PROIECT SISTEME DE GESTIUNE A BAZELOR DE DATE

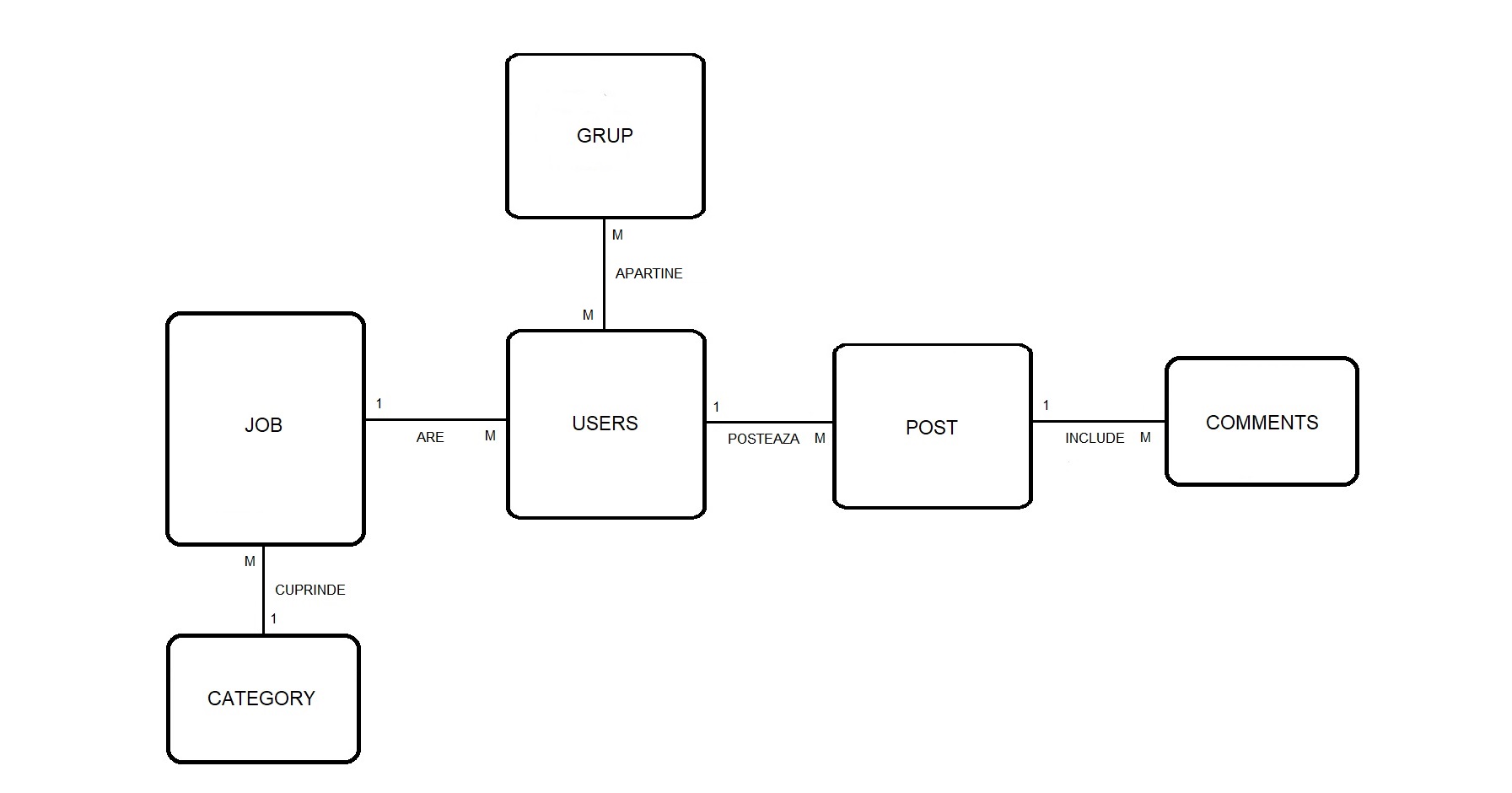
RETEA DE SOCIALIZARE PENTRU ANGAJARE

HORCEAG CRISTIAN-ANDREI – GRUPA 241

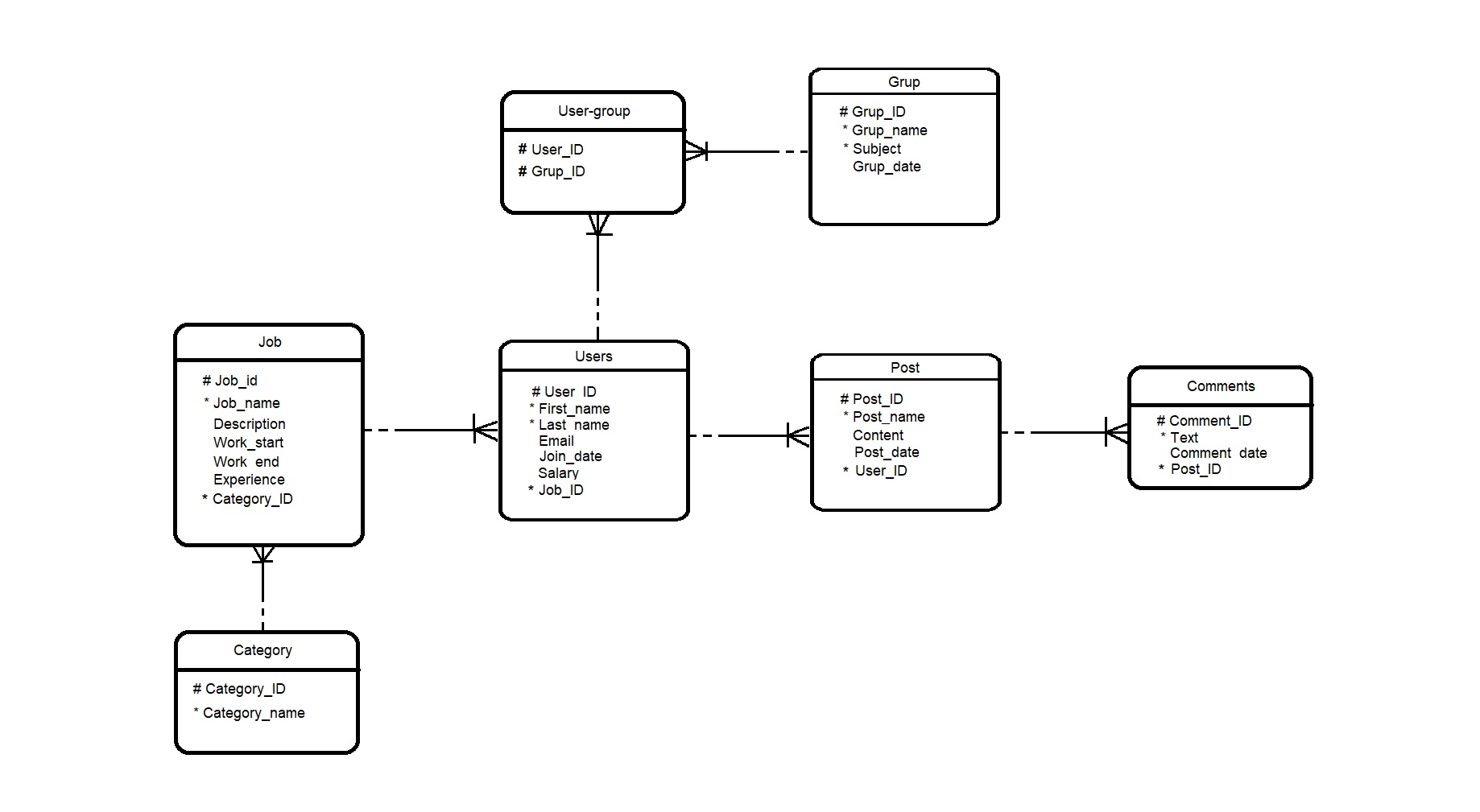
1. Prezentati pe scurt baza de date (utilitatea ei) :

Baza de date are rolul de a stoca informatiile unei retele de socializare create cu scopul gasirii unui loc de munca. In cadrul acesteia utilizatorii se pot alatura mai multor grupuri cu subiecte potrivite intereselor lor si pot publica postari la care pot adauga comentarii. Joburile vor fi impartite pe categorii pentru a oferi utilizatorilor posibilitatea de a vizualiza locurile de munca din cadrul categoriei preferate.

2. Realizati diagrama entitate-relatie (ERD) :



3.Pornind de la diagrama entitate-relatie realizati diagrama conceptuala a modelului propus, integrand toate atributele necesare :



4. Implementati în Oracle diagrama conceptuala realizata: definiti toate tabelele, implementând toate constrângerile de integritate necesare (chei primare, cheile externe etc). :

-- Creare tabele + primary keys

-- Category

CREATE TABLE category (

category\_id NUMBER(9) NOT NULL,

category\_name VARCHAR2(30) NOT NULL

);

ALTER TABLE category ADD CONSTRAINT category\_pk PRIMARY KEY ( category\_id );

-- Comments

CREATE TABLE Comments (

comment\_id NUMBER(9) NOT NULL,

text VARCHAR2(200) NOT NULL,

comment\_date DATE,

post\_id NUMBER(9) NOT NULL

);

ALTER TABLE Comments

ADD CONSTRAINT comment\_pk PRIMARY KEY ( comment\_id );

-- Grup

CREATE TABLE Grup (

grup\_id NUMBER(9) NOT NULL,

grup\_name VARCHAR2(30) NOT NULL,

subject VARCHAR2(40) NOT NULL,

grup\_date DATE

);

ALTER TABLE Grup ADD CONSTRAINT grup\_pk PRIMARY KEY ( grup\_id );

-- JOB

CREATE TABLE job (

job\_id NUMBER(9) NOT NULL,

job\_name VARCHAR2(40) NOT NULL,

description VARCHAR2(300),

work\_start NUMBER(3),

work\_end NUMBER(3),

experience NUMBER(3),

category\_id NUMBER(9) NOT NULL

);

ALTER TABLE job ADD CONSTRAINT job\_pk PRIMARY KEY ( job\_id);

-- Post

CREATE TABLE post (

post\_id NUMBER(9) NOT NULL,

post\_name VARCHAR2(50) NOT NULL,

content VARCHAR2(300),

post\_date DATE,

user\_id NUMBER(9) NOT NULL

);

ALTER TABLE post

ADD CONSTRAINT post\_pk PRIMARY KEY ( post\_id );

-- User

CREATE TABLE Users (

user\_id NUMBER(9) NOT NULL,

first\_name VARCHAR2(30) NOT NULL,

last\_name VARCHAR2(30) NOT NULL,

email VARCHAR2(30),

join\_date DATE,

salary NUMBER(9),

job\_id NUMBER(9) NOT NULL

);

ALTER TABLE Users

ADD CONSTRAINT user\_pk PRIMARY KEY ( user\_id);

-- User-Group

CREATE TABLE user\_group (

user\_id NUMBER(9) NOT NULL,

grup\_id NUMBER(9) NOT NULL

);

ALTER TABLE user\_group

ADD CONSTRAINT user\_group\_pk PRIMARY KEY ( user\_id, grup\_id);

-- Foreign Keys

-- Comments - Post

ALTER TABLE Comments

ADD CONSTRAINT comment\_post\_fk FOREIGN KEY ( post\_id )

REFERENCES post( post\_id );

-- Job - Category

ALTER TABLE job

ADD CONSTRAINT job\_category\_fk FOREIGN KEY ( category\_id )

REFERENCES category ( category\_id );

-- Post - Users

ALTER TABLE post

ADD CONSTRAINT post\_user\_fk FOREIGN KEY ( user\_id )

REFERENCES Users ( user\_id );

-- User\_group - Grup

ALTER TABLE user\_group

ADD CONSTRAINT user\_group\_group\_fk FOREIGN KEY ( grup\_id )

REFERENCES Grup ( grup\_id );

-- User\_group - Users

ALTER TABLE user\_group

ADD CONSTRAINT user\_group\_user\_fk FOREIGN KEY ( user\_id )

REFERENCES Users ( user\_id );

-- Users - Job

ALTER TABLE Users

ADD CONSTRAINT user\_job\_fk FOREIGN KEY ( job\_id )

REFERENCES job ( job\_id );

5. Adaugati informatii coerente în tabelele create (minim 5 înregistrari pentru fiecare entitate independente; minim 10 înregistrari pentru tabela asociativa). :

-- Inserare date

-- Category

INSERT INTO CATEGORY VALUES (1, 'IT');

INSERT INTO CATEGORY VALUES (2, 'Marketing');

INSERT INTO CATEGORY VALUES (3, 'Sales');

INSERT INTO CATEGORY VALUES (4, 'Politics');

INSERT INTO CATEGORY VALUES (5, 'Education');

-- Job

INSERT INTO Job VALUES (1, 'FrontEnd Developer', 'Developing Web Applications', 8, 16, 2, 1);

INSERT INTO Job VALUES (2, 'BackEnd Developer', 'Developing Servers', 10, 16, 1, 1);

INSERT INTO Job VALUES (3, 'Software Engineer', 'Working to improve Google', 9, 20, 4, 1);

INSERT INTO Job VALUES (4, 'Shop Owner', 'The head of a small company', 9, 20, 4, 2);

INSERT INTO Job VALUES (5, 'Accountant', 'Solving all the paperwork', 8, 16, 1, 2);

INSERT INTO Job VALUES (6, 'Real Estate Agent', 'The man to call when you need a new home', 9, 20, 3, 3);

INSERT INTO Job VALUES (7, 'Lawyer', 'Fighting for justice', 8, 14, 1, 4);

INSERT INTO Job VALUES (8, 'Judge', 'Keeping peace', 8, 14, 5, 4);

INSERT INTO Job VALUES (9, 'Governator', 'One of the most desired jobs in the parliament', 8, 16, 2, 4);

-- Users

INSERT INTO Users VALUES (1, 'Mircea', 'Podani', 'mirceapodani@gmail.com', SYSDATE, 5000, 8);

INSERT INTO Users VALUES (2, 'Alex', 'Popa', 'mirceapodani@gmail.com', '10-07-2007', 5000, 8);

INSERT INTO Users VALUES (3, 'Ionut', 'Dumitru', 'ionutdumitru@gmail.com', '20-10-1970', 2450, 9);

INSERT INTO Users VALUES (4, 'Adrian', 'Murgoci', 'adrianmurgoci@gmail.com', '30-11-2010', 1850, 9);

INSERT INTO Users VALUES (5, 'Ioana', 'Gheorghe', 'ioanagheorghe@gmail.com', '05-01-2017', 5680, 1);

INSERT INTO Users VALUES (6, 'Ana', 'Tanase', 'anatanase@gmail.com', '15-04-2000', 5460, 1);

INSERT INTO Users VALUES (7, 'Bianca', 'Ursu', 'biancaursu@gmail.com', '20-09-2013', 4670, 6);

INSERT INTO Users VALUES (8, 'Eugen', 'Grigore', 'eugengrigore@gmail.com', '16-05-2011', 1210, 2);

INSERT INTO Users VALUES (9, 'Mihai', 'Pralea', 'mihaipralea@gmail.com', '10-08-2019', 1150, 2);

INSERT INTO Users VALUES (10, 'Gabriel', 'Turnu', 'gabrielturnu@gmail.com', '29-11-2021', 1760, 3);

INSERT INTO Users VALUES (11, 'Mihnea', 'Ionescu', 'mihneaionescu@gmail.com', '13-09-2022', 8950, 4);

INSERT INTO Users VALUES (12, 'Daria', 'Jugaru', 'dariajugaru@gmail.com', '01-04-2020', 5760, 4);

INSERT INTO Users VALUES (13, 'Cristian', 'Istrate', 'cristianistrate@gmail.com', '07-03-2004', 9590, 5);

INSERT INTO Users VALUES (14, 'Miruna', 'Adam', 'mirunaadam@gmail.com', '08-09-1994', 2650, 5);

INSERT INTO Users VALUES (15, 'Mihaela', 'Florea', 'mihaelaflorea@gmail.com', '06-12-2016', 2950, 6);

INSERT INTO Users VALUES (16, 'Ecaterina', 'Dragu', 'ecaterinadragu@gmail.com', '23-10-1999', 9870, 7);

INSERT INTO Users VALUES (17, 'Oana', 'Sava', 'oanasava@gmail.com', '21-02-2020', 4680, 7);

INSERT INTO Users VALUES (18, 'Andrei', 'Moldoveanu', 'andreimoldoveanu@gmail.com', '15-09-1997', 3690, 4);

INSERT INTO Users VALUES (19, 'Robert', 'Rusu', 'robertrusu@gmail.com', '10-10-2002', 7830, 3);

INSERT INTO Users VALUES (20, 'Maria', 'Velea', 'mariavelea@gmail.com', '20-05-2022', 6380, 3);

--Post

INSERT INTO POST VALUES (1, 'World Cup', 'The 2022 World Cup will be held in Qatar', '20-05-2022', 20);

INSERT INTO POST VALUES (2, 'Formula 1', 'The next race will take place in Maranello', '24-09-2022', 10);

INSERT INTO POST VALUES (3, 'Sports News', 'Messi won the world cup', '29-08-2022', 14);

INSERT INTO POST VALUES (4, 'Global Warming', 'We are more and more concerned about the rising temperatures', '29-07-2022', 8);

INSERT INTO POST VALUES (5, 'Endangered species', 'The white tiger is in great danger of becoming extinct', '27-10-2022', 18);

INSERT INTO POST VALUES (6, 'Rolland Garros', 'Simona Halep is expected to win the next tournament', '20-01-2022', 2);

INSERT INTO POST VALUES (7, 'Mountain Rally', 'The rally cars have been prepared 2 weeks ahead', '23-09-2022', 9);

--Comments

INSERT INTO COMMENTS VALUES (1, 'Messi is the GOAT', '30-10-2022', 3);

INSERT INTO COMMENTS VALUES (2, 'Hamilton is the GOAT', '31-11-2021', 2);

INSERT INTO COMMENTS VALUES (3, 'Help animals !', '03-12-2020', 4);

INSERT INTO COMMENTS VALUES (4, 'Halep is the GOAT', '06-08-2015', 6);

INSERT INTO COMMENTS VALUES (5, 'I love rally cars !', '17-04-2018', 7);

INSERT INTO COMMENTS VALUES (6, 'Such a great competition ahead', '19-04-2018', 7);

-- Grup

INSERT INTO GRUP VALUES (1, 'Pasionati de masini', 'Masini', '16-05-2011');

INSERT INTO GRUP VALUES (2, 'Fani Real Madrid', 'Fotbal', '19-05-2018');

INSERT INTO GRUP VALUES (3, 'Organizatori Evenimente', 'Petreceri', '19-09-2006');

INSERT INTO GRUP VALUES (4, 'Sneaker Community', 'Adidasi', '16-05-2010');

INSERT INTO GRUP VALUES (5, 'Cycling Community', 'Bicycles', '18-07-2016');

-- User\_group

INSERT INTO USER\_GROUP VALUES (1, 2);

INSERT INTO USER\_GROUP VALUES (2, 3);

INSERT INTO USER\_GROUP VALUES (4, 4);

INSERT INTO USER\_GROUP VALUES (10, 1);

INSERT INTO USER\_GROUP VALUES (11, 2);

INSERT INTO USER\_GROUP VALUES (14, 3);

INSERT INTO USER\_GROUP VALUES (8, 4);

INSERT INTO USER\_GROUP VALUES (9, 3);

INSERT INTO USER\_GROUP VALUES (5, 1);

INSERT INTO USER\_GROUP VALUES (7, 2);

INSERT INTO USER\_GROUP VALUES (13, 4);

INSERT INTO USER\_GROUP VALUES (16, 1);

INSERT INTO USER\_GROUP VALUES (17, 2);

INSERT INTO USER\_GROUP VALUES (12, 4);

INSERT INTO USER\_GROUP VALUES (3, 3);

6. Formulati în limbaj natural o problema pe care sa o rezolvati folosind un subprogram stocat independent care sa utilizeze doua tipuri diferite de colectii studiate. Apelati subprogramul :

-- 6. Sa se retina intr-un record textul comentariului din '17-04-2018' al unei postari specificate prin id

-- si sa se stocheze id urile tuturor comentariilor postarii precizate intr-un alt tip de colectie

create or replace procedure p\_comentarii(cod\_postare post.post\_id % type)

is

TYPE comm\_ids IS VARRAY(100) OF Number;

TYPE comentariu is record (continut comments.text % type);

v\_com comentariu;

v\_comm\_ids comm\_ids;

begin

select text into v\_com

from comments where post\_id = cod\_postare and comment\_date = '17-04-2018';

DBMS\_OUTPUT.PUT\_LINE('Textul comentariului din 17-04-2018 este : ' || v\_com.continut);

select comment\_id bulk collect into v\_comm\_ids

from comments where post\_id = cod\_postare;

DBMS\_OUTPUT.PUT\_LINE('');

DBMS\_OUTPUT.PUT\_LINE('Afisarea id-urilor comentariilor de la postarea ' || cod\_postare || ' :');

FOR i IN v\_comm\_ids.FIRST..v\_comm\_ids.LAST LOOP

DBMS\_OUTPUT.PUT\_LINE('Id-ul cu numarul ' || i || ' este : ' || v\_comm\_ids(i) );

end loop;

end p\_comentarii;

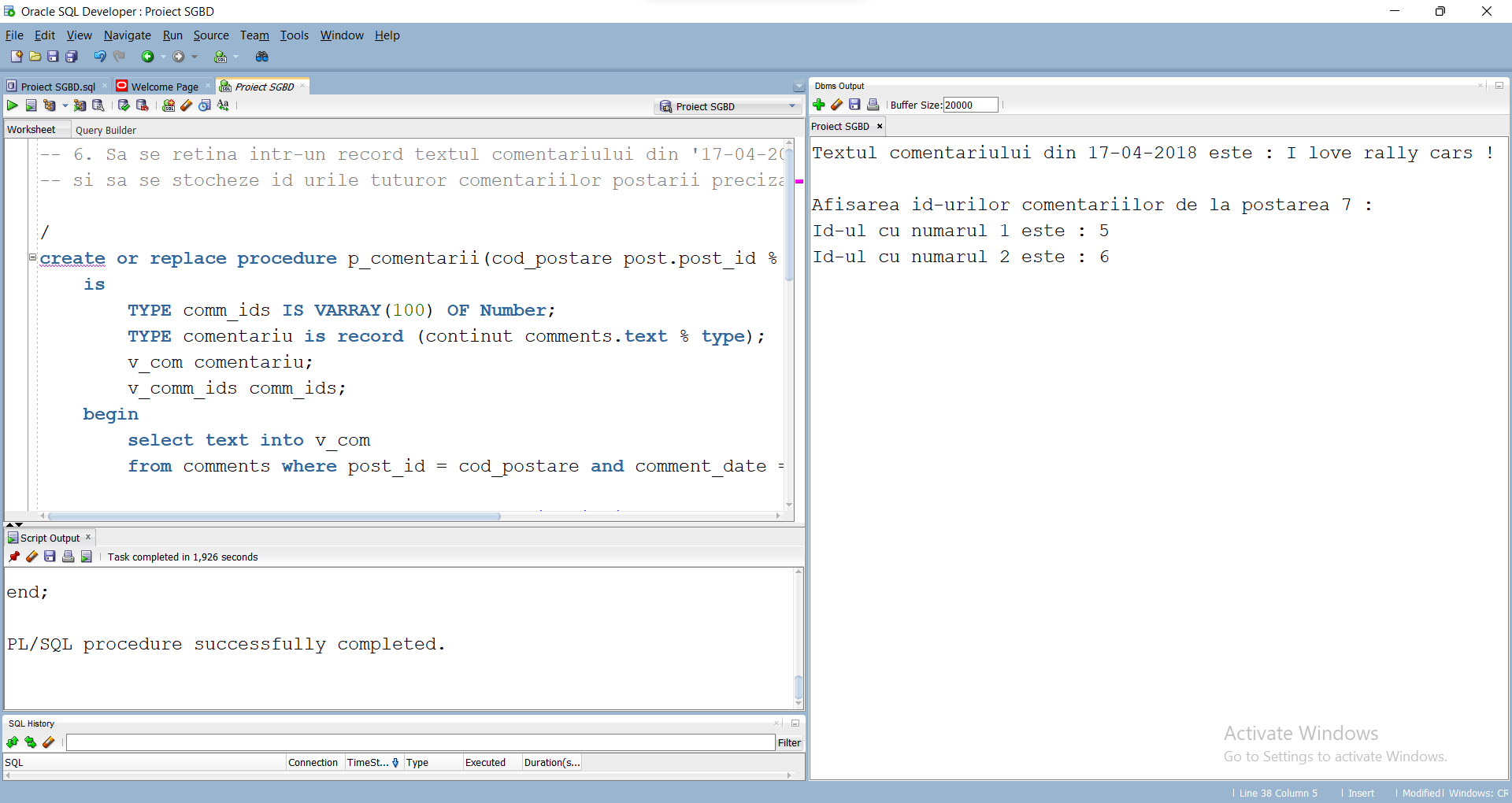
declare

v\_cod\_postare post.post\_id % type := '&p\_cod\_postare';

begin

p\_comentarii(v\_cod\_postare);

end;



7. Formulati în limbaj natural o problema pe care sa o rezolvati folosind un subprogram stocat independent care sa utilizeze 2 tipuri diferite de cursoare studiate, unul dintre acestea fiind cursor parametrizat. Apelati subprogramul. :

-- 7. Obtineti pentru fiecare categorie numele acesteia si numarul de joburi folosind un cursor explicit.

-- Apoi, folositi un ciclu cursor pentru a afisa numele si numele jobului userilor angajati la joburi cu programul de lucru dat de la tastatura

create or replace procedure p\_categorie

is

v\_count number(4);

v\_nume\_categ category.category\_name % TYPE;

CURSOR c1 IS

SELECT category\_name, COUNT(job\_id)

FROM category c join job j on c.category\_id = j.category\_id GROUP BY category\_name;

v\_first\_name\_user users.first\_name % TYPE;

v\_last\_name\_user users.last\_name % TYPE;

v\_nume\_job job.job\_name % TYPE;

v\_work\_start job.work\_start % type := &p\_work\_start;

v\_work\_end job.work\_end % type := &p\_work\_end;

CURSOR c2(v1 number, v2 number) IS

SELECT first\_name f\_name, last\_name l\_name, job\_name j\_name

FROM users u join job j on u.job\_id = j.job\_id where j.work\_start = v1 and j.work\_end = v2;

begin

OPEN c1;

LOOP

FETCH c1 INTO v\_nume\_categ, v\_count;

EXIT WHEN c1 % NOTFOUND;

IF v\_count = 0 THEN

DBMS\_OUTPUT.PUT\_LINE('In categoria '|| v\_nume\_categ ||' nu sunt inscrise joburi');

ELSIF v\_count = 1 THEN

DBMS\_OUTPUT.PUT\_LINE('In categoria '|| v\_nume\_categ ||' este inscris un job');

ELSE

DBMS\_OUTPUT.PUT\_LINE('In categoria '|| v\_nume\_categ ||' sunt inscrise '|| v\_count ||' joburi');

END IF;

END LOOP;

CLOSE c1;

DBMS\_OUTPUT.PUT\_LINE(' ');

DBMS\_OUTPUT.PUT\_LINE('Lista angajatilor cu joburi cu programul dorit');

FOR i in c2(v\_work\_start, v\_work\_end) LOOP

DBMS\_OUTPUT.PUT\_LINE('Angajatul ' || i.f\_name || ' ' || i.l\_name || ' lucreaza ca ' || i.j\_name);

END LOOP;

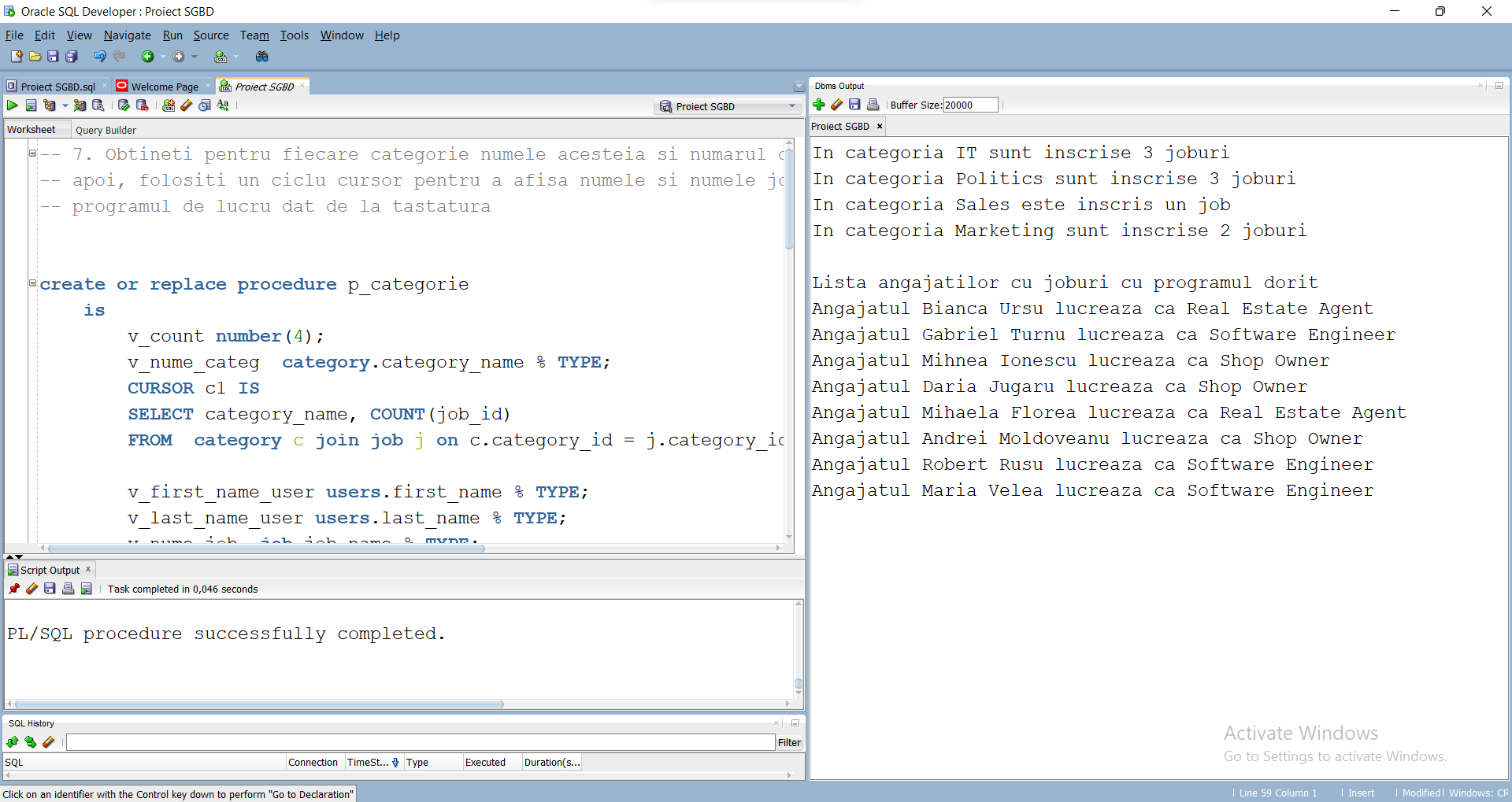
end p\_categorie;

DECLARE

BEGIN

p\_categorie;

END;



8. Formulati în limbaj natural o problema pe care sa o rezolvati folosind un subprogram stocat independent de tip functie care sa utilizeze într-o singura comanda SQL 3 dintre tabelele definite. Definiti minim 2 exceptii. Apelati subprogramul astfel încât sa evidentiati toate cazurile tratate. :

-- 8. Sa se se afiseze media salariilor userilor dintr-un grup cu subiect introdus de la tastatura

CREATE OR REPLACE FUNCTION f\_medie\_salarii(subiect VARCHAR2) RETURN NUMBER IS

salariu users.salary % type;

nume\_grup grup.grup\_name % type;

BEGIN

select grup\_name into nume\_grup from grup

where upper(subject) = upper(subiect);

SELECT avg(salary) into salariu from

users u join user\_group ug on u.user\_id = ug.user\_id

join grup g on ug.grup\_id = g.grup\_id

WHERE upper(subject) = upper(subiect);

return salariu;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

DBMS\_OUTPUT.PUT\_LINE('Nu exista grup cu subiectul introdus !');

WHEN TOO\_MANY\_ROWS THEN

DBMS\_OUTPUT.PUT\_LINE('Exista mai multe grupuri cu subiectul introdus !');

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE('Alta eroare!');

END f\_medie\_salarii;

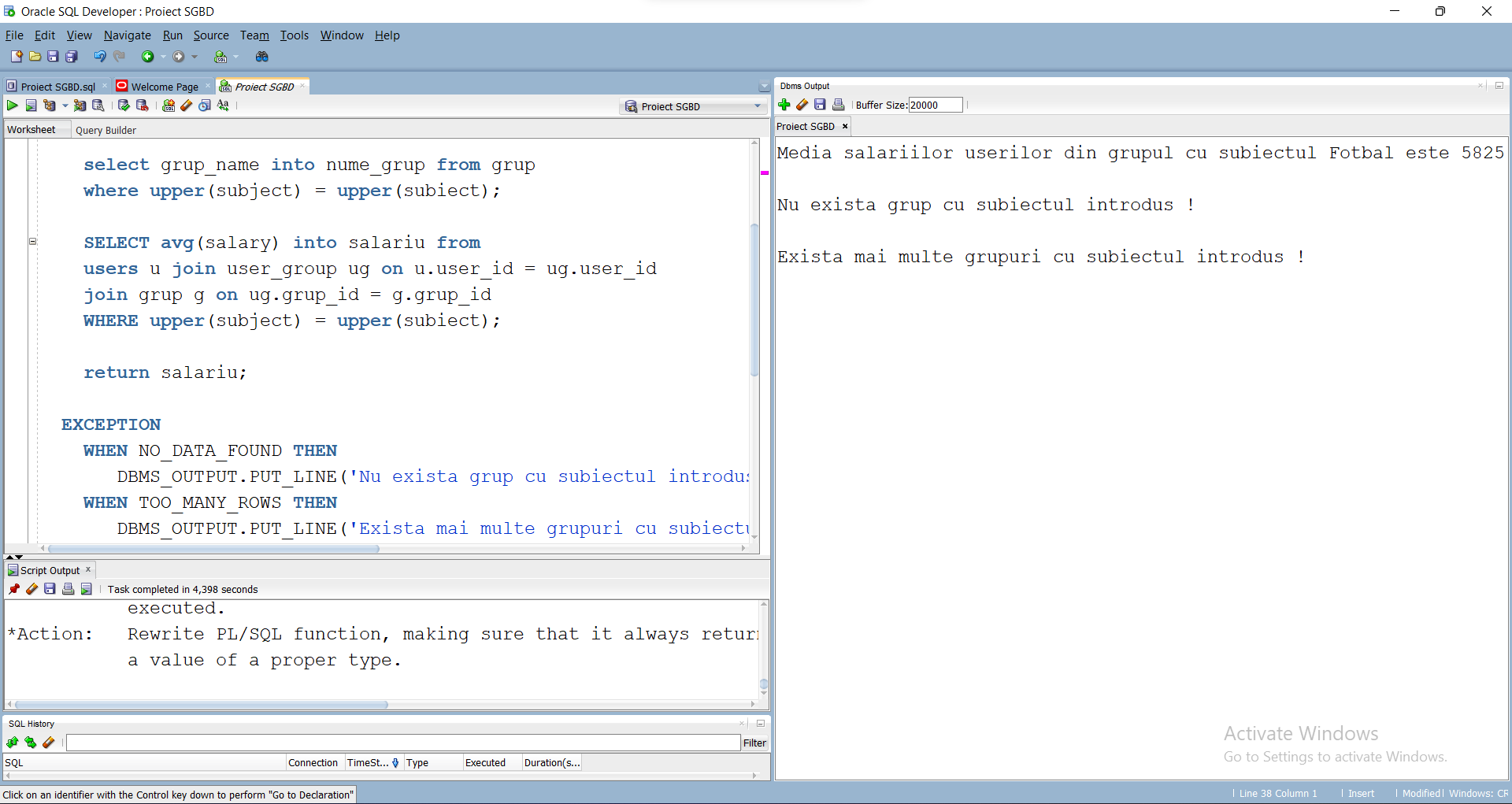
DECLARE

v\_subiect grup.subject % TYPE := '&p\_subiect';

BEGIN

DBMS\_OUTPUT.PUT\_LINE('Media salariilor userilor din grupul cu subiectul ' || v\_subiect || ' este ' || f\_medie\_salarii(v\_subiect));

END;



9. Formulati în limbaj natural o problema pe care sa o rezolvati folosind un subprogram stocat independent de tip procedura care sa utilizeze într-o singura comanda SQL 5 dintre tabelele definite. Tratati toate exceptiile care pot aparea, incluzând exceptiile NO\_DATA\_FOUND si TOO\_MANY\_ROWS. Apelati subprogramul astfel încât sa evidentiati toate cazurile tratate :

-- 9. Sa se afiseze numele, numele grupului, numele postarii si continutul comentariului userilor dintr-un grup precizat prin subiect

CREATE OR REPLACE procedure proc(subiect VARCHAR2)

IS

nume\_grup2 grup.grup\_name % type;

prenume users.first\_name % type;

nume users.last\_name % type;

nume\_grup grup.grup\_name % type;

nume\_postare post.post\_name %type;

comm\_text comments.text % type;

BEGIN

select grup\_name into nume\_grup2 from grup

where upper(subject) = upper(subiect);

SELECT

first\_name, last\_name, grup\_name, post\_name, text

into prenume, nume, nume\_grup, nume\_postare, comm\_text

FROM grup g

join user\_group ug on g.grup\_id = ug.grup\_id

join users u on ug.user\_id = u.user\_id

join post p on u.user\_id = p.user\_id

join comments c on c.post\_id = p.post\_id

where upper(subject) = upper(subiect);

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

DBMS\_OUTPUT.PUT\_LINE(nume\_grup) ;

DBMS\_OUTPUT.PUT\_LINE(nume\_postare);

DBMS\_OUTPUT.PUT\_LINE(comm\_text);

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

DBMS\_OUTPUT.PUT\_LINE('Nu exista un grup cu subiectul introdus / un user cu toate campurile cerute nenule');

RAISE\_APPLICATION\_ERROR(-20003,'Nu exista un grup cu subiectul introdus / un user cu toate campurile cerute nenule');

WHEN TOO\_MANY\_ROWS THEN

DBMS\_OUTPUT.PUT\_LINE('Exista mai multe grupuri cu subiectul introdus / mai multi useri care indeplinesc conditiile');

RAISE\_APPLICATION\_ERROR(-20001,'Exista mai multe grupuri cu subiectul introdus / mai multi useri care indeplinesc conditiile');

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE('Alta eroare!');

RAISE\_APPLICATION\_ERROR(-20002,'Alta eroare!');

END proc;

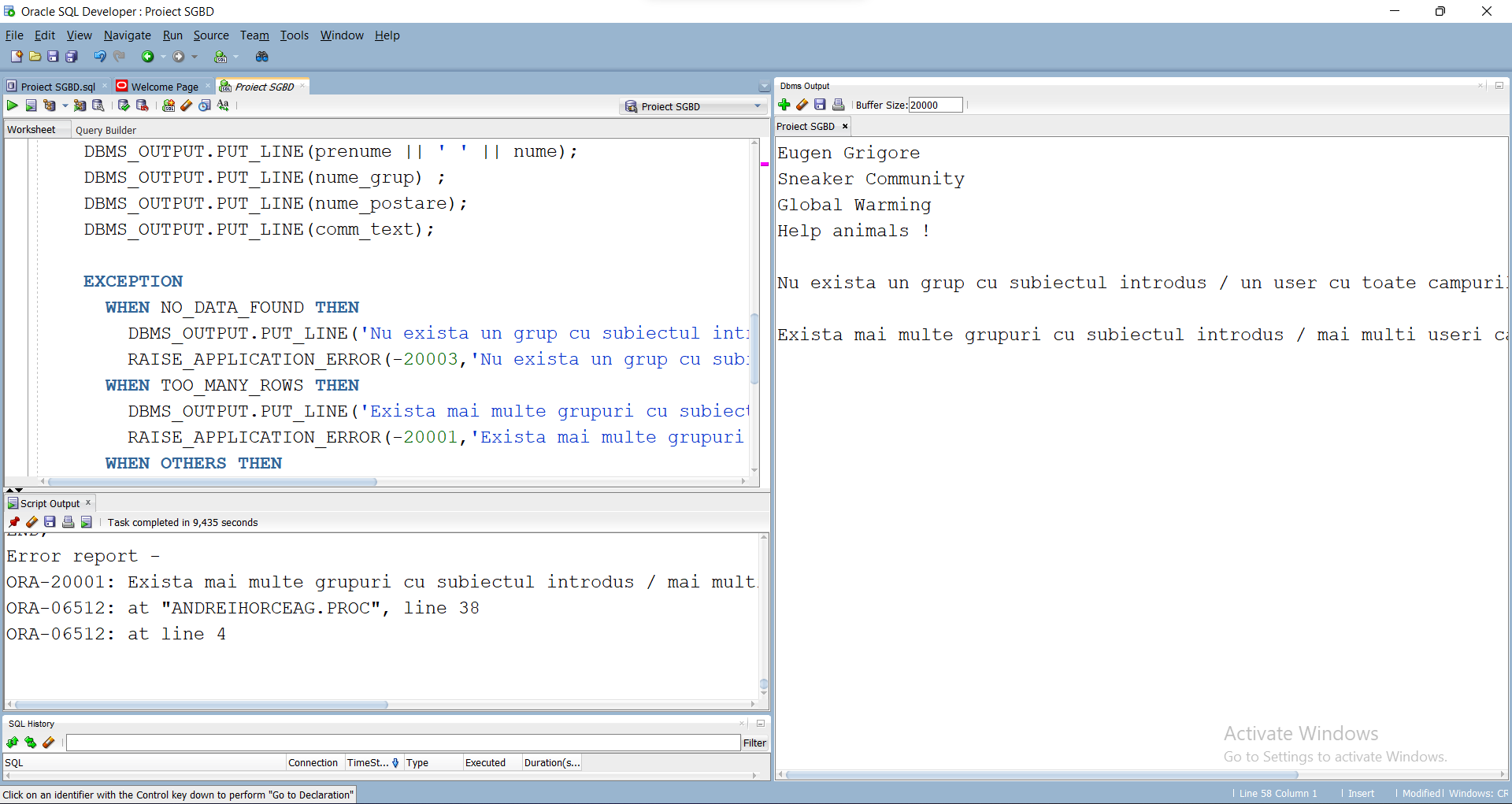
DECLARE

v\_subiect grup.subject % TYPE := '&p\_subiect';

BEGIN

proc(v\_subiect);

END;



10. Definiti un trigger de tip LMD la nivel de comanda. Declansati trigger-ul :

-- 10. Definiti un trigger care va fi declansat afisand un mesaj corespunzator

-- in cazul in care se incearca inserarea, stergerea sau actualizarea tabelei category dupa primele 5 zile ale lunii.

CREATE OR REPLACE TRIGGER trigger\_category

BEFORE INSERT OR DELETE OR UPDATE on category

BEGIN

IF (TO\_CHAR(SYSDATE,'DD') NOT BETWEEN 1 AND 5)

THEN

IF INSERTING THEN

DBMS\_OUTPUT.PUT\_LINE('Inserarea in tabel este permisa doar in primele 5 zile ale lunii !');

RAISE\_APPLICATION\_ERROR(-20001,'Inserarea in tabel este permisa doar in primele 5 zile ale lunii !');

ELSIF DELETING THEN

DBMS\_OUTPUT.PUT\_LINE('Stergerea din tabel este permisa doar in primele 5 zile ale lunii !');

RAISE\_APPLICATION\_ERROR(-20002,'Stergerea din tabel este permisa doar in primele 5 zile ale lunii !');

ELSE

DBMS\_OUTPUT.PUT\_LINE('Actualizarile in tabel sunt permise doar in primele 5 zile ale lunii !');

RAISE\_APPLICATION\_ERROR(-20003,'Actualizarile in tabel sunt permise doar in primele 5 zile ale lunii !');

END IF;

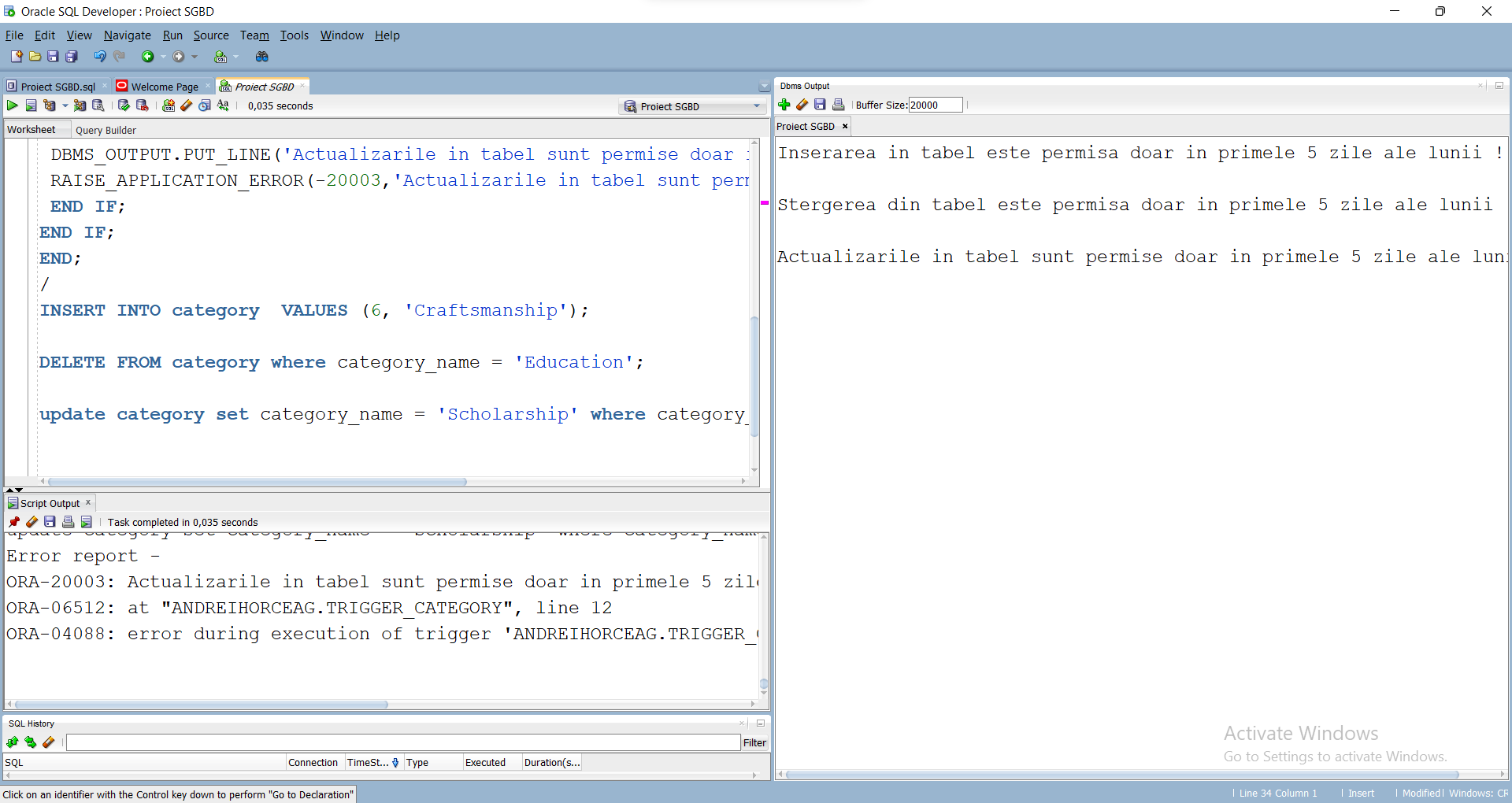
END IF;

END;

INSERT INTO category VALUES (6, 'Craftsmanship');

DELETE FROM category where category\_name = 'Education';

update category set category\_name = 'Scholarship' where category\_name = 'Education';



11. Definiti un trigger de tip LMD la nivel de linie. Declansati trigger-ul :

-- 11. Definiti un trigger care sa fie declansat atunci cand

-- se incearca modificarea subiectului unui grup

CREATE OR REPLACE TRIGGER trigger\_grup

BEFORE UPDATE OF subject ON grup

FOR EACH ROW

WHEN (NEW.subject <> OLD.subject)

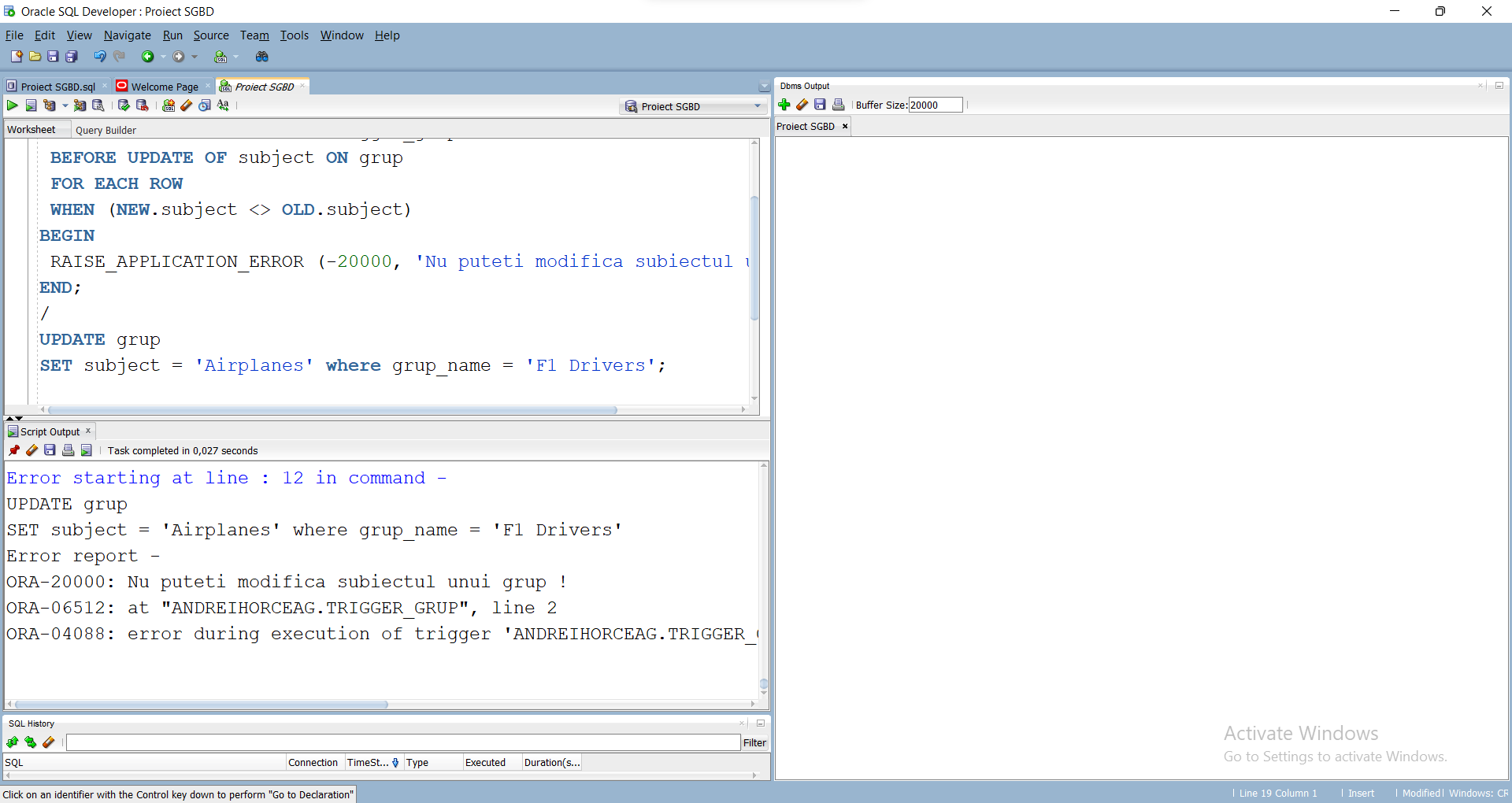
BEGIN

RAISE\_APPLICATION\_ERROR (-20000, 'Nu puteti modifica subiectul unui grup !');

END;

UPDATE grup

SET subject = 'Airplanes' where grup\_name = 'F1 Drivers';



12. Definiti un trigger de tip LDD. Declansati trigger-ul.

-- 12. Sa se defineasca un trigger care sa stocheze datele despre o modificare

-- de tip LDD pe Schema bazei de date

CREATE TABLE schema\_info

(

ddl\_data DATE,

ddl\_user VARCHAR2(30),

tip\_obiect VARCHAR(30),

nume\_obiect VARCHAR2(30),

ddl\_eveniment VARCHAR2(30)

);

CREATE OR REPLACE TRIGGER trigger\_schema

AFTER CREATE OR DROP OR ALTER ON SCHEMA

BEGIN

INSERT INTO schema\_info VALUES

(SYSDATE , SYS.LOGIN\_USER, SYS.DICTIONARY\_OBJ\_TYPE,

SYS.DICTIONARY\_OBJ\_NAME, SYS.SYSEVENT);

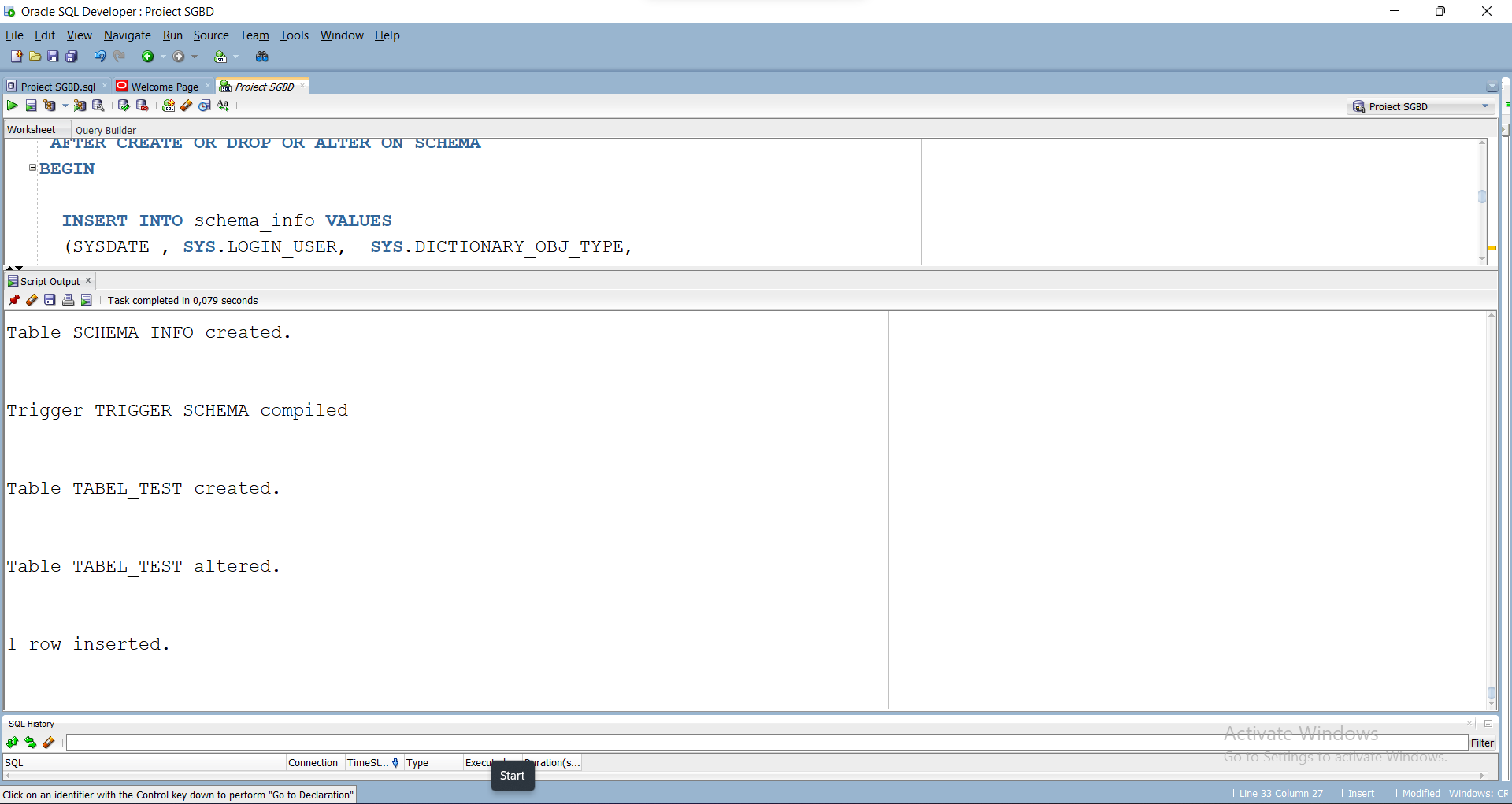
END;

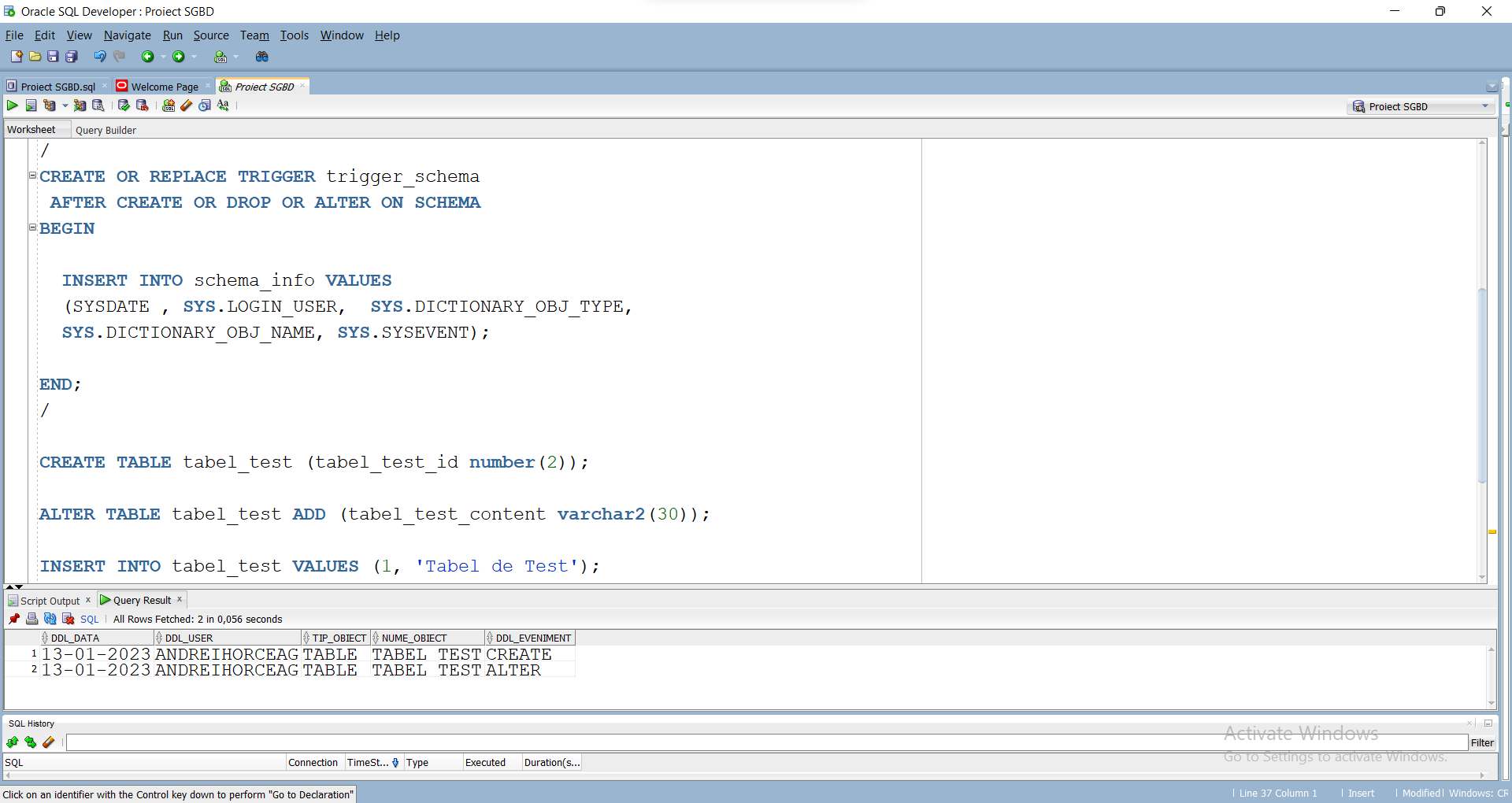
CREATE TABLE tabel\_test (tabel\_test\_id number(2));

ALTER TABLE tabel\_test ADD (tabel\_test\_content varchar2(30));

INSERT INTO tabel\_test VALUES (1, 'Tabel de Test');

select \* from schema\_info;





13. Definiti un pachet care sa contina toate obiectele definite în cadrul proiectului.

CREATE OR REPLACE PACKAGE pachet\_proiect\_SGBD as

-- ex 6

procedure p\_comentarii(cod\_postare post.post\_id % type);

-- ex 7

procedure p\_categorie;

-- ex 8

FUNCTION f\_medie\_salarii(subiect VARCHAR2) RETURN NUMBER;

-- ex 9

procedure proc(subiect VARCHAR2);

end pachet\_proiect\_SGBD;

CREATE OR REPLACE PACKAGE BODY pachet\_proiect\_SGBD AS

-- ex 6

procedure p\_comentarii(cod\_postare post.post\_id % type)

is

TYPE comm\_ids IS VARRAY(100) OF Number;

TYPE comentariu is record (continut comments.text % type);

v\_com comentariu;

v\_comm\_ids comm\_ids;

begin

select text into v\_com

from comments where post\_id = cod\_postare and comment\_date = '17-04-2018';

DBMS\_OUTPUT.PUT\_LINE('Textul comentariului din 17-04-2018 este : ' || v\_com.continut);

select comment\_id bulk collect into v\_comm\_ids

from comments where post\_id = cod\_postare;

DBMS\_OUTPUT.PUT\_LINE('');

DBMS\_OUTPUT.PUT\_LINE('Afisarea id-urilor comentariilor de la postarea ' || cod\_postare || ' :');

FOR i IN v\_comm\_ids.FIRST..v\_comm\_ids.LAST LOOP

DBMS\_OUTPUT.PUT\_LINE('Id-ul cu numarul ' || i || ' este : ' || v\_comm\_ids(i) );

end loop;

end p\_comentarii;

-- ex 7

procedure p\_categorie

is

v\_count number(4);

v\_nume\_categ category.category\_name % TYPE;

CURSOR c1 IS

SELECT category\_name, COUNT(job\_id)

FROM category c join job j on c.category\_id = j.category\_id GROUP BY category\_name;

v\_first\_name\_user users.first\_name % TYPE;

v\_last\_name\_user users.last\_name % TYPE;

v\_nume\_job job.job\_name % TYPE;

v\_work\_start job.work\_start % type := &p\_work\_start;

v\_work\_end job.work\_end % type := &p\_work\_end;

CURSOR c2(v1 number, v2 number) IS

SELECT first\_name f\_name, last\_name l\_name, job\_name j\_name

FROM users u join job j on u.job\_id = j.job\_id where j.work\_start = v1 and j.work\_end = v2;

begin

OPEN c1;

LOOP

FETCH c1 INTO v\_nume\_categ, v\_count;

EXIT WHEN c1 % NOTFOUND;

IF v\_count = 0 THEN

DBMS\_OUTPUT.PUT\_LINE('In categoria '|| v\_nume\_categ ||' nu sunt inscrise joburi');

ELSIF v\_count = 1 THEN

DBMS\_OUTPUT.PUT\_LINE('In categoria '|| v\_nume\_categ ||' este inscris un job');

ELSE

DBMS\_OUTPUT.PUT\_LINE('In categoria '|| v\_nume\_categ ||' sunt inscrise '|| v\_count ||' joburi');

END IF;

END LOOP;

CLOSE c1;

DBMS\_OUTPUT.PUT\_LINE(' ');

DBMS\_OUTPUT.PUT\_LINE('Lista angajatilor cu joburi cu programul dorit');

FOR i in c2(v\_work\_start, v\_work\_end) LOOP

DBMS\_OUTPUT.PUT\_LINE('Angajatul ' || i.f\_name || ' ' || i.l\_name || ' lucreaza ca ' || i.j\_name);

END LOOP;

end p\_categorie;

-- ex 8

FUNCTION f\_medie\_salarii(subiect VARCHAR2) RETURN NUMBER IS

salariu users.salary % type;

nume\_grup grup.grup\_name % type;

BEGIN

select grup\_name into nume\_grup from grup

where upper(subject) = upper(subiect);

SELECT avg(salary) into salariu from

users u join user\_group ug on u.user\_id = ug.user\_id

join grup g on ug.grup\_id = g.grup\_id

WHERE upper(subject) = upper(subiect);

return salariu;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

DBMS\_OUTPUT.PUT\_LINE('Nu exista grup cu subiectul introdus !');

WHEN TOO\_MANY\_ROWS THEN

DBMS\_OUTPUT.PUT\_LINE('Exista mai multe grupuri cu subiectul introdus !');

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE('Alta eroare!');

END f\_medie\_salarii;

-- ex 9

procedure proc(subiect VARCHAR2)

IS

nume\_grup2 grup.grup\_name % type;

prenume users.first\_name % type;

nume users.last\_name % type;

nume\_grup grup.grup\_name % type;

nume\_postare post.post\_name %type;

comm\_text comments.text % type;

BEGIN

select grup\_name into nume\_grup2 from grup

where upper(subject) = upper(subiect);

SELECT

first\_name, last\_name, grup\_name, post\_name, text

into prenume, nume, nume\_grup, nume\_postare, comm\_text

FROM grup g

join user\_group ug on g.grup\_id = ug.grup\_id

join users u on ug.user\_id = u.user\_id

join post p on u.user\_id = p.user\_id

join comments c on c.post\_id = p.post\_id

where upper(subject) = upper(subiect);

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

DBMS\_OUTPUT.PUT\_LINE(nume\_grup) ;

DBMS\_OUTPUT.PUT\_LINE(nume\_postare);

DBMS\_OUTPUT.PUT\_LINE(comm\_text);

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RAISE\_APPLICATION\_ERROR(-20003,'Nu exista un grup cu subiectul introdus / un user cu toate campurile cerute nenule');

WHEN TOO\_MANY\_ROWS THEN

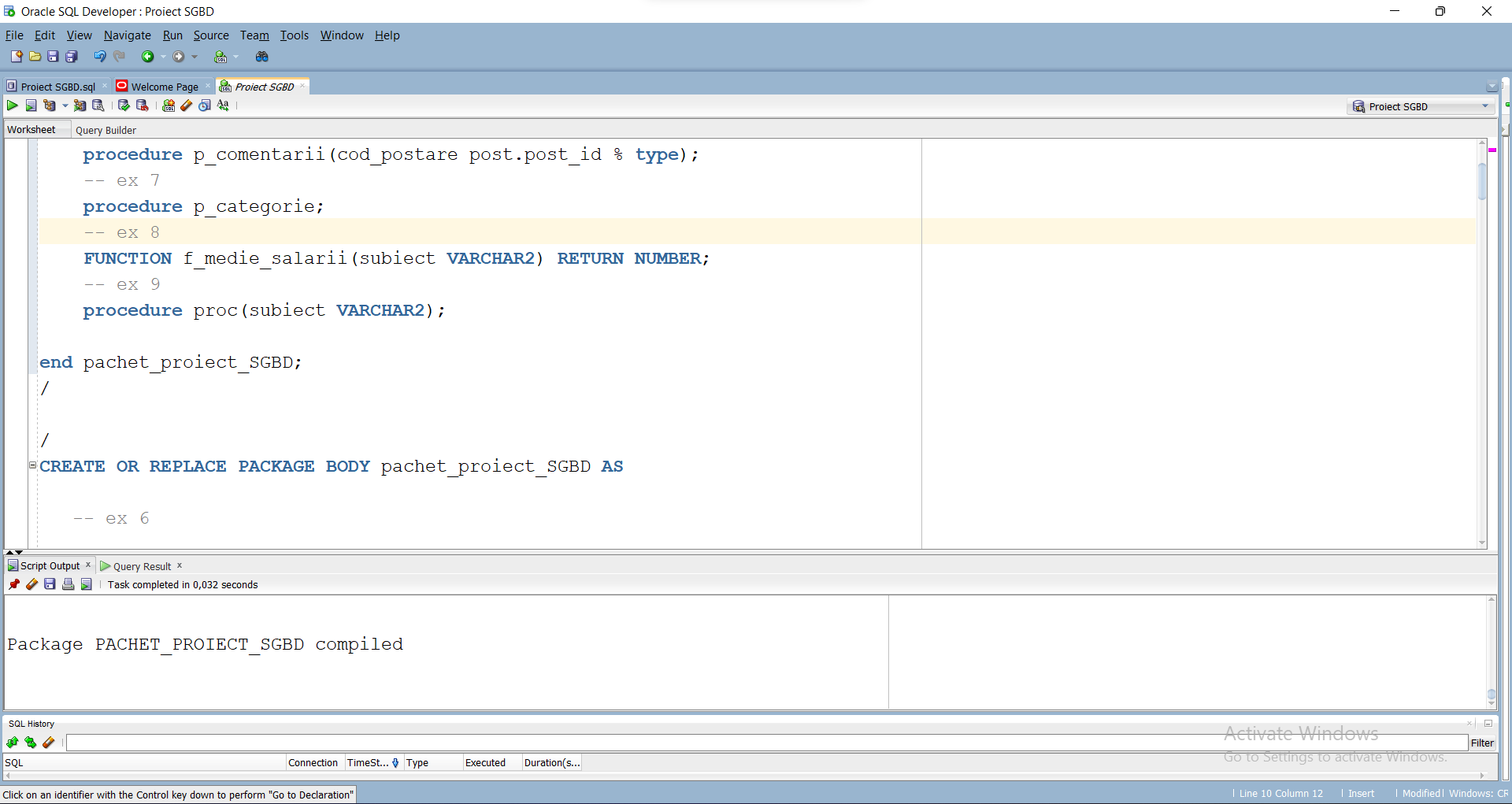
RAISE\_APPLICATION\_ERROR(-20001,'Exista mai multe grupuri cu subiectul introdus');

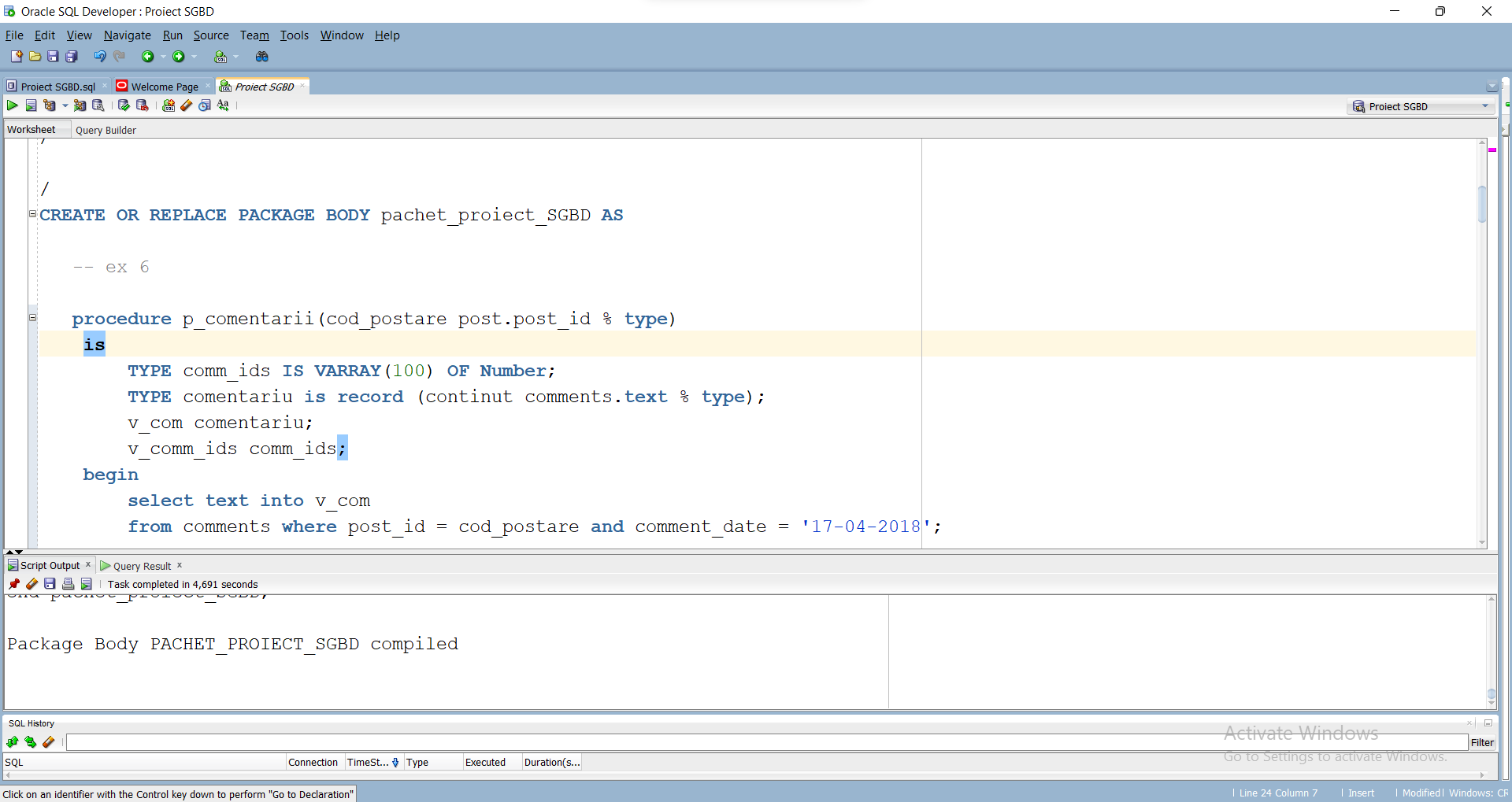
WHEN OTHERS THEN

RAISE\_APPLICATION\_ERROR(-20002,'Alta eroare!');

END proc;

end pachet\_proiect\_SGBD;





14. Definiti un pachet care sa includa tipuri de date complexe si obiecte necesare unui flux de actiuni integrate, specifice bazei de date definite (minim 2 tipuri de date, minim 2 functii, minim 2 proceduri).

CREATE OR REPLACE PACKAGE pachet\_proiect as

procedure p\_1;

procedure p\_2(cod\_grup number);

function f\_grup(cod\_grup number) RETURN VARCHAR2;

function f\_postare(cod\_postare number) RETURN VARCHAR2;

TYPE tablou\_indexat IS TABLE OF NUMBER INDEX BY PLS\_INTEGER;

TYPE inregistrare IS RECORD (salariu users.salary % type);

end pachet\_proiect;

CREATE OR REPLACE PACKAGE BODY pachet\_proiect AS

--///////////////////////////////////////////////

-- sa se afiseze numele unui grup introdus prin cod

function f\_grup(cod\_grup number) RETURN VARCHAR2

IS

nume\_grup grup.grup\_name % type;

BEGIN

select grup\_name into nume\_grup

from grup where grup\_id = cod\_grup;

return nume\_grup;

END f\_grup;

--///////////////////////////////////////////////

-- sa se afiseze numele unei postari introduse prin cod

function f\_postare(cod\_postare number) RETURN VARCHAR2

IS

nume\_postare post.post\_name %type;

BEGIN

select post\_name into nume\_postare

from post where post\_id = cod\_postare;

return nume\_postare;

END f\_postare;

--///////////////////////////////////////////////

-- sa se adauge salariile angajatilor cu id impar si cu programul jobului de la 9 la 20

-- intr-un tabel indexat si apoi sa se afiseze acestea

procedure p\_1

is

t tablou\_indexat;

indice number := 0;

CURSOR c2 IS

SELECT user\_id cod\_user, salary salariu

FROM users u join job j on u.job\_id = j.job\_id where j.work\_start = 9 and j.work\_end = 20;

begin

for i in c2 loop

if i.cod\_user mod 2 = 1 then

indice := indice + 1;

t(indice) := i.salariu;

end if;

end loop;

for i in t.first..t.last loop

DBMS\_OUTPUT.PUT\_LINE('Salariul angajatului este ' || t(i));

end loop;

end p\_1;

--///////////////////////////////////////////////

-- sa se retina intr-un record media salariilor userilor dintr-un grup introdus prin cod

-- si sa se afiseze aceasta

procedure p\_2(cod\_grup number)

is

rec inregistrare;

salariu users.salary % type;

begin

select avg(salary) into rec.salariu

from users u join user\_group ug on u.user\_id = ug.user\_id

join grup g on ug.grup\_id = g.grup\_id

WHERE g.grup\_id = cod\_grup;

DBMS\_OUTPUT.PUT\_LINE('Media salariilor userilor din grupul ' || f\_grup(cod\_grup) || ' este ' || rec.salariu);

end p\_2;

END pachet\_proiect;

begin

pachet\_proiect.p\_1;

DBMS\_OUTPUT.PUT\_LINE('');

pachet\_proiect.p\_2(2);

end;

