Proiect
Baze de date

Rădulescu Mihai-Alexandru
-- Grupa 234 --

Cuprins

Cerința 1 Prezentarea modelului

Pentru acest proiect am ales sa creez modelul unei baze de date utilizata de un magazin online.

Cu ajutorul acestui model putem gestiona informații despre utilizatorii magazinului, despre produsele vandute, stocul disponibil și depozitele vanzatorilor. Modelul ne permite și reținerea unui istoric al tuturor comenzilor plasate de utilizatorii din baza de date.

Cerința 2 Prezentarea constrangerilor

În implementarea acestui model am impus următoarele constrangeri:

- primary key: pentru id-urile unice ale linilor din tabele
- not null: pentru valori obligatorii (ex: numele unui utilizator sau locația unui depozit)
- foreign key: pentru a accesa date din alte tabele mai ușor (ex: locația unui utilizator sau depozit)
- on delete set null: în cazul în care o locație este ștearsă din tabel, acolo unde id-ul locației apare cu rol de cheie externă, id-ul va deveni null
- on delete cascade: în cazul în care un produs sau utilizator este șters, se vor șterge restul de linii care au drept cheie externă id-ul respectiv

Cerința 3 Descrierea entitatilor

Modelul creat contine 6 tabele și 2 tabele asociative.

Tabelul *Utilizator*: contine date de baza despre un utilizator, ca de exemplu nume, prenume, date de contact(email, telefon). Cheia primara este un indice unic Utilizator_id.

Tabelul *Comanda:* contine id-ul utilizatorului care a plasat comanda și data la care a fost plasată. Cheia primara este un indice unic Comanda_id.

Tabelul *Locație*: conține informații despre o adresa, țara și oraș. Cheia primara este un indice unic Locatie_id.

Tabelul *Categorie*: conține denumirea unei categori, prețul maxim și prețul minim al produselor din categoria respectivă. Cheia primara este un indice unic Categorie_id.

Tabelul *Depozit*: conține un id către locația unde se afla depozitul și datele de contact(email, telefon). Cheia primara este un id unic Depozit_id.

Tabelul *Produs*: conține denumirea, descriera, prețul și rating-ul produsului. La acestea se adaugă 2 chei externe care indica utilizatorul care vandut produsul și categoria din care face parte.

Cheia primara este un id unic Produs_id.

Tabela asociativă *PlasareComanda*: conține cantitatea de produs vanduta la o anumită comandă. Cheia primara este compusa din 2 chei externe Produs_id și Comanda_id.

Tabela asociativă *StocDepozit*: contine stocul disponibil într-un depozit x, pentru un produs y. Cheia primară este compusă din 2 chei externe Depozit_id si Produs_id.

Cerința 4 Descrierea relațiilor

În modelul curent exista 6 relații:

Utilizator plasează Comenzi: one to many

Utilizatori se afla la Locație: many to one

Depozite se afla la Locație: many to one

Categoria contine Produse: one to many

Comanda contine Produse: one to many

Depozit are în stoc Produse: one to many

Cerința 5 Descrierea atributelor

Locație:

Locatie_id: reprezinta cheia primara a tabelului, de tip numar.

Adresa: reprezinta detalile unei adrese (strada, numar, apartament), de tip șir de caractere de lungime maxima 100.

Oras: reprezinta orașul unde se găsește adresa, de tip șir de caractere de lungime 100.

Tara: reprezinta tara unde se gaseste adresa, de tip șir de caractere de lungime 100.

Utilizator:

Utilizator_id: reprezinta cheia primara a tabelului, de tip numar.

Nume: reprezinta numele unui utilizator, de tip sir de caractere de lungime 20, nu poate fii null.

Prenume: reprezinta prenumele unui utilizator, de tip sir de caractere de lungime 20, nu poate fii null.

Tip: reprezint rolul unui utilizator: utilizator, partener, administrator. Este de tip șir de caractere de lungime 20.

Email: reprezint adresa de email a unui utilizator, de tip șir de caractere de lungime 60.

Telefon: reprezinta numarul de telefon al unui utilizator, de tip șir de caractere de lungime 10.

DataInregistrare: reprezinta data la care sa inregistrat un utilizator, de tip data.

Locatie_id: reprezinta o cheie externa catre adresa utilizatorului, de tip numar, nu poate fii null.

Categorie:

Categorie_id: reprezinda cheia primara a tabelului, de tip numar.

NumeCategorie: reprezinta numele unei categorii de produse, de tip șir de caractere de lungime 50.

PretMinim: reprezinta pretul celui mai ieftin produs din categoria respectivă, de tip număr.

PretMaxim: reprezinta pretul celui mai scump produs din categoria respectivă, de tip număr.

Produs:

Produs_id: reprezinda cheia primara a tabelului, de tip numar.

Vanzator_id: reprezinta o cheie externa catre utilizatorul care vinde produsul respectiv, de tip număr, nu poate fii null.

Categorie_id: reprezinta o cheie externa catre categoria din care face parte produsul respectiv, de tip numar, nu poate fii null.

Titlu: reprezinta denumirea produsului, de tip sir de caractere de lungime 200.

Descriere: reprezinta descrierea produsului, de tip sir de caractere de lungime 3000.

Pret: reprezinta pretul produsului, de tip numar, de maxim 10 cifre, rotunjit la 2 zecimale.

Rating: reprezinta rating-ul produsului respectiv, de tip număr, de maxim 2 cifre, rotunjit la o zecimala.

Comanda:

Comanda_id: reprezinda cheia primara a tabelului, de tip numar.

Utilizator_id: reprezinta o cheie externa catre utilizatorul care a plasat comanda respectivă, de tip numar, nu poate fii null.

Data: reprezinta data plasarii comenzii, de tip data.

PlasareComanda:

Produs_id: reprezinta o cheie externa catre un produs comandat, de tip numar, nu poate fii null.

Comanda_id: reprezinta o cheie externa catre o comanda plasata, de tip numar, nu poate fii null.

Cantitate: reprezinta numarul de produse cu id-ul Produs_id comandate, de tip număr.

Cheia primara a tabelei asociative este compusa din cele 2 chei externe Produs_id și Comanda_id.

Depozit:

Depozit_id: reprezinda cheia primara a tabelului, de tip numar.

Locatie_id: reprezinta o cheie externa catre adresa depozitului, de tip numar, nu poate fii null.

Telefon: reprezinta numarul de telefon prin care se poate lua legatura cu angajații depozitului, de tip șir de caractere de lungime 10.

Email: reprezint adresa de email prin care se poate lua legatura cu angajații depozitului, de tip șir de caractere de lungime 60.

StocDepozit:

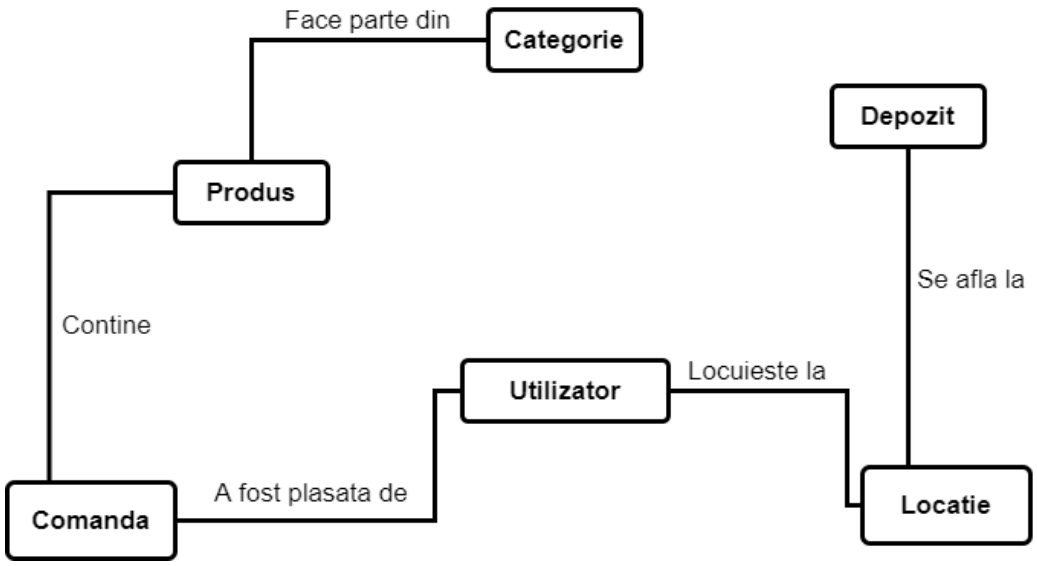
Produs_id: reprezinta o cheie externa catre un produs care este in stoc în depozit, de tip numar, nu poate fii null.

Depozit_id: reprezinta o cheie externa catre depozitul unde produsul cu id-ul Produs_id se afla în stoc.

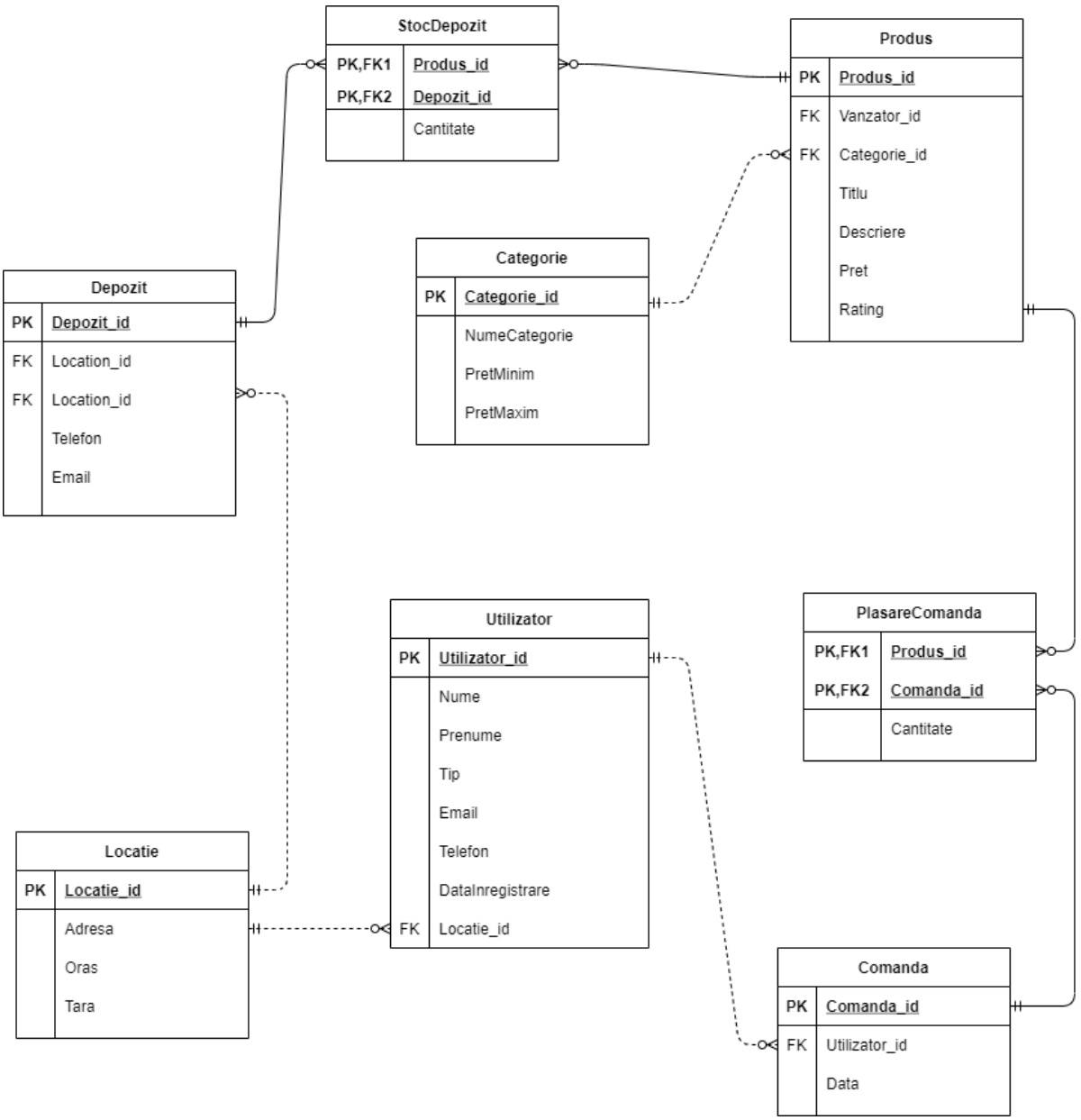
Cantitate: reprezinta numarul de produse cu id-ul Produs_id in stoc, de tip număr.

Cheia primara a tabelei asociative este compusa din cele 2 chei externe Produs_id și Depozit_id.

Cerința 6 Diagrama E/R



Cerința 7 Diagrama conceptuală



Cerința 8 Scheme relaționale

- Utilizator (#utilizator_id, nume, prenume, tip, email, telefon, datainregistrare, locatie_id)
- Comanda (#comanda_id, utilizator_id, data)
- Locație (#locaite_id, adresa, oraș, țara)
- Categorie (#categorie_id, numecategorie, pretminim, pretmaxim)
- Depozit (#depozit_id, locatie_id, telefon, email)

- Produs (#produs_id, vanzator_id, categorie_id, titlu, descriere, preț, rating)

- PlasareComanda (#produs_id, #comanda_id, cantitate)
- StocDepozit (#produs_id, #depozit_id, cantitate)

Cerința 9 Normalizarea până în forma normala 3

Modelul realizat la cerința 6 este în forma normal 3. Din acest motiv voie exemplifica transformarile de la fiecare forma normala.

FN1:

O comanda plasată poate aveam mai multe produse diferite, necesitand memorarea id-ului pentru fiecare produs, dar și cantitatea fiecărui produs . Acest lucru ar însemna ca o coloana sa contina mai mult de o valoare.

Comanda

#Comanda_id	Utilizator_id	Data	Produs_id	Cantitate
1	1	15/03/2021	5, 3	2, 1
2	3	14/04/2021	5, 8	3, 15

Pentru a remedia problema am ales sa creez tabela asociativa PlasareComanda care face legatura intre o comanda, un produs si cantitatea ceruta.

Comanda

#Comanda_id	Utilizator_id	Data
1	1	15/03/2021
2	3	14/04/2021

Plasare Comanda

#Produs_id	#Comanda_id	Cantitate
5	1	2
3	1	1
5	2	3
8	2	15

FN2:

Pentru a reține numărul de produse disponibile dintr-un depozit putem creea tabelul depozit astfel incat sa avem o cheie compusă din atributele depozit_id și produs_id. Acest lucru ar însemna sa avem un rand pentru fiecare produs, în timp ce restul atributelor ar depinde doar de cheia primara depozit_id, nu si de produs_id.

Depozit

#Depozit_id	Locatie_id	Telefon	#Produs_id	Stoc
1	1	0123654789	5	2
1	1	0123654789	3	1
2	3	0123654789	5	3
2	3	0123654789	8	15

Putem reține stocul într-o tabela asociativă.

Depozit

#Depozit_id	Locatie_id	Telefon
1	1	0123654789
2	3	0123654789

Stoc

#Depozit_id	#Produs_id	Stoc
1	5	2
1	3	1
2	5	3
2	8	15

FN3:

Produsele vandute sunt împărțite pe categorii. Pentru fiecare produs am putea reține categoria din care face parte, prețul celui mai ieftin și celui mai scump produs din categoria respectivă, însă acest lucru devine problematic cand este introdus un nou produs al cărui preț este mai mic sau mai mare decât valorile salvate până în acel moment. Pentru a modifica valorile ar însemna sa facem o operație de update pe fiecare rand din tabelul produs. De asemenea prețul minim și cel maxim sunt dependente de categoria unui produs, dar acesta nu este cheie primara.

Produs

#Produs_id	Titlu	Categorie	Preț minim	Preț maxim
1	A	Tech	100	2500
2	B	Incaltaminte	80	500
3	C	Tech	100	2500
4	D	Tech	100	2500

Pentru a rezolva problema am adaugat un tabel Categorie în care reţin numele, preţul minim şi preţul maxim, iar în tabelul produs exista o cheie externă categorie_id care indică din ce categorie face parte produsul.

Produs

#Produs_id	Titlu	Categorie_id
1	A	1
2	B	2
3	C	1
4	D	1

Categorie

#Categorie_id	Nume Categorie	Preţ minim	Preţ maxim
1	Tech	100	2500
2	Incaltaminte	80	500

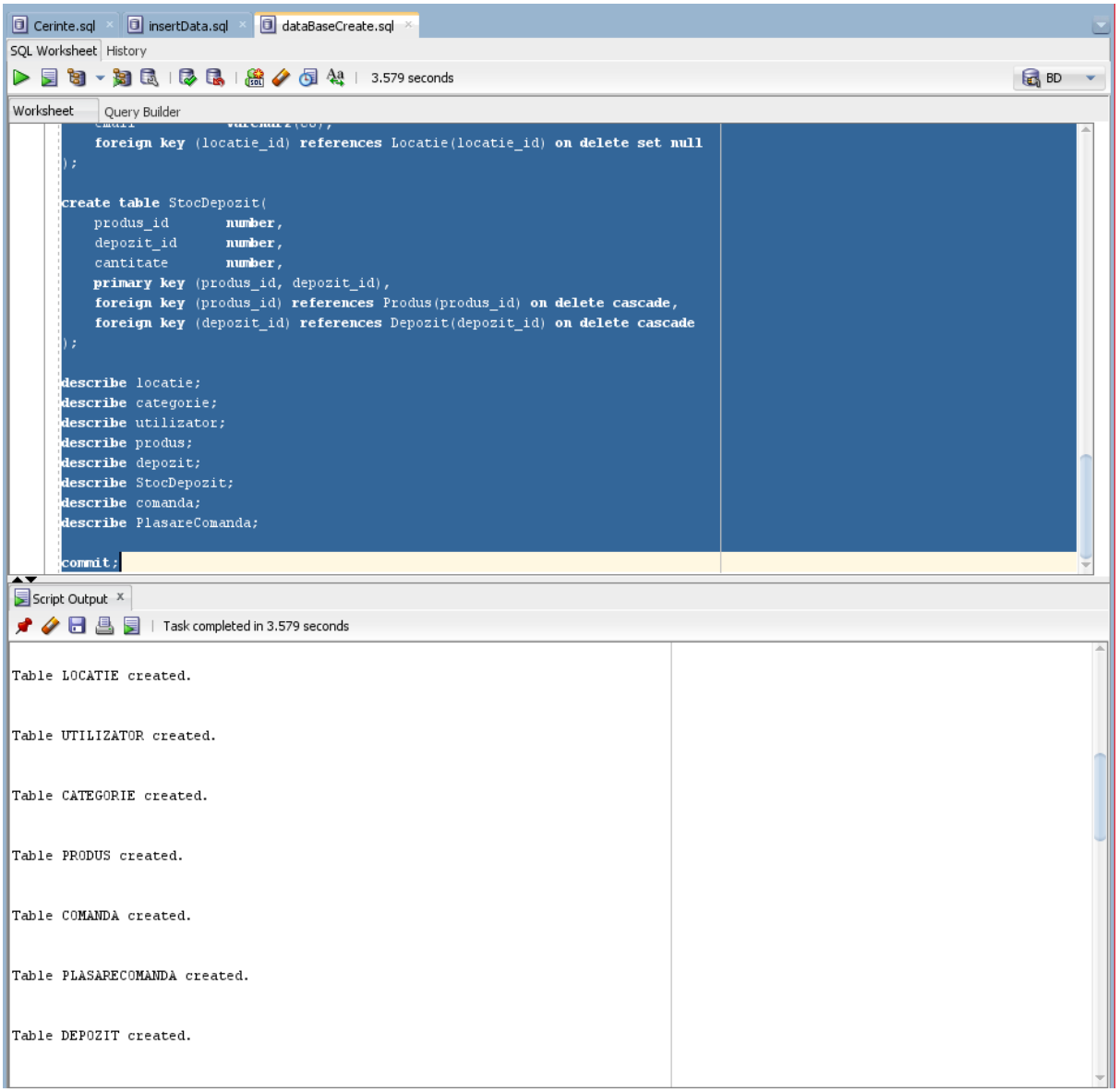
Cerinţa 10 Crearea tabelelor şi inserare datelor

Crearea tabelelor

```
-- Creare tabele --
drop table Utilizator cascade constraints;
drop table Depozit cascade constraints;
drop table StocDepozit cascade constraints;
drop table Produs cascade constraints;
drop table Locatie cascade constraints;
drop table Categorie cascade constraints;
drop table Comanda cascade constraints;
drop table PlasareComanda cascade constraints;

create table Locatie(
      locatie_id number primary key,
      adresa     varchar2(100),
```

```
      oras       varchar2(100),
      tara     varchar2(100)
);

create table Utilizator(
      utilizator_id           number primary key,
      nume                    varchar2(20) not null,
      prenume                       varchar2(20) not null,
      tip                     varchar2(20),
      email                   varchar2(60),
      telefon                       varchar2(10),
      DataInregistrare  date,
      locatie_id              number not null,
      foreign key (locatie_id) references Locatie(locatie_id) on delete
set null
);

create table Categorie(
      categorie_id      number primary key,
      numeCategorie     varchar2(50),
      PretMinim         number,
      pretMaxim         number
);

create table Produs(
      produs_id         number primary key,
      vanzator_id       number not null,
      categorie_id      number not null,
      titlu             varchar2(200),
      descriere         varchar2(3000),
      pret              number(10, 2),
      rating                  number(2, 1),
      foreign key (vanzator_id) references Utilizator(utilizator_id) on
delete cascade,
      foreign key (categorie_id) references Categorie(categorie_id) on
delete cascade
);

create table Comanda(
      comanda_id        number primary key,
      utilizator_id     number not null,
      data              date,
      foreign key (utilizator_id) references Utilizator(utilizator_id)
on delete cascade
);
```

```
create table PlasareComanda(
      produs_id         number,
      comanda_id        number,
      cantitate         number,
      primary key (produs_id, comanda_id),
      foreign key (produs_id) references Produs(produs_id) on delete
cascade,
      foreign key (comanda_id) references Comanda(comanda_id) on delete
cascade
);

create table Depozit(
      depozit_id        number primary key,
      locatie_id        number not null,
      telefon                 varchar2(10),
      email             varchar2(60),
      foreign key (locatie_id) references Locatie(locatie_id) on delete
set null
);

create table StocDepozit(
      produs_id         number,
      depozit_id        number,
      cantitate         number,
      primary key (produs_id, depozit_id),
      foreign key (produs_id) references Produs(produs_id) on delete
cascade,
      foreign key (depozit_id) references Depozit(depozit_id) on delete
cascade
);

describe locatie;
describe categorie;
describe utilizator;
describe produs;
describe depozit;
describe StocDepozit;
describe comanda;
describe PlasareComanda;

commit;
```

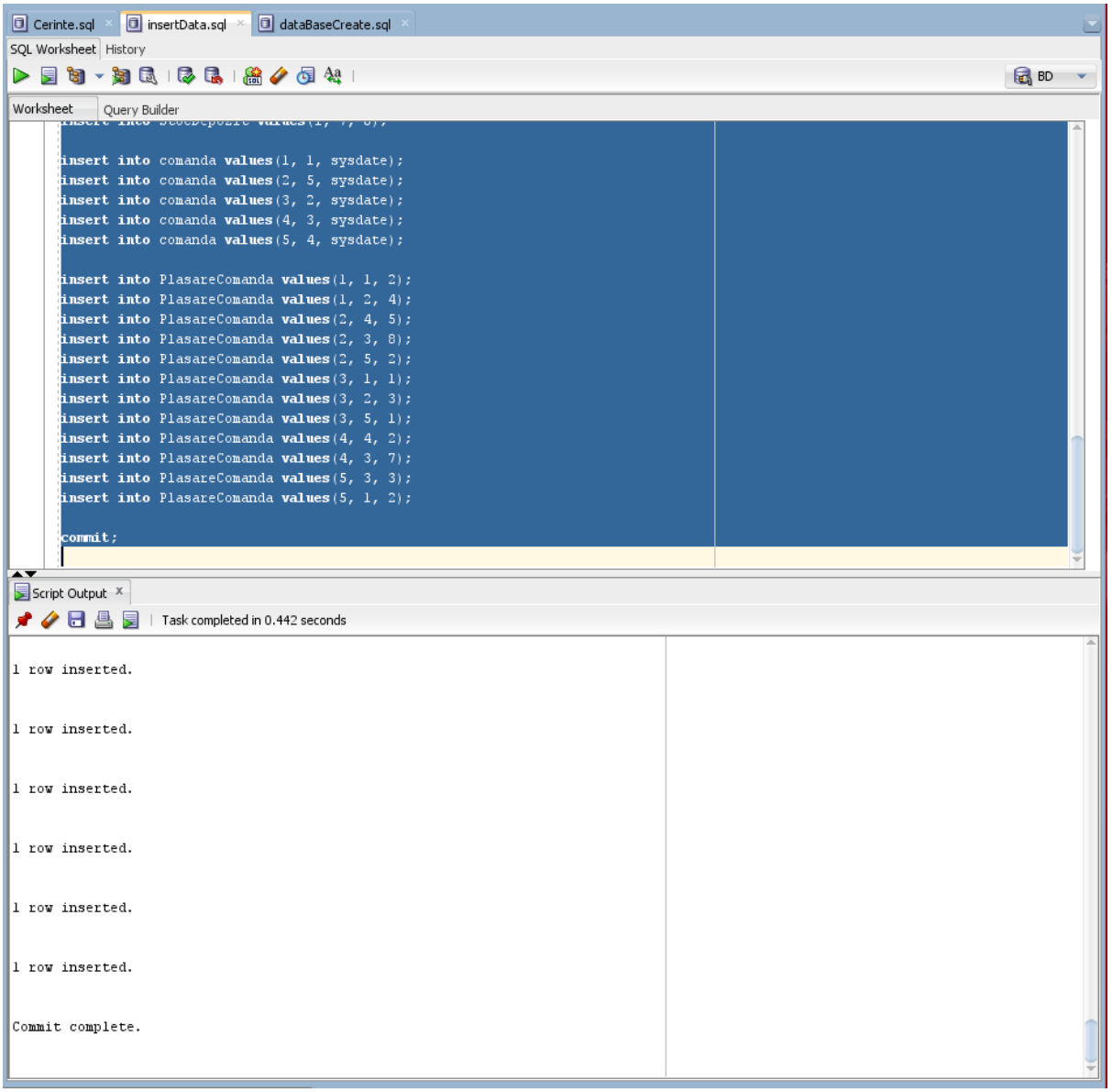


Inserarea datelor

```
-- Inserare date --
insert into locatie values(1, 'Splaiul Independentei 204', 'Bucuresti',
'Romania');
insert into locatie values(2, 'Bulevardul Iuliu Maniua 104',
'Bucuresti', 'Romania');
insert into locatie values(3, 'Strada Zavoiului 39', 'Chisinau',
'Republica Moldova');
insert into locatie values(4, 'Bergen Strasse 3', 'Berlin', 'Germania');
insert into locatie values(5, 'Bulevardul Alba Iulia 69', 'Cluj-Napoca',
'Romania');

insert into depozit values(1, 1, '0747111111', 'depozit1@email.com');
```

```
insert into depozit values(2, 2, '0737222222', 'depozit2@email.com');
insert into depozit values(3, 4, '0737222322', 'depozit3@email.com');
insert into depozit values(4, 5, '0737222422', 'depozit4@email.com');
insert into depozit values(5, 3, '0737252222', 'depozit5@email.com');
insert into depozit values(6, 4, '0737252226', 'depozit6@email.com');
insert into depozit values(7, 4, '0737252227', 'depozit7@email.com');

insert into utilizator values(1, 'Ion', 'Popescu', 'Administrator',
'ion.popescu@email.com', '0737111111', sysdate, 4);
insert into utilizator values(2, 'Gicu', 'Gigel', 'Utilizator',
'gicu.gigel@email.com', '0737222222', sysdate, 3);
insert into utilizator values(3, 'Dorel', 'Dori', 'Partener',
'dorel.dori@email.com', '0737333333', sysdate, 1);
insert into utilizator values(4, 'Teo', 'Costel', 'Utilizator',
'teo.costel@email.com', '0737204222', sysdate, 4);
insert into utilizator values(5, 'Ana', 'Maria', 'Utilizator',
'ana.maria@email.com', '0737202202', sysdate, 2);
insert into utilizator values(6, 'Popescu', 'Stefania', null,
'popescu.stefania@email.com', '0737202252', sysdate, 1);

insert into categorie values(1, 'Tech', null, null);
insert into categorie values(2, 'Religie', null, null);
insert into categorie values(3, 'Pantaloni', null, null);
insert into categorie values(4, 'Incaltaminte', null, null);
insert into categorie values(5, 'Utilitare', null, null);
insert into categorie values(6, 'Pantofi', null, null);

insert into produs values(1, 3, 1, 'Iphone 115 Pro X Max', 'Max',
1099.99, 5);
insert into produs values(2, 1, 2, 'Biblia', 'Efectiv cartea cartilor',
333, 3);
insert into produs values(3, 3, 4, 'Crose alergat', 'Adidas cel mai bun
pentru alergat, tripaloski', 699, 4);
insert into produs values(4, 3, 5, 'Banda adeziva', 'Repara orice', 15,
5);
insert into produs values(5, 3, 3, 'Pantaloni negri', 'O pereche de
pantaloni', 49.99, 2.5);
insert into produs values(6, 3, 6, 'Pantofi negri', 'O pereche de
pantofi', 69.99, 3.5);
insert into produs values(7, 3, 6, 'Pantofi maro', 'O pereche de
pantofi', 59.99, 3);
insert into produs values(8, 3, 1, 'Samasug A20e', 'ieftin', 400.00,
3.5);

insert into StocDepozit values(1, 4, 40);
insert into StocDepozit values(1, 5, 50);
```

```
insert into StocDepozit values(1, 2, 100);
insert into StocDepozit values(1, 1, 220);
insert into StocDepozit values(2, 3, 100);
insert into StocDepozit values(2, 2, 10);
insert into StocDepozit values(2, 4, 25);
insert into StocDepozit values(3, 5, 69);
insert into StocDepozit values(3, 2, 420);
insert into StocDepozit values(3, 1, 12);
insert into StocDepozit values(4, 1, 55);
insert into StocDepozit values(4, 2, 69420);
insert into StocDepozit values(4, 3, 34);
insert into StocDepozit values(4, 4, 21);
insert into StocDepozit values(4, 5, 9);
insert into StocDepozit values(5, 3, 8);
insert into StocDepozit values(5, 5, 13);
insert into StocDepozit values(6, 3, 40);
insert into StocDepozit values(7, 2, 30);
insert into StocDepozit values(1, 7, 0);

insert into comanda values(1, 1, sysdate);
insert into comanda values(2, 5, sysdate);
insert into comanda values(3, 2, sysdate);
insert into comanda values(4, 3, sysdate);
insert into comanda values(5, 4, sysdate);

insert into PlasareComanda values(1, 1, 2);
insert into PlasareComanda values(1, 2, 4);
insert into PlasareComanda values(2, 4, 5);
insert into PlasareComanda values(2, 3, 8);
insert into PlasareComanda values(2, 5, 2);
insert into PlasareComanda values(3, 1, 1);
insert into PlasareComanda values(3, 2, 3);
insert into PlasareComanda values(3, 5, 1);
insert into PlasareComanda values(4, 4, 2);
insert into PlasareComanda values(4, 3, 7);
insert into PlasareComanda values(5, 3, 3);
insert into PlasareComanda values(5, 1, 2);

commit;
```

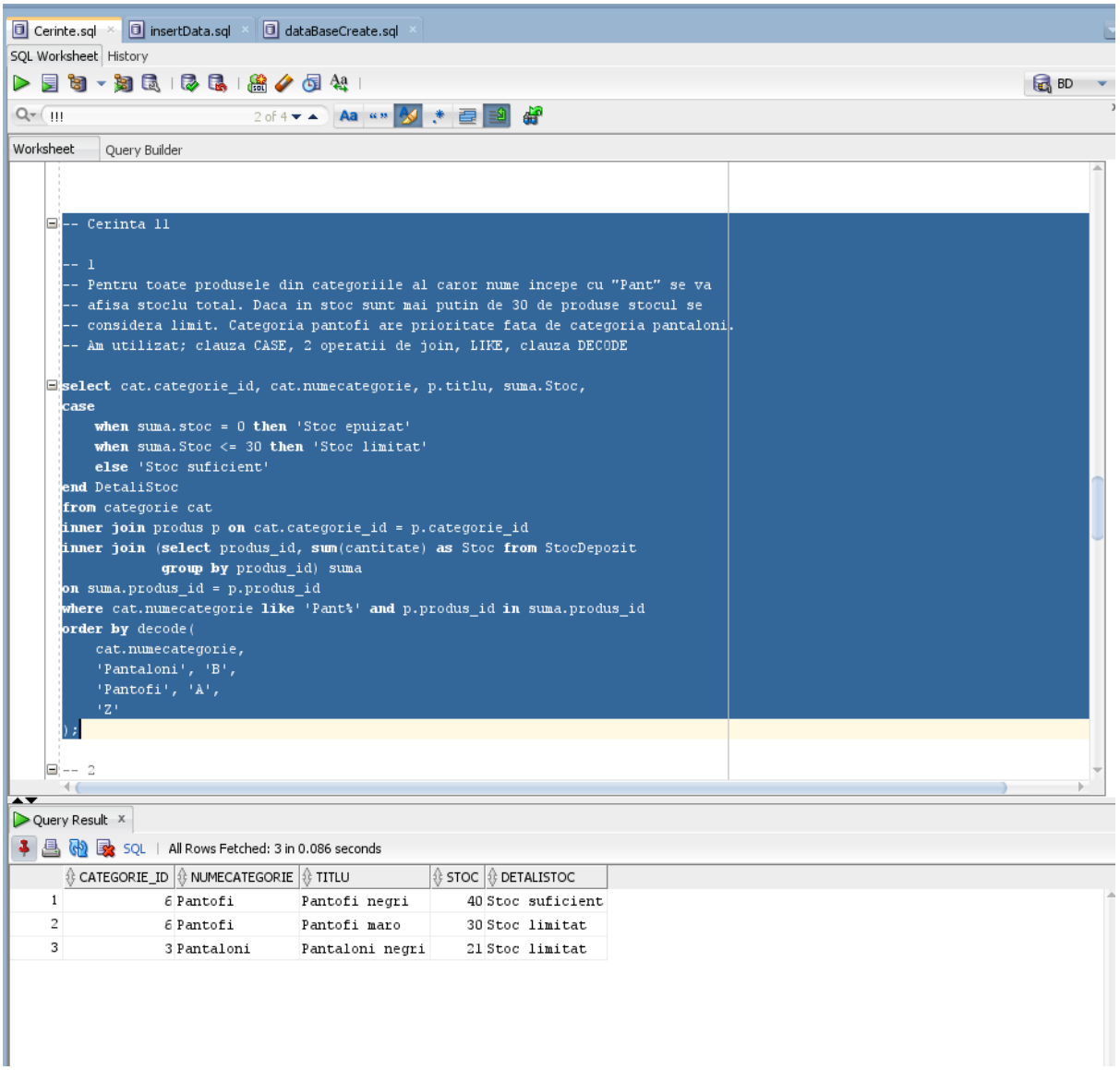


Cerința 11 Cereri SQL complexe

1.

```
-- 1
-- Pentru toate produsele din categoriile al caror nume incepe cu "Pant"
se va
-- afisa stoclu total. Daca in stoc sunt mai putin de 30 de produse
stocul se
-- considera limit. Categoria pantofi are prioritate fata de categoria
pantaloni.
-- Am utilizat; clauza CASE, 2 operatii de join, LIKE, clauza DECODE
```

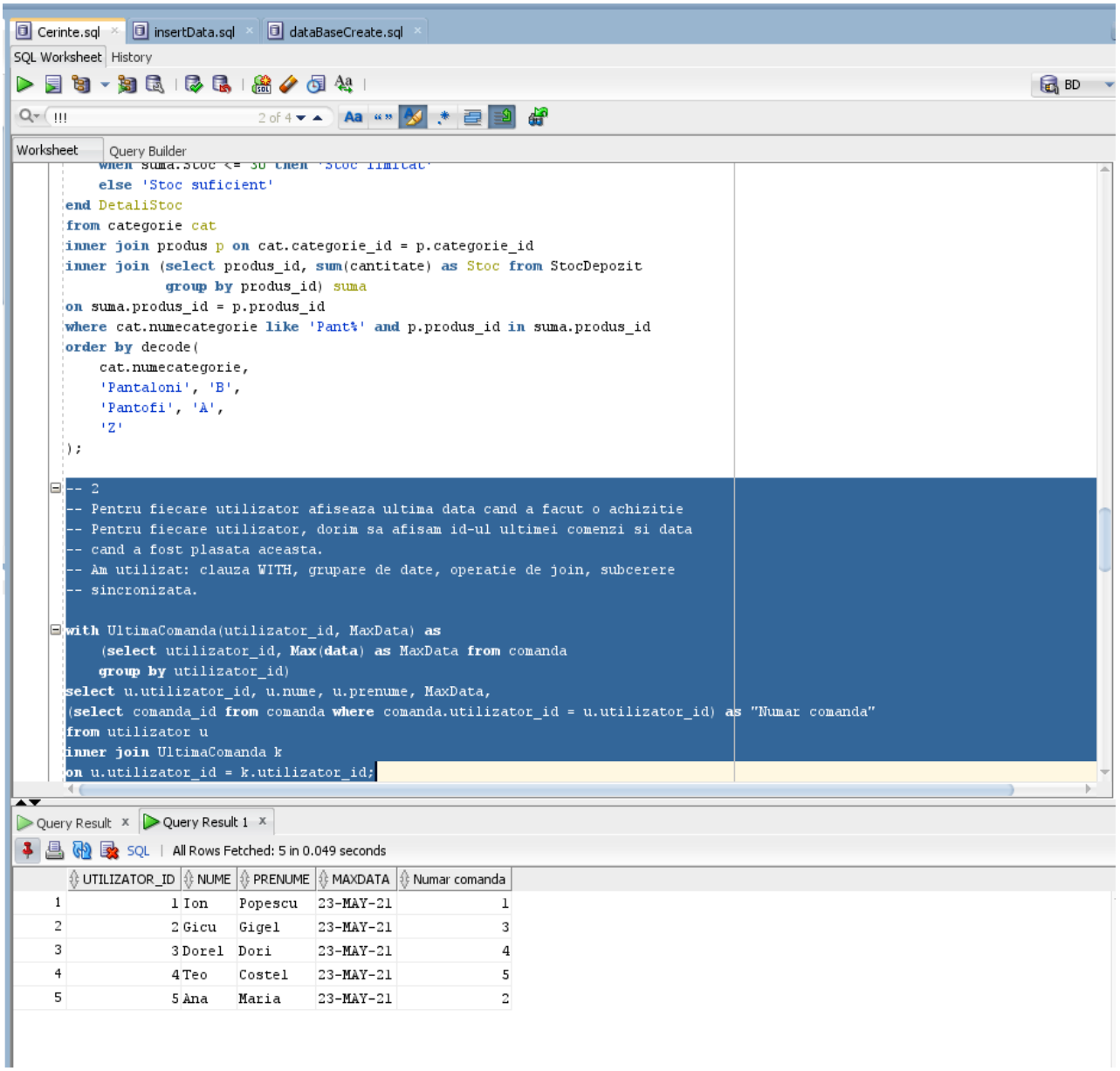
```sql
select cat.categorie_id, cat.numecategorie, p.titlu, suma.Stoc,
case
    when suma.stoc = 0 then 'Stoc epuizat'
    when suma.Stoc <= 30 then 'Stoc limitat'
    else 'Stoc suficient'
end DetaliStoc
from categorie cat
inner join produs p on cat.categorie_id = p.categorie_id
inner join (select produs_id, sum(cantitate) as Stoc from StocDepozit
            group by produs_id) suma
on suma.produs_id = p.produs_id
where cat.numecategorie like 'Pant%' and p.produs_id in suma.produs_id
order by decode(
    cat.numecategorie,
    'Pantaloni', 'B',
    'Pantofi', 'A',
    'Z'
);
```



2.

```
-- 2
-- Pentru fiecare utilizator afiseaza ultima data cand a facut o
achizitie
-- Pentru fiecare utilizator, dorim sa afisam id-ul ultimei comenzi si
data
-- cand a fost plasata aceasta.
-- Am utilizat: clauza WITH, grupare de date, operatie de join,
subcerere
-- sincronizata.

with UltimaComanda(utilizator_id, MaxData) as
    (select utilizator_id, Max(data) as MaxData from comanda
    group by utilizator_id)
```

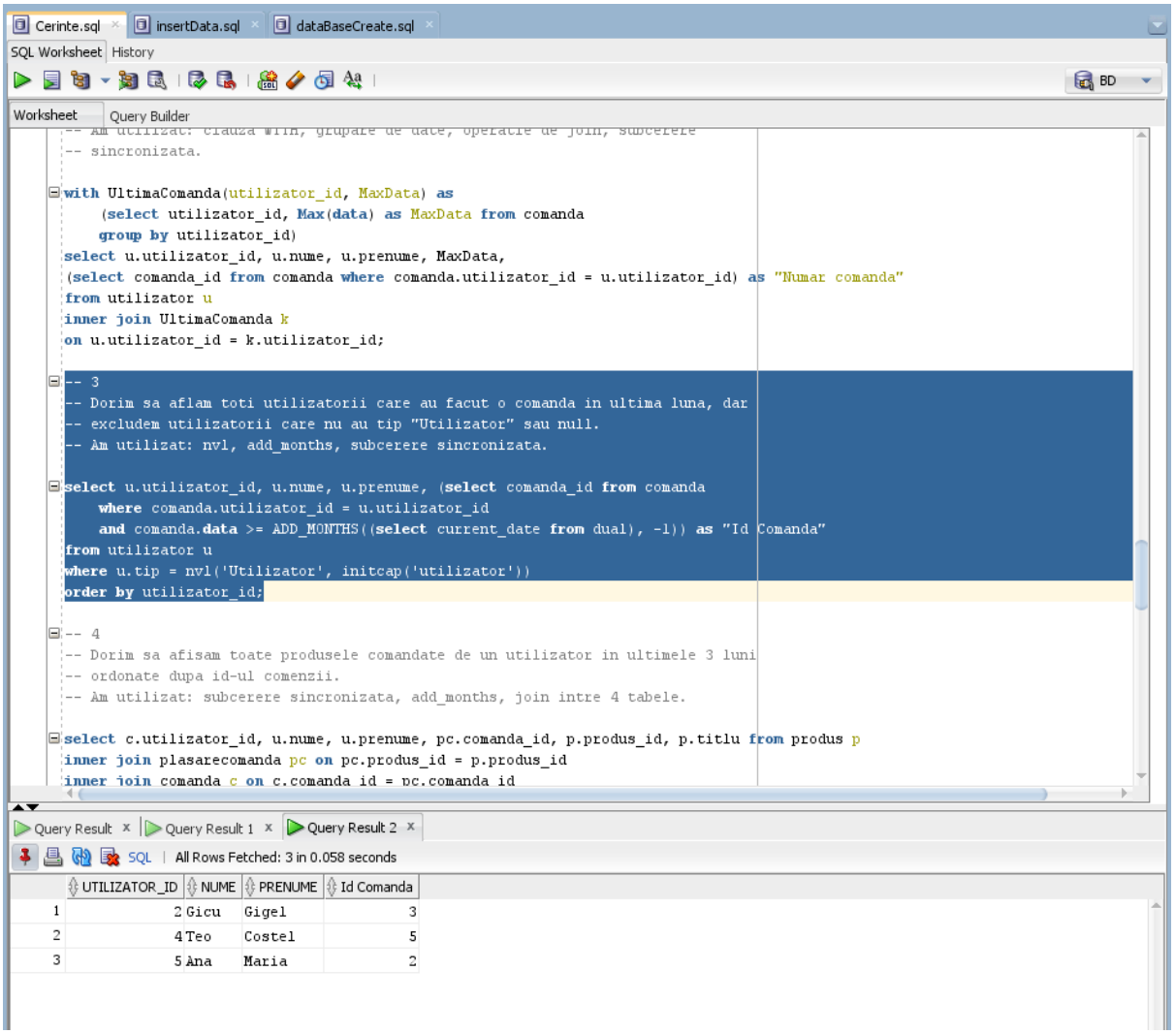
```
select u.utilizator_id, u.nume, u.prenume, MaxData,
(select comanda_id from comanda where comanda.utilizator_id =
u.utilizator_id) as "Numar comanda"
from utilizator u
inner join UltimaComanda k
on u.utilizator_id = k.utilizator_id;
```



3.

```
-- 3
-- Dorim sa aflam toti utilizatorii care au facut o comanda in ultima
luna, dar
-- excludem utilizatorii care nu au tip "Utilizator" sau null.
```

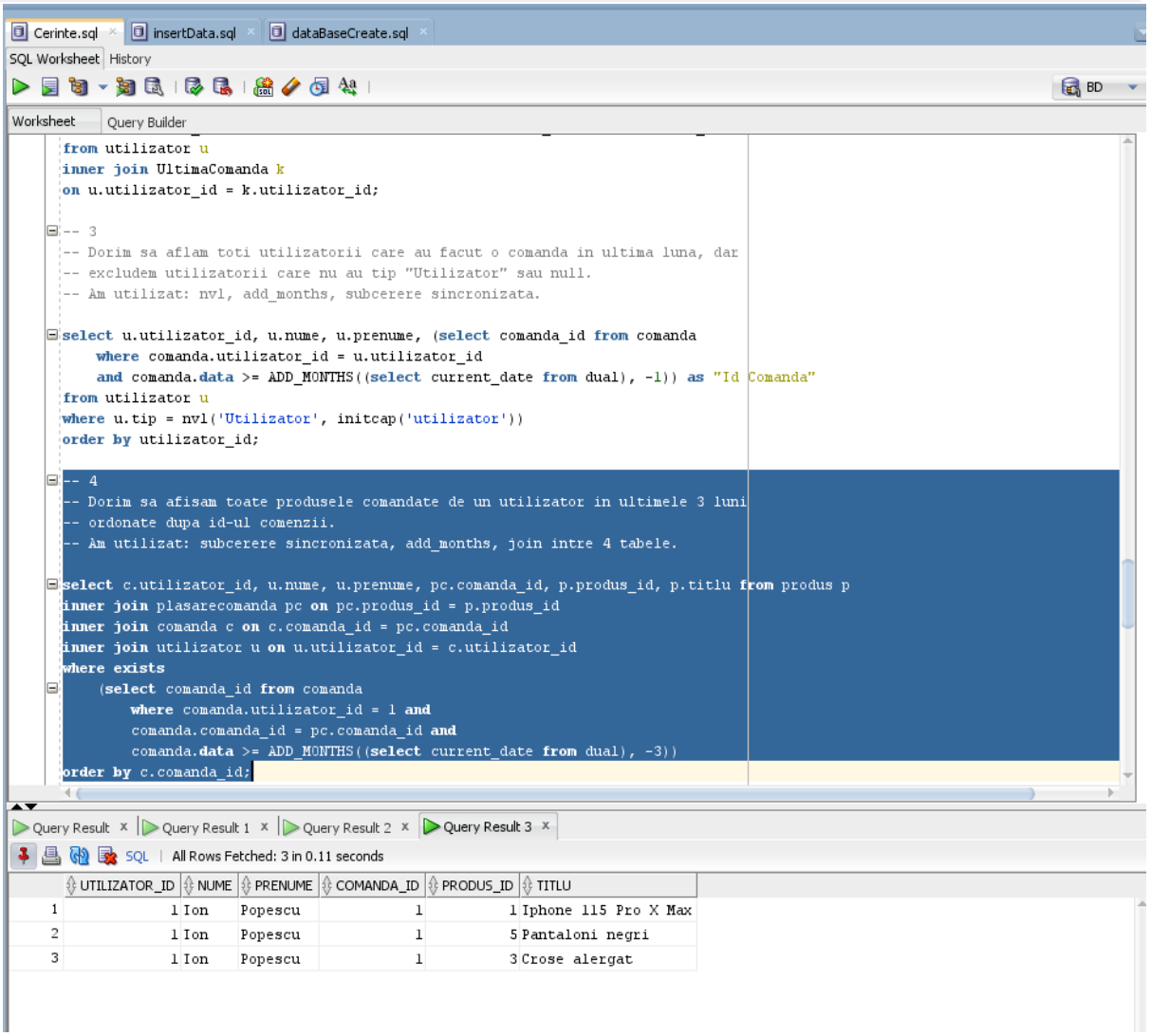
```
-- Am utilizat: nvl, add_months, subcerere sincronizata.

select u.utilizator_id, u.nume, u.prenume, (select comanda_id from
comanda
    where comanda.utilizator_id = u.utilizator_id
    and comanda.data >= ADD_MONTHS((select current_date from dual), -1))
as "Id Comanda"
from utilizator u
where u.tip = nvl('Utilizator', initcap('utilizator'))
order by utilizator_id;
```



4.

```
-- 4
-- Dorim sa afisam toate produsele comandate de un utilizator in
ultimele 3 luni
-- ordonate dupa id-ul comenzii.
```

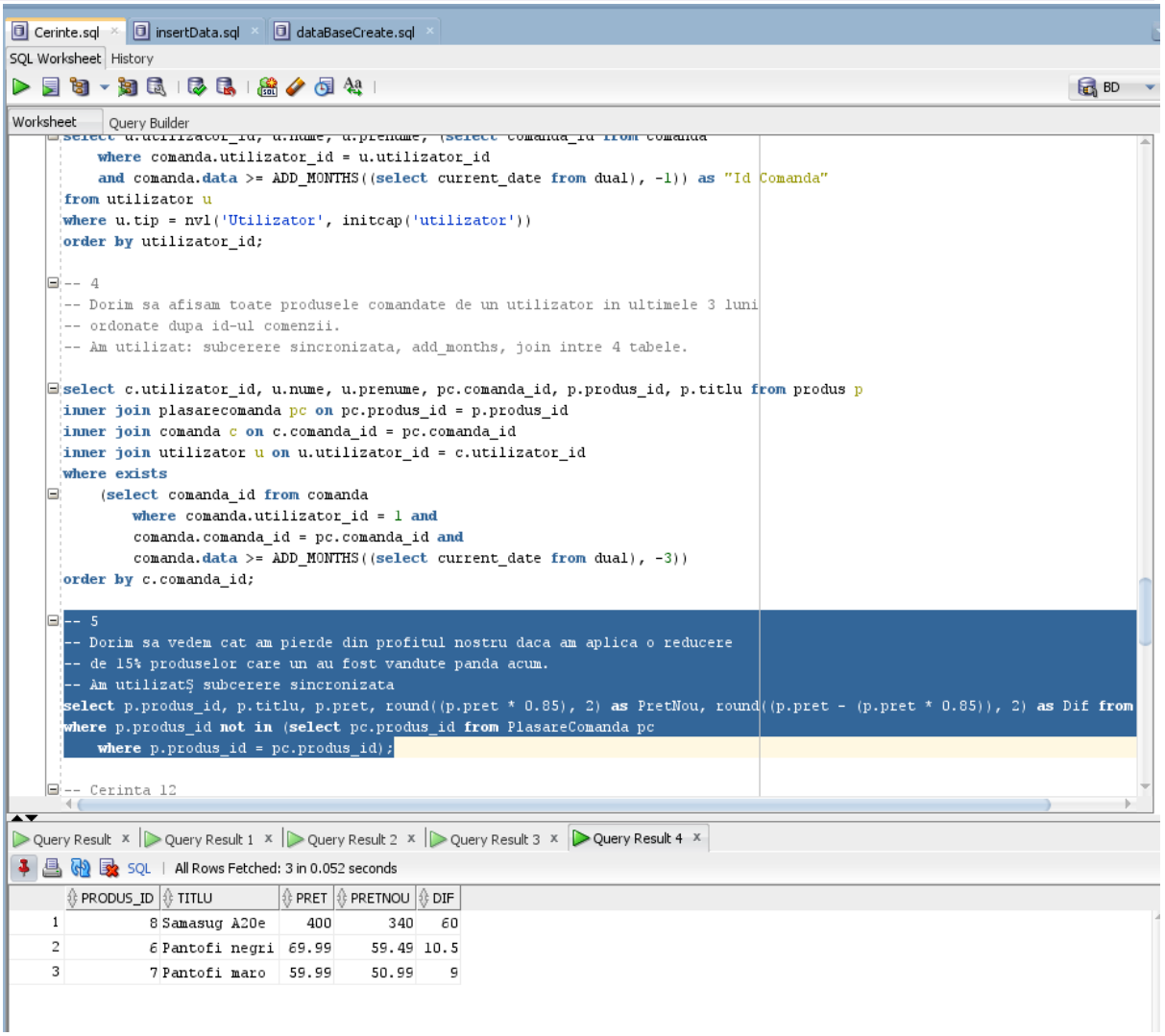
```sql
-- Am utilizat: subcerere sincronizata, add_months, join intre 4 tabele.

select c.utilizator_id, u.nume, u.prenume, pc.comanda_id, p.produs_id,
p.titlu from produs p
inner join plasarecomanda pc on pc.produs_id = p.produs_id
inner join comanda c on c.comanda_id = pc.comanda_id
inner join utilizator u on u.utilizator_id = c.utilizator_id
where exists
    (select comanda_id from comanda
        where comanda.utilizator_id = 1 and
        comanda.comanda_id = pc.comanda_id and
        comanda.data >= ADD_MONTHS((select current_date from dual), -3))
order by c.comanda_id;
```



5.

```sql
-- 5
```

```
-- Dorim sa vedem cat am pierde din profitul nostru daca am aplica o
reducere
-- de 15% produselor care un au fost vandute panda acum.
-- Am utilizatȘ subcerere sincronizata
select p.produs_id, p.titlu, p.pret, round((p.pret * 0.85), 2) as
PretNou, round((p.pret - (p.pret * 0.85)), 2) as Dif from produs p
where p.produs_id not in (select pc.produs_id from PlasareComanda pc
    where p.produs_id = pc.produs_id);
```

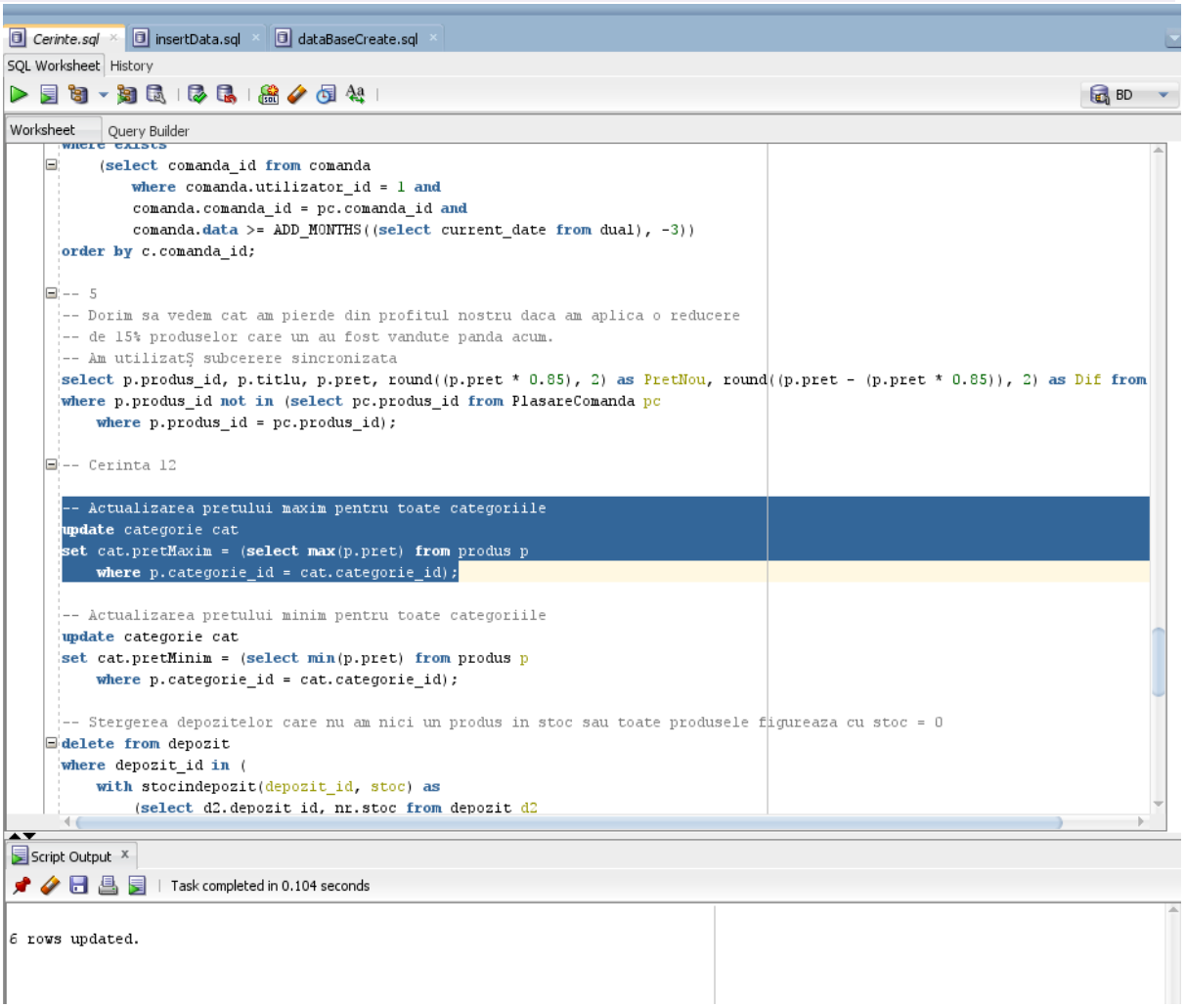


Cerința 12 Operații de actualizare sau suprimare a datelor

1.

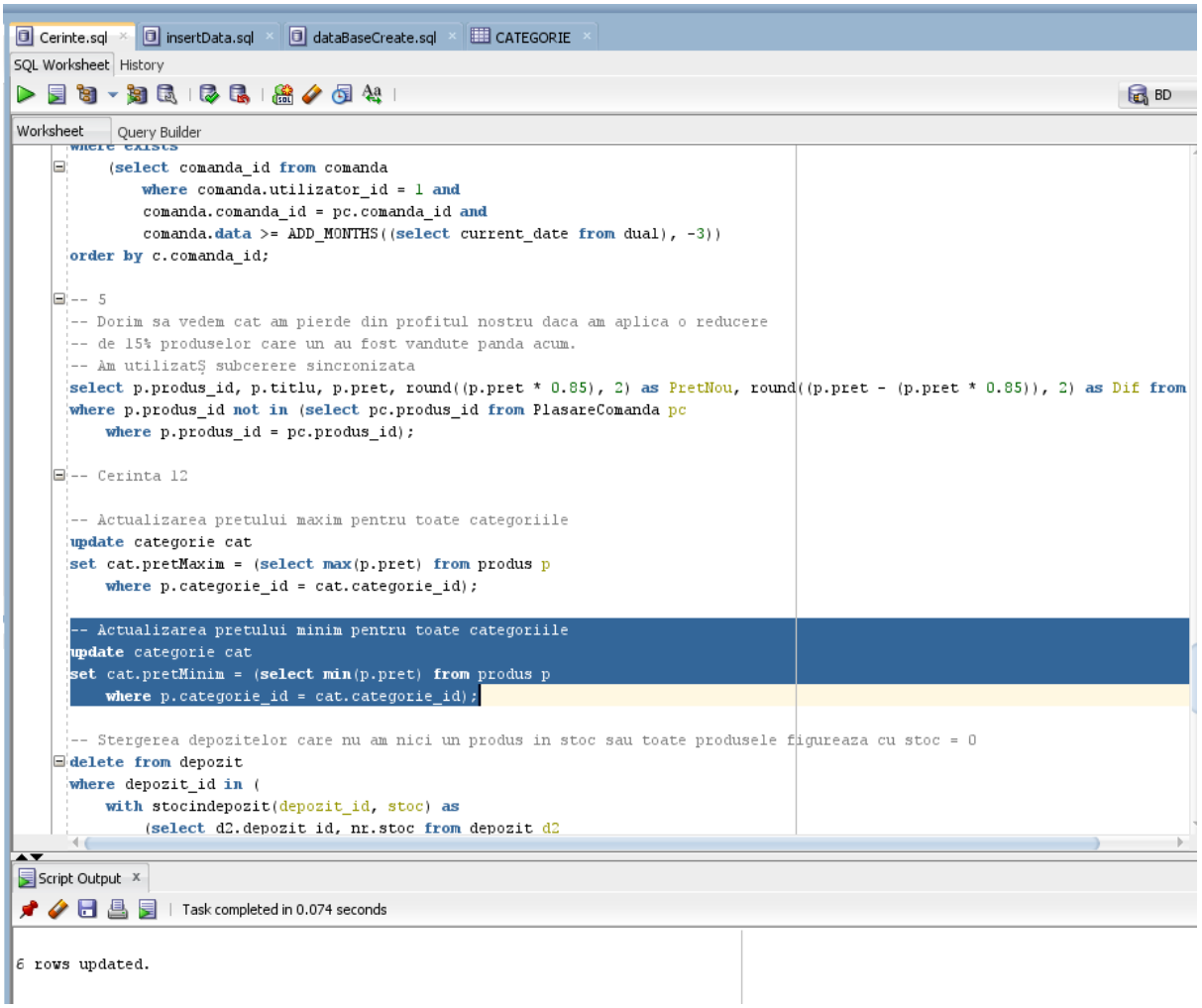
```
-- Actualizarea pretului maxim pentru toate categoriile
update categorie cat
set cat.pretMaxim = (select max(p.pret) from produs p
```

```
    where p.categorie_id = cat.categorie_id);
```



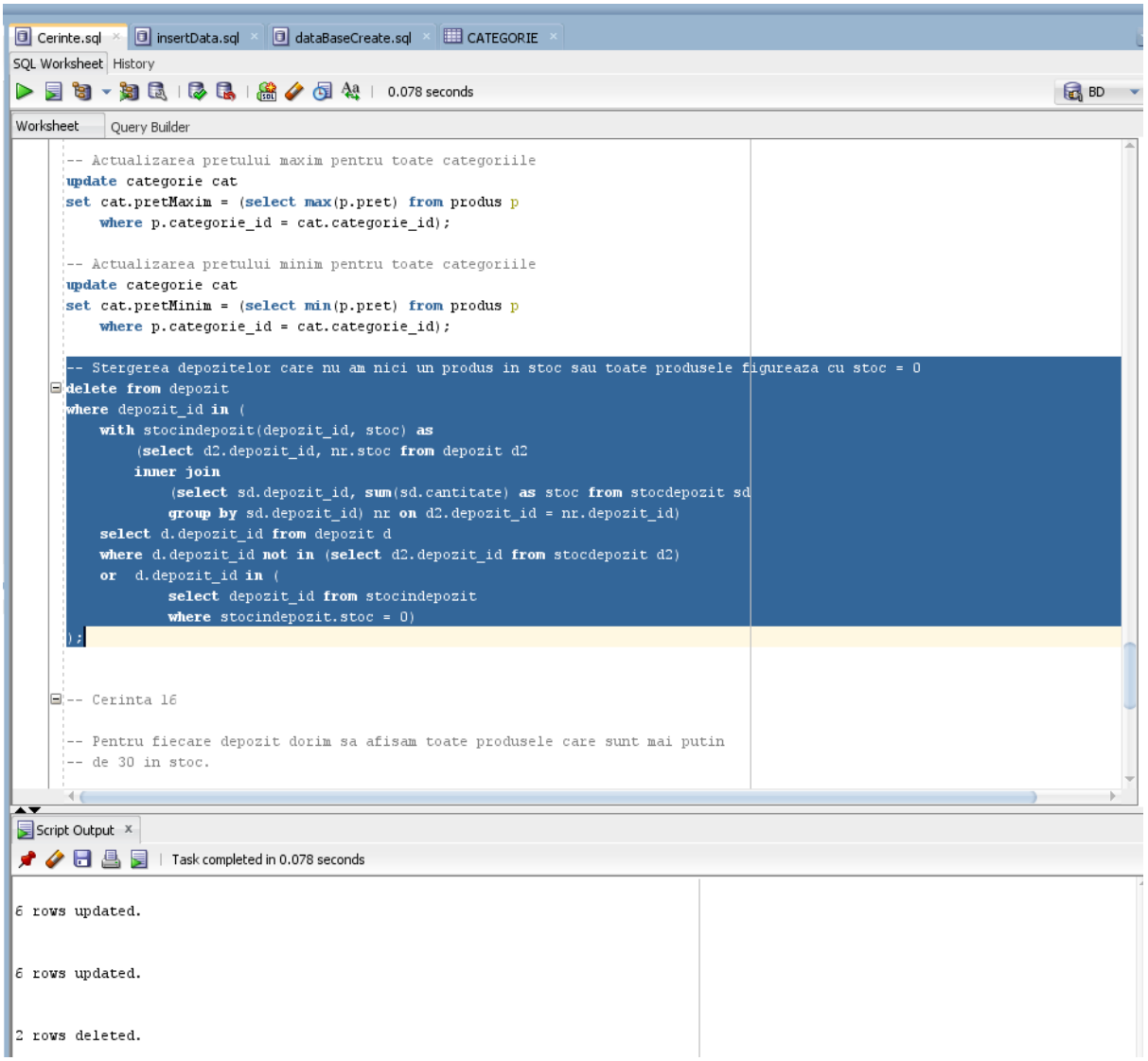
2.

```
-- Actualizarea pretului minim pentru toate categoriile
update categorie cat
set cat.pretMinim = (select min(p.pret) from produs p
    where p.categorie_id = cat.categorie_id);
```



3.

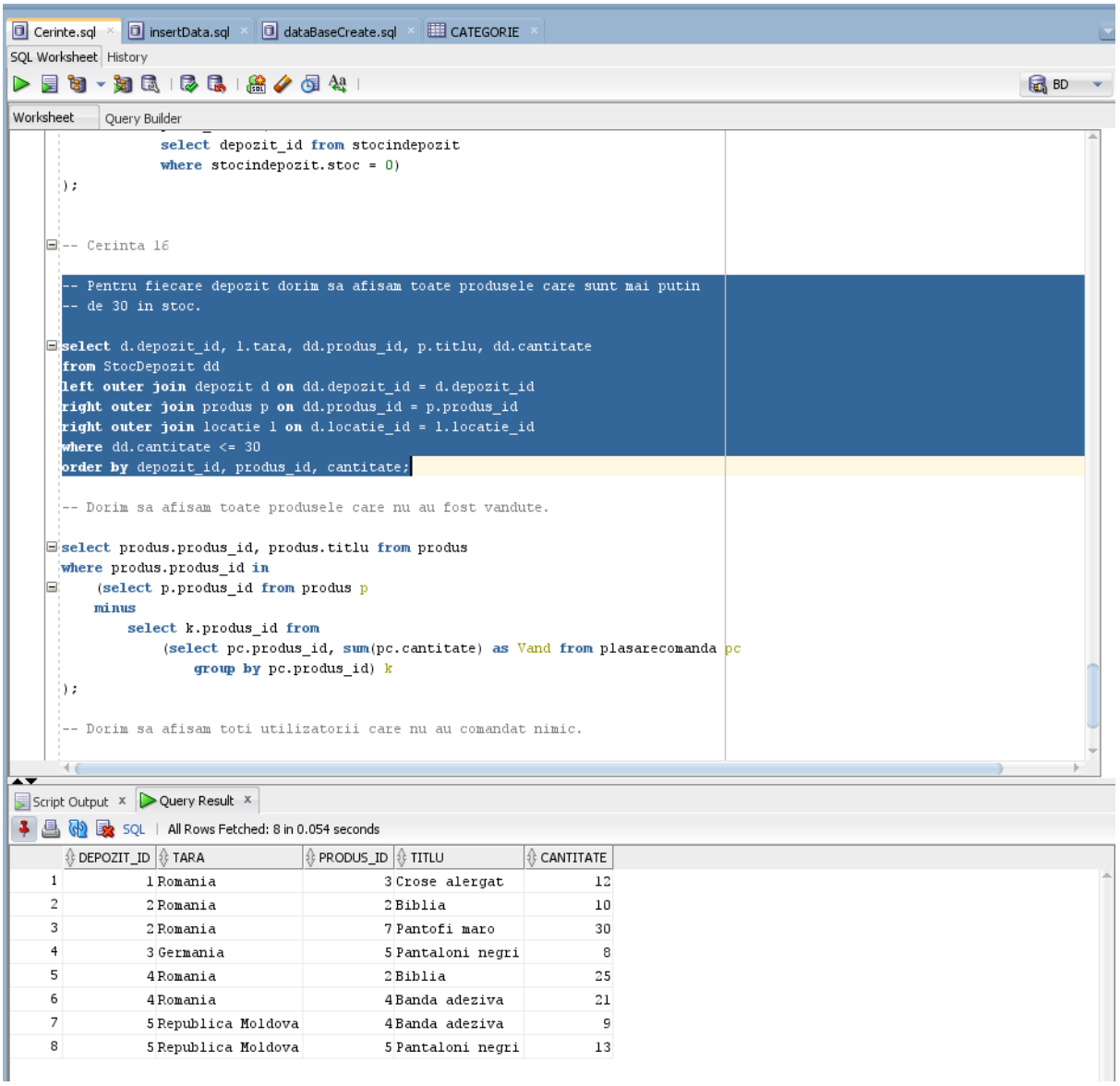
```
-- Stergerea depozitelor care nu am nici un produs in stoc sau toate
produsele figureaza cu stoc = 0
delete from depozit
where depozit_id in (
    with stocindepozit(depozit_id, stoc) as
        (select d2.depozit_id, nr.stoc from depozit d2
        inner join
            (select sd.depozit_id, sum(sd.cantitate) as stoc from
stocdepozit sd
            group by sd.depozit_id) nr on d2.depozit_id = nr.depozit_id)
    select d.depozit_id from depozit d
    where d.depozit_id not in (select d2.depozit_id from stocdepozit d2)
    or  d.depozit_id in (
            select depozit_id from stocindepozit
            where stocindepozit.stoc = 0));
```



Cerința 16

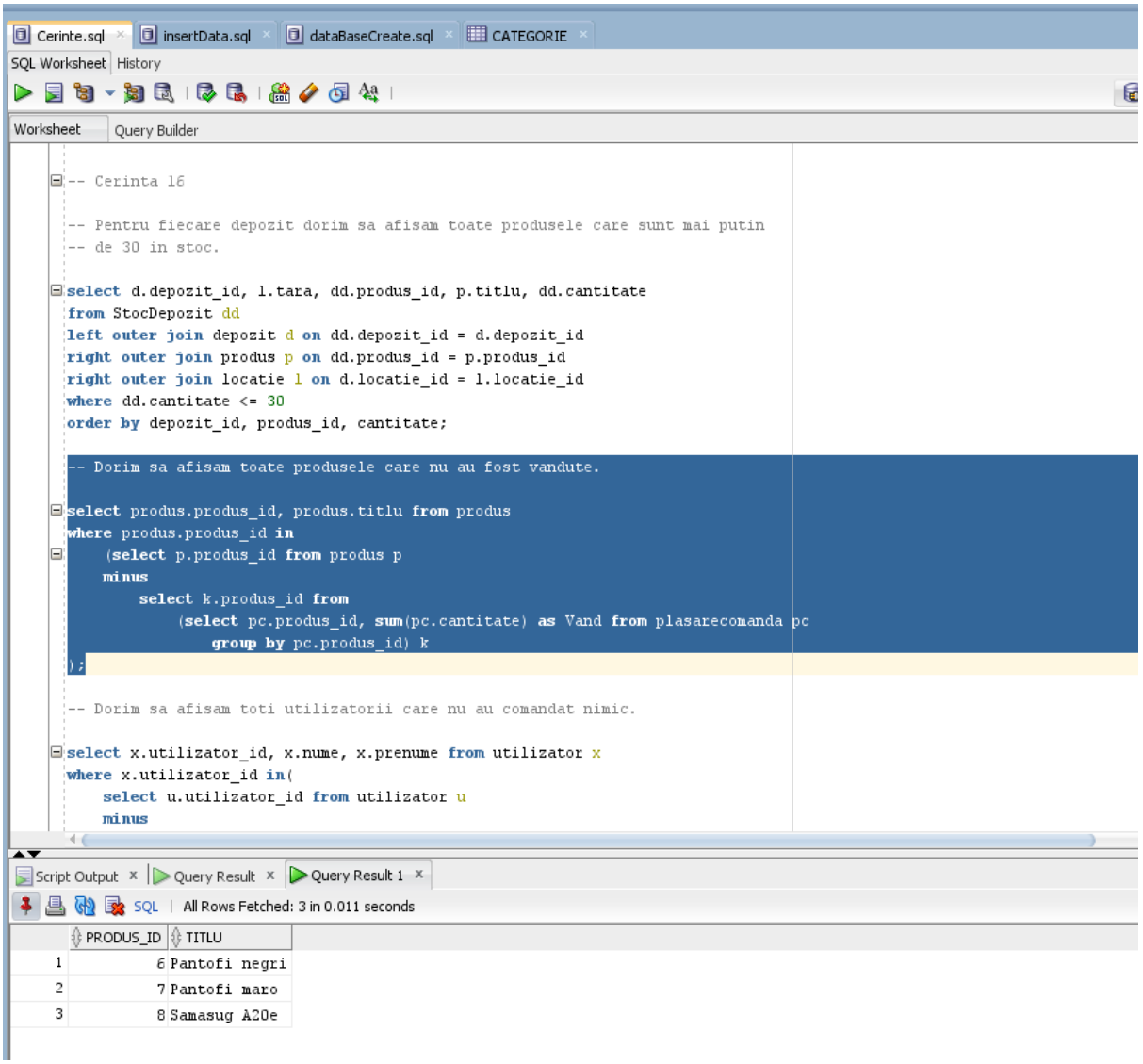
Utilizarea operației de outer join pe 4 tabele

```
-- Pentru fiecare depozit dorim sa afisam toate produsele care sunt mai
putin
-- de 30 in stoc.
select d.depozit_id, l.tara, dd.produs_id, p.titlu, dd.cantitate
from StocDepozit dd
left outer join depozit d on dd.depozit_id = d.depozit_id
right outer join produs p on dd.produs_id = p.produs_id
right outer join locatie l on d.locatie_id = l.locatie_id
where dd.cantitate <= 30
order by depozit_id, produs_id, cantitate;
```

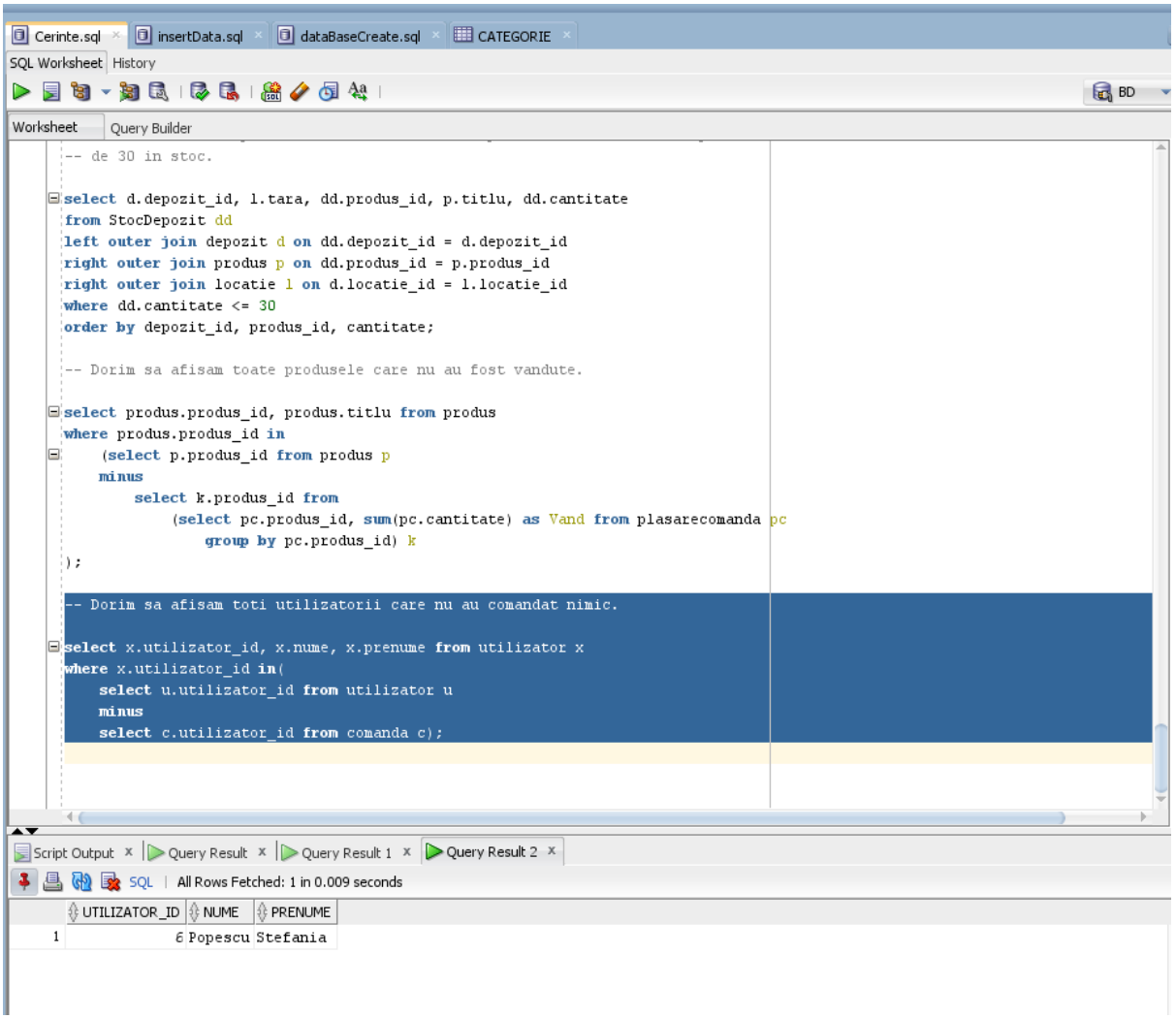


Utilizarea operației division

```
-- Dorim sa afisam toate produsele care nu au fost vandute.
select produs.produs_id, produs.titlu from produs
where produs.produs_id in
    (select p.produs_id from produs p
    minus
        select k.produs_id from
            (select pc.produs_id, sum(pc.cantitate) as Vand from
plasarecomanda pc
                group by pc.produs_id) k);
```




```
-- Dorim sa afisam toti utilizatorii care nu au comandat nimic.
select x.utilizator_id, x.nume, x.prenume from utilizator x
where x.utilizator_id in(
    select u.utilizator_id from utilizator u
    minus
    select c.utilizator_id from comanda c);
```


Cerinte.sql
insertData.sql
dataBaseCreate.sql
CATEGORIE
SQL Worksheet
History
BD
Worksheet
Query Builder
-- de 30 in stoc.
select d.depozit_id, l.tara, dd.produs_id, p.titlu, dd.cantitate
from StocDepozit dd
left outer join depozit d on dd.depozit_id = d.depozit_id
right outer join produs p on dd.produs_id = p.produs_id
right outer join locatie l on d.locatie_id = l.locatie_id
where dd.cantitate <= 30
order by depozit_id, produs_id, cantitate;
-- Dorim sa afisam toate produsele care nu au fost vandute.
select produs.produs_id, produs.titlu from produs
where produs.produs_id in
(select p.produs_id from produs p
minus
select k.produs_id from
(select pc.produs_id, sum(pc.cantitate) as Vand from plasarecomanda pc
group by pc.produs_id) k
);
-- Dorim sa afisam toti utilizatorii care nu au comandat nimic.
select x.utilizator_id, x.nume, x.prenume from utilizator x
where x.utilizator_id in(
select u.utilizator_id from utilizator u
minus
select c.utilizator_id from comanda c);
Script Output
Query Result
Query Result 1
Query Result 2
SQL
All Rows Fetched: 1 in 0.009 seconds
UTILIZATOR_ID
NUME
PRENUME
1
6
Popescu
Stefania