Mihai Andrei - Alexandru

grupa 241  12.01.2023

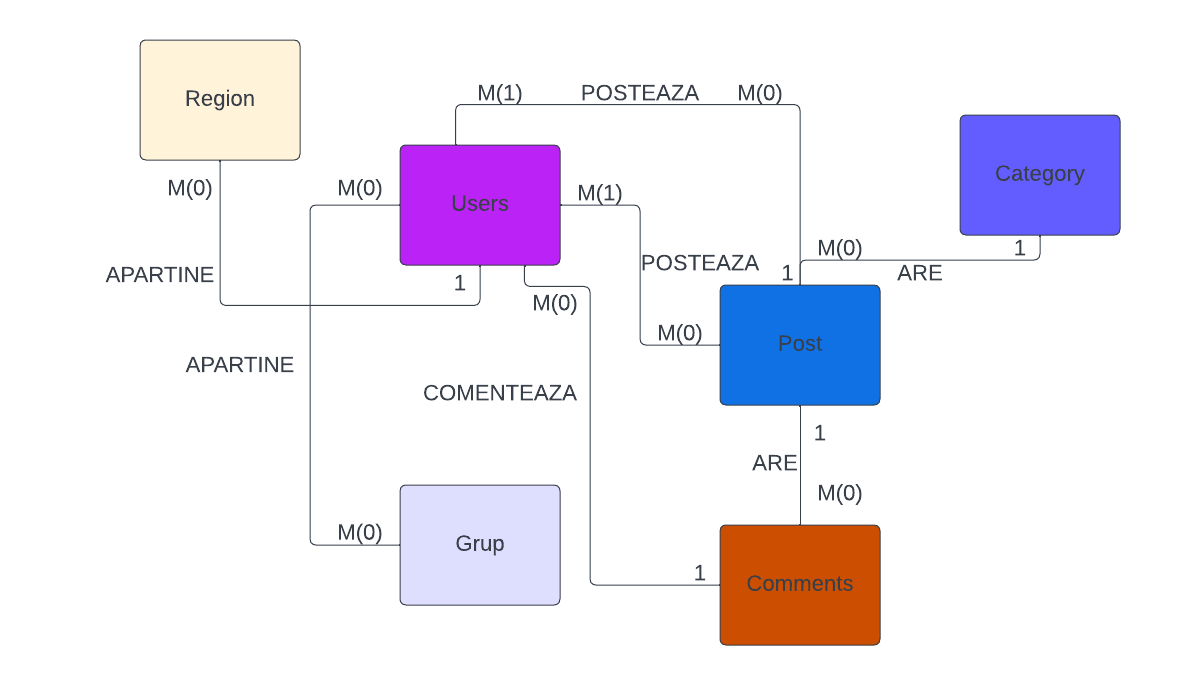
sisteme de gestiune ale bazelor de date - proiect

Cerințe

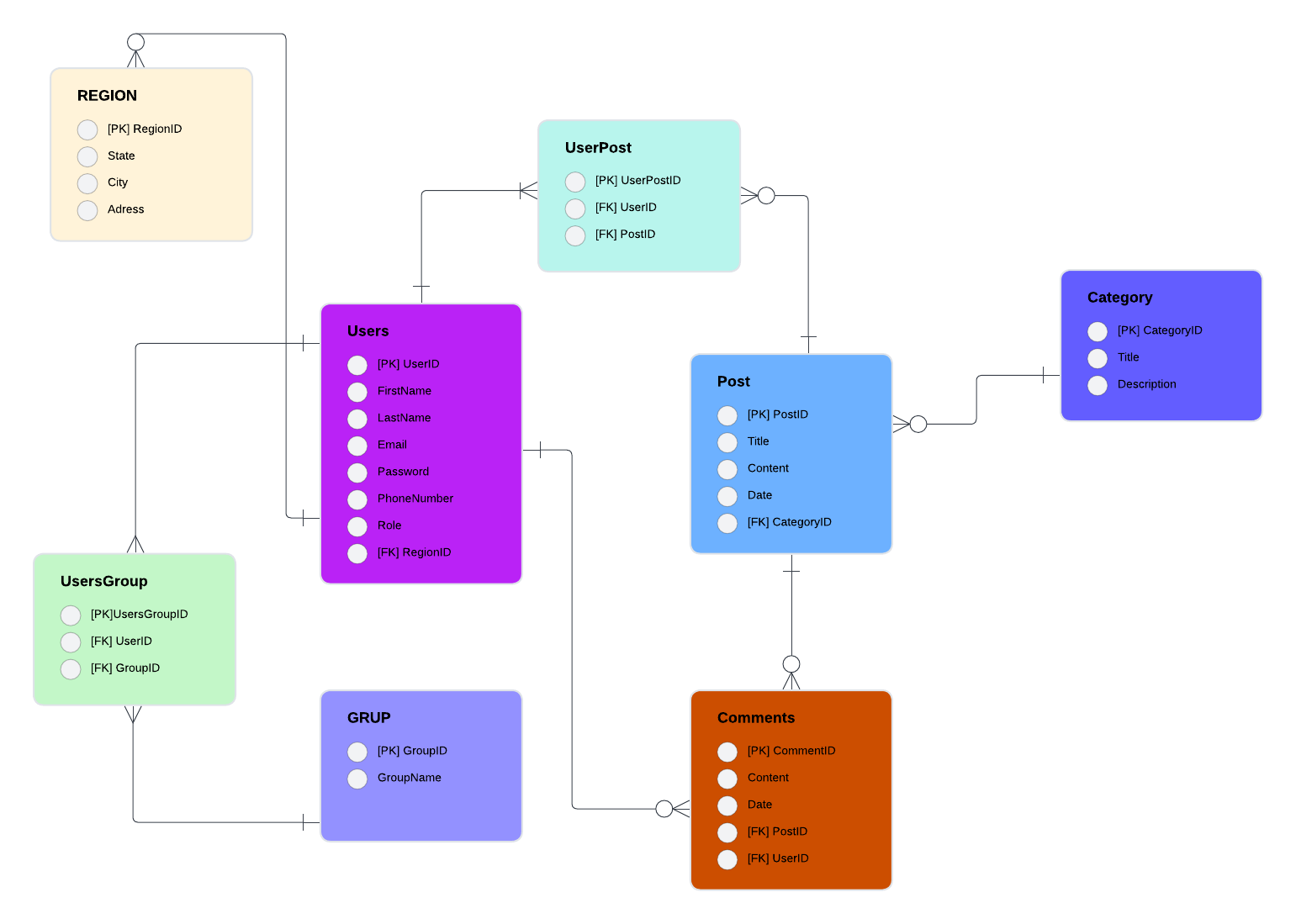
**1. Prezentați pe scurt baza de date (utilitatea ei).**

Baza de date pe care am implementat-o are rol în utilitatea unei platforme de tip „Open Discussion”. Diferiți utilizatori pot adăuga postări noi, pot lăsa comentarii la postări deja existente. Fiecare utilizator are un rol în cadrul administrării site-ului. Postările sunt grupate pe categorii, acestea fiind manageriate de către administratorul platformei. Pot exista si postări colaborative, între mai mulți useri. De asemenea, locația fiecărui user este cunoscută.

**2. Realizați diagrama entitate-relație (ERD).**



**3. Pornind de la diagrama entitate-relație realizați diagrama conceptuală a modelului propus, integrând toate atributele necesare.**

****

**4. Implementați în Oracle diagrama conceptuală realizată: definiți toate tabelele, implementând toate constrângerile de integritate necesare (chei primare, cheile externe etc).**

create table USERS(

UserID integer primary key,

FirstName varchar2(30),

LastName varchar2(30),

Username varchar2(30) not null unique,

Email varchar2(30) unique,

Password varchar2(20),

PhoneNumber varchar2(10) unique,

Role varchar2(10) not null,

check (Role in('Admin','Editor','User'))

);

create table GRUP(

GroupID integer primary key,

GroupName varchar2(20)

);

create table USERSGROUP(

UsersGroupID integer primary key,

UserID integer,

GroupID integer

);

create table CATEGORY(

CategoryID integer primary key,

Title varchar2(101) not null,

Description varchar2(101)

);

create table POST(

PostID integer primary key,

Title varchar2(101) not null,

PostContent varchar2(1000),

PostDate date default sysdate,

CategoryID integer

-- constraint PostCategoryFK foreign key(CategoryID) references CATEGORY(CategoryID),

);

create table COMMENTS(

CommentID integer primary key,

CommentContent varchar2(1000),

CommentDate date default sysdate,

PostID integer,

-- constraint CommentPostFK foreign key(PostID) references Post(PostID),

UserID integer

-- constraint CommentUserFK foreign key(UserID) references USERS(UserID)

);

create table USERSPOST(

UserPostID integer primary key,

UserID integer,

PostID integer

);

create table REGION(

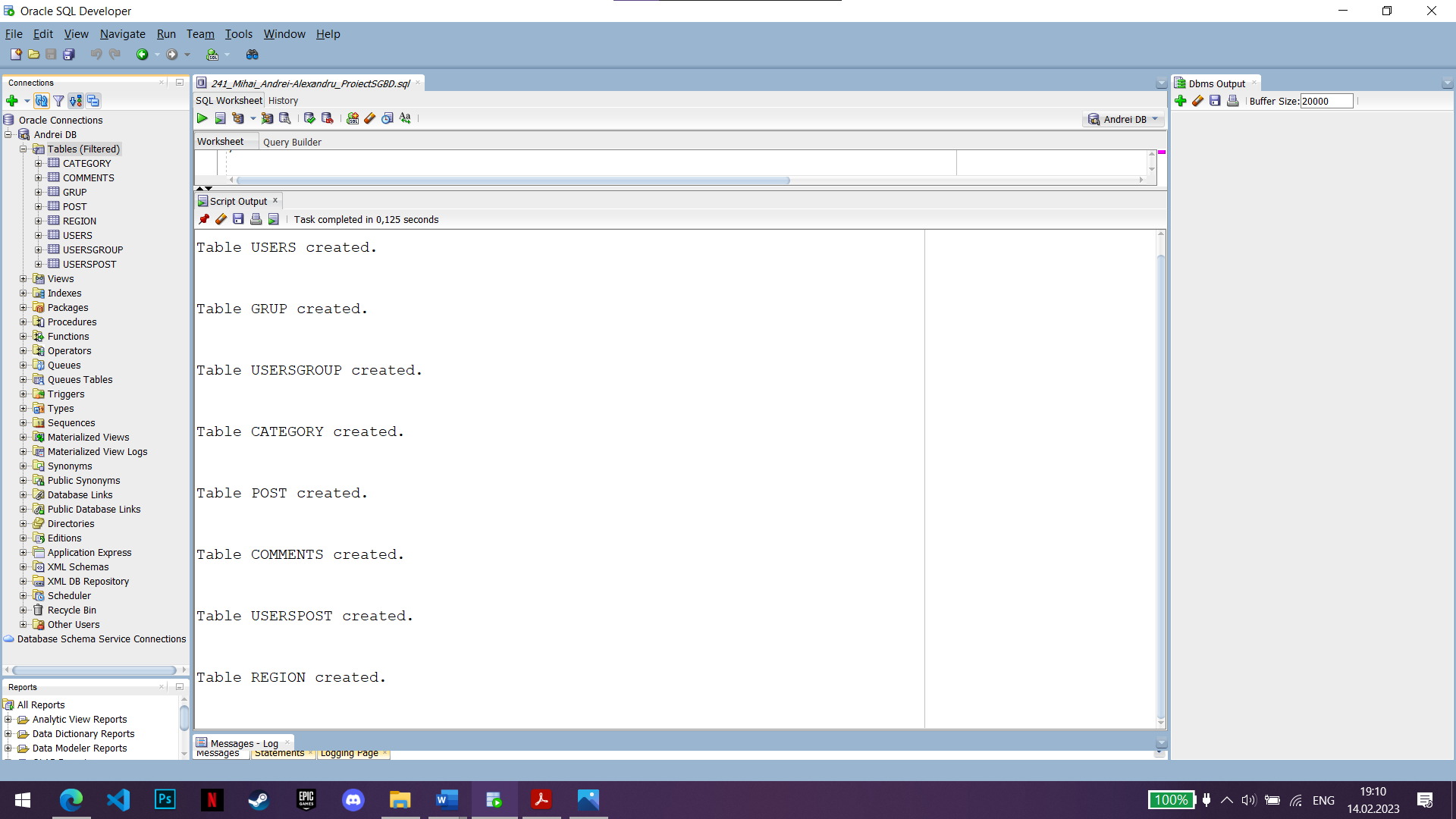
RegionID integer primary key,

State varchar2(50),

City varchar2(50),

Adress varchar2(75)

);



*\*Am creat întâi structura tabelelor (toate coloanele), apoi am adăugat constrângeri pentru cheile externe prin intermediul comenzii* ***ALTER TABLE****.*

alter table POST

add constraint PostCategoryFK foreign key(CategoryID) referencesCATEGORY(CategoryID);

alter table COMMENTS

add constraint CommentPostFK foreign key(PostID) references Post(PostID)

add constraint CommentUserFK foreign key(UserID) references USERS(UserID);alter table USERSPOST

add constraint UserPostPostFK foreign key(PostID) references POST(PostID)

add constraint UserPostUserFK foreign key(UserID) references USERS(UserID);

alter table USERSGROUP

add constraint UsersGroupUserFK foreign key(UserID) references USERS(UserID)

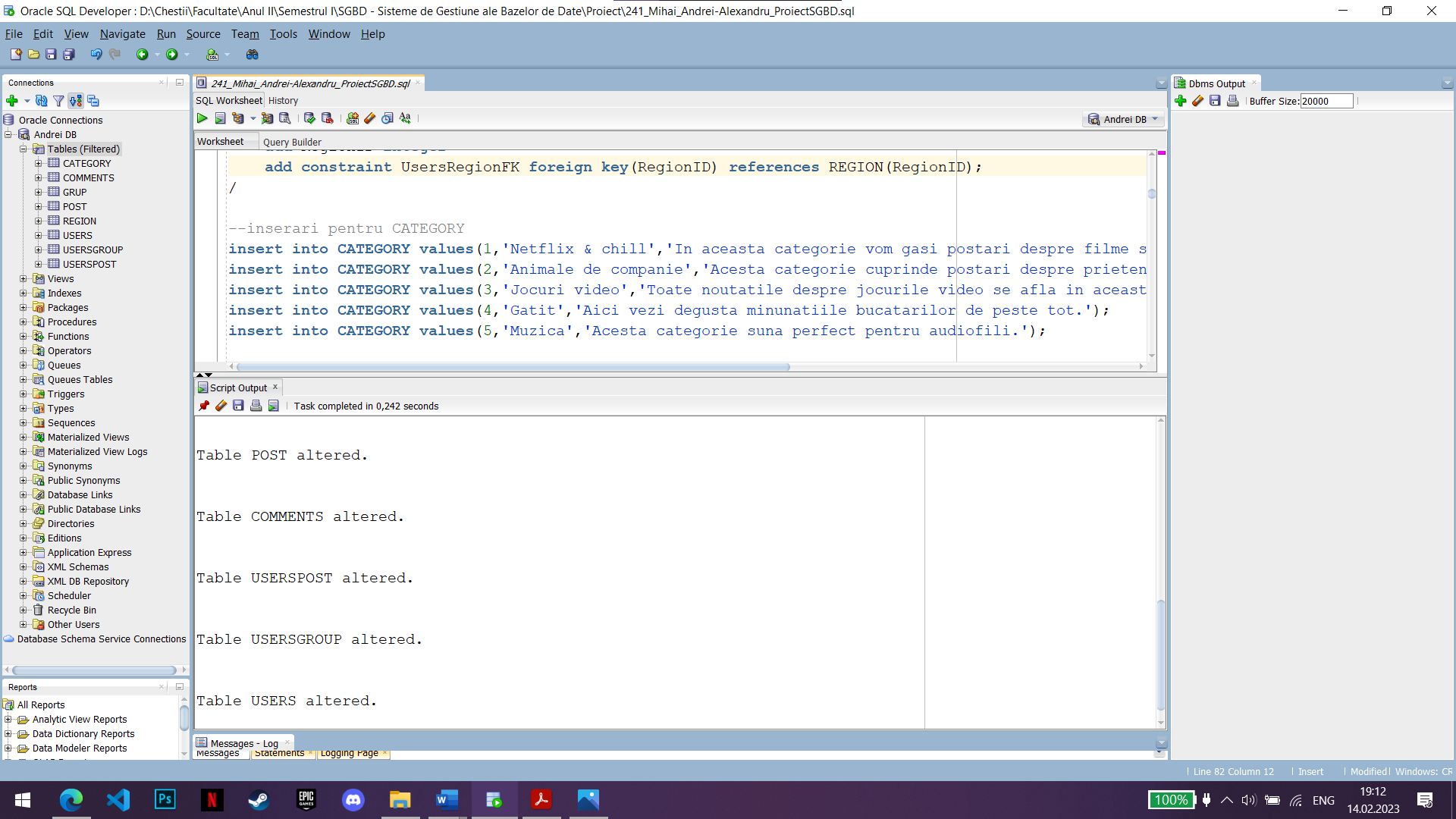
add constraint UsersGroupGroupFK foreign key(GroupID) references GRUP(GroupID);

alter table USERS

add RegionID integer

add constraint UsersRegionFK foreign key(RegionID) references REGION(RegionID);

/



**5. Adăugați informații coerente în tabelele create (minim 5 înregistrări pentru fiecare entitate independentă; minim 10 înregistrări pentru tabela asociativă).**

--inserari pentru CATEGORY

insert into CATEGORY values(1,'Netflix And Chill','In aceasta categorie vom gasi postari despre filme si seriale.');

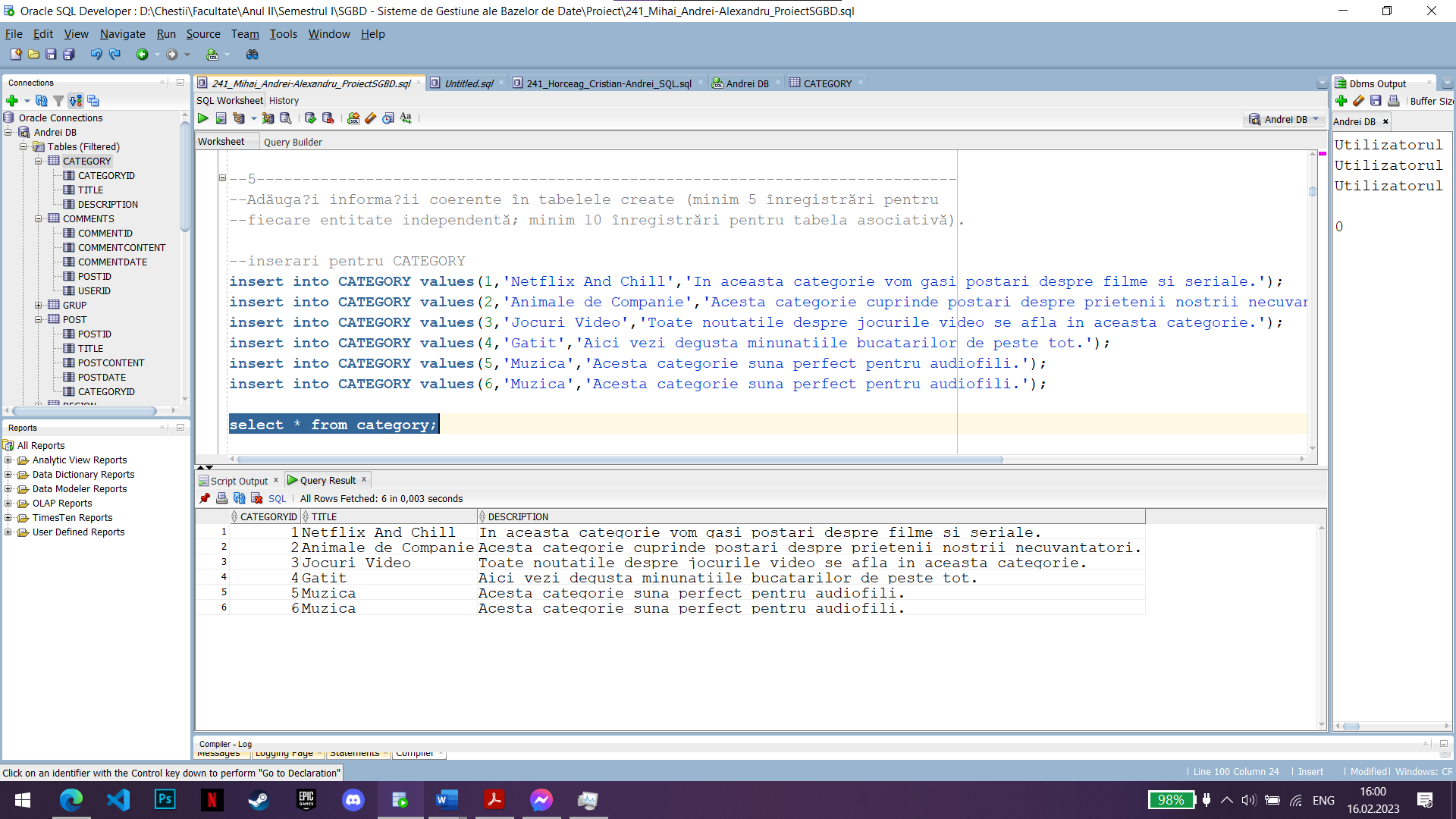
insert into CATEGORY values(2,'Animale de companie','Acesta categorie cuprinde postari despre prietenii nostrii necuvantatori.');

insert into CATEGORY values(3,'Jocuri video','Toate noutatile despre jocurile video se afla in aceasta categorie.');

insert into CATEGORY values(4,'Gatit','Aici vezi degusta minunatiile bucatarilor de peste tot.');

insert into CATEGORY values(5,'Muzica','Acesta categorie suna perfect pentru audiofili.');

insert into CATEGORY values(6,'Muzica','Acesta categorie suna perfect pentru audiofili.');



--inserari pentru POST

insert into POST values(1,'Top filme 2022','Shrek 5',default,1);

insert into POST values(2,'Recomandare de serial?','Am nevoie de un serial relativ scurt','11.02.2022',1);

insert into POST values(3,'Noua mea pisica','Acesta este noul meu motan! Cum vi se pare?','13.04.2019',2);

insert into POST values(4,'Donez caine','Caut stapan pentru acest catelus','13.04.2019',2);

insert into POST values(5,'Castigatorii turneului de CS:GO','Fiecare membru al echipei a primit cate 200 euro',default,3);

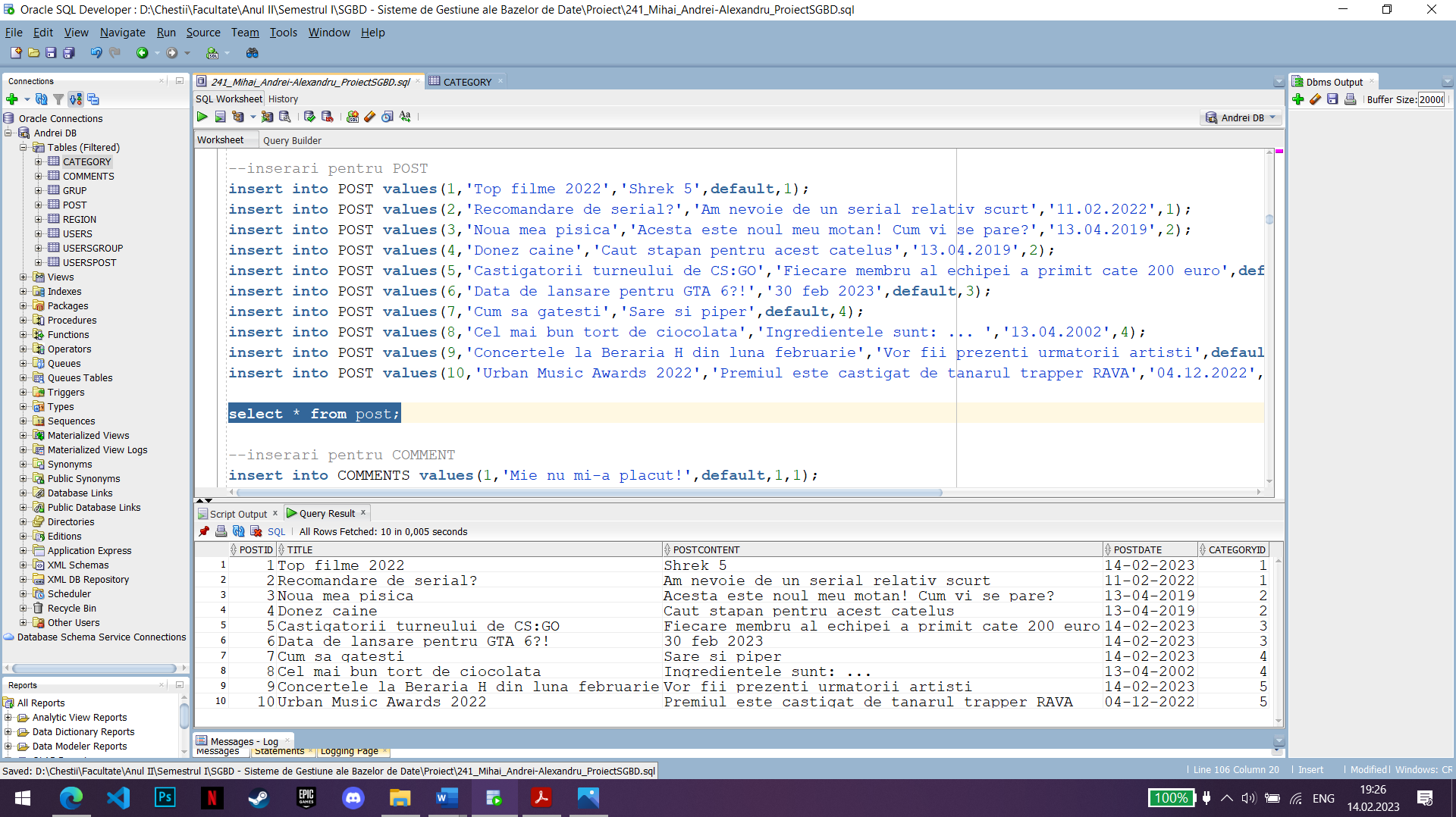
insert into POST values(6,'Data de lansare pentru GTA 6?!','30 feb 2023',default,3);

insert into POST values(7,'Cum sa gatesti','Sare si piper',default,4);

insert into POST values(8,'Cel mai bun tort de ciocolata','Ingredientele sunt: ... ','13.04.2002',4);

insert into POST values(9,'Concertele la Beraria H din luna februarie','Vor fii prezenti urmatorii artisti',default,5);

insert into POST values(10,'Urban Music Awards 2022','Premiul este castigat de tanarul trapper RAVA','04.12.2022',5);



--inserari pentru COMMENT

insert into COMMENTS values(1,'Mie nu mi-a placut!',default,1,1);

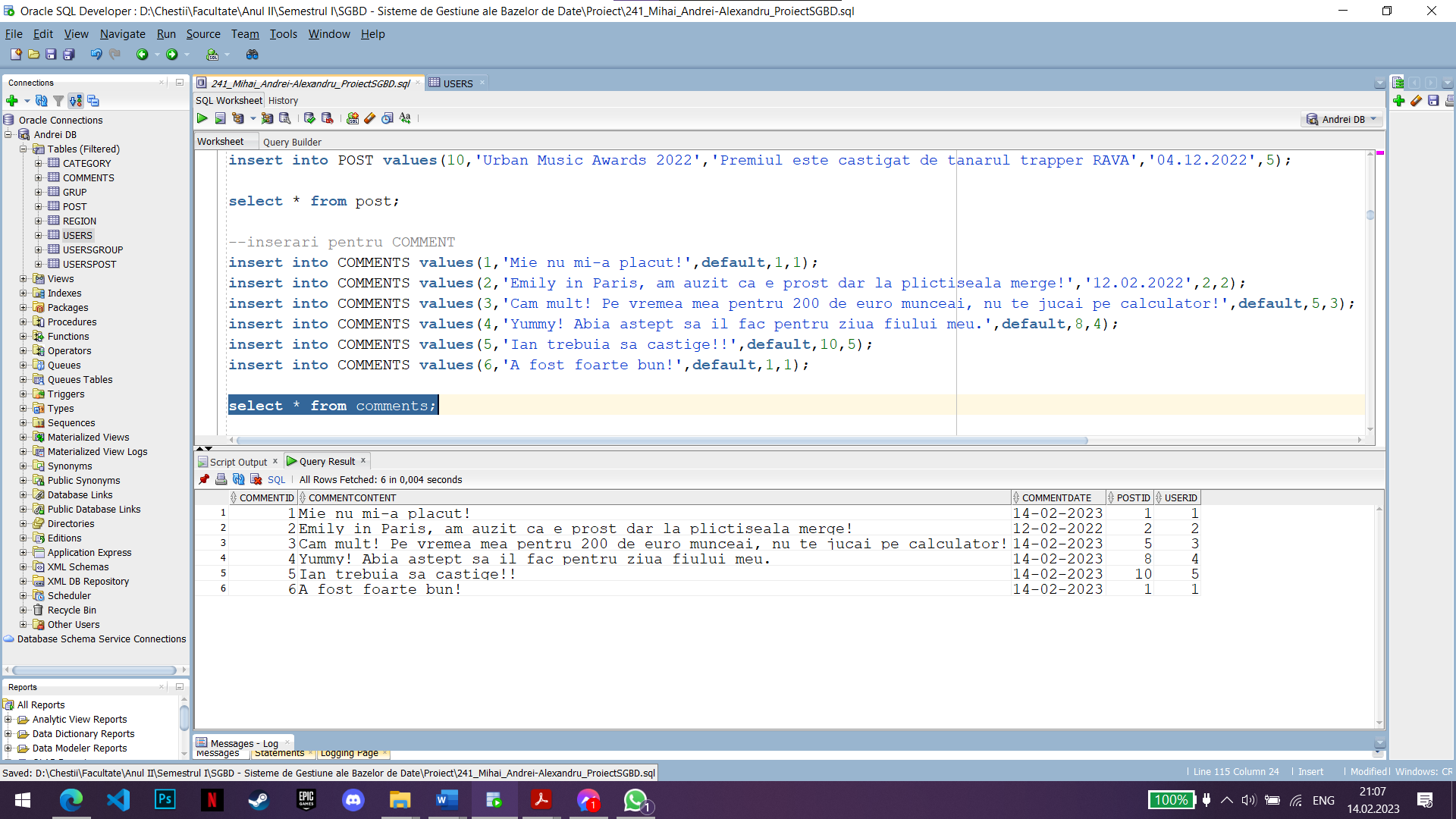
insert into COMMENTS values(2,'Emily in Paris, am auzit ca e prost dar la plictiseala merge!','12.02.2022',2,2);

insert into COMMENTS values(3,'Cam mult! Pe vremea mea pentru 200 de euro munceai, nu te jucai pe calculator!',default,5,3);

insert into COMMENTS values(4,'Yummy! Abia astept sa il fac pentru ziua fiului meu.',default,8,4);

insert into COMMENTS values(5,'Ian trebuia sa castige!!',default,10,5);

insert into COMMENTS values(6,'A fost foarte bun!',default,1,1);



--inserari pentru USERS

insert into USERS values(1,'Teodor','Podani','Podani69','xifahic990@vingood.com','Parola1','0716358423','User',1);

insert into USERS values(2,'Florin','Salam','Cptn','abdad342@vingood.com','flrnslm1','0711358423','Editor',3);

insert into USERS values(3,'Alexandru','Moraru','calculatoristul','eererv12t@vingood.com','coalitia123','0758310567','User',2);

insert into USERS values(4,'Rava','Ravanelli','rava.rva','312312fdast@vingood.com','reginanoptii123','0756758912','Admin',1);

insert into USERS values(5,'Andrei','Horceag','andreihorceag','diasbd2@vingood.com','urzcclctrst1','0706198645','Admin',4);

insert into USERS values(6,'Alexandra','Birzila','alexxndra','hvcs123@vingood.com','astaEParolaMea312','0778122345','Editor',5);

insert into USERS values(7,'Bogdan','Juganaru','juganarub20','dasdbd2@vingood.com','haiSAMergemPeTeren1','0723017395','User',3);

insert into USERS values(8,'Razvan','Burlan','brln123','diadasd233@vingood.com','BurlanGen','0723568974','User',2);

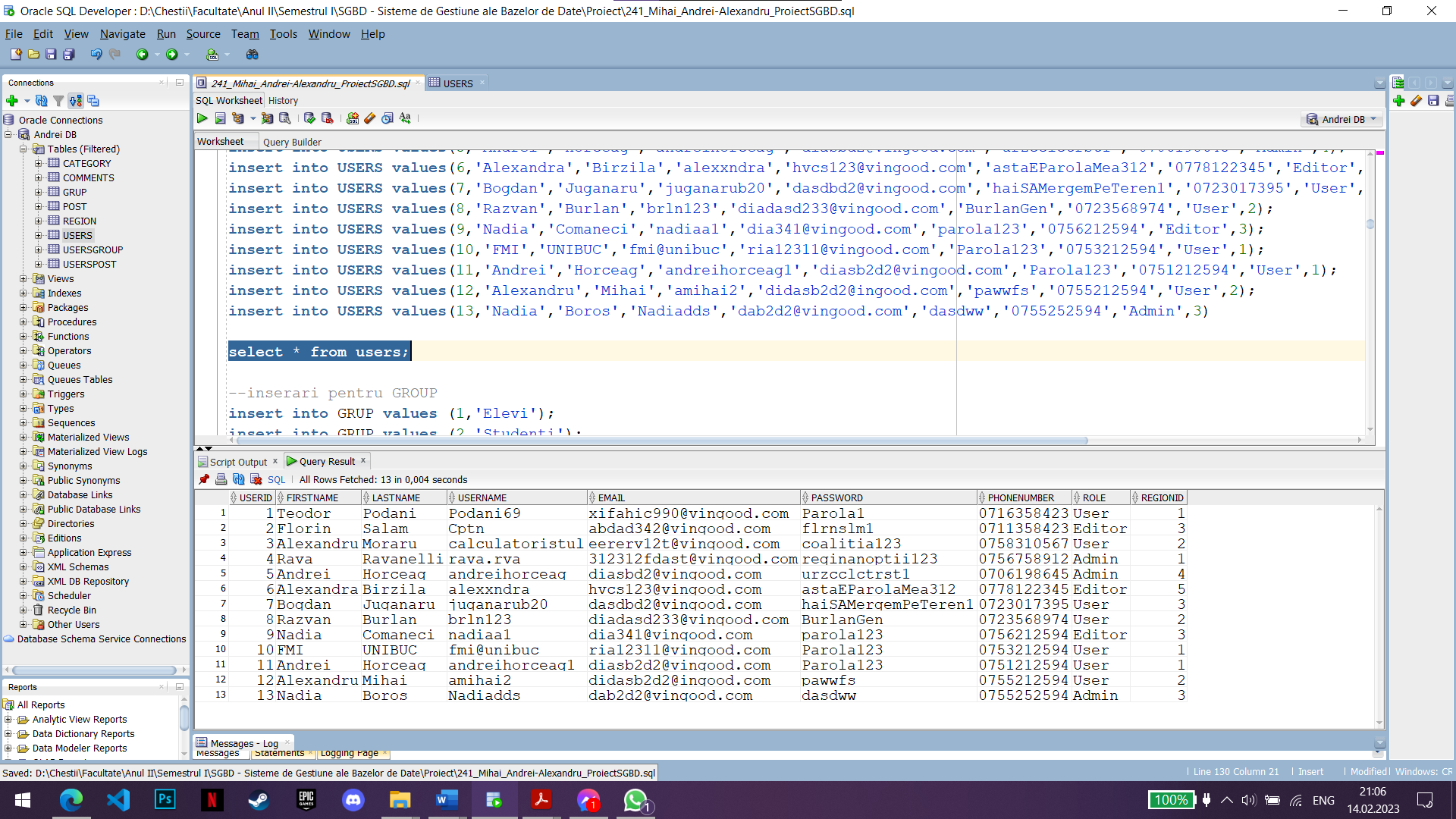
insert into USERS values(9,'Nadia','Comaneci','nadiaa1','dia341@vingood.com','parola123','0756212594','Editor',3);

insert into USERS values(10,'FMI','UNIBUC','fmi@unibuc','ria12311@vingood.com','Parola123','0753212594','User',1);

insert into USERS values(11,'Andrei','Horceag','andreihorceag1','diasb2d2@vingood.com','Parola123','0751212594','User',1);

insert into USERS values(12,'Alexandru','Mihai','amihai2','didasb2d2@ingood.com','pawwfs','0755212594','User',2);

insert into USERS values(13,'Nadia','Boros','Nadiadds','dab2d2@vingood.com','dasdww','0755252594','Admin',3)



--inserari pentru REGION

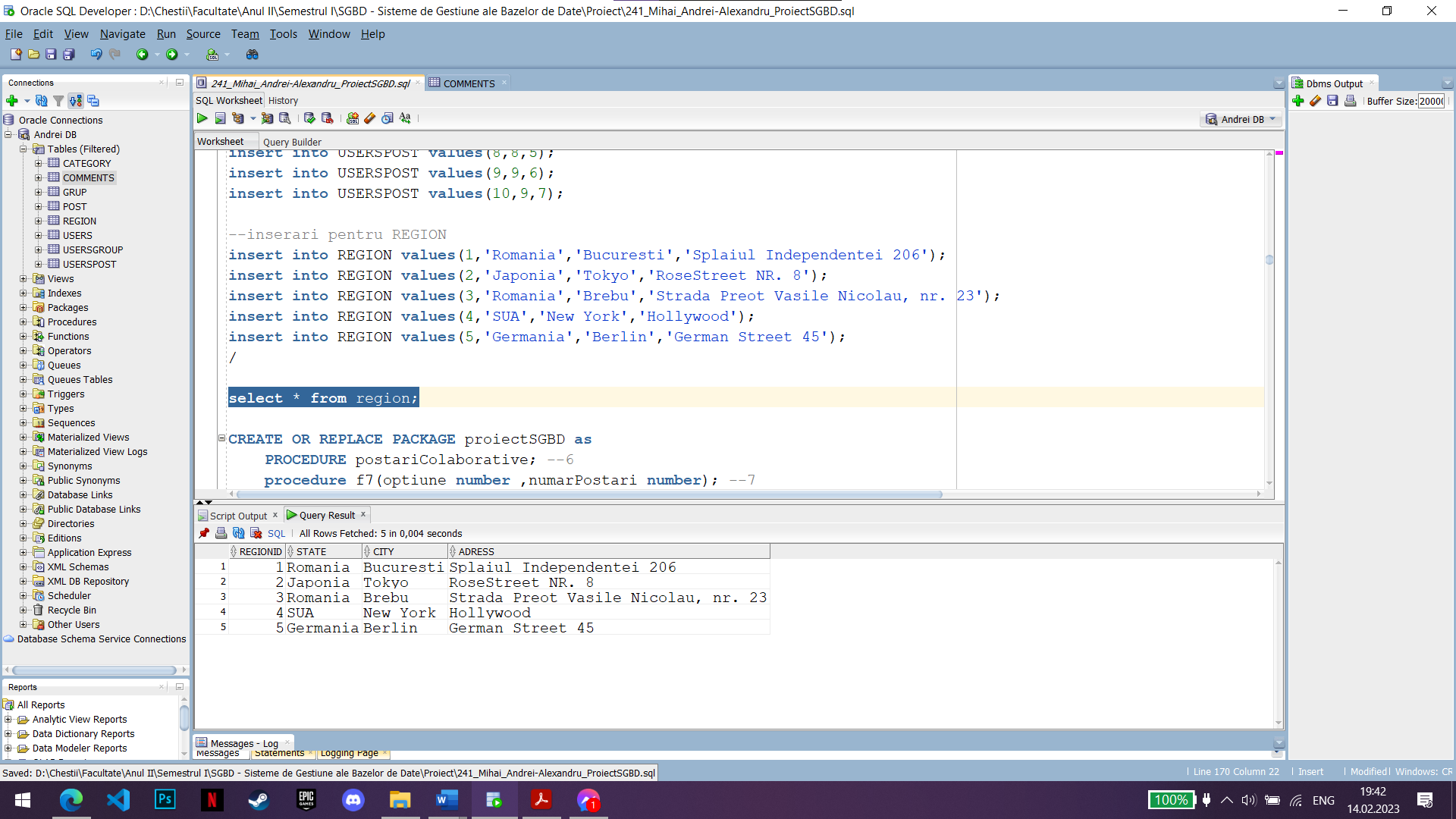
insert into REGION values(1,'Romania','Bucuresti','Splaiul Independentei 206');

insert into REGION values(2,'Japonia','Tokyo','RoseStreet NR. 8');

insert into REGION values(3,'Romania','Brebu','Strada Preot Vasile Nicolau, nr. 23');

insert into REGION values(4,'SUA','New York','Hollywood');

insert into REGION values(5,'Germania','Berlin','German Street 45');



--inserari pentru GRUP

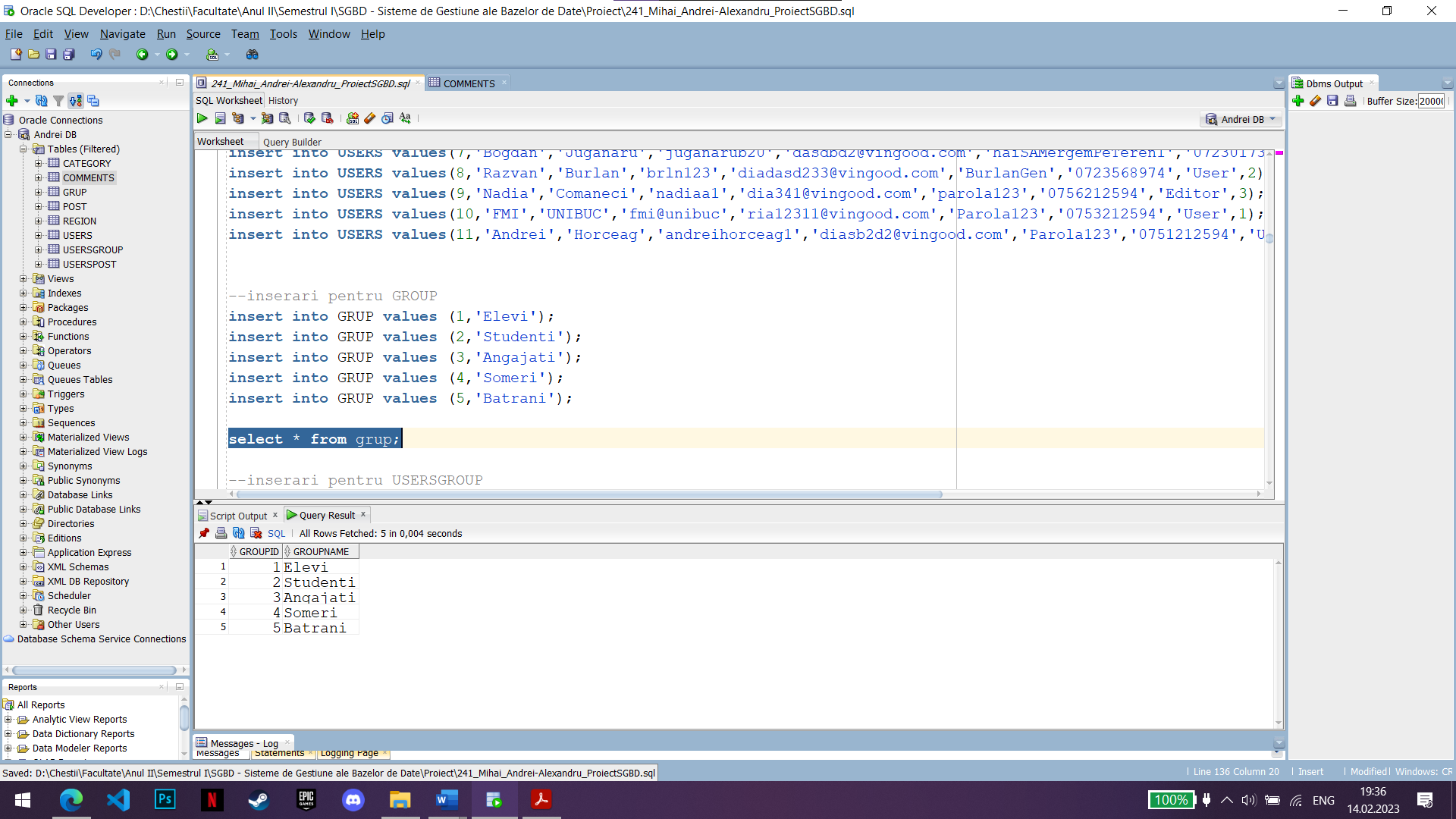
insert into GRUP values (1,'Elevi');

insert into GRUP values (2,'Studenti');

insert into GRUP values (3,'Angajati');

insert into GRUP values (4,'Someri');

insert into GRUP values (5,'Batrani');



--inserari pentru USERSGROUP

insert into USERSGROUP values(1,2,2);

insert into USERSGROUP values(2,1,1);

insert into USERSGROUP values(3,3,3);

insert into USERSGROUP values(4,4,4);

insert into USERSGROUP values(5,5,1);

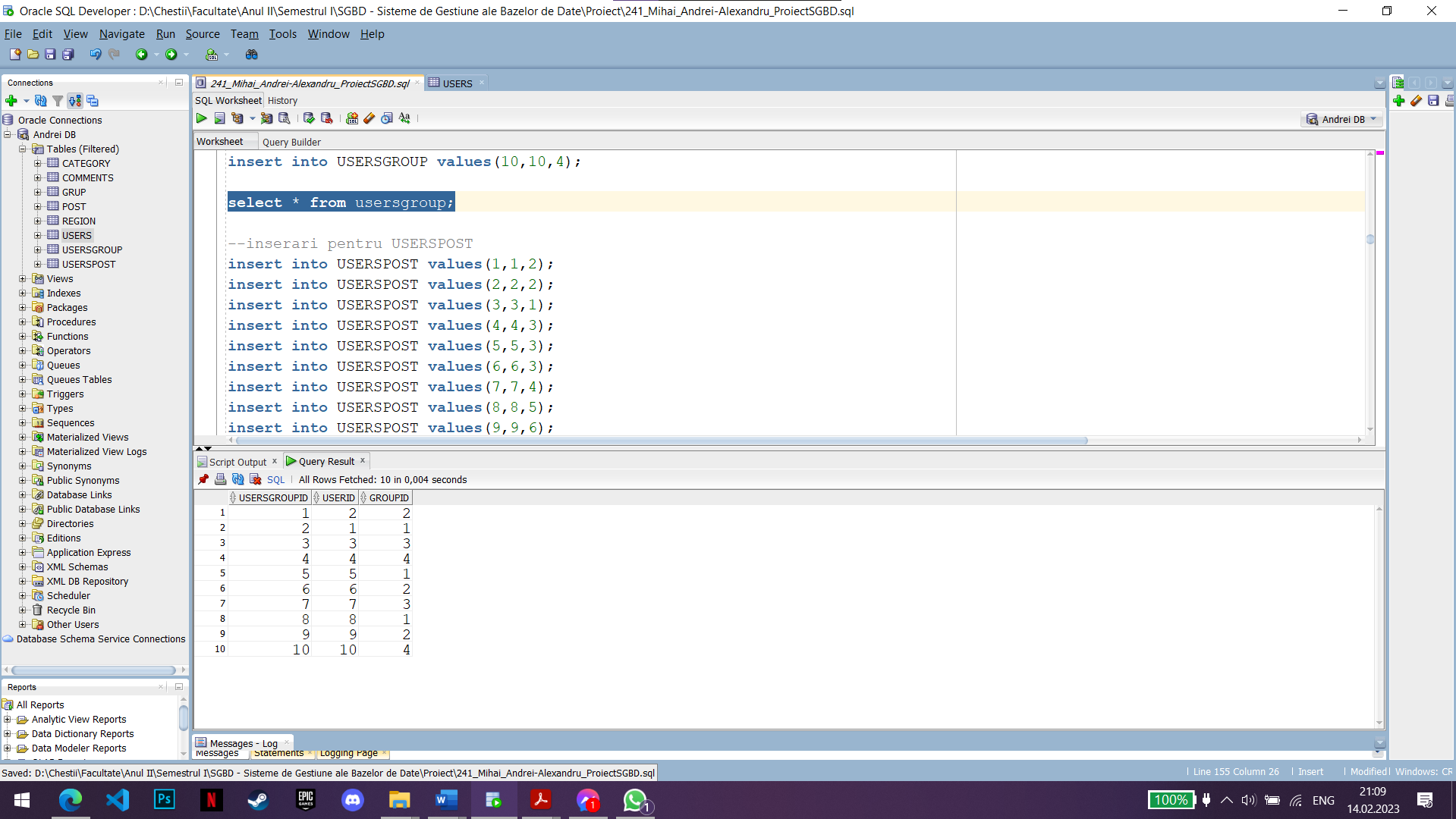
insert into USERSGROUP values(6,6,2);

insert into USERSGROUP values(7,7,3);

insert into USERSGROUP values(8,8,1);

insert into USERSGROUP values(9,9,2);

insert into USERSGROUP values(10,10,4);



--inserari pentru USERSPOST

insert into USERSPOST values(1,1,2);

insert into USERSPOST values(2,2,2);

insert into USERSPOST values(3,3,1);

insert into USERSPOST values(4,4,3);

insert into USERSPOST values(5,5,3);

insert into USERSPOST values(6,6,3);

insert into USERSPOST values(7,7,4);

insert into USERSPOST values(8,8,5);

insert into USERSPOST values(9,9,6);

insert into USERSPOST values(10,9,7);

****

**6. Formulați în limbaj natural o problemă pe care să o rezolvați folosind un subprogram stocat independent care să utilizeze două tipuri diferite de colecții studiate. Apelați subprogramul.**

--6. Sa se afiseze pentru toate postarile colaborative

--care sunt facute de mai mult de 2 useri ID-ul, titlul si numarul de useri.

--De asemenea, sa se afiseze pentru fiecare postare colaborativa username-ul celor care au --creat-o

--sub forma textului "Postarea \*titlu\* a fost creata de catre \*numeuser1\*, \*numeuser2\* ... ---si \*numeuserN\*.

create or replace PROCEDURE postariColaborative

IS

TYPE v\_postari is varray(101) of number;

--am folosit tipul record care retine pentru o postare id-ul, titlul si numarul de useri

type r\_post is record

(id post.postid%type,

titlu post.title%type,

nrUseri integer);

TYPE indexedTable IS TABLE OF VARCHAR2(200) INDEX BY VARCHAR2(100);

record r\_post;

useriPostari indexedTable;

postari v\_postari;

numarUseri number(2);

counter number(2):=1;

concatUsernames varchar2(200):='';

BEGIN

--selectam postarile care au mai mult de 2 useri intr-un vector

select distinct

p.POSTID

bulk collect into

postari

from USERSPOST p

where (select count(u.UserID)

from USERSPOST u

where u.PostID=p.PostID) >2;

FOR I IN postari.FIRST..postari.LAST LOOP

--memoram numarul de useri

select

count(USERID)

into

numarUseri

from USERSPOST

where

PostID=postari(i);

--memoram in record datele pentru postarea curenta

select PostID,title,numarUseri

into record

from post

where PostID=postari(i);

DBMS\_OUTPUT.PUT\_LINE('ID: ' || record.ID || ', titlu: ' || record.titlu || ', numar useri: ' || record.nrUseri);

--concatenam in variabila concatUsernames username-ul userilor postarilor colaborative

for j in (select u.username usr

from users u, userspost up

where PostID=postari(i) and u.userid=up.userid) loop

--concatenam numele la lista numelor

select concat(concatUsernames,j.usr) into concatUsernames

from dual;

if counter=numarUseri-1 then

select concat(concatUsernames,' si ') into concatUsernames

from dual;

elsif counter <> numarUseri then

select concat(concatUsernames,', ') into concatUsernames

from dual;

else

select concat(concatUsernames,'.') into concatUsernames

from dual;

end if;

counter:=counter+1;

end loop;

--adaugam in tabloul indexat pentru fiecare postare stringul format din username-urile concatenate.

useriPostari(record.titlu):=concatUsernames;

--reinitializam variabilele

counter:=1;

concatUsernames:='';

end loop;

--afisam pentru fiecare postare colaborativa userii sai

--reutilizez concatUsernames pentru a itera prin tabloul indexat

concatUsernames := useriPostari.FIRST;

WHILE concatUsernames IS NOT null LOOP

dbms\_output.put\_line ('Postarea ' || concatUsernames || ' a fost creata de catre ' || useriPostari(concatUsernames));

concatUsernames := useriPostari.NEXT(concatUsernames);

END LOOP;

end postariColaborative;

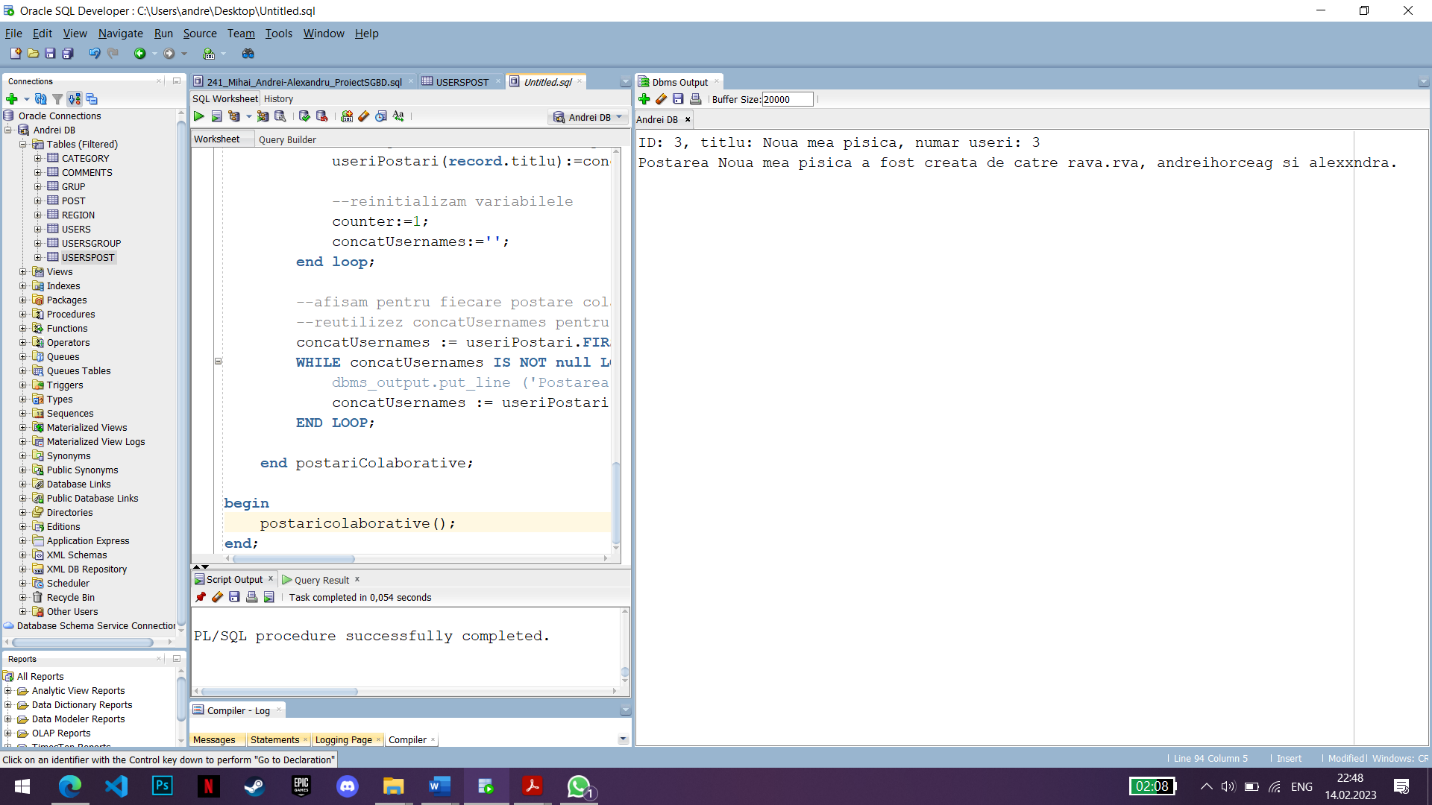
/

begin

postariColaborative();

end;

/



**7. Formulați în limbaj natural o problemă pe care să o rezolvați folosind un subprogram stocat independent care să utilizeze 2 tipuri diferite de cursoare studiate, unul dintre acestea fiind cursor parametrizat. Apelați subprogramul.**

-- 7.Dati de la tastatura un numar natural, reprezentand numarul unei optiuni:

-- Daca optiunea este 1, sa se afiseze numele grupurilor cu un numar de participanti < 5.

-- Daca optiunea este 2, sa se afiseze username-ul utilizatorilor care sunt administratori si au postat de minim n ori, n fiind dat de la tastatura.

-- Daca se da alta optiune de la tastatura, tratati exceptia in mod corespunzator.

Create or Replace procedure f7(optiune number,numarPostari number)

is

numeGrup grup.groupname%type;

username users.username%type;

counter number :=0;

--Primul cursor folosit este un cursor explicit, care rezolva cerinta pentru optiunea 1.

cursor cGrup is

select distinct

g.groupname

from GRUP g, USERSGROUP u

where g.GroupID=u.GroupID and (

select

count(USERID)

from

USERSGROUP

where

g.GroupID=GroupID

) <5;

begin

if optiune=1 then

DBMS\_OUTPUT.PUT\_LINE('Numele grupurilor cu un numar de participanti < 5:');

open cGrup;

loop

counter:=counter+1;

fetch cGrup into numeGrup;

exit when cGrup%notfound;

DBMS\_OUTPUT.PUT\_LINE(counter || '. ' || numeGrup);

end loop;

close cGrup;

elsif optiune=2 then

DBMS\_OUTPUT.PUT\_LINE('Username-ul utilizatorilor care sunt administratori si au postat de minim ' || numarPostari || ' ori:');

--al doilea cursor folosit este un ciclu cursor cu subcereri, care rezolva cerinta pentru optiunea 2.

for i in (select distinct username numeUtilizator

from USERS u, USERSPOST up

where lower(u.ROLE)='admin' and (

select

count(PostID)

from

USERSPOST

where u.USERID=UserID

)>=numarPostari) loop

DBMS\_OUTPUT.PUT\_LINE(counter+1 || '. ' || i.numeUtilizator);

counter:=counter+1;

end loop;

if counter=0 then

DBMS\_OUTPUT.PUT\_LINE('Nu exista!');

end if;

--daca utilizatorul introduce altceva decat 1 si 2 ==> eroare.

else

RAISE\_APPLICATION\_ERROR(-20007,'Optiune invalida! Alegeti o optiune intre 1 si 2');

end if;

end f7;

/

begin

f7(1,2); --7, aici al doilea parametru poate fi orice, nu influenteaza.

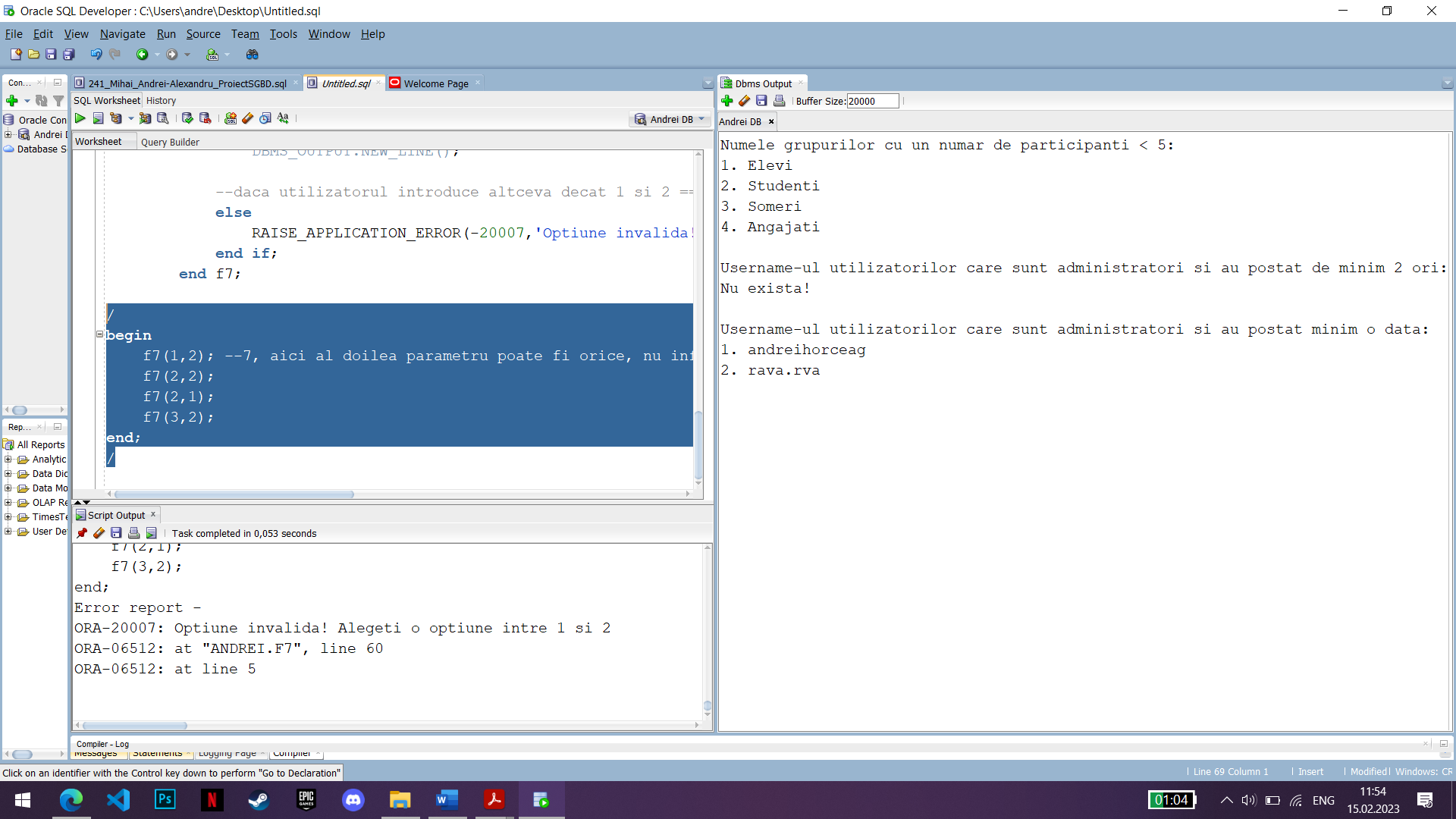
f7(2,2);

f7(2,1);

f7(3,2);

end;

/



*\*După cum se poate observa, primele 3 apeluri au output-uri corespunzătoare, dar pentru ultimul apel (unde numarul opțiunii este 3) am primit eroare!*

**8. Formulați în limbaj natural o problemă pe care să o rezolvați folosind un subprogram stocat independent de tip funcție care să utilizeze într-o singură comandă SQL 3 dintre tabelele definite.**

--8. Afisati titlul pentru cele mai comentate postari dintr-o categorie data ca parametru.

--Returnati prin intermediul functiei numarul de postari gasite.

create or replace function mostCommentedPost(categorie CATEGORY.categoryid%type)

return number is

TYPE mPosts is varray(101) of POST.postid%type;

maxPosts mPosts;

titlu post.title%type;

negCat exception;

NDF exception;

existaCat number;

begin

--verificam daca exista categoria data ca parametru

select count(categoryid) into existaCat

from category

where categoryid=categorie;

if existaCat=0 and categorie>0 then

raise NDF;

elsif categorie<0 then

raise negCat;

else

--selectam id-ul celor mai comentate postari din categoria "categorie".

select p.PostID bulk collect into maxPosts

from POST p

where (select count(COMMENTID)

from COMMENTS c

where p.PostID=c.postid)=(select

max(counter)

from

(

select count(COMMENTID) counter

from COMMENTS

group by PostID

)) and CategoryID=categorie;

--parcurgem id-urile postarilor gasite

FOR I IN maxPosts.FIRST..maxPosts.LAST LOOP

select Title into titlu

from POST

where

PostID=maxPosts(i);

DBMS\_OUTPUT.PUT\_LINE('Postarea ' || i || ' cu titlul ' || titlu);

end loop;

return maxPosts.COUNT;

end if;

EXCEPTION

--NO DATA FOUND

when NDF then

RAISE\_APPLICATION\_ERROR(-20003,'Nu exista categoria introdusa!');

return -1;

--daca user-ul a introdus o categorie negativa o sa presupunem ca a fost

--o greseala de sintaxa, iar output-ul corespuzator va fii pentru

--valoarea respectiva in modul.

when negCat then

DBMS\_OUTPUT.PUT\_LINE('Nu exista categorii negative! Prusupunem ca ati vrut sa introduceti ' || -categorie || '.');

return mostCommentedPost(-categorie);

end mostCommentedPost;

/

begin

**--Cazul 1) in care categoria exista:;**

DBMS\_OUTPUT.put\_line('1) Numarul postarilor este: ' || mostCommentedPost(1));

DBMS\_OUTPUT.NEW\_LINE();

**--Cazul 2) in care categoria nu exista, dar este negativa: - exceptia NegCat**

DBMS\_OUTPUT.put\_line('2) Numarul postarilor este: ' || mostCommentedPost(-1));

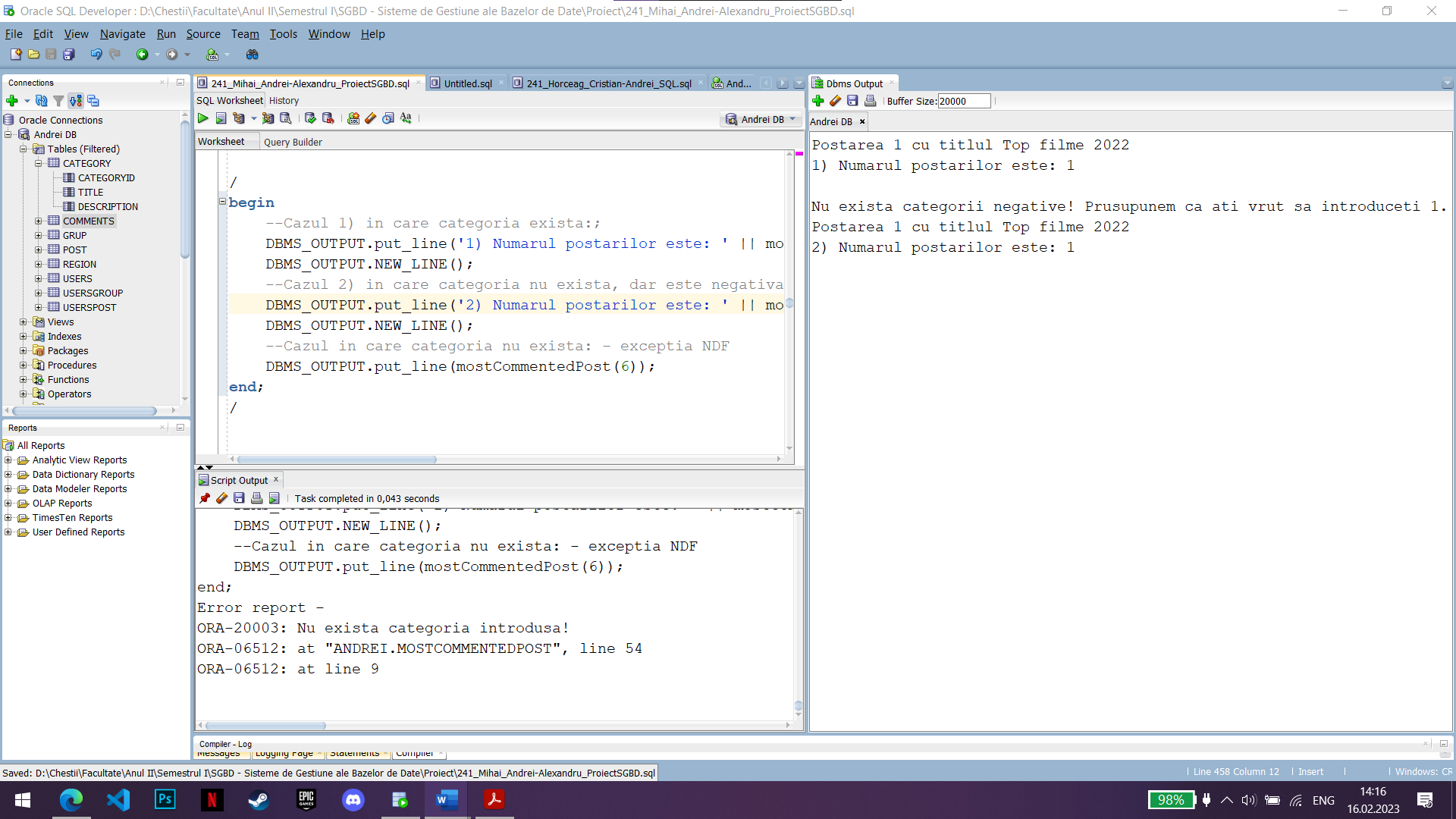
DBMS\_OUTPUT.NEW\_LINE();

**--Cazul in care categoria nu exista: - exceptia NDF**

DBMS\_OUTPUT.put\_line(mostCommentedPost(6));

end;

/



*\*după cum se poate observa, primele 2 cazuri au output-urile corespunzătoare:*

* *Cazul 1 primește un ID de categorie care există deja în DB*
* *Cazul 2 primește un ID negativ de categorie care exista în baza de date, deci output-ul este corespunzător ID-ului în modul.*
* *Cazul 3 primește un ID de categorie care NU există în DB, astfel am primit eroarea corespunzătoare.*

**9. Formulați în limbaj natural o problemă pe care să o rezolvați folosind un subprogram stocat independent de tip procedură care să utilizeze într-o singură comandă SQL 5 dintre tabelele definite. Tratați toate excepțiile care pot apărea, incluzând excepțiile NO\_DATA\_FOUND și TOO\_MANY\_ROWS. Apelați subprogramul astfel încât să evidențiați toate cazurile tratate.**

--9.Sa se afiseze numele, prenumele si grupul de care apartine fiecare user a carui

--postare are cele mai putine comentarii si apartine unei categorii specificate

--(titlul categoriei este input).

/

create or replace procedure f9(categorie category.title%type)

is

catNr number(1) := -1;

nume users.lastname%type;

prenume users.firstname%type;

numeGrup grup.groupname%type;

idPostare post.postid%type;

cursor cf9 is

select u.lastname nume,u.firstname prenume,g.groupname numeGrup,p.postid idPostare

from POST p, users u, grup g, userspost up, usersgroup ug, category ct

where (select count(COMMENTID)

from COMMENTS c

where p.PostID=c.postid)=(select

min(counter)

from

(

select count(COMMENTID) counter

from COMMENTS

group by PostID

)) and lower(ct.title)=lower(categorie) and u.userid=ug.userid and g.groupid=ug.groupid and up.postid=p.postid and up.userid=u.userid;

begin

--verificam daca exista categoria data de la tastatura /

--daca exista mai multe categorii cu acealasi nume.

select

count(categoryid) into catNr

from category

where lower(title)=lower(categorie);

if catNr=0 then

raise no\_data\_found;

elsif catNr>1 then

raise too\_many\_rows;

end if;

DBMS\_OUTPUT.PUT\_LINE(catNr);

for i in cf9 loop

DBMS\_OUTPUT.PUT\_LINE('Utilizatorul ' || i.nume || ' ' || i.prenume || ' din grupul ' || i.numeGrup || ' a facut postarea cu numarul ' || i.idpostare);

end loop;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RAISE\_APPLICATION\_ERROR(-20001,'Nu exista categoria specificata in baza de date');

WHEN TOO\_MANY\_ROWS THEN

RAISE\_APPLICATION\_ERROR(-20002,'Exista mai multe categorii cu numele specificat in baza de date!');

end f9;

/

/

begin

--cazul 1, pentru o categorie care apare o singura data in DB.

f9('Netflix And Chill');

--cazul 2, pentru o categorie care NU apare in DB.

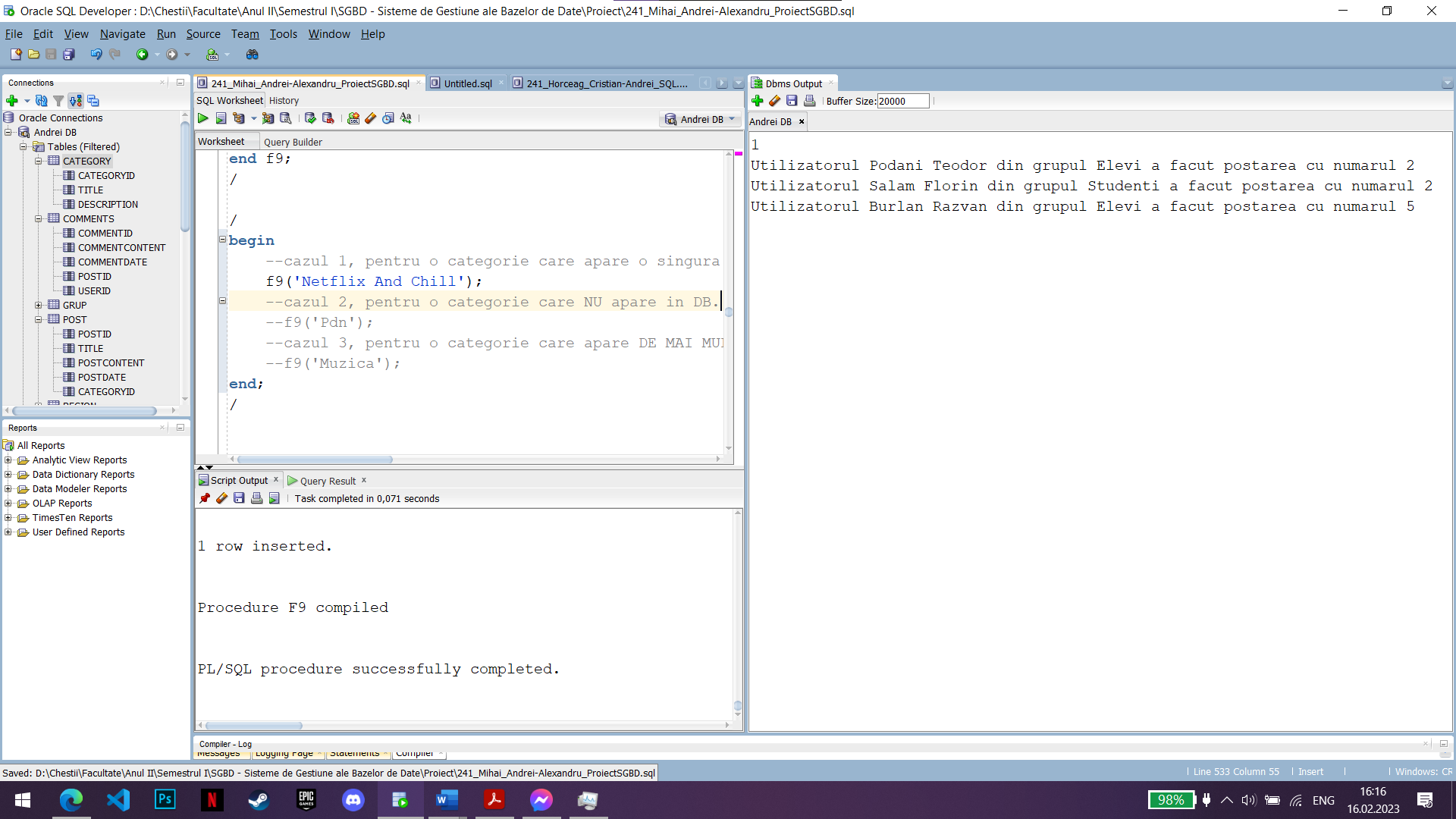
f9('Pdn');

--cazul 3, pentru o categorie care apare DE MAI MULTE ORI in DB

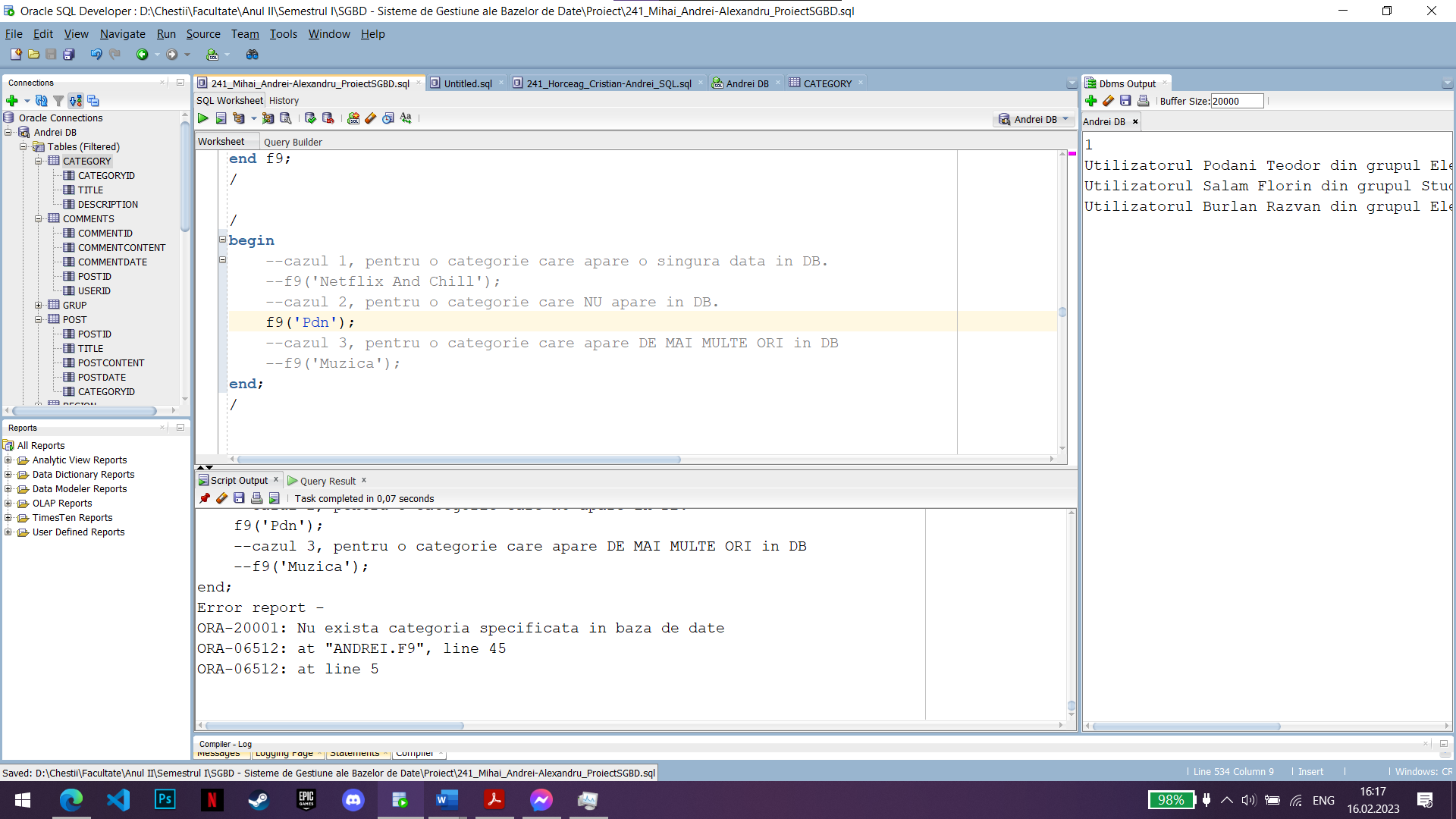
f9('Muzica');

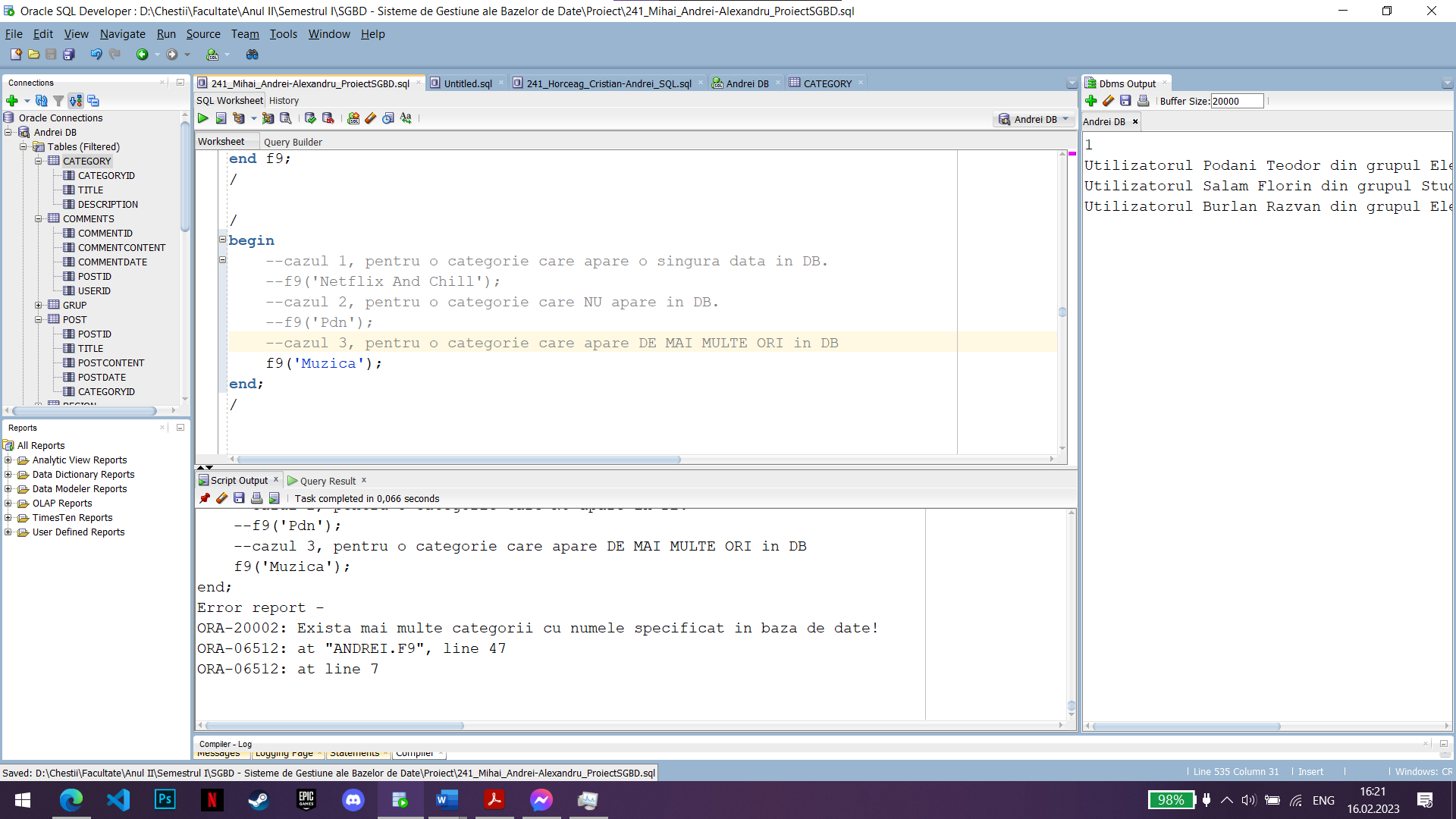
end;

/



*\*pentru cazul 1, s-a afișat output-ul corespunzător.*





*\*pentru cazurile 2 si 3, am primit erorile corespunzătoare!*

**10. Definiți un *trigger* de tip LMD la nivel de comandă. Declanșați *trigger*-ul.**

--10. Trigger care nu permite crearea unui user nou intre orele 20:00-21:00.

/

create or replace TRIGGER ora

BEFORE INSERT ON users

BEGIN

IF (TO\_CHAR(SYSDATE,'HH24') BETWEEN 20 AND 21)

THEN RAISE\_APPLICATION\_ERROR(-20007,'Nu se pot creea useri noi intre 20:00-21:00');

END IF;

END;

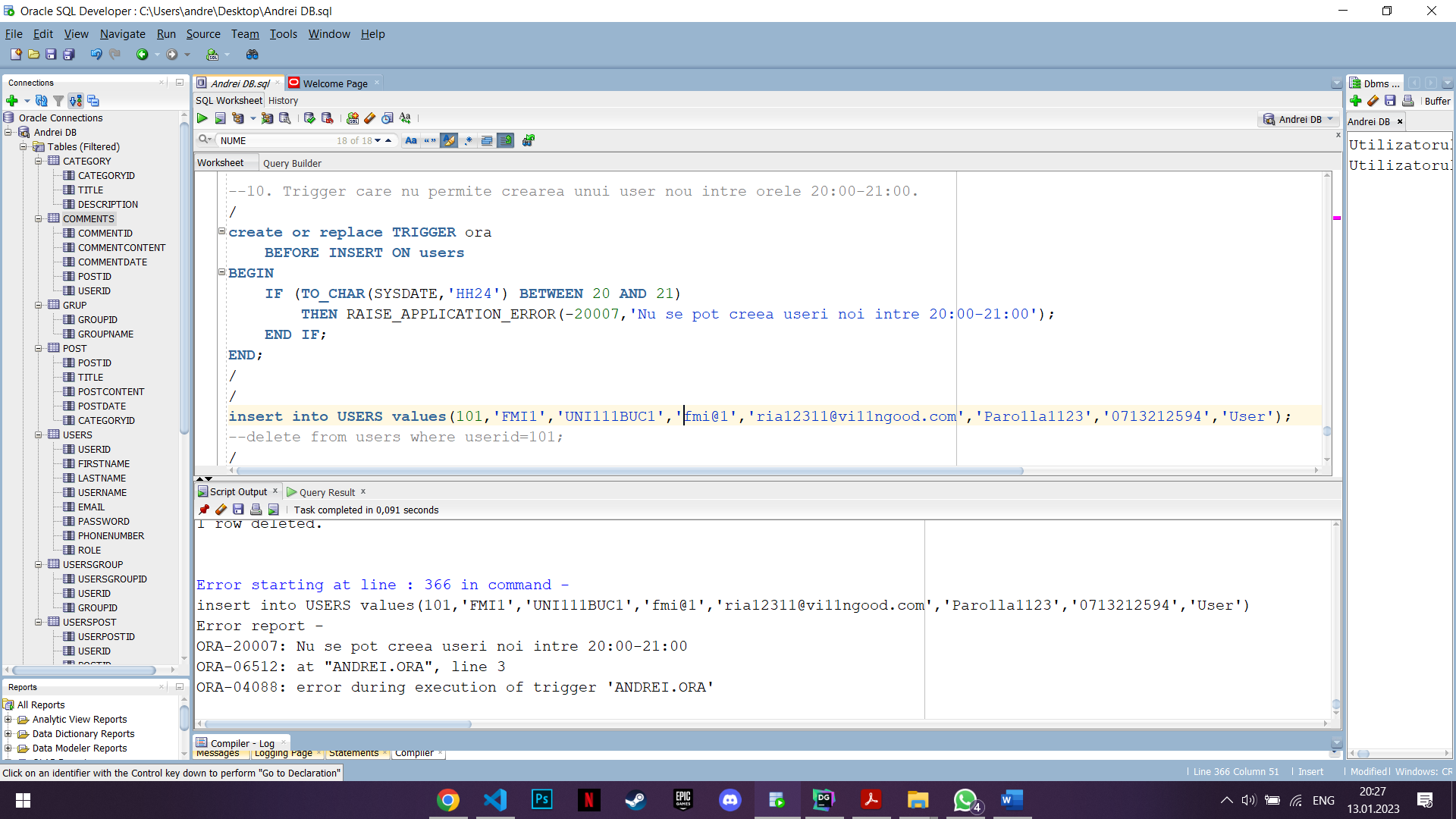
/

/

insert into USERS values(101,'FMI1','UNI111BUC1','fmi@1','ria12311@vi11ngood.com','Paro1la1123','0713212594','User');

--delete from users where userid=101;

/



*\*Inserarea liniei in tabela USERS nu a funcționat datorită triggerului care nu permite inserarea în tabelă între orele 20:00-21:00.*

**11. Definiți un *trigger* de tip LMD la nivel de linie. Declanșați *trigger*-ul.**

/

--11.Sa nu se poata modifica titlul postarilor cu unul nou care are mai putin de 3 caractere.

create or replace trigger actualizarePostari

before update of title on post

for each row

begin

if(length(:new.title)<3) then

RAISE\_APPLICATION\_ERROR(-20006,'Noul titlu nu poate avea lungime <3!!!');

end if;

end;

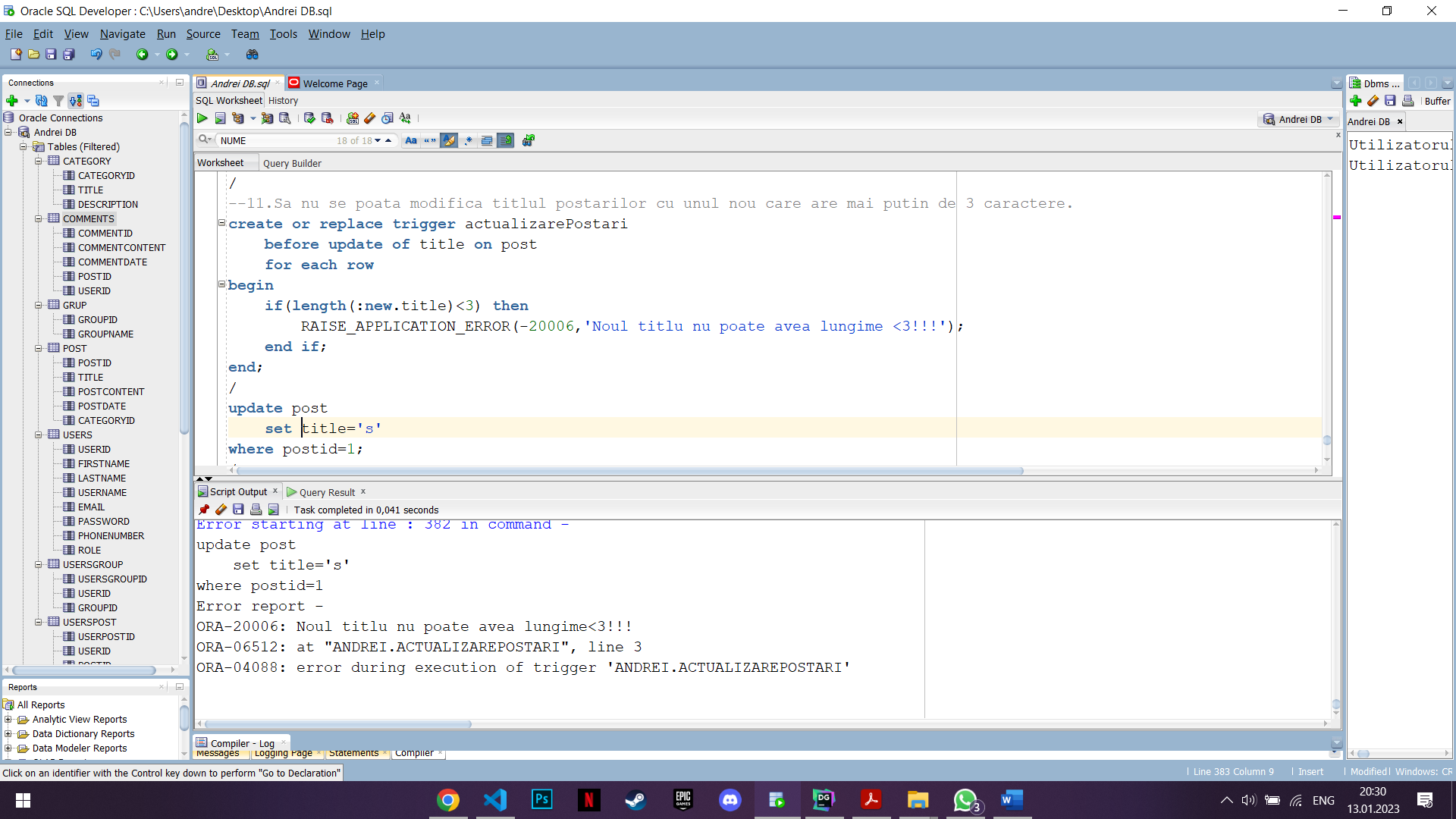
/

update post

set title='s'

where postid=1;

/



*\*Lungimea noului titlu este prea mică, deci titlul nu a putut fii actualizat.*

**12. Definiți un trigger de tip LDD. Declanșați trigger-ul.**

/

--12. Trigger ca nu permite stergerea tabelului POST.

create or replace trigger dropTrigger

before drop on database

begin

if(ORA\_DICT\_OBJ\_NAME = 'POST') then

RAISE\_APPLICATION\_ERROR(-20005,'Nu puteti sterge tabela POST.');

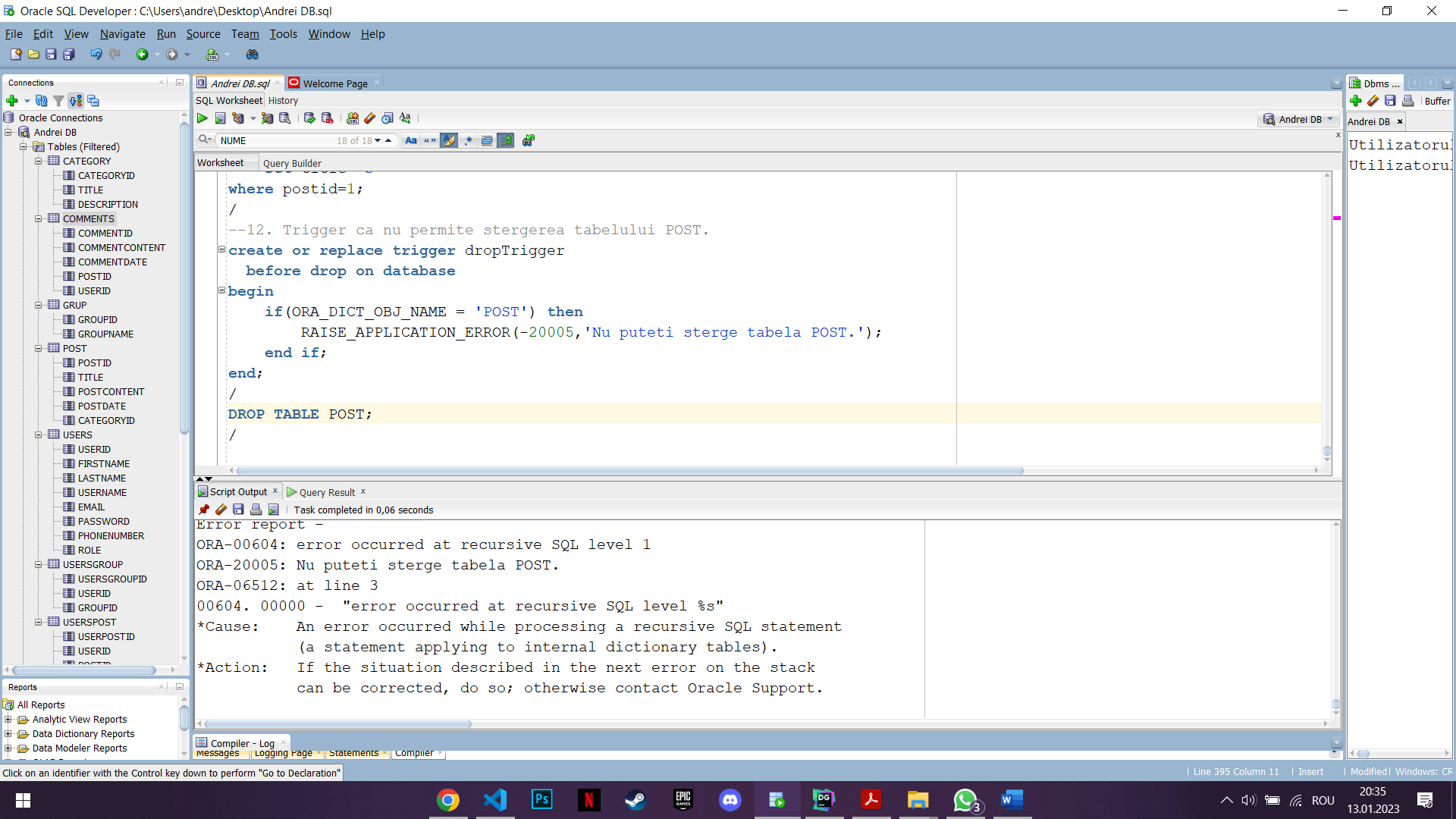
end if;

end;

/

DROP TABLE POST;

/



După cum se poate observa, datorită triggerului tabela POST nu a putut fii ștearsă.

**13. Definiți un pachet care să conțină toate obiectele definite în cadrul proiectului**.

/

CREATE OR REPLACE PACKAGE proiectSGBD as

PROCEDURE postariColaborative; --6

procedure f7(optiune number ,numarPostari number); --7

function mostCommentedPost(categorie CATEGORY.categoryid%type) --8

return number;

procedure f9(categorie category.title%type); --9

end proiectSGBD;

/

/

CREATE OR REPLACE PACKAGE BODY proiectSGBD AS

--6. Sa se afiseze pentru toate postarile colaborative

--care sunt facute de mai mult de 2 useri ID-ul, titlul si numarul de useri.

--De asemenea, sa se afiseze pentru fiecare postare colaborativa username-ul celor care au creat-o

--sub forma textului "Postarea \*titlu\* a fost creata de catre \*numeuser1\*, \*numeuser2\* ... si \*numeuserN\*.

PROCEDURE postariColaborative

IS

TYPE v\_postari is varray(101) of number;

--am folosit tipul record care retine pentru o postare id-ul, titlul si numarul de useri

type r\_post is record

(id post.postid%type,

titlu post.title%type,

nrUseri integer);

TYPE indexedTable IS TABLE OF VARCHAR2(200) INDEX BY VARCHAR2(100);

record r\_post;

useriPostari indexedTable;

postari v\_postari;

numarUseri number(2);

counter number(2):=1;

concatUsernames varchar2(200):='';

BEGIN

--selectam postarile care au mai mult de 2 useri intr-un vector

select distinct

p.POSTID

bulk collect into

postari

from USERSPOST p

where (select count(u.UserID)

from USERSPOST u

where u.PostID=p.PostID) >2;

FOR I IN postari.FIRST..postari.LAST LOOP

--memoram numarul de useri

select

count(USERID)

into

numarUseri

from USERSPOST

where

PostID=postari(i);

--memoram in record datele pentru postarea curenta

select PostID,title,numarUseri

into record

from post

where PostID=postari(i);

DBMS\_OUTPUT.PUT\_LINE('ID: ' || record.ID || ', titlu: ' || record.titlu || ', numar useri: ' || record.nrUseri);

--concatenam in variabila concatUsernames username-ul userilor postarilor colaborative

for j in (select u.username usr

from users u, userspost up

where PostID=postari(i) and u.userid=up.userid) loop

-- --concatenam numele la lista numelor

select concat(concatUsernames,j.usr) into concatUsernames

from dual;

if counter=numarUseri-1 then

select concat(concatUsernames,' si ') into concatUsernames

from dual;

elsif counter <> numarUseri then

select concat(concatUsernames,', ') into concatUsernames

from dual;

else

select concat(concatUsernames,'.') into concatUsernames

from dual;

end if;

counter:=counter+1;

end loop;

--adaugam in tabloul indexat pentru fiecare postare stringul format din username-urile concatenate.

useriPostari(record.titlu):=concatUsernames;

--reinitializam variabilele

counter:=1;

concatUsernames:='';

end loop;

--afisam pentru fiecare postare colaborativa userii sai

--reutilizez concatUsernames pentru a itera prin tabloul indexat

concatUsernames := useriPostari.FIRST;

WHILE concatUsernames IS NOT null LOOP

dbms\_output.put\_line ('Postarea ' || concatUsernames || ' a fost creata de catre ' || useriPostari(concatUsernames));

concatUsernames := useriPostari.NEXT(concatUsernames);

END LOOP;

end postariColaborative;

-- 7.Dati de la tastatura un numar natural, reprezentand numarul unei optiuni:

-- Daca optiunea este 1, sa se afiseze numele grupurilor cu un numar de participanti < 5.

-- Daca optiunea este 2, sa se afiseze username-ul utilizatorilor care sunt administratori si au postat de minim n ori, n fiind dat de la tastatura.

-- Daca se da alta optiune de la tastatura, tratati exceptia in mod corespunzator.

procedure f7(optiune number,numarPostari number)

is

numeGrup grup.groupname%type;

username users.username%type;

counter number :=0;

--Primul cursor folosit este un cursor explicit, care rezolva cerinta pentru optiunea 1.

cursor cGrup is

select distinct

g.groupname

from GRUP g, USERSGROUP u

where g.GroupID=u.GroupID and (

select

count(USERID)

from

USERSGROUP

where

g.GroupID=GroupID

) <5;

begin

if optiune=1 then

DBMS\_OUTPUT.PUT\_LINE('Numele grupurilor cu un numar de participanti < 5:');

open cGrup;

loop

counter:=counter+1;

fetch cGrup into numeGrup;

exit when cGrup%notfound;

DBMS\_OUTPUT.PUT\_LINE(counter || '. ' || numeGrup);

end loop;

close cGrup;

DBMS\_OUTPUT.NEW\_LINE();

elsif optiune=2 then

if numarPostari<>1 then

DBMS\_OUTPUT.PUT\_LINE('Username-ul utilizatorilor care sunt administratori si au postat de minim ' || numarPostari || ' ori:');

else

DBMS\_OUTPUT.PUT\_LINE('Username-ul utilizatorilor care sunt administratori si au postat minim o data:');

end if;

--al doilea cursor folosit este un ciclu cursor cu subcereri, care rezolva cerinta pentru optiunea 2.

for i in (select distinct username numeUtilizator

from USERS u, USERSPOST up

where lower(u.ROLE)='admin' and (

select

count(PostID)

from

USERSPOST

where u.USERID=UserID

)>=numarPostari) loop

DBMS\_OUTPUT.PUT\_LINE(counter+1 || '. ' || i.numeUtilizator);

counter:=counter+1;

end loop;

if counter=0 then

DBMS\_OUTPUT.PUT\_LINE('Nu exista!');

end if;

DBMS\_OUTPUT.NEW\_LINE();

--daca utilizatorul introduce altceva decat 1 si 2 ==> eroare.

else

RAISE\_APPLICATION\_ERROR(-20007,'Optiune invalida! Alegeti o optiune intre 1 si 2');

end if;

end f7;

--8. Afisati titlul pentru cele mai comentate postari dintr-o categorie data ca parametru.

-- Returnati prin intermediul functiei numarul de postari gasite.

function mostCommentedPost(categorie CATEGORY.categoryid%type)

return number is

TYPE mPosts is varray(101) of POST.postid%type;

maxPosts mPosts;

titlu post.title%type;

negCat exception;

NDF exception;

existaCat number;

begin

--verificam daca exista categoria data ca parametru

select count(categoryid) into existaCat

from category

where categoryid=categorie;

if existaCat=0 and categorie>0 then

raise NDF;

elsif categorie<0 then

raise negCat;

else

--selectam id-ul celor mai comentate postari din categoria "categorie".

select p.PostID bulk collect into maxPosts

from POST p

where (select count(COMMENTID)

from COMMENTS c

where p.PostID=c.postid)=(select

max(counter)

from

(

select count(COMMENTID) counter

from COMMENTS

group by PostID

)) and CategoryID=categorie;

--parcurgem id-urile postarilor gasite

FOR I IN maxPosts.FIRST..maxPosts.LAST LOOP

select Title into titlu

from POST

where

PostID=maxPosts(i);

DBMS\_OUTPUT.PUT\_LINE('Postarea ' || i || ' cu titlul ' || titlu);

end loop;

return maxPosts.COUNT;

end if;

EXCEPTION

--NO DATA FOUND

when NDF then

RAISE\_APPLICATION\_ERROR(-20003,'Nu exista categoria introdusa!');

return -1;

--daca user-ul a introdus o categorie negativa o sa presupunem ca a fost

--o greseala de sintaxa, iar output-ul corespuzator va fii pentru

--valoarea respectiva in modul.

when negCat then

DBMS\_OUTPUT.PUT\_LINE('Nu exista categorii negative! Prusupunem ca ati vrut sa introduceti ' || -categorie || '.');

return mostCommentedPost(-categorie);

end mostCommentedPost;

--9.Sa se afiseze numele, prenumele si grupul de care apartine fiecare user a carui

--postare are cele mai putine comentarii si apartine unei categorii specificate

--(titlul categoriei este input).

procedure f9(categorie category.title%type)

is

catNr number(1) := -1;

nume users.lastname%type;

prenume users.firstname%type;

numeGrup grup.groupname%type;

idPostare post.postid%type;

cursor cf9 is

select u.lastname nume,u.firstname prenume,g.groupname numeGrup,p.postid idPostare

from POST p, users u, grup g, userspost up, usersgroup ug, category ct

where (select count(COMMENTID)

from COMMENTS c

where p.PostID=c.postid)=(select

min(counter)

from

(

select count(COMMENTID) counter

from COMMENTS

group by PostID

)) and lower(ct.title)=lower(categorie) and u.userid=ug.userid and g.groupid=ug.groupid and up.postid=p.postid and up.userid=u.userid;

begin

--verificam daca exista categoria data de la tastatura /

--daca exista mai multe categorii cu acealasi nume.

select

count(categoryid) into catNr

from category

where lower(title)=lower(categorie);

if catNr=0 then

raise no\_data\_found;

elsif catNr>1 then

raise too\_many\_rows;

end if;

DBMS\_OUTPUT.PUT\_LINE(catNr);

for i in cf9 loop

DBMS\_OUTPUT.PUT\_LINE('Utilizatorul ' || i.nume || ' ' || i.prenume || ' din grupul ' || i.numeGrup || ' a facut postarea cu numarul ' || i.idpostare);

end loop;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RAISE\_APPLICATION\_ERROR(-20001,'Nu exista categoria specificata in baza de date');

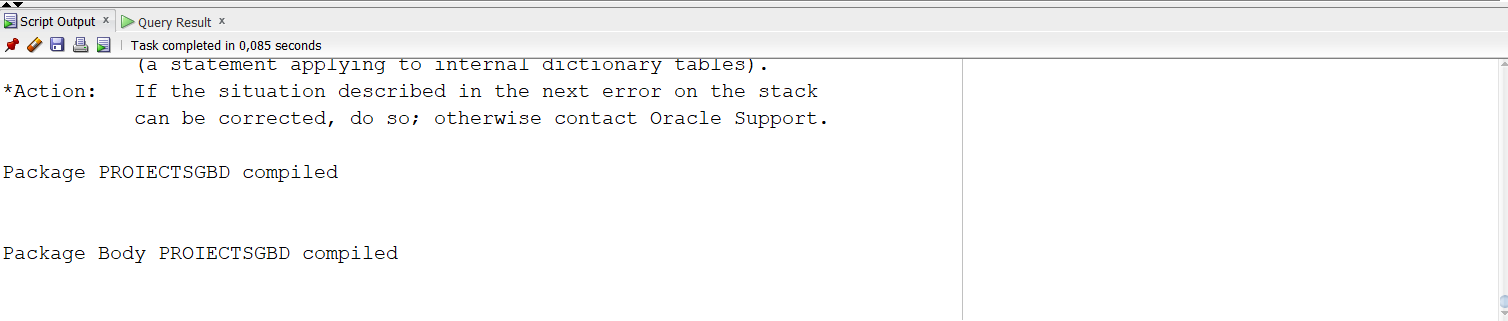
WHEN TOO\_MANY\_ROWS THEN

RAISE\_APPLICATION\_ERROR(-20002,'Exista mai multe categorii cu numele specificat in baza de date!');

end f9;

end proiectSGBD;

/



**14. Definiți un pachet care să includă tipuri de date complexe și obiecte necesare unui flux de acțiuni integrate, specifice bazei de date definite (minim 2 tipuri de date, minim 2 funcții, minim 2 proceduri).**

/

create or replace package ex14 as

TYPE useri IS TABLE OF users.userid%type;

type postari is record(

ptitlu post.title%type,

pdata post.postdate%type

);

procedure useriDinRomania;

procedure modificaNume;

function postariDinLunaAceasta return number;

function titluSiData(cat category.categoryid%type) return number;

end ex14;

/

/

CREATE OR REPLACE PACKAGE BODY ex14 AS

--sa se afiseze utilizatorii care locuiesc in Romania

procedure useriDinRomania

is

nume users.lastname%type;

prenume users.firstname%type;

numarUseri number;

idUseri useri;

cursor Cusers is

select u.userid

from users u, region r

where u.regionid=r.regionid and lower(r.state)='romania';

begin

--adaugam userii in tablou

open Cusers;

fetch Cusers bulk collect into iduseri;

close Cusers;

numarUseri := iduseri.count;

for i in 1..numarUseri loop

select firstname,lastname into prenume,nume

from users

where userid=iduseri(i);

DBMS\_OUTPUT.PUT\_LINE(prenume || ' ' || nume);

end loop;

end useriDinRomania;

--sa se modifice titlul postarilor care nu au cometarii. Titlul le va aparea in DB cu majuscule.

--sa se afiseze numarul de postari modificate.

procedure modificaNume

is

nr number;

begin

nr:=0;

for i in (select title titlu

from post

where postid not in (

select postid

from comments

)) loop

nr := nr+1;

update post

set title = upper(title)

where title=i.titlu;

end loop;

DBMS\_OUTPUT.PUT\_LINE('S-au modificat ' || nr || ' postari!');

end modificaNume;

--Functie care intorce numarul postarilor din luna curenta.

function postariDinLunaAceasta return number is

numar number;

begin

select count(postid) into numar

from post,dual

where to\_char(postdate,'MM') = (to\_char(sysdate,'MM'));

return numar;

end postariDinLunaAceasta;

--Sa se afiseze titlul si data postarilor dintr-o categorie data ca parametru. Sa se returneze numarul lor.

function titluSiData(cat category.categoryid%type) return number is

postrec postari;

nr number :=0;

cursor cPost is

select title, postdate

from post

where categoryid=cat;

begin

open cpost;

loop

fetch cpost into postrec;

DBMS\_OUTPUT.PUT\_LINE('Postare cu titlul ' || postrec.ptitlu || ' din data de ' || postrec.pdata);

nr:=nr+1;

exit when cpost%notfound;

end loop;

close cpost;

return nr;

end titlusidata;

end ex14;

/

/

BEGIN

ex14.useridinromania();

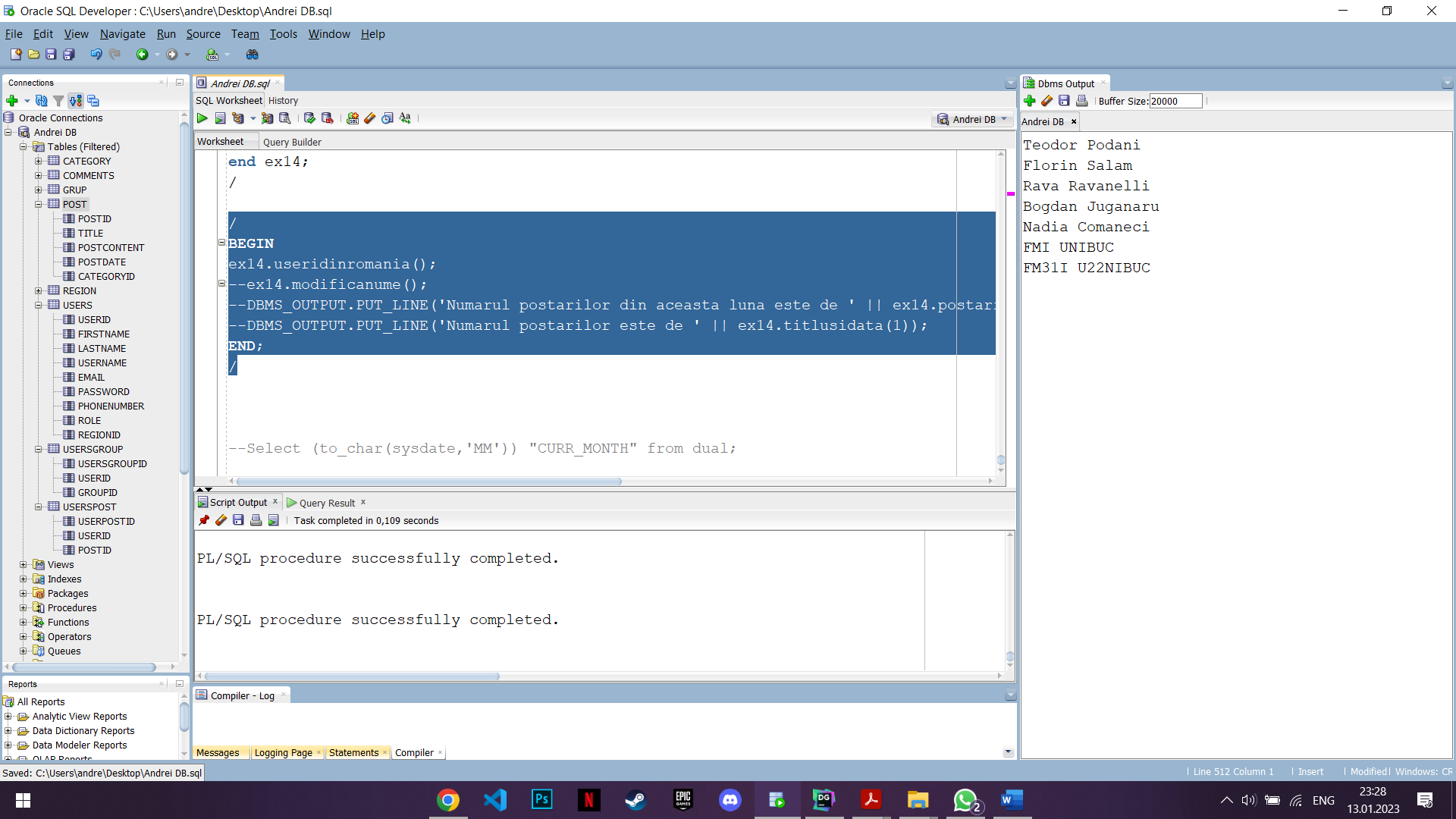
--ex14.modificanume();

--DBMS\_OUTPUT.PUT\_LINE('Numarul postarilor din aceasta luna este de ' || ex14.postaridinlunaaceasta());

--DBMS\_OUTPUT.PUT\_LINE('Numarul postarilor este de ' || ex14.titlusidata(1));

END;

/



/

BEGIN

--ex14.useridinromania();

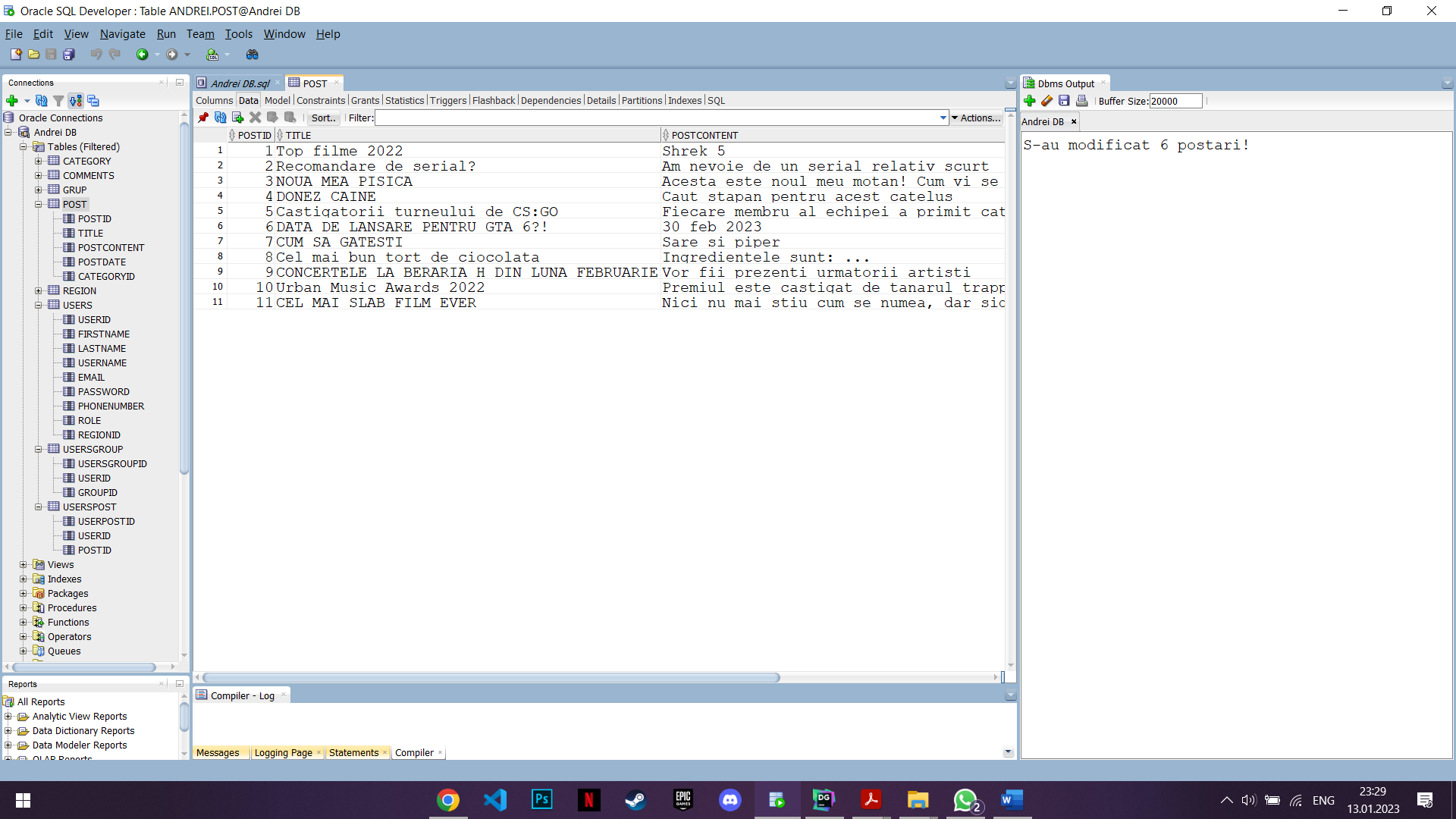
ex14.modificanume();

--DBMS\_OUTPUT.PUT\_LINE('Numarul postarilor din aceasta luna este de ' || ex14.postaridinlunaaceasta());

--DBMS\_OUTPUT.PUT\_LINE('Numarul postarilor este de ' || ex14.titlusidata(1));

END;

/



Se poate că titlul postărilor a fost modificat, fiind scris cu majuscule.

/

BEGIN

--ex14.useridinromania();

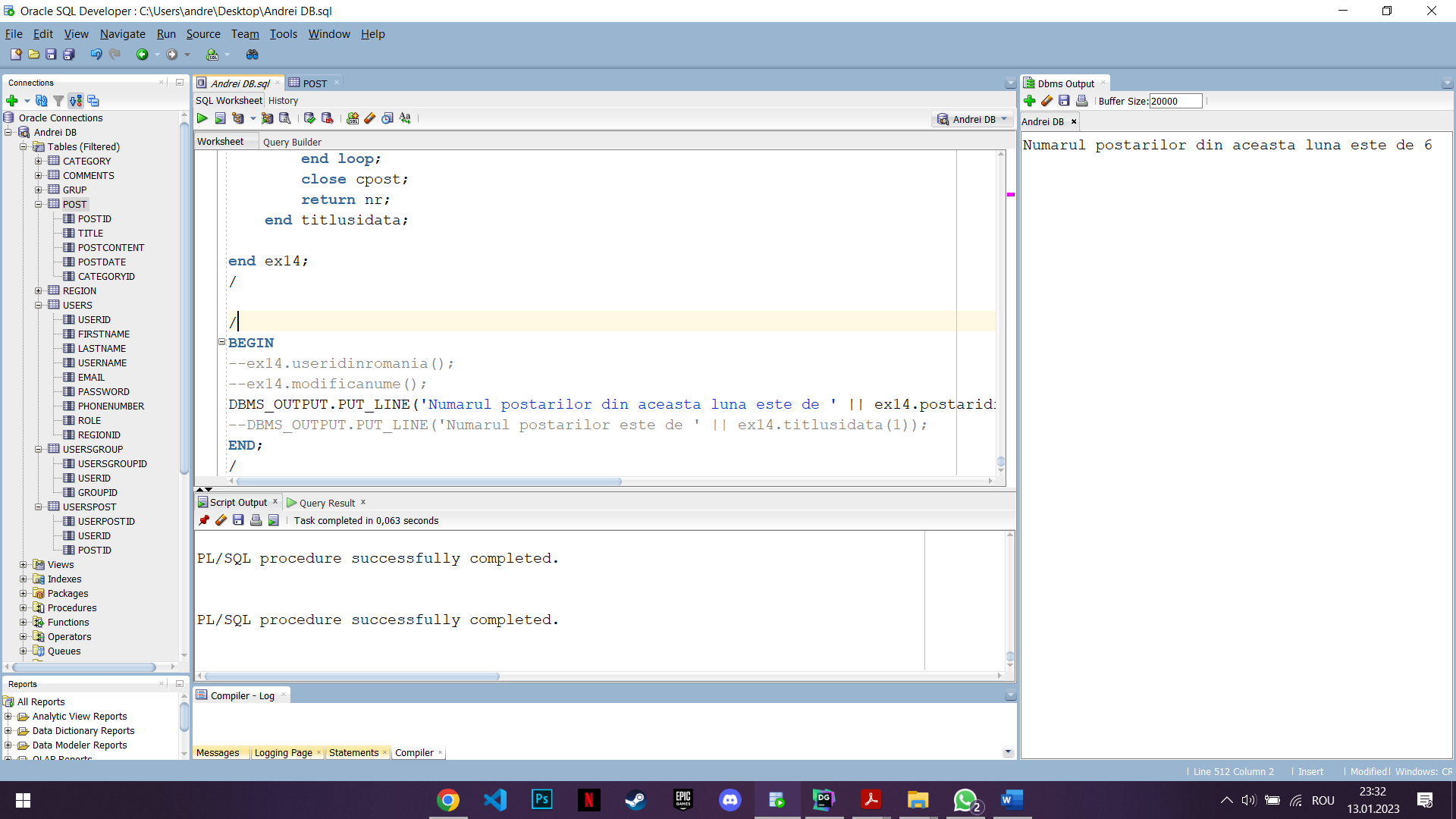
--ex14.modificanume();

DBMS\_OUTPUT.PUT\_LINE('Numarul postarilor din aceasta luna este de ' || ex14.postaridinlunaaceasta());

--DBMS\_OUTPUT.PUT\_LINE('Numarul postarilor este de ' || ex14.titlusidata(1));

END;

/



/

BEGIN

--ex14.useridinromania();

--ex14.modificanume();

--DBMS\_OUTPUT.PUT\_LINE('Numarul postarilor din aceasta luna este de ' || ex14.postaridinlunaaceasta());

DBMS\_OUTPUT.PUT\_LINE('Numarul postarilor este de ' || ex14.titlusidata(1));

END;

/

