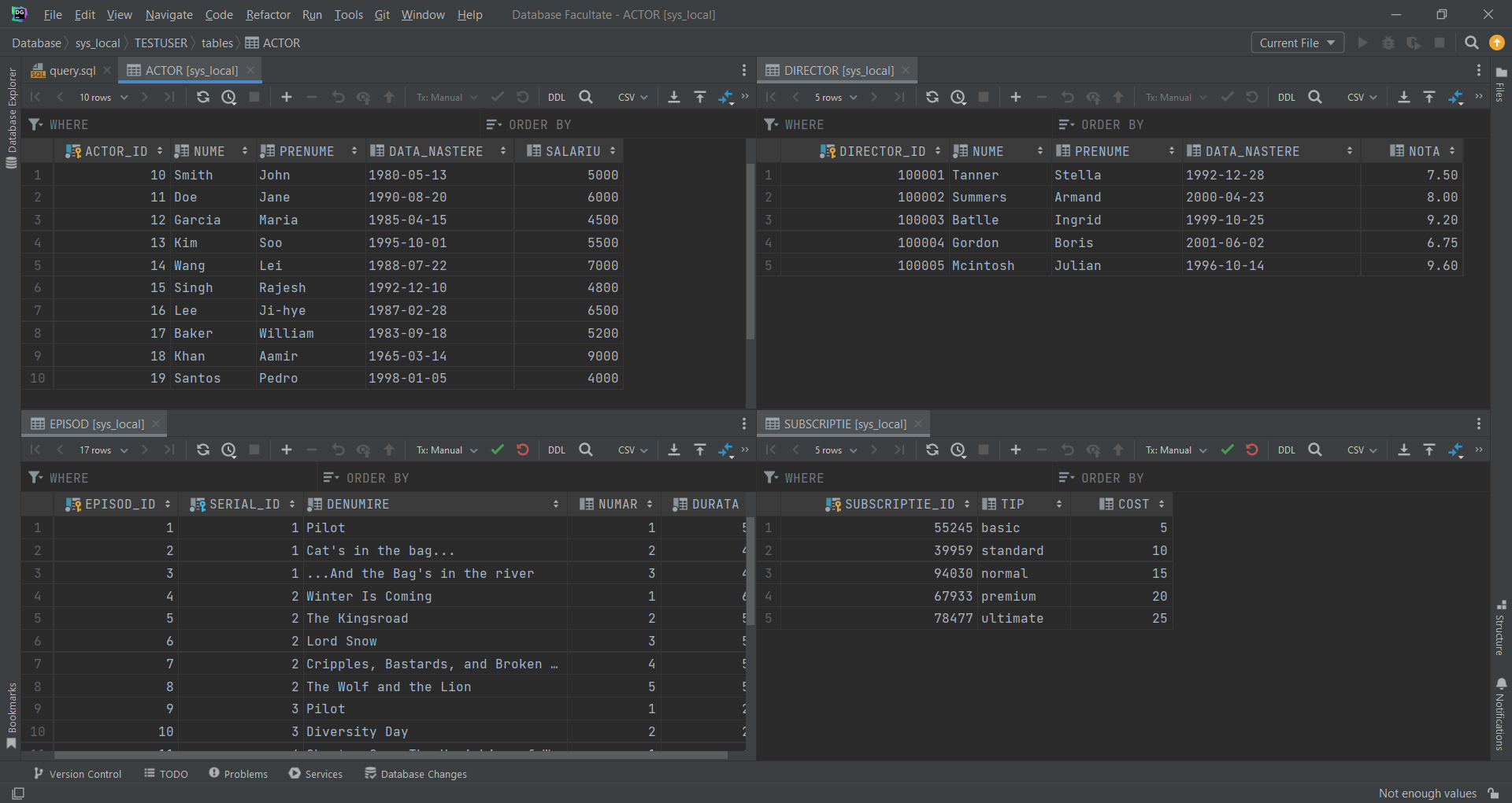
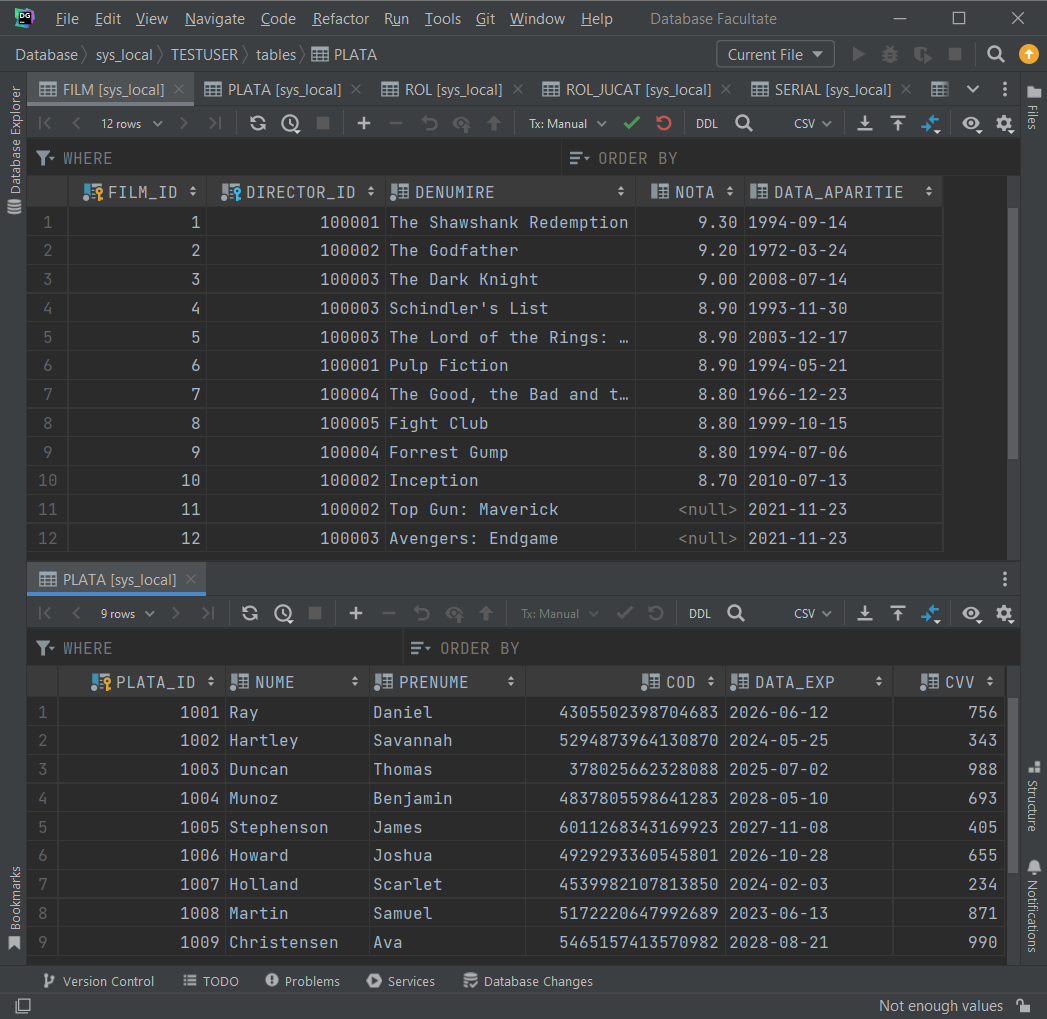
create table SUBSCRIPTIE (  
 subscriptie\_id number(6) constraint pk\_subscritpie primary key ,  
 tip varchar2(20) default 'basic' check ( *lower*(tip) in ('basic','standard', 'normal', 'premium','ultimate') ) ,  
 cost number(2)  
);  
  
create table UTILIZATOR (  
 utilizator\_id number(6) constraint pk\_utilizator primary key ,  
 subscriptie\_id number(6) not null ,  
 plata\_id number(6) not null ,  
 porecla varchar2(50) constraint porecla\_null not null ,  
 mail varchar2(255) constraint mail\_null not null ,  
 parola varchar2(255) constraint parola\_null not null,  
 data\_creare date default *current\_date* ,  
 data\_exp\_sub date not null ,  
 constraint fk\_subscriptie foreign key (subscriptie\_id) references SUBSCRIPTIE(subscriptie\_id),  
 constraint fk\_plata foreign key (plata\_id) references PLATA(plata\_id)  
);  
  
create table SERIAL (  
 serial\_id number(6) constraint pk\_serial primary key ,  
 denumire varchar2(50) constraint denumire\_null not null ,  
 nota number(4,2) constraint nota\_check check ( nota >= 1 and nota <= 10 ),  
 data\_aparitie date  
);  
  
create table EPISOD (  
 episod\_id number(6) constraint pk\_episod primary key ,  
 serial\_id number(6) not null ,  
 denumire varchar2(50)constraint denumire\_episod\_null not null ,  
 numar number(2),  
 durata number(3) constraint durata\_null not null ,  
 constraint fk\_serial foreign key (serial\_id) references SERIAL(serial\_id)  
);

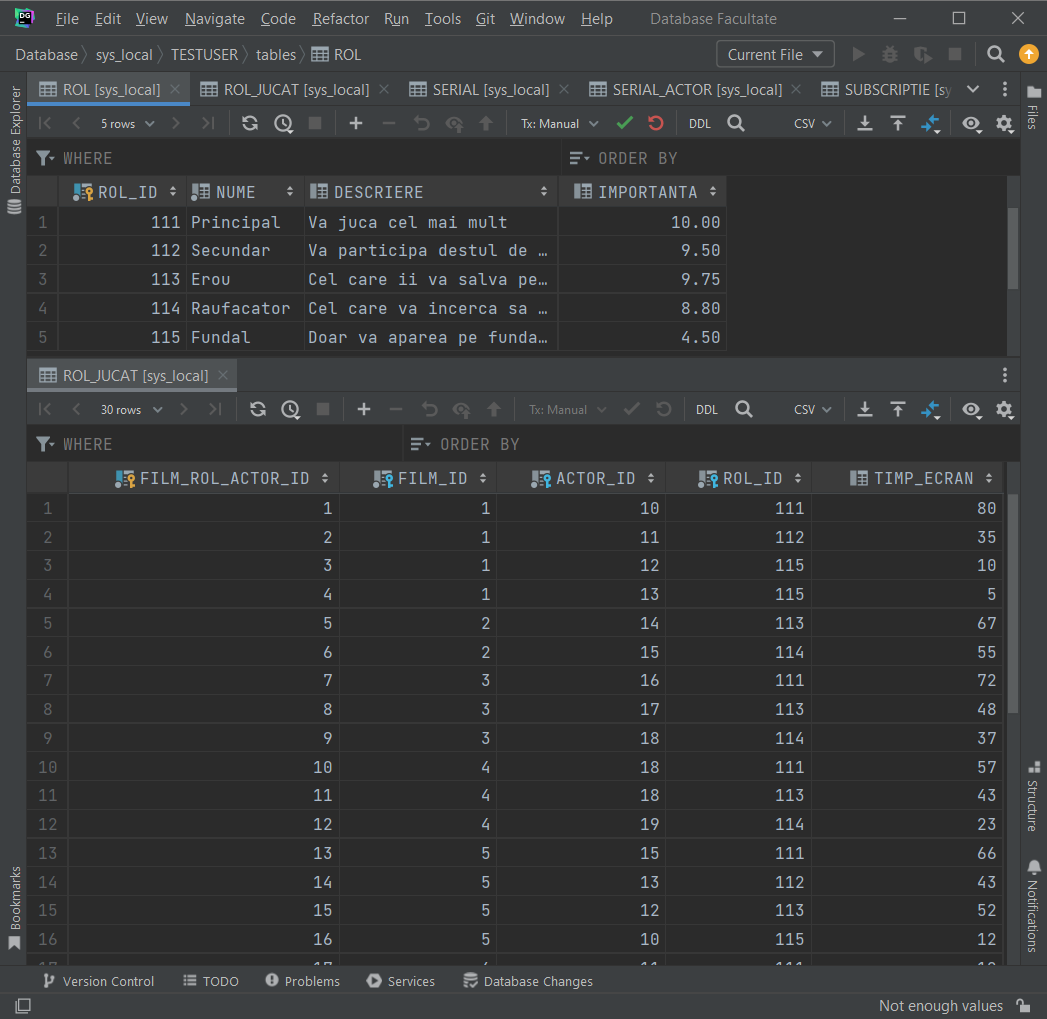
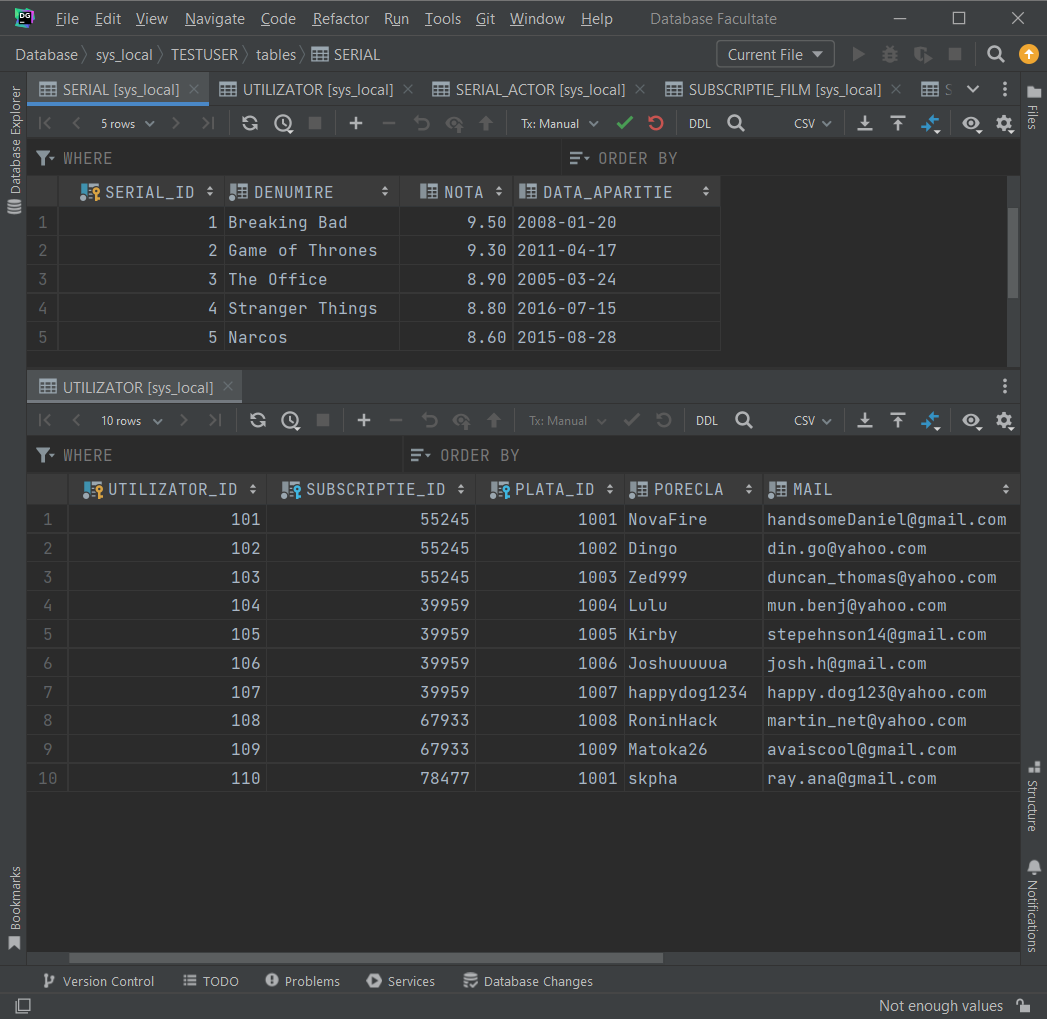
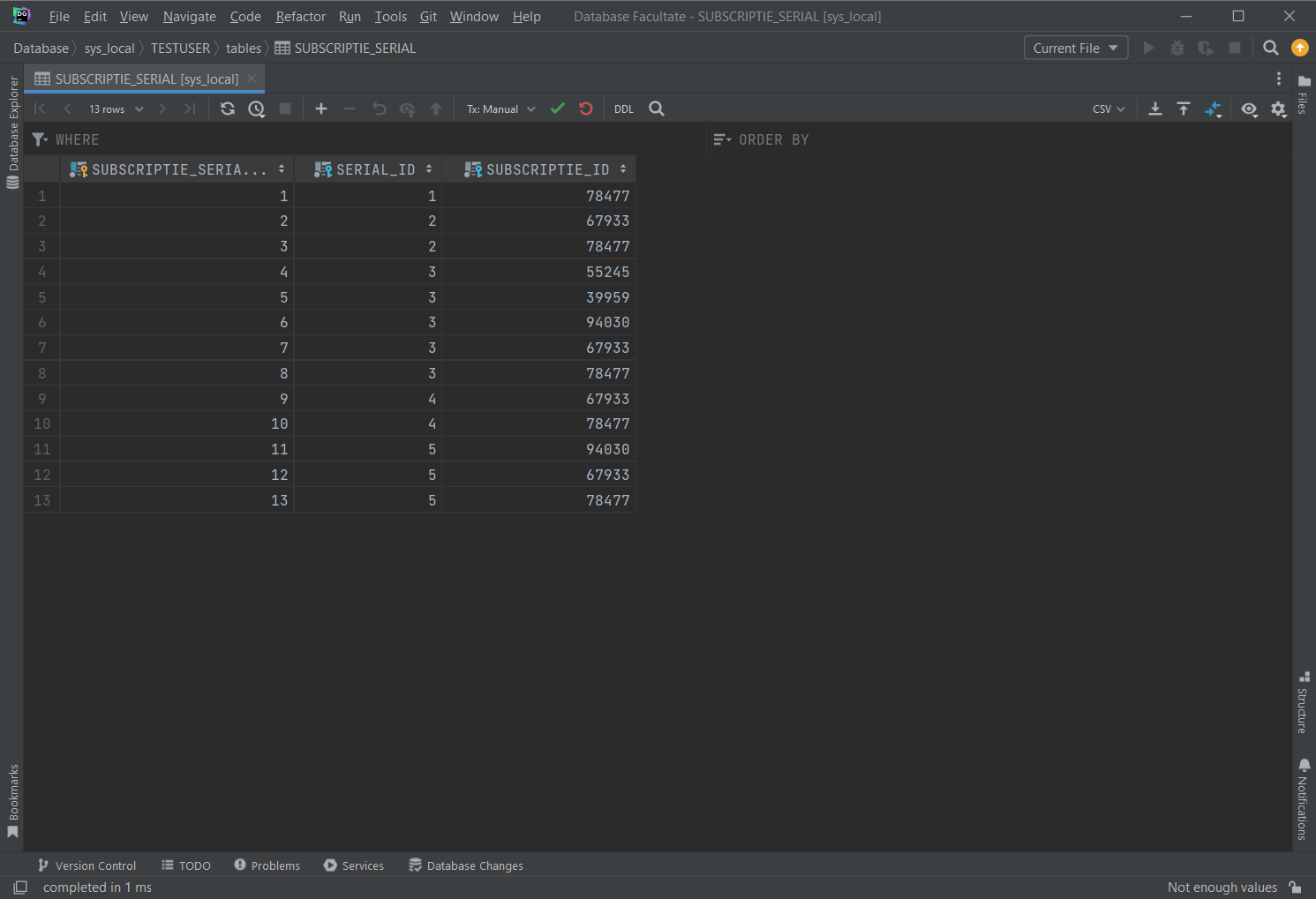
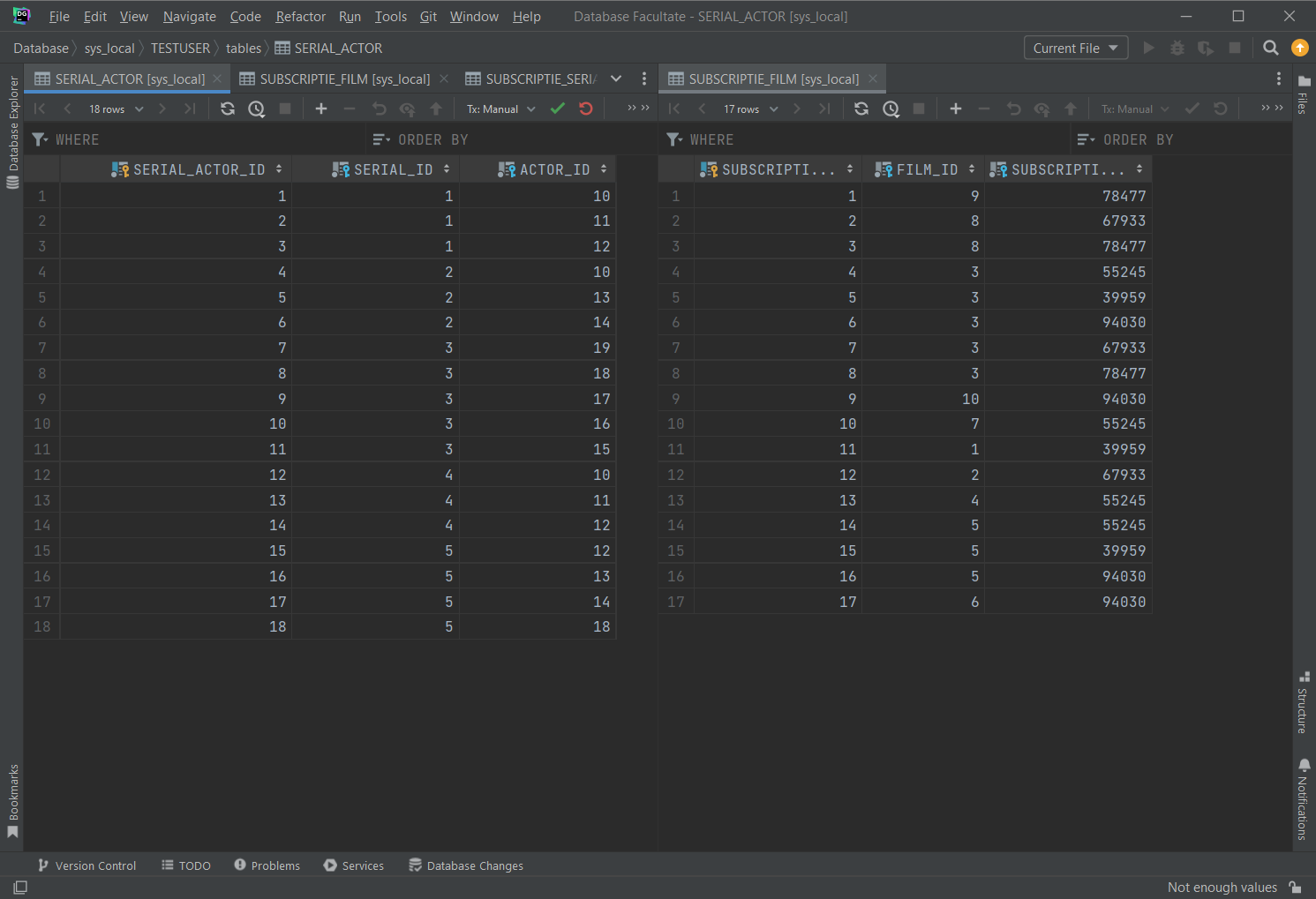
create table SUBSCRIPTIE\_SERIAL (  
 subscriptie\_serial\_id number(6) constraint pk\_sub\_ser primary key ,  
 serial\_id number(6) not null ,  
 subscriptie\_id number(6) not null ,  
 constraint fk\_subscriptie\_asoc foreign key (subscriptie\_id) references SUBSCRIPTIE(subscriptie\_id) ,  
 constraint fk\_serial\_asoc foreign key (serial\_id) references SERIAL(serial\_id)  
);  
  
create table DIRECTOR (  
 director\_id number(6) constraint pk\_director primary key ,  
 nume varchar2(50) constraint nume\_dir\_null not null ,  
 prenume varchar2(50) ,  
 data\_nastere date ,  
 nota number(4,2) constraint nota\_director\_check check ( nota >= 1 and nota <= 10 )  
);  
  
create table FILM (  
 film\_id number(6) constraint pk\_film primary key ,  
 director\_id number(6) not null ,  
 denumire varchar2(50) constraint denumire\_film\_null not null ,  
 nota number(4,2) constraint nota\_film\_check check ( nota >= 1 and nota <= 10 ) ,  
 data\_aparitie date ,  
 constraint fk\_director foreign key (director\_id) references DIRECTOR(director\_id)  
);  
  
create table SUBSCRIPTIE\_FILM (  
 subscriptie\_film\_id number(6) constraint pk\_sub\_film primary key ,  
 film\_id number(6) not null ,  
 subscriptie\_id number(6) not null ,  
 constraint fk\_subscriptie\_asocf foreign key (subscriptie\_id) references SUBSCRIPTIE(subscriptie\_id) ,  
 constraint fk\_serial\_asocf foreign key (film\_id) references FILM(film\_id)  
);

create table ROL (  
 rol\_id number(6) constraint pk\_rol primary key ,  
 nume varchar2(50) constraint caracter\_null not null ,  
 descriere varchar2(255) ,  
 importanta number(4,2) constraint imp\_check check ( importanta >= 1 and importanta <= 10 )  
);  
  
create table ACTOR (  
 actor\_id number(6) constraint pk\_actor primary key ,  
 nume varchar2(50) constraint nume\_act\_null not null ,  
 prenume varchar2(50) constraint prenume\_act\_null not null ,  
 data\_nastere date ,  
 salariu number(7) default 3000  
);  
  
create table SERIAL\_ACTOR (  
 serial\_actor\_id number(6) constraint pk\_ser\_act primary key ,  
 serial\_id number(6) not null ,  
 actor\_id number(6) not null ,  
 constraint fk\_actor\_asocA foreign key (actor\_id) references ACTOR(actor\_id) ,  
 constraint fk\_serial\_asocA foreign key (serial\_id) references SERIAL(serial\_id)  
);  
  
create table ROL\_JUCAT (  
 film\_rol\_actor\_id number(6) constraint pk\_rol\_jucat primary key ,  
 film\_id number(6) not null ,  
 actor\_id number(6) not null ,  
 rol\_id number(6) not null ,  
 timp\_ecran number(3) ,  
 constraint fk\_film\_asocRJ foreign key (film\_id) references FILM(film\_id) ,  
 constraint fk\_actor\_asocRJ foreign key (actor\_id) references ACTOR(actor\_id) ,  
 constraint fk\_rol foreign key (rol\_id) references ROL(rol\_id)  
);

## inserari:

insert all  
 into PLATA(plata\_id, nume, prenume, cod, data\_exp, cvv) values (1001,'Ray','Daniel',4305502398704683,'12/06/2026',756)  
 into PLATA(plata\_id, nume, prenume, cod, data\_exp, cvv) values (1002,'Hartley','Savannah',5294873964130870,'25/05/2024',343)  
 into PLATA(plata\_id, nume, prenume, cod, data\_exp, cvv) values (1003,'Duncan','Thomas',378025662328088,'02/07/2025',988)  
 into PLATA(plata\_id, nume, prenume, cod, data\_exp, cvv) values (1004,'Munoz','Benjamin',4837805598641283,'10/05/2028',693)  
 into PLATA(plata\_id, nume, prenume, cod, data\_exp, cvv) values (1005,'Stephenson','James',6011268343169923,'08/11/2027',405)  
 into PLATA(plata\_id, nume, prenume, cod, data\_exp, cvv) values (1006,'Howard','Joshua',4929293360545801,'28/10/2026',655)  
 into PLATA(plata\_id, nume, prenume, cod, data\_exp, cvv) values (1007,'Holland','Scarlet',4539982107813850,'03/02/2024',234)  
 into PLATA(plata\_id, nume, prenume, cod, data\_exp, cvv) values (1008,'Martin','Samuel',5172220647992689,'13/06/2023',871)  
 into PLATA(plata\_id, nume, prenume, cod, data\_exp, cvv) values (1009,'Christensen','Ava',5465157413570982,'21/08/2028',990)  
select 'worked' from dual;  
  
insert all  
 into SUBSCRIPTIE(subscriptie\_id, tip, cost) values (55245,'basic',5)  
 into SUBSCRIPTIE(subscriptie\_id, tip, cost) values (39959,'standard',10)  
 into SUBSCRIPTIE(subscriptie\_id, tip, cost) values (94030,'normal',15)  
 into SUBSCRIPTIE(subscriptie\_id, tip, cost) values (67933,'premium',20)  
 into SUBSCRIPTIE(subscriptie\_id, tip, cost) values (78477,'ultimate',25)  
select 'worked' from dual;  
  
insert all  
 into UTILIZATOR(utilizator\_id, subscriptie\_id, plata\_id, porecla, mail, parola, data\_exp\_sub) values (101,55245,1001,'NovaFire','handsomeDaniel@gmail.com','RD\_2003','14/06/2023')  
 into UTILIZATOR(utilizator\_id, subscriptie\_id, plata\_id, porecla, mail, parola, data\_exp\_sub) values (102,55245,1002,'Dingo','din.go@yahoo.com','39hsrda9','05/05/2023')  
 into UTILIZATOR(utilizator\_id, subscriptie\_id, plata\_id, porecla, mail, parola, data\_exp\_sub) values (103,55245,1003,'Zed999','duncan\_thomas@yahoo.com','Zed\_MY\_life','27/05/2023')  
 into UTILIZATOR(utilizator\_id, subscriptie\_id, plata\_id, porecla, mail, parola, data\_exp\_sub) values (104,39959,1004,'Lulu','mun.benj@yahoo.com','Aida\_2022','22/05/2023')  
 into UTILIZATOR(utilizator\_id, subscriptie\_id, plata\_id, porecla, mail, parola, data\_exp\_sub) values (105,39959,1005,'Kirby','stepehnson14@gmail.com','kirby1306','14/05/2023')  
 into UTILIZATOR(utilizator\_id, subscriptie\_id, plata\_id, porecla, mail, parola, data\_exp\_sub) values (106,39959,1006,'Joshuuuuua','josh.h@gmail.com','netflixPassword','18/05/2023')  
 into UTILIZATOR(utilizator\_id, subscriptie\_id, plata\_id, porecla, mail, parola, data\_exp\_sub) values (107,39959,1007,'happydog1234','happy.dog123@yahoo.com','dogggsAREsweet','10/06/2023')  
 into UTILIZATOR(utilizator\_id, subscriptie\_id, plata\_id, porecla, mail, parola, data\_exp\_sub) values (108,67933,1008,'RoninHack','martin\_net@yahoo.com','genshin001','01/06/2023')  
 into UTILIZATOR(utilizator\_id, subscriptie\_id, plata\_id, porecla, mail, parola, data\_exp\_sub) values (109,67933,1009,'Matoka26','avaiscool@gmail.com','Mihai1234568','08/06/2023')  
 into UTILIZATOR(utilizator\_id, subscriptie\_id, plata\_id, porecla, mail, parola, data\_exp\_sub) values (110,78477,1001,'skpha','ray.ana@gmail.com','VulturViteaz','15/06/2023')  
select 'worked' from dual;  
  
insert all  
 into DIRECTOR(director\_id, nume, prenume, data\_nastere, nota) VALUES (100001,'Tanner','Stella','28/12/1992',7.5)  
 into DIRECTOR(director\_id, nume, prenume, data\_nastere, nota) VALUES (100002,'Summers','Armand','23/04/2000',8)  
 into DIRECTOR(director\_id, nume, prenume, data\_nastere, nota) VALUES (100003,'Batlle','Ingrid','25/10/1999',9.2)  
 into DIRECTOR(director\_id, nume, prenume, data\_nastere, nota) VALUES (100004,'Gordon','Boris','02/06/2001',6.75)  
 into DIRECTOR(director\_id, nume, prenume, data\_nastere, nota) VALUES (100005,'Mcintosh','Julian','14/10/1996',9.6)  
select 'worked' from dual;  
  
insert all  
 into ROL(rol\_id, nume, descriere, importanta) VALUES (111,'Principal','Va juca cel mai mult',10)  
 into ROL(rol\_id, nume, descriere, importanta) VALUES (112,'Secundar','Va participa destul de des',9.5)  
 into ROL(rol\_id, nume, descriere, importanta) VALUES (113,'Erou','Cel care ii va salva pe toti',9.75)  
 into ROL(rol\_id, nume, descriere, importanta) VALUES (114,'Raufacator','Cel care va incerca sa ii distruga pe toti',8.8)  
 into ROL(rol\_id, nume, descriere, importanta) VALUES (115,'Fundal','Doar va aparea pe fundalul filmului pentru a popula scena cu oameni',4.5)  
select 'worked' from dual;  
  
insert all  
 into ACTOR(actor\_id, nume, prenume, data\_nastere, salariu) VALUES (10, 'Smith', 'John', '13/05/1980', 5000)  
 into ACTOR(actor\_id, nume, prenume, data\_nastere, salariu) VALUES (11, 'Doe', 'Jane', '20/08/1990', 6000)  
 into ACTOR(actor\_id, nume, prenume, data\_nastere, salariu) VALUES (12, 'Garcia', 'Maria', '15/04/1985', 4500)  
 into ACTOR(actor\_id, nume, prenume, data\_nastere, salariu) VALUES (13, 'Kim', 'Soo', '01/10/1995', 5500)  
 into ACTOR(actor\_id, nume, prenume, data\_nastere, salariu) VALUES (14, 'Wang', 'Lei', '22/07/1988', 7000)  
 into ACTOR(actor\_id, nume, prenume, data\_nastere, salariu) VALUES (15, 'Singh', 'Rajesh', '10/12/1992', 4800)  
 into ACTOR(actor\_id, nume, prenume, data\_nastere, salariu) VALUES (16, 'Lee', 'Ji-hye', '28/02/1987', 6500)  
 into ACTOR(actor\_id, nume, prenume, data\_nastere, salariu) VALUES (17, 'Baker', 'William', '18/09/1983', 5200)  
 into ACTOR(actor\_id, nume, prenume, data\_nastere, salariu) VALUES (18, 'Khan', 'Aamir', '14/03/1965', 9000)  
 into ACTOR(actor\_id, nume, prenume, data\_nastere, salariu) VALUES (19, 'Santos', 'Pedro', '05/01/1998', 4000)  
select 'worked' from dual;  
  
insert all  
 into FILM(film\_id, director\_id, denumire, nota, data\_aparitie) VALUES (1, 100001, 'The Shawshank Redemption', 9.3, '14/09/1994')  
 into FILM(film\_id, director\_id, denumire, nota, data\_aparitie) VALUES (2, 100002, 'The Godfather', 9.2, '24/03/1972')  
 into FILM(film\_id, director\_id, denumire, nota, data\_aparitie) VALUES (3, 100003, 'The Dark Knight', 9.0, '14/07/2008')  
 into FILM(film\_id, director\_id, denumire, nota, data\_aparitie) VALUES (4, 100003, 'Schindler''s List', 8.9, '30/11/1993') -- 2 '' pentru a avea doar unul  
 into FILM(film\_id, director\_id, denumire, nota, data\_aparitie) VALUES (5, 100003, 'The Lord of the Rings: The Return of the King', 8.9, '17/12/2003')  
 into FILM(film\_id, director\_id, denumire, nota, data\_aparitie) VALUES (6, 100001, 'Pulp Fiction', 8.9, '21/05/1994')  
 into FILM(film\_id, director\_id, denumire, nota, data\_aparitie) VALUES (7, 100004, 'The Good, the Bad and the Ugly', 8.8, '23/12/1966')  
 into FILM(film\_id, director\_id, denumire, nota, data\_aparitie) VALUES (8, 100005, 'Fight Club', 8.8, '15/10/1999')  
 into FILM(film\_id, director\_id, denumire, nota, data\_aparitie) VALUES (9, 100004, 'Forrest Gump', 8.8, '06/07/1994')  
 into FILM(film\_id, director\_id, denumire, nota, data\_aparitie) VALUES (10, 100002, 'Inception', 8.7, '13/07/2010')  
select 'worked' from dual;  
  
insert all  
 into SERIAL(serial\_id, denumire, nota, data\_aparitie) VALUES (1, 'Breaking Bad', 9.5, '20/01/2008')  
 into SERIAL(serial\_id, denumire, nota, data\_aparitie) VALUES (2, 'Game of Thrones', 9.3, '17/04/2011')  
 into SERIAL(serial\_id, denumire, nota, data\_aparitie) VALUES (3, 'The Office', 8.9, '24/03/2005')  
 into SERIAL(serial\_id, denumire, nota, data\_aparitie) VALUES (4, 'Stranger Things', 8.8, '15/07/2016')  
 into SERIAL(serial\_id, denumire, nota, data\_aparitie) VALUES (5, 'Narcos', 8.6, '28/08/2015')  
select 'worked' from dual;  
  
insert all  
 into EPISOD(episod\_id, serial\_id, denumire, numar, durata) VALUES (1,1,'Pilot',1,58)  
 into EPISOD(episod\_id, serial\_id, denumire, numar, durata) VALUES (2,1,'Cat''s in the bag...',2,48)  
 into EPISOD(episod\_id, serial\_id, denumire, numar, durata) VALUES (3,1,'...And the Bag''s in the river',3,47)  
 into EPISOD(episod\_id, serial\_id, denumire, numar, durata) VALUES (4,2,'Winter Is Coming',1,60)  
 into EPISOD(episod\_id, serial\_id, denumire, numar, durata) VALUES (5,2,'The Kingsroad',2,56)  
 into EPISOD(episod\_id, serial\_id, denumire, numar, durata) VALUES (6,2,'Lord Snow',3,58)  
 into EPISOD(episod\_id, serial\_id, denumire, numar, durata) VALUES (7,2,'Cripples, Bastards, and Broken Things',4,54)  
 into EPISOD(episod\_id, serial\_id, denumire, numar, durata) VALUES (8,2,'The Wolf and the Lion',5,55)  
 into EPISOD(episod\_id, serial\_id, denumire, numar, durata) VALUES (9,3,'Pilot',1,22)  
 into EPISOD(episod\_id, serial\_id, denumire, numar, durata) VALUES (10,3,'Diversity Day',2,23)  
 into EPISOD(episod\_id, serial\_id, denumire, numar, durata) VALUES (11,4,'Chapter One: The Vanishing of Will Byers',1,48)  
 into EPISOD(episod\_id, serial\_id, denumire, numar, durata) VALUES (12,4,'Chapter Two: The Weirdo on Maple Street',2,55)  
 into EPISOD(episod\_id, serial\_id, denumire, numar, durata) VALUES (13,4,'Chapter Three: Holly, Jolly',3,51)  
 into EPISOD(episod\_id, serial\_id, denumire, numar, durata) VALUES (14,4,'Chapter Four: The Body',4,58)  
 into EPISOD(episod\_id, serial\_id, denumire, numar, durata) VALUES (15,5,'Descenso',1,57)  
 into EPISOD(episod\_id, serial\_id, denumire, numar, durata) VALUES (16,5,'The Sword of Simón Bolivar',2,50)  
 into EPISOD(episod\_id, serial\_id, denumire, numar, durata) VALUES (17,5,'The Men of Always',3,46)  
select 'worked' from dual;  
  
-- secventa nu merge cu insert all, da aceeasi valoare  
insert into SUBSCRIPTIE\_SERIAL(subscriptie\_serial\_id, serial\_id, subscriptie\_id) VALUES (INCREMENTARE\_SERIAL.nextval,1,78477);  
insert into SUBSCRIPTIE\_SERIAL(subscriptie\_serial\_id, serial\_id, subscriptie\_id) VALUES (INCREMENTARE\_SERIAL.nextval,2,67933);  
insert into SUBSCRIPTIE\_SERIAL(subscriptie\_serial\_id, serial\_id, subscriptie\_id) VALUES (INCREMENTARE\_SERIAL.nextval,2,78477);  
insert into SUBSCRIPTIE\_SERIAL(subscriptie\_serial\_id, serial\_id, subscriptie\_id) VALUES (INCREMENTARE\_SERIAL.nextval,3,55245);  
insert into SUBSCRIPTIE\_SERIAL(subscriptie\_serial\_id, serial\_id, subscriptie\_id) VALUES (INCREMENTARE\_SERIAL.nextval,3,39959);  
insert into SUBSCRIPTIE\_SERIAL(subscriptie\_serial\_id, serial\_id, subscriptie\_id) VALUES (INCREMENTARE\_SERIAL.nextval,3,94030);  
insert into SUBSCRIPTIE\_SERIAL(subscriptie\_serial\_id, serial\_id, subscriptie\_id) VALUES (INCREMENTARE\_SERIAL.nextval,3,67933);  
insert into SUBSCRIPTIE\_SERIAL(subscriptie\_serial\_id, serial\_id, subscriptie\_id) VALUES (INCREMENTARE\_SERIAL.nextval,3,78477);  
insert into SUBSCRIPTIE\_SERIAL(subscriptie\_serial\_id, serial\_id, subscriptie\_id) VALUES (INCREMENTARE\_SERIAL.nextval,4,67933);  
insert into SUBSCRIPTIE\_SERIAL(subscriptie\_serial\_id, serial\_id, subscriptie\_id) VALUES (INCREMENTARE\_SERIAL.nextval,4,78477);  
insert into SUBSCRIPTIE\_SERIAL(subscriptie\_serial\_id, serial\_id, subscriptie\_id) VALUES (INCREMENTARE\_SERIAL.nextval,5,94030);  
insert into SUBSCRIPTIE\_SERIAL(subscriptie\_serial\_id, serial\_id, subscriptie\_id) VALUES (INCREMENTARE\_SERIAL.nextval,5,67933);  
insert into SUBSCRIPTIE\_SERIAL(subscriptie\_serial\_id, serial\_id, subscriptie\_id) VALUES (INCREMENTARE\_SERIAL.nextval,5,78477);  
  
insert into SUBSCRIPTIE\_film(subscriptie\_film\_id, film\_id, subscriptie\_id) VALUES (INCREMENTARE\_film.nextval,9,78477);  
insert into SUBSCRIPTIE\_film(subscriptie\_film\_id, film\_id, subscriptie\_id) VALUES (INCREMENTARE\_film.nextval,8,67933);  
insert into SUBSCRIPTIE\_film(subscriptie\_film\_id, film\_id, subscriptie\_id) VALUES (INCREMENTARE\_film.nextval,8,78477);  
insert into SUBSCRIPTIE\_film(subscriptie\_film\_id, film\_id, subscriptie\_id) VALUES (INCREMENTARE\_film.nextval,3,55245);  
insert into SUBSCRIPTIE\_film(subscriptie\_film\_id, film\_id, subscriptie\_id) VALUES (INCREMENTARE\_film.nextval,3,39959);  
insert into SUBSCRIPTIE\_film(subscriptie\_film\_id, film\_id, subscriptie\_id) VALUES (INCREMENTARE\_film.nextval,3,94030);  
insert into SUBSCRIPTIE\_film(subscriptie\_film\_id, film\_id, subscriptie\_id) VALUES (INCREMENTARE\_film.nextval,3,67933);  
insert into SUBSCRIPTIE\_film(subscriptie\_film\_id, film\_id, subscriptie\_id) VALUES (INCREMENTARE\_film.nextval,3,78477);  
insert into SUBSCRIPTIE\_film(subscriptie\_film\_id, film\_id, subscriptie\_id) VALUES (INCREMENTARE\_film.nextval,10,94030);  
insert into SUBSCRIPTIE\_film(subscriptie\_film\_id, film\_id, subscriptie\_id) VALUES (INCREMENTARE\_film.nextval,7,55245);  
insert into SUBSCRIPTIE\_film(subscriptie\_film\_id, film\_id, subscriptie\_id) VALUES (INCREMENTARE\_film.nextval,1,39959);  
insert into SUBSCRIPTIE\_film(subscriptie\_film\_id, film\_id, subscriptie\_id) VALUES (INCREMENTARE\_film.nextval,2,67933);  
insert into SUBSCRIPTIE\_film(subscriptie\_film\_id, film\_id, subscriptie\_id) VALUES (INCREMENTARE\_film.nextval,4,55245);  
insert into SUBSCRIPTIE\_film(subscriptie\_film\_id, film\_id, subscriptie\_id) VALUES (INCREMENTARE\_film.nextval,5,55245);  
insert into SUBSCRIPTIE\_film(subscriptie\_film\_id, film\_id, subscriptie\_id) VALUES (INCREMENTARE\_film.nextval,5,39959);  
insert into SUBSCRIPTIE\_film(subscriptie\_film\_id, film\_id, subscriptie\_id) VALUES (INCREMENTARE\_film.nextval,5,94030);  
insert into SUBSCRIPTIE\_film(subscriptie\_film\_id, film\_id, subscriptie\_id) VALUES (INCREMENTARE\_film.nextval,6,94030);  
  
insert into SERIAL\_ACTOR(serial\_actor\_id, serial\_id, actor\_id) VALUES (INCREMENTARE\_ACTOR.nextval,1,10);  
insert into SERIAL\_ACTOR(serial\_actor\_id, serial\_id, actor\_id) VALUES (INCREMENTARE\_ACTOR.nextval,1,11);  
insert into SERIAL\_ACTOR(serial\_actor\_id, serial\_id, actor\_id) VALUES (INCREMENTARE\_ACTOR.nextval,1,12);  
insert into SERIAL\_ACTOR(serial\_actor\_id, serial\_id, actor\_id) VALUES (INCREMENTARE\_ACTOR.nextval,2,10);  
insert into SERIAL\_ACTOR(serial\_actor\_id, serial\_id, actor\_id) VALUES (INCREMENTARE\_ACTOR.nextval,2,13);  
insert into SERIAL\_ACTOR(serial\_actor\_id, serial\_id, actor\_id) VALUES (INCREMENTARE\_ACTOR.nextval,2,14);  
insert into SERIAL\_ACTOR(serial\_actor\_id, serial\_id, actor\_id) VALUES (INCREMENTARE\_ACTOR.nextval,3,19);  
insert into SERIAL\_ACTOR(serial\_actor\_id, serial\_id, actor\_id) VALUES (INCREMENTARE\_ACTOR.nextval,3,18);  
insert into SERIAL\_ACTOR(serial\_actor\_id, serial\_id, actor\_id) VALUES (INCREMENTARE\_ACTOR.nextval,3,17);  
insert into SERIAL\_ACTOR(serial\_actor\_id, serial\_id, actor\_id) VALUES (INCREMENTARE\_ACTOR.nextval,3,16);  
insert into SERIAL\_ACTOR(serial\_actor\_id, serial\_id, actor\_id) VALUES (INCREMENTARE\_ACTOR.nextval,3,15);  
insert into SERIAL\_ACTOR(serial\_actor\_id, serial\_id, actor\_id) VALUES (INCREMENTARE\_ACTOR.nextval,4,10);  
insert into SERIAL\_ACTOR(serial\_actor\_id, serial\_id, actor\_id) VALUES (INCREMENTARE\_ACTOR.nextval,4,11);  
insert into SERIAL\_ACTOR(serial\_actor\_id, serial\_id, actor\_id) VALUES (INCREMENTARE\_ACTOR.nextval,4,12);  
insert into SERIAL\_ACTOR(serial\_actor\_id, serial\_id, actor\_id) VALUES (INCREMENTARE\_ACTOR.nextval,5,12);  
insert into SERIAL\_ACTOR(serial\_actor\_id, serial\_id, actor\_id) VALUES (INCREMENTARE\_ACTOR.nextval,5,13);  
insert into SERIAL\_ACTOR(serial\_actor\_id, serial\_id, actor\_id) VALUES (INCREMENTARE\_ACTOR.nextval,5,14);  
insert into SERIAL\_ACTOR(serial\_actor\_id, serial\_id, actor\_id) VALUES (INCREMENTARE\_ACTOR.nextval,5,18);  
  
insert into ROL\_JUCAT(film\_rol\_actor\_id, film\_id, rol\_id, actor\_id, timp\_ecran) VALUES (INCREMENTARE\_ROL\_JUCAT.nextval,1,111,10,80);  
insert into ROL\_JUCAT(film\_rol\_actor\_id, film\_id, rol\_id, actor\_id, timp\_ecran) VALUES (INCREMENTARE\_ROL\_JUCAT.nextval,1,112,11,35);  
insert into ROL\_JUCAT(film\_rol\_actor\_id, film\_id, rol\_id, actor\_id, timp\_ecran) VALUES (INCREMENTARE\_ROL\_JUCAT.nextval,1,115,12,10);  
insert into ROL\_JUCAT(film\_rol\_actor\_id, film\_id, rol\_id, actor\_id, timp\_ecran) VALUES (INCREMENTARE\_ROL\_JUCAT.nextval,1,115,13,5);  
insert into ROL\_JUCAT(film\_rol\_actor\_id, film\_id, rol\_id, actor\_id, timp\_ecran) VALUES (INCREMENTARE\_ROL\_JUCAT.nextval,2,113,14,67);  
insert into ROL\_JUCAT(film\_rol\_actor\_id, film\_id, rol\_id, actor\_id, timp\_ecran) VALUES (INCREMENTARE\_ROL\_JUCAT.nextval,2,114,15,55);  
insert into ROL\_JUCAT(film\_rol\_actor\_id, film\_id, rol\_id, actor\_id, timp\_ecran) VALUES (INCREMENTARE\_ROL\_JUCAT.nextval,3,111,16,72);  
insert into ROL\_JUCAT(film\_rol\_actor\_id, film\_id, rol\_id, actor\_id, timp\_ecran) VALUES (INCREMENTARE\_ROL\_JUCAT.nextval,3,113,17,48);  
insert into ROL\_JUCAT(film\_rol\_actor\_id, film\_id, rol\_id, actor\_id, timp\_ecran) VALUES (INCREMENTARE\_ROL\_JUCAT.nextval,3,114,18,37);  
insert into ROL\_JUCAT(film\_rol\_actor\_id, film\_id, rol\_id, actor\_id, timp\_ecran) VALUES (INCREMENTARE\_ROL\_JUCAT.nextval,4,111,18,57);  
insert into ROL\_JUCAT(film\_rol\_actor\_id, film\_id, rol\_id, actor\_id, timp\_ecran) VALUES (INCREMENTARE\_ROL\_JUCAT.nextval,4,113,18,43);  
insert into ROL\_JUCAT(film\_rol\_actor\_id, film\_id, rol\_id, actor\_id, timp\_ecran) VALUES (INCREMENTARE\_ROL\_JUCAT.nextval,4,114,19,23);  
insert into ROL\_JUCAT(film\_rol\_actor\_id, film\_id, rol\_id, actor\_id, timp\_ecran) VALUES (INCREMENTARE\_ROL\_JUCAT.nextval,5,111,15,66);  
insert into ROL\_JUCAT(film\_rol\_actor\_id, film\_id, rol\_id, actor\_id, timp\_ecran) VALUES (INCREMENTARE\_ROL\_JUCAT.nextval,5,112,13,43);  
insert into ROL\_JUCAT(film\_rol\_actor\_id, film\_id, rol\_id, actor\_id, timp\_ecran) VALUES (INCREMENTARE\_ROL\_JUCAT.nextval,5,113,12,52);  
insert into ROL\_JUCAT(film\_rol\_actor\_id, film\_id, rol\_id, actor\_id, timp\_ecran) VALUES (INCREMENTARE\_ROL\_JUCAT.nextval,5,115,10,12);  
insert into ROL\_JUCAT(film\_rol\_actor\_id, film\_id, rol\_id, actor\_id, timp\_ecran) VALUES (INCREMENTARE\_ROL\_JUCAT.nextval,6,111,11,12);  
insert into ROL\_JUCAT(film\_rol\_actor\_id, film\_id, rol\_id, actor\_id, timp\_ecran) VALUES (INCREMENTARE\_ROL\_JUCAT.nextval,6,112,16,12);  
insert into ROL\_JUCAT(film\_rol\_actor\_id, film\_id, rol\_id, actor\_id, timp\_ecran) VALUES (INCREMENTARE\_ROL\_JUCAT.nextval,6,115,17,12);  
insert into ROL\_JUCAT(film\_rol\_actor\_id, film\_id, rol\_id, actor\_id, timp\_ecran) VALUES (INCREMENTARE\_ROL\_JUCAT.nextval,6,115,15,22);  
insert into ROL\_JUCAT(film\_rol\_actor\_id, film\_id, rol\_id, actor\_id, timp\_ecran) VALUES (INCREMENTARE\_ROL\_JUCAT.nextval,6,115,13,10);  
insert into ROL\_JUCAT(film\_rol\_actor\_id, film\_id, rol\_id, actor\_id, timp\_ecran) VALUES (INCREMENTARE\_ROL\_JUCAT.nextval,7,111,14,57);  
insert into ROL\_JUCAT(film\_rol\_actor\_id, film\_id, rol\_id, actor\_id, timp\_ecran) VALUES (INCREMENTARE\_ROL\_JUCAT.nextval,7,114,14,63);  
insert into ROL\_JUCAT(film\_rol\_actor\_id, film\_id, rol\_id, actor\_id, timp\_ecran) VALUES (INCREMENTARE\_ROL\_JUCAT.nextval,7,112,13,44);  
insert into ROL\_JUCAT(film\_rol\_actor\_id, film\_id, rol\_id, actor\_id, timp\_ecran) VALUES (INCREMENTARE\_ROL\_JUCAT.nextval,8,113,19,77);  
insert into ROL\_JUCAT(film\_rol\_actor\_id, film\_id, rol\_id, actor\_id, timp\_ecran) VALUES (INCREMENTARE\_ROL\_JUCAT.nextval,8,114,18,82);  
insert into ROL\_JUCAT(film\_rol\_actor\_id, film\_id, rol\_id, actor\_id, timp\_ecran) VALUES (INCREMENTARE\_ROL\_JUCAT.nextval,9,111,12,93);  
insert into ROL\_JUCAT(film\_rol\_actor\_id, film\_id, rol\_id, actor\_id, timp\_ecran) VALUES (INCREMENTARE\_ROL\_JUCAT.nextval,9,115,17,23);  
insert into ROL\_JUCAT(film\_rol\_actor\_id, film\_id, rol\_id, actor\_id, timp\_ecran) VALUES (INCREMENTARE\_ROL\_JUCAT.nextval,9,115,13,44);  
insert into ROL\_JUCAT(film\_rol\_actor\_id, film\_id, rol\_id, actor\_id, timp\_ecran) VALUES (INCREMENTARE\_ROL\_JUCAT.nextval,10,111,19,123);  
  
insert into FILM(film\_id, director\_id, denumire, nota, data\_aparitie) VALUES (11, 100002, 'Top Gun: Maverick', NULL, '23/11/2021');  
insert into FILM(film\_id, director\_id, denumire, nota, data\_aparitie) VALUES (12, 100003, 'Avengers: Endgame', NULL, '23/11/2021');  
  
insert into DIRECTOR(director\_id, nume, prenume, data\_nastere, nota) VALUES (100006,'William','Anderson','10/05/1990',5);  
  
insert into SUBSCRIPTIE\_film(subscriptie\_film\_id, film\_id, subscriptie\_id) VALUES (INCREMENTARE\_film.nextval,11,78477);  
insert into SUBSCRIPTIE\_film(subscriptie\_film\_id, film\_id, subscriptie\_id) VALUES (INCREMENTARE\_film.nextval,12,78477);

****

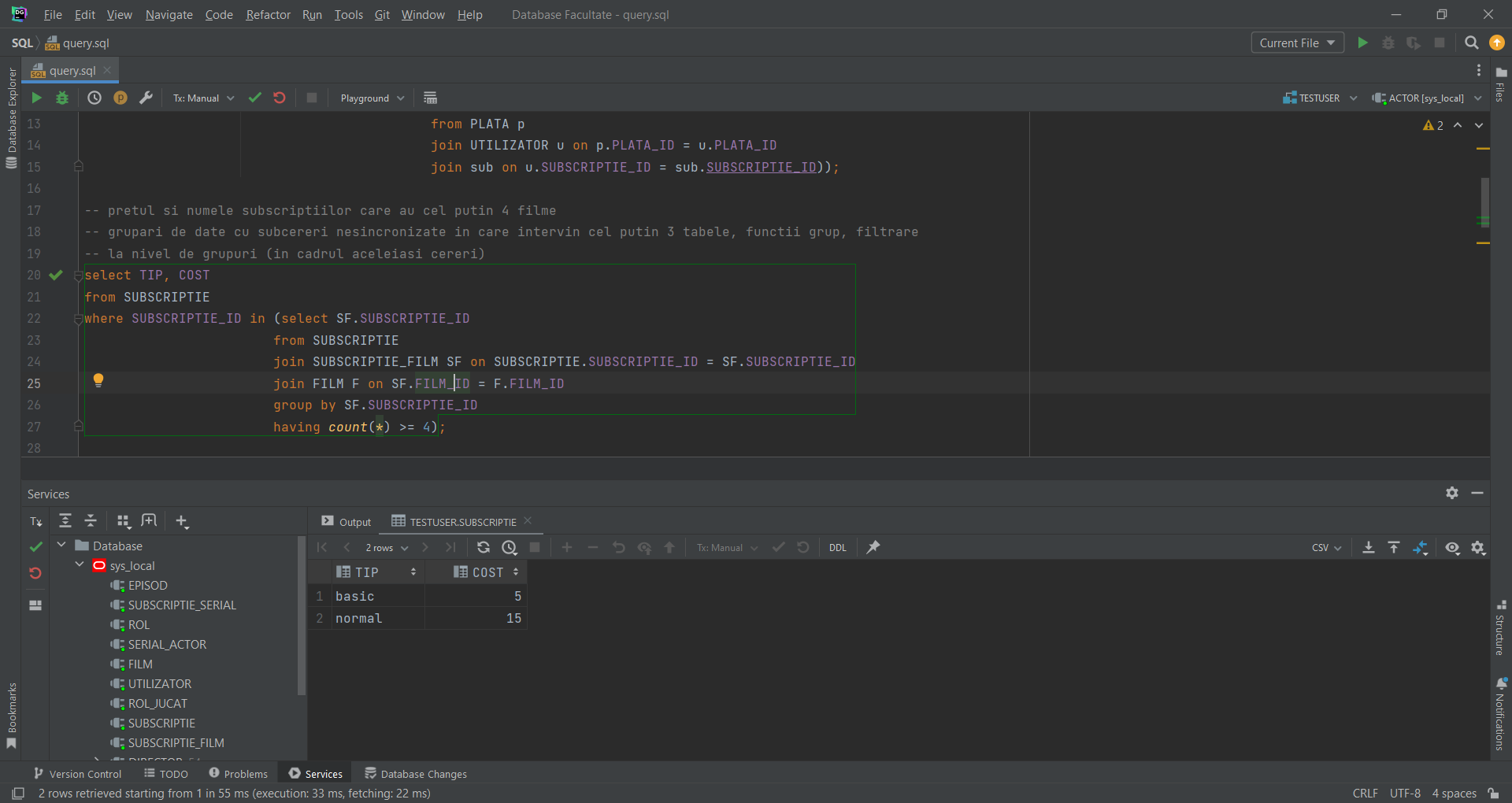
****

# Query-uri

## Grupări de date cu subcereri nesincronizate în care intervin cel puțin 3 tabele, funcții grup, filtrare la nivel de grupuri

**Prețul și numele subscripțiilor care au cel puțin 4 filme**

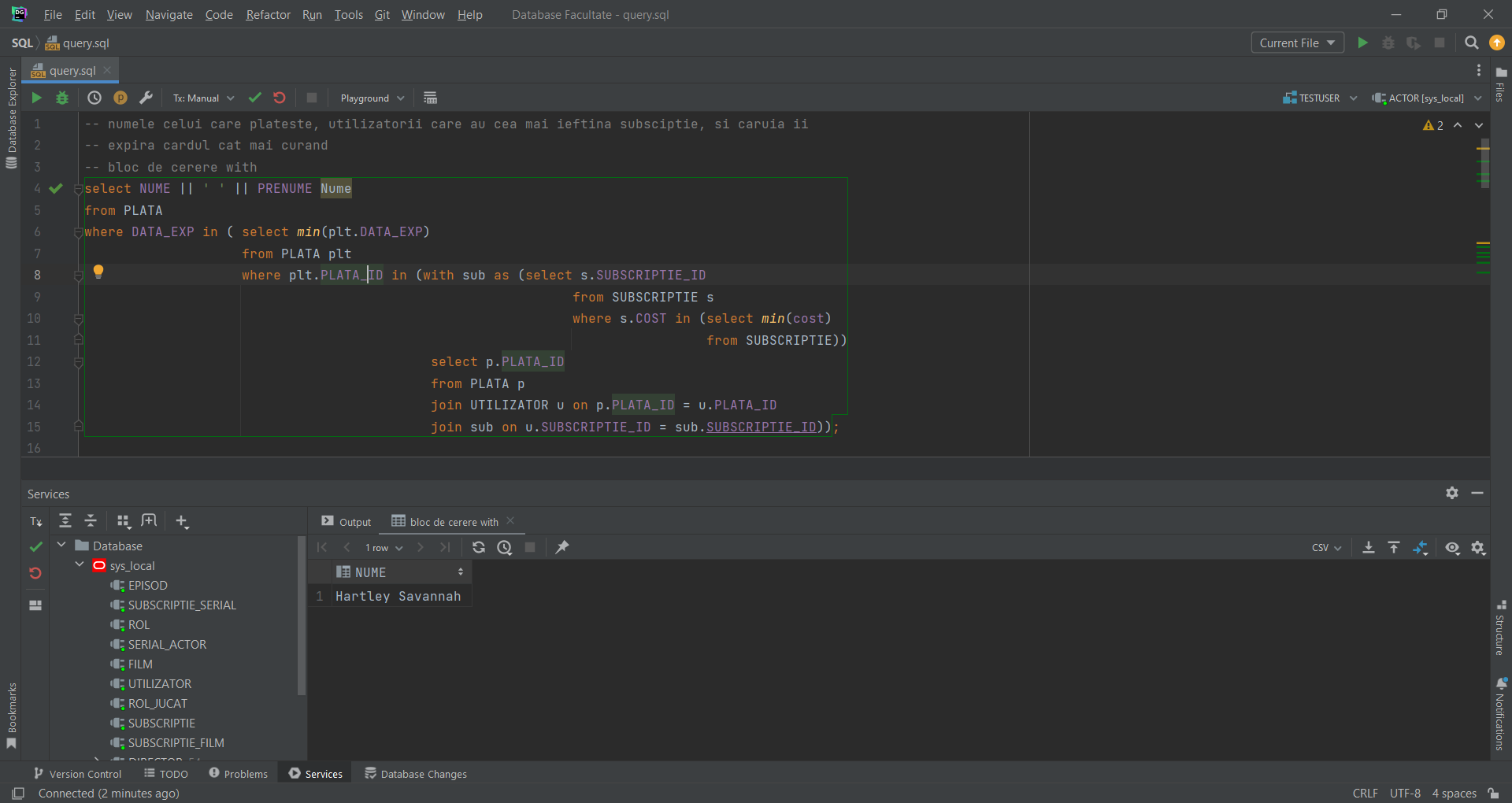
select TIP, COST  
from SUBSCRIPTIE  
where SUBSCRIPTIE\_ID in (select SF.SUBSCRIPTIE\_ID  
 from SUBSCRIPTIE  
 join SUBSCRIPTIE\_FILM SF on SUBSCRIPTIE.SUBSCRIPTIE\_ID = SF.SUBSCRIPTIE\_ID  
 join FILM F on SF.FILM\_ID = F.FILM\_ID  
 group by SF.SUBSCRIPTIE\_ID  
 having *count*(\*) >= 4);



## Bloc de cerere with

**Numele celui care platește, utilizatorii care au cea mai ieftină subscipție, și căruia îi expiră cardul cât mai curând**

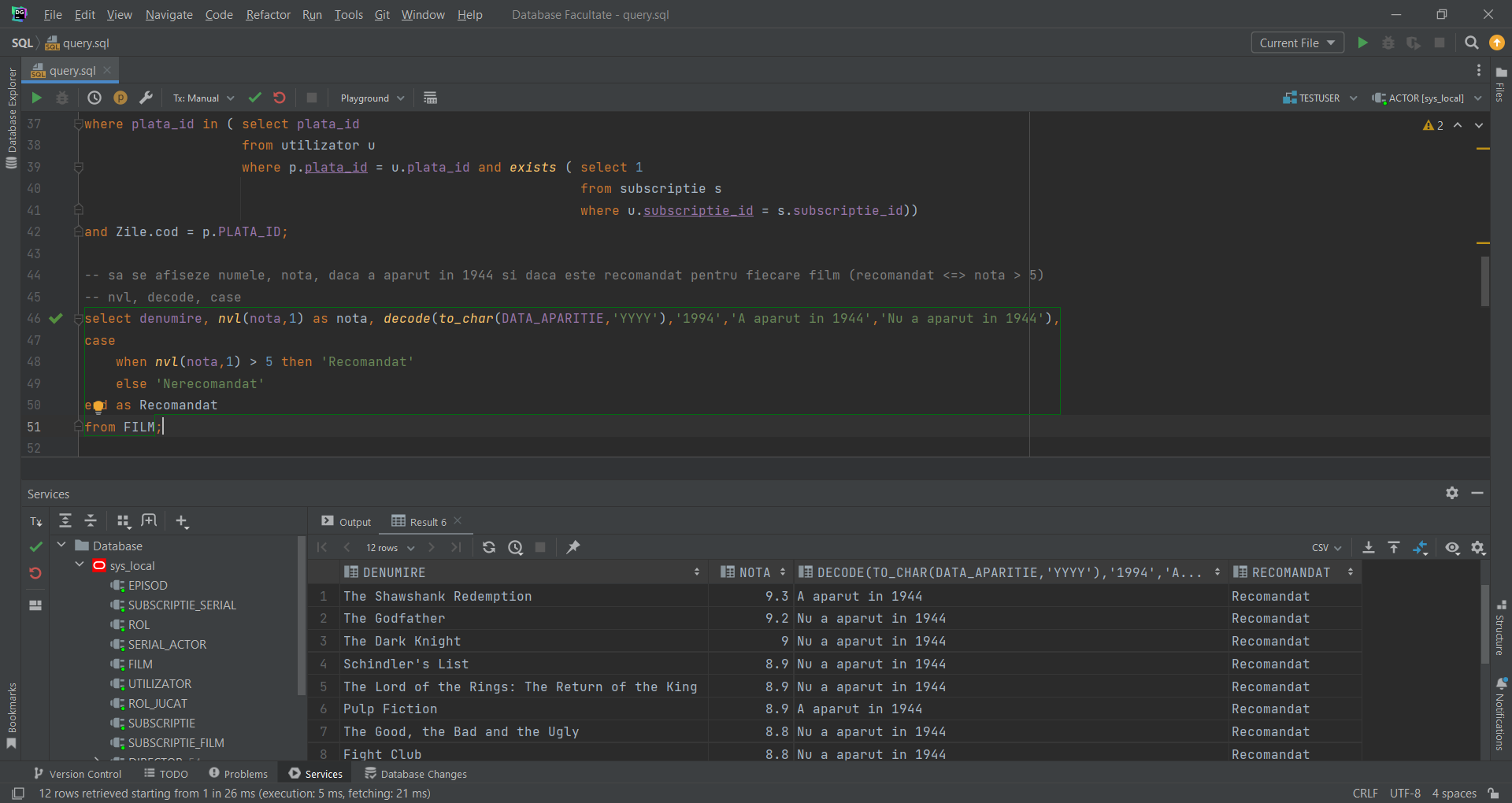
select NUME || ' ' || PRENUME Nume  
from PLATA  
where DATA\_EXP in ( select *min*(plt.DATA\_EXP)  
 from PLATA plt  
 where plt.PLATA\_ID in (with sub as (select s.SUBSCRIPTIE\_ID  
 from SUBSCRIPTIE s  
 where s.COST in (select *min*(cost)  
 from SUBSCRIPTIE))  
 select p.PLATA\_ID  
 from PLATA p  
 join UTILIZATOR u on p.PLATA\_ID = u.PLATA\_ID  
 join sub on u.SUBSCRIPTIE\_ID = sub.SUBSCRIPTIE\_ID));



## nvl, decode, case

**Să se afișeze numele, nota, dacă a aparut in 1944 și dacă este recomandat pentru fiecare film (recomandat <=> nota > 5)**

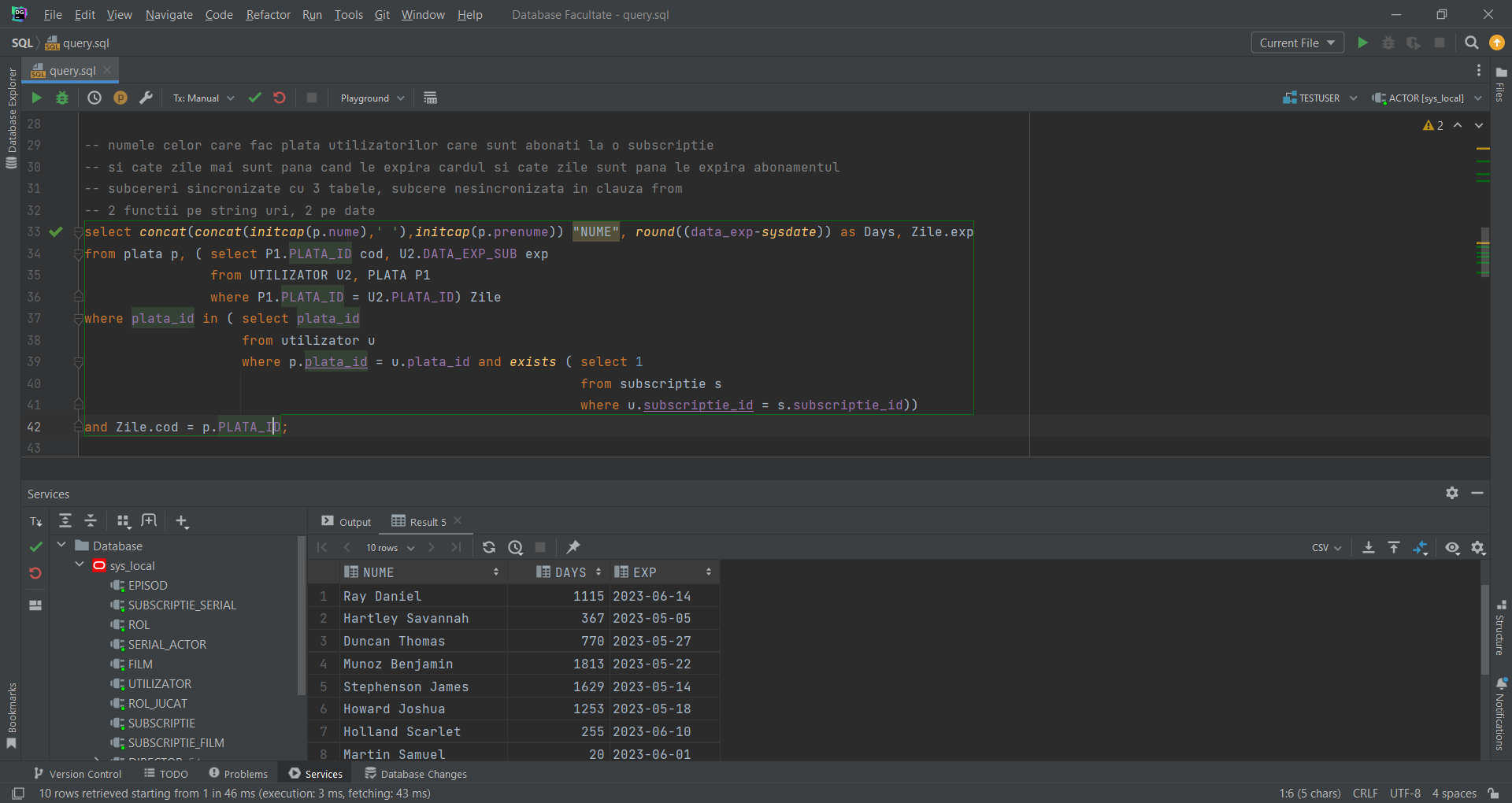
select denumire, *nvl*(nota,1) as nota, *decode*(*to\_char*(DATA\_APARITIE,'YYYY'),'1994','A aparut in 1944','Nu a aparut in 1944'),  
case  
 when *nvl*(nota,1) > 5 then 'Recomandat'  
 else 'Nerecomandat'  
end as Recomandat  
from FILM;

****

## Subcereri sincronizate cu 3 tabele, subcere nesincronizata în clauza from 2 funcții pe string uri, 2 pe date

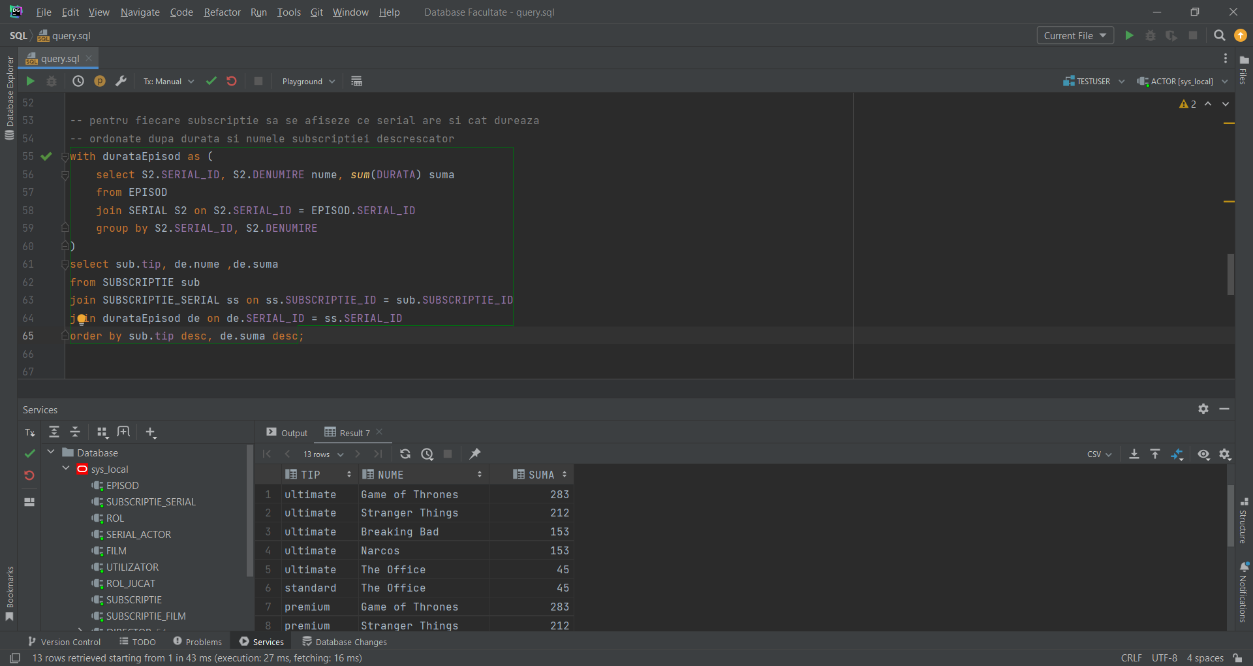
**Numele celor care fac plata utilizatorilor care sunt abonați la o subscripție și câte zile mai sunt până când le expiră cardul și câte zile sunt până le expiră abonamentul**

select *concat*(*concat*(*initcap*(p.nume),' '),*initcap*(p.prenume)) "NUME", *round*((data\_exp-*sysdate*)) as Days, Zile.exp  
from plata p, ( select P1.PLATA\_ID cod, U2.DATA\_EXP\_SUB exp  
 from UTILIZATOR U2, PLATA P1  
 where P1.PLATA\_ID = U2.PLATA\_ID) Zile  
where plata\_id in ( select plata\_id  
 from utilizator u  
 where p.plata\_id = u.plata\_id and *exists* ( select 1  
 from subscriptie s  
 where u.subscriptie\_id = s.subscriptie\_id))  
and Zile.cod = p.PLATA\_ID;

****

## Pentru fiecare subscripție să se afișeze ce serial și cât durează ordonate după durată și numele subscripției descrescător

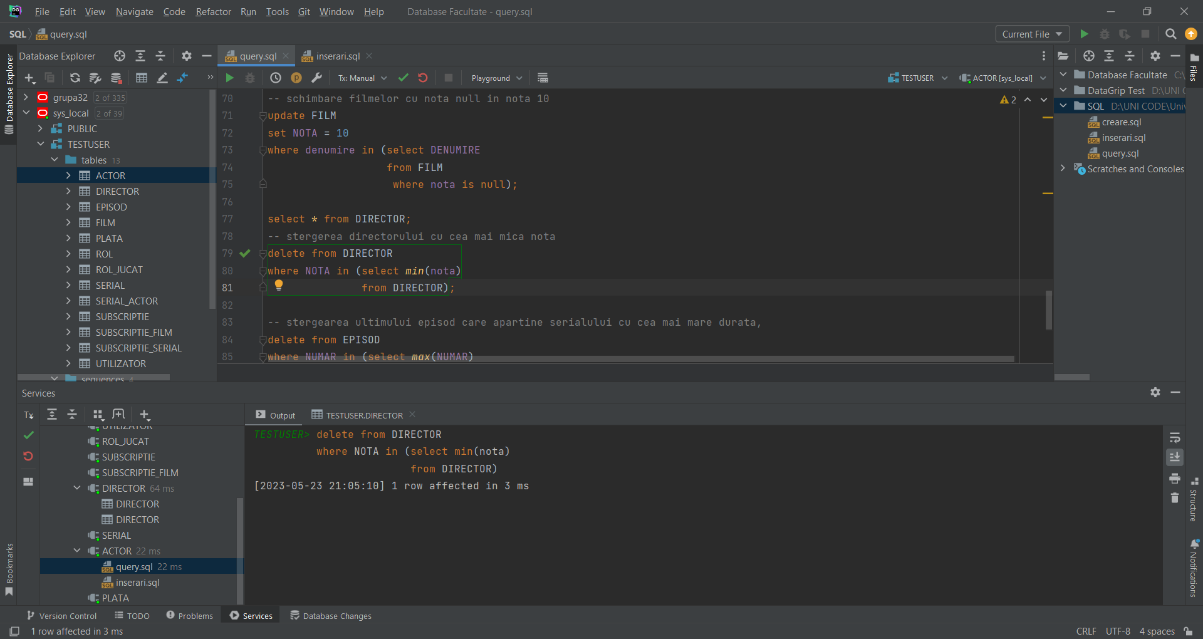
with durataEpisod as (  
 select S2.SERIAL\_ID, S2.DENUMIRE nume, *sum*(DURATA) suma  
 from EPISOD  
 join SERIAL S2 on S2.SERIAL\_ID = EPISOD.SERIAL\_ID  
 group by S2.SERIAL\_ID, S2.DENUMIRE  
)  
select sub.tip, de.nume ,de.suma  
from SUBSCRIPTIE sub  
join SUBSCRIPTIE\_SERIAL ss on ss.SUBSCRIPTIE\_ID = sub.SUBSCRIPTIE\_ID  
join durataEpisod de on de.SERIAL\_ID = ss.SERIAL\_ID  
order by sub.tip desc, de.suma desc;



# Operații

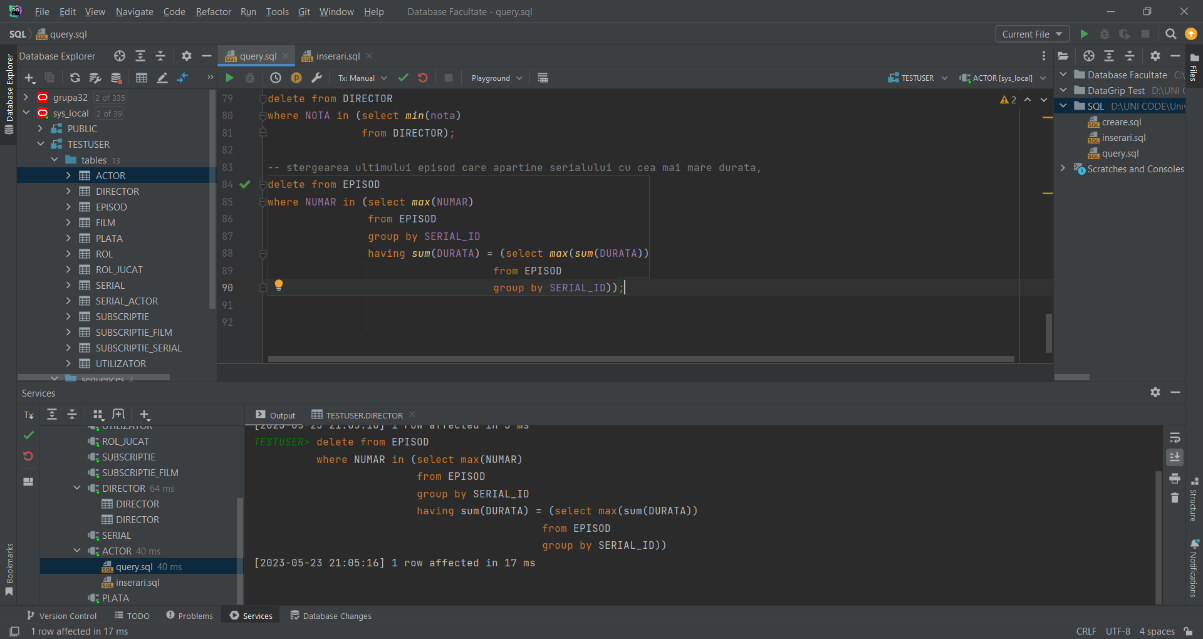
## ștergerea directorului cu cea mai mică notă

delete from DIRECTOR  
where NOTA in (select *min*(nota)  
 from DIRECTOR);

****

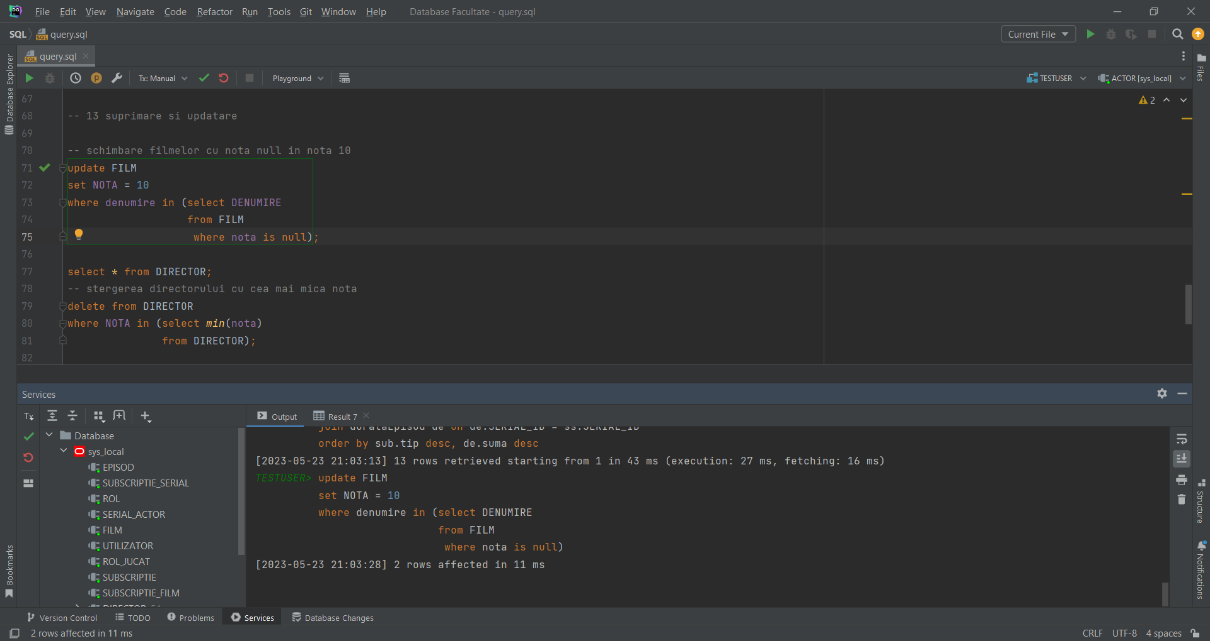
## ștergearea ultimului episod care aparține serialului cu cea mai mare durată

delete from EPISOD  
where NUMAR in (select *max*(NUMAR)  
 from EPISOD  
 group by SERIAL\_ID  
 having *sum*(DURATA) = (select *max*(*sum*(DURATA))  
 from EPISOD  
 group by SERIAL\_ID));

****

## schimbare filmelor cu nota null in nota 10

update FILM  
set NOTA = 10  
where denumire in (select DENUMIRE  
 from FILM  
 where nota is null);

****

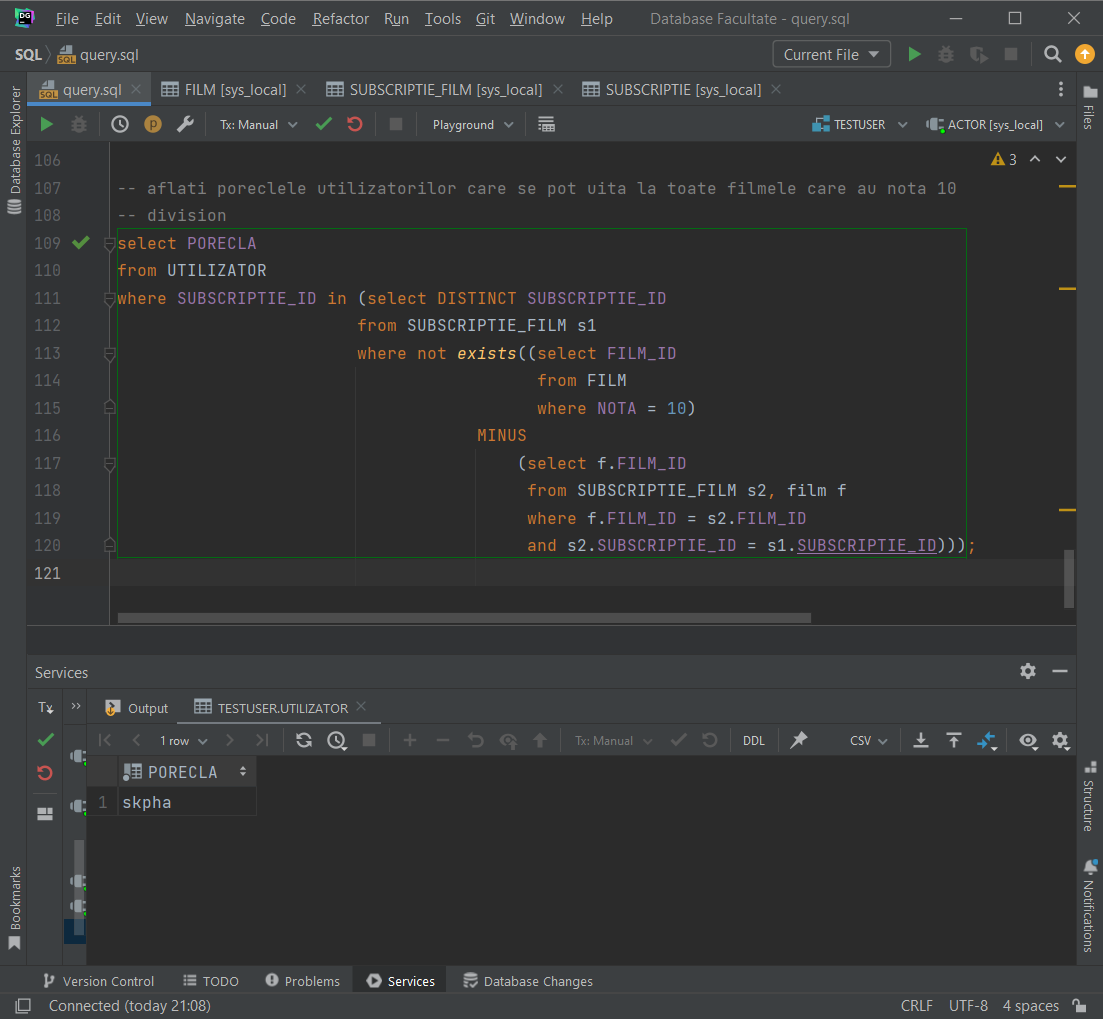
1. **C**

# Cereri

## Division

**Aflați poreclele utilizatorilor care se pot uita la toate filmele care au nota 10**

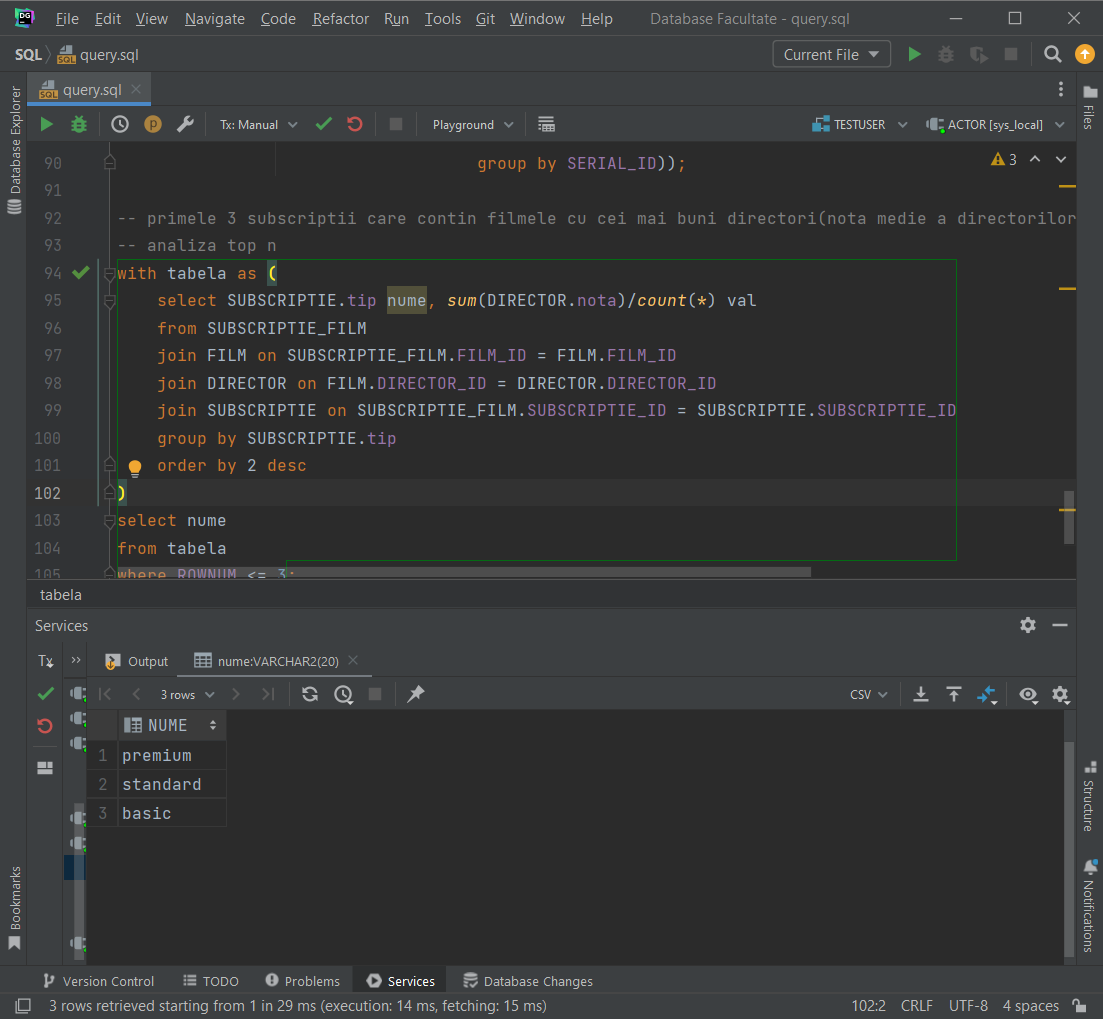
select PORECLA  
from UTILIZATOR  
where SUBSCRIPTIE\_ID in (select DISTINCT SUBSCRIPTIE\_ID  
 from SUBSCRIPTIE\_FILM s1  
 where not *exists*((select FILM\_ID  
 from FILM  
 where NOTA = 10)  
 MINUS  
 (select f.FILM\_ID  
 from SUBSCRIPTIE\_FILM s2, film f  
 where f.FILM\_ID = s2.FILM\_ID  
 and s2.SUBSCRIPTIE\_ID = s1.SUBSCRIPTIE\_ID)));

****

## Top-n

**Primele 3 subscripții care conțin filmele cu cei mai buni directori (notă medie a directorilor)**

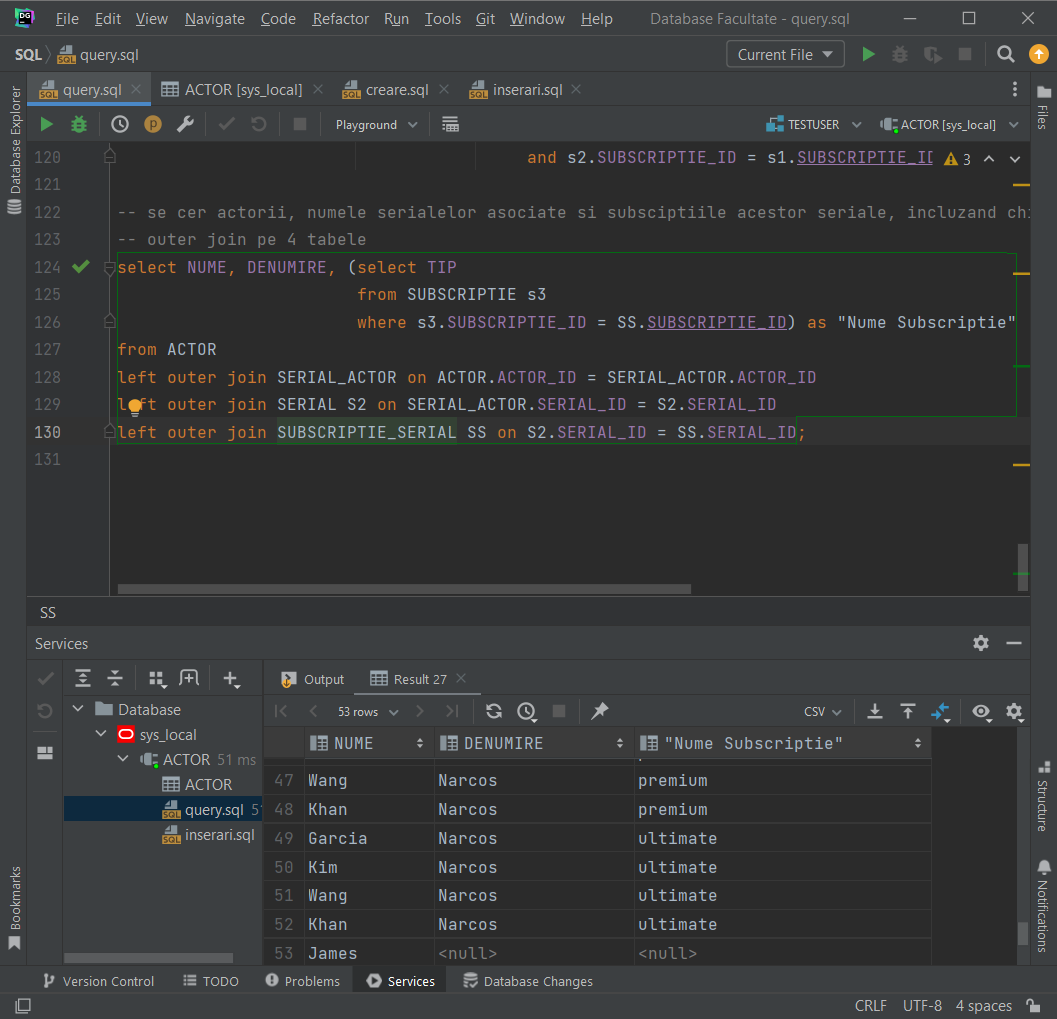
with tabela as (  
 select SUBSCRIPTIE.tip nume, *avg*(DIRECTOR.nota) val  
 from SUBSCRIPTIE\_FILM  
 join FILM on SUBSCRIPTIE\_FILM.FILM\_ID = FILM.FILM\_ID  
 join DIRECTOR on FILM.DIRECTOR\_ID = DIRECTOR.DIRECTOR\_ID  
 join SUBSCRIPTIE on SUBSCRIPTIE\_FILM.SUBSCRIPTIE\_ID = SUBSCRIPTIE.SUBSCRIPTIE\_ID  
 group by SUBSCRIPTIE.tip  
 order by 2 desc  
)  
select nume  
from tabela  
where ROWNUM <= 3;

****

## Outer Join

**Se cer actorii, numele serialelor asociate si subscipțiile acestor seriale, incluzand și actorii care nu joacă în niciun serial**

select NUME, DENUMIRE, (select TIP  
 from SUBSCRIPTIE s3  
 where s3.SUBSCRIPTIE\_ID = SS.SUBSCRIPTIE\_ID) as "Nume Subscriptie"  
from ACTOR  
left outer join SERIAL\_ACTOR on ACTOR.ACTOR\_ID = SERIAL\_ACTOR.ACTOR\_ID  
left outer join SERIAL S2 on SERIAL\_ACTOR.SERIAL\_ID = S2.SERIAL\_ID  
left outer join SUBSCRIPTIE\_SERIAL SS on S2.SERIAL\_ID = SS.SERIAL\_ID;

****

# Optimizarea unei cereri

## Cererea neoptimizată

**Se cere numele directorilor care au filme cu nota peste 9 și au apărut după anul 2000**

-- 16  
-- numele directorilor care au filme cu nota peste 9 si au aparut dupa anul 2000  
select distinct d.NUME  
from DIRECTOR D, FILM F  
where f.FILM\_ID in (  
 select FILM\_ID  
 from FILM  
 where NOTA >= 9 and f.DATA\_APARITIE >= '01/01/2000'  
 ) and d.DIRECTOR\_ID = f.DIRECTOR\_ID;  
  
-- R1 = PRODUCT(DIRECTOR, FILM)  
-- R2 = SELECT(R1, f.nota >= 9 and data\_aparitie >= 01/01/2000 and d.director\_id = f.director\_id)  
-- REZULTAT = R3 = PROJECT(R3, nume)

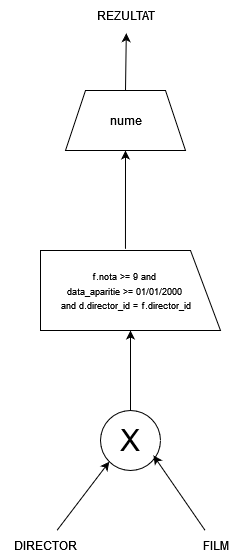
### Expresie algebrică:

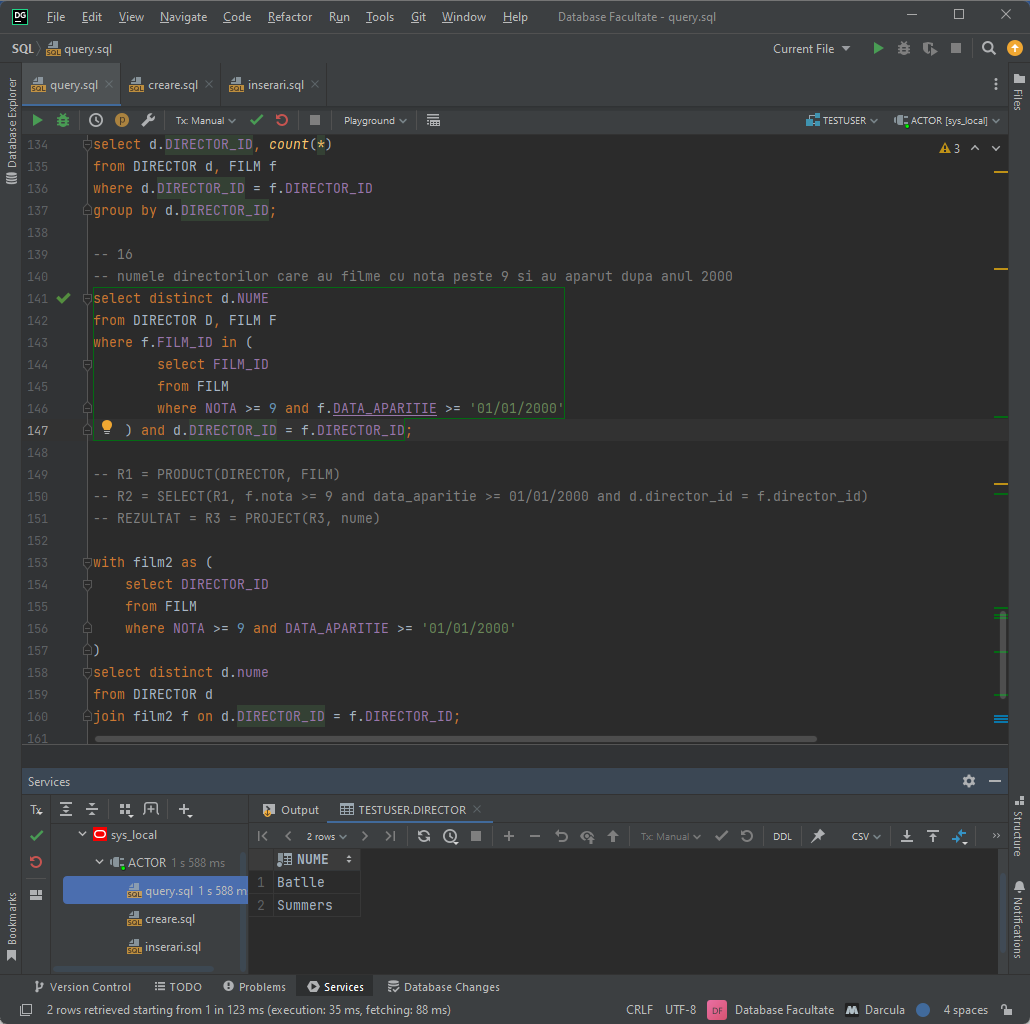
R1 = PRODUCT(DIRECTOR, FILM)

R2 = SELECT(R1, f.nota >= 9 and data\_aparitie >= 01/01/2000 and d.director\_id = f.director\_id)

REZULTAT = R3 = PROJECT(R3, nume)

### Arbore algebric:





## Cererea optimizată:

**Se cere numele directorilor care au filme cu nota peste 9 și au apărut după anul 2000**

with film2 as (  
 select DIRECTOR\_ID  
 from FILM  
 where NOTA >= 9 and DATA\_APARITIE >= '01/01/2000'  
)  
select distinct d.nume  
from DIRECTOR d  
join film2 f on d.DIRECTOR\_ID = f.DIRECTOR\_ID;  
  
-- R1 = SELECT(FILM, nota >= 9 and data\_aparitie >= 01/01/2000)  
-- R2 = PROJECT(DIRECTOR, nume)  
-- REZULTAT = R3 = JOIN(R1,R2)

### 

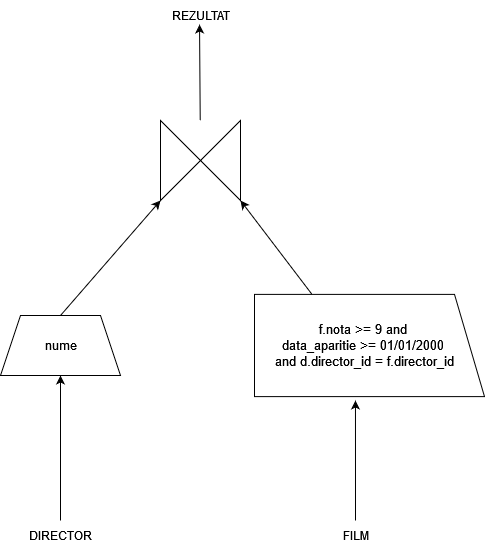
### Expresie algebrică:

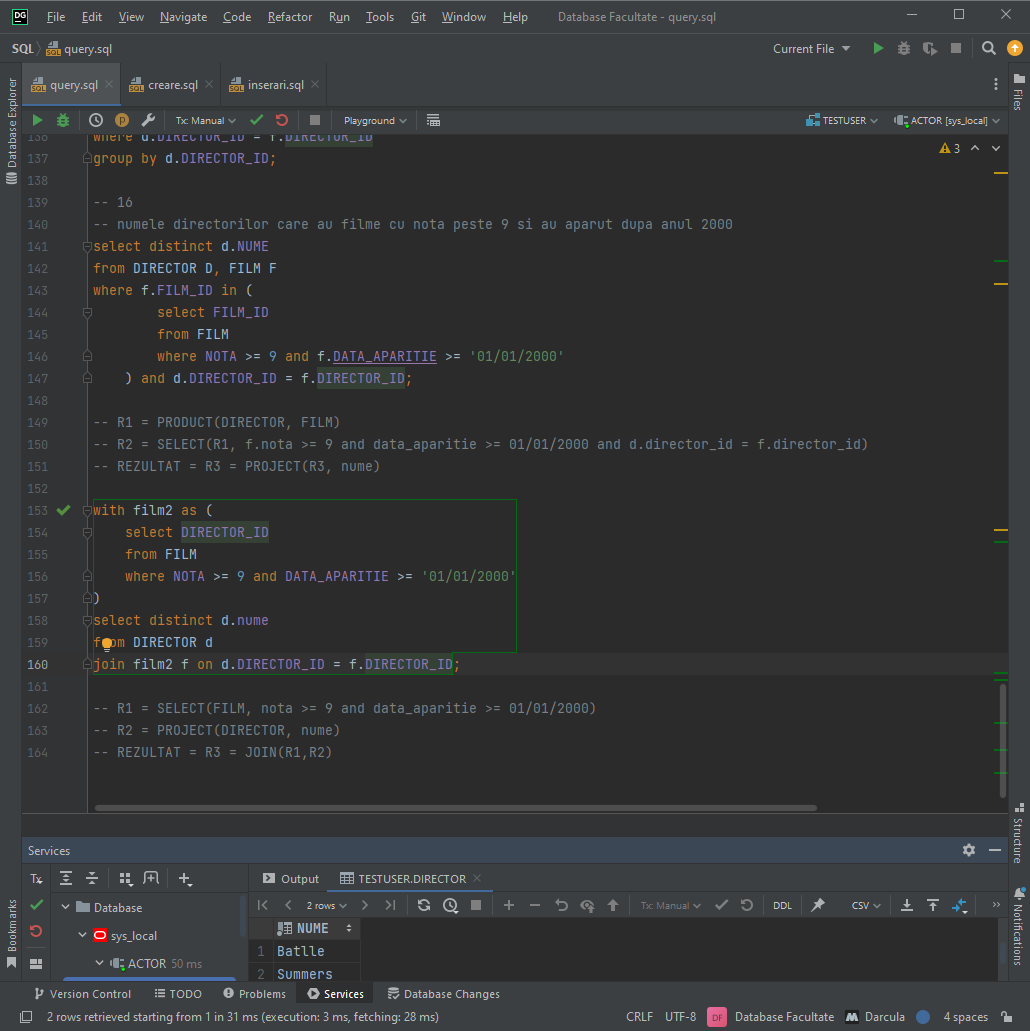
R1 = SELECT(FILM, nota >= 9 and data\_aparitie >= 01/01/2000)

R2 = PROJECT(DIRECTOR, nume)

REZULTAT = R3 = JOIN(R1, R2)

### Arbore algebric:





Realizarea normalizării

## Normalizări

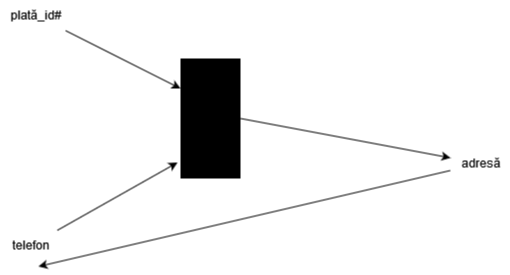
### BCNF:

Exemplu non-BCNF: Plată(plată\_id#, telefon#, adresă)

În dependența adresă->telefon determinantul nu este o cheie candidat deci relația se descompune in:

Plată\_1(plată\_id#, adresă)

Plată\_2(adresă#, telefon)



### FN4:

Exemplu non-FN4: Subscripție\_Serial(SubscripțieSerial\_id#,Serial\_id#,Episod\_id#,Subscripție\_id#), unde un serial poate avea mai multe episoade și poate aparține mai multor subscripții.

Deci există multidependențele:

- serial\_id -> -> episod\_id

- serial\_id -> -> subscripție\_id

Pentru a transforma relația în FN4 o descompunem prin proiecție în două relații:

Subscripție\_Serial\_1(serial\_id#, episod\_id#)

Subscripție\_Serial\_2(serial\_id#, subscripție\_id#)

Subscripție\_Serial = JOIN(Subscripție\_Serial\_1, Subscripție\_Serial\_2)

### FN5:

Exemplu non-FN5: Producții\_preferate\_de\_utilizator(utilizator\_id#,film\_id#,actor\_id#). Nu se află în FN5 deoarece un utilizator poate avea mai multe filme preferate, și actori preferați, iar cu această relație dorim să aflăm producțiile preferate în care ținem cont de aceste 2 criterii.

Așa că o desfacem în trei relații:

- Actori\_care\_joacă\_în\_filme(actor\_id#, film\_id#)

- Actori\_preferați\_de\_utilizator(utilizator\_id#, actor\_id#)

- Filme\_preferate\_de\_utilizator(utilizator\_id#, film\_id#)

De unde se poate observa relația Producții\_preferate\_de\_utilizator = JOIN(JOIN(Filme\_preferate\_de\_utilizator, Actori\_preferați\_de\_utilizator), Actori\_care\_joacă\_în\_filme)

## Denormalizări

În tabela: Subscripție\_serial(subscripțieSerial\_id#, serial\_id#, subscripție\_id#)

Putem adăuga redundanță (atribute în plus), și anume: denumire și tip, pentru a putea răspunde mai rapid la interogări.

Tabel denormalizat: Sub\_serial\_Denormalizat(subscripțieSerial\_id#, serial\_id#, subscripție\_id#, denumire, tip)

Astfel prin denormalizare scăpăm de operațiile de join pe alte tabele, micșorând astfel timpul de execuție, dar avem ca și cost mărirea memoriei în baza de date