PROIECT SGBD

Football – Baciu Rares grupa 243

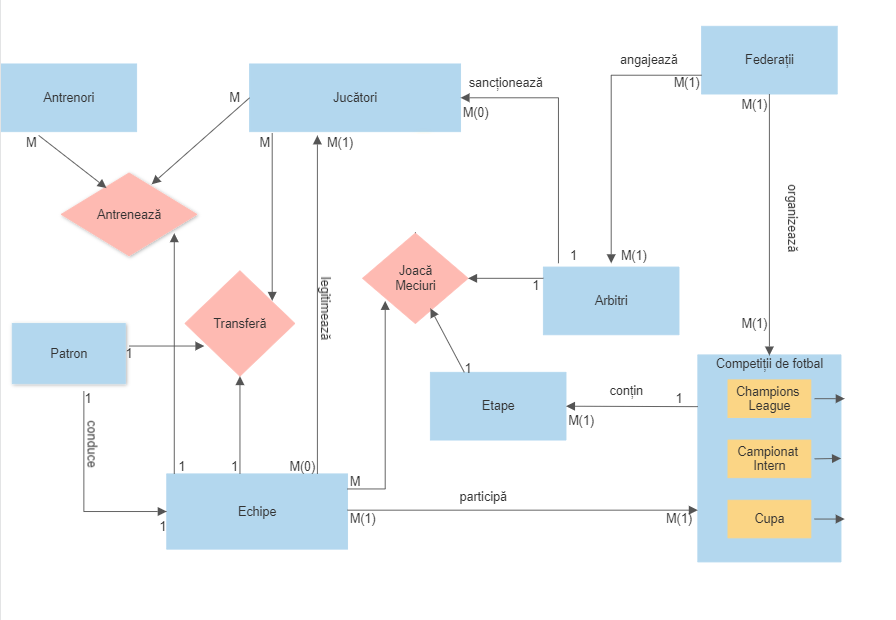
1. Prezentați pe scurt baza de date (utilitatea ei).

În lumea fotbalului există federații ce organizează diferite competiții de fotbal: campionate interne, cupe și Champions League. Competițiile sunt organizate în etape. La fiecare competiție participă echipe de fotbal.

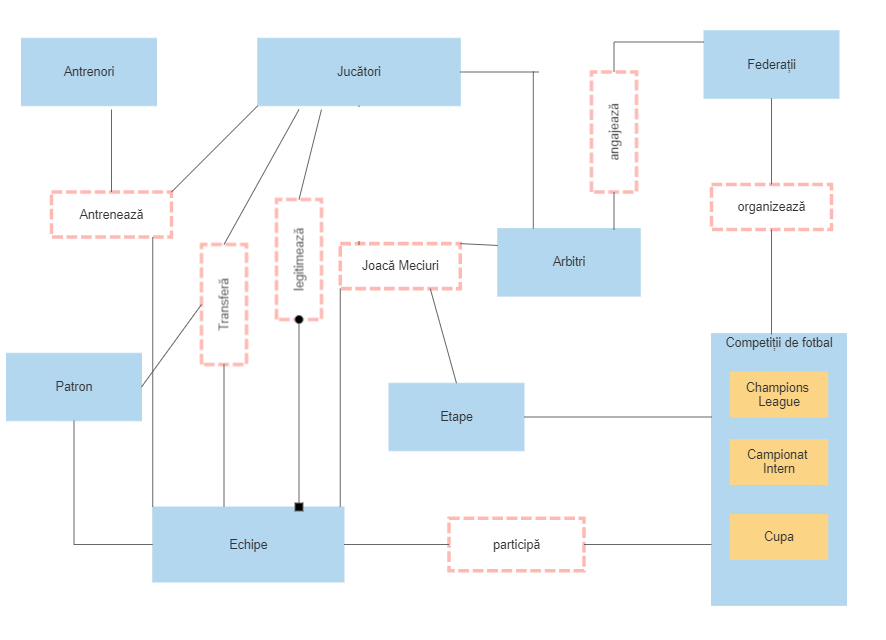
Fiecare echipă de fotbal este condusă de un patron. Acesta face transferuri și aduce jucători pentru a îmbunătăți performanța. Jucătorii sunt legitimati pentru a putea fi pregătiți de antrenorii echipei și pentru a participa la meciuri.

Toate meciurile de fotbal se încadrează într-o etapa dintr-o anumită competiție de fotbal.  
Fiecare partidă este asistată de către arbitri, aceștia fiind angajați de către federații. La meciuri participă echipe alături de jucătorii lor.

1. Realizați diagrama entitate-relație (ERD): entitățile, relațiile și atributele trebuie definite în limba română (vezi curs SGBD / model de diagrama ERD; nu se va accepta alt format).



1. Pornind de la diagrama entitate-relație realizați diagrama conceptuală a modelului propus, integrând toate atributele necesare: entitățile, relațiile și atributele trebuie definite în limba română.



**Fiecare tabel si atributele sale:**

Jucatori(#jucator\_id, nume, prenume, pozitie, numar\_tricou);

Echipe(#echipa\_id. nume, stadion, nr\_trofee, patron\_id);

Patron(#patron\_id, nume, prenume);

Etape(#etapa\_id, competitie\_id);

Antrenori(#antrenor\_id, nume, prenume, specializare);

Arbitri(#arbitru\_id, nume, prenume);

Federatii(#federatie\_id, nume, presedinte);

Competitii de fotbal(#competitie\_id, tara, data\_inceput, data\_final, sponsor);

Antreneaza(#echipa\_id, #antrenor\_id, #jucator\_id, data\_antrenament, tip\_antrenament);

Transfera(#echipa\_id, #patron\_id, #jucator\_id, suma\_transfer, nume\_agent);

Joaca Meciuri(#etapa\_id, #arbitru\_id, gazda, oaspete, data\_meci, stadion);

legitimeaza(#echipa\_id, #jucatori\_id, salariu, ani\_contract);

participa(#echipa\_id, #competitie\_id, poz\_clasament, puncte, goluri\_marcate, goluri\_primtie);

organizeaza(#federatie\_id, #competitie\_id);

angajeaza(#federatie\_id, #arbitru\_id, salariu, ani\_contract);

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

CREATE TABLE JUCATORI(

jucator\_id NUMBER(10) PRIMARY KEY,

nume VARCHAR(20) NOT NULL,

prenume VARCHAR(20) NOT NULL,

pozitie VARCHAR(20) NOT NULL,

numar\_tricou NUMBER(5) NOT NULL

);

ALTER TABLE JUCATORI

ADD CHECK (numar\_tricou >= 1 AND numar\_tricou <= 99);

CREATE TABLE ECHIPE(

echipa\_id NUMBER(10) PRIMARY KEY,

nume VARCHAR(20) NOT NULL,

stadion VARCHAR(20) NOT NULL,

nr\_trofee NUMBER(5),

patron\_id NUMBER(10) NOT NULL

);

CREATE TABLE PATRON(

patron\_id NUMBER(10) PRIMARY KEY,

nume VARCHAR(20) NOT NULL,

prenume VARCHAR(20) NOT NULL

);

CREATE TABLE ANTRENORI(

antrenor\_id NUMBER(10) PRIMARY KEY,

nume VARCHAR(20) NOT NULL,

prenume VARCHAR(20) NOT NULL,

specializare VARCHAR(20) NOT NULL

);

CREATE TABLE ARBITRI(

arbitru\_id NUMBER(10) PRIMARY KEY,

nume VARCHAR(20) NOT NULL,

prenume VARCHAR(20) NOT NULL

);

CREATE TABLE FEDERATII(

federatie\_id NUMBER(10) PRIMARY KEY,

nume VARCHAR(20) NOT NULL,

presedinte VARCHAR(20) NOT NULL

);

CREATE TABLE COMPETITII(

competitie\_id NUMBER(10) PRIMARY KEY,

tara VARCHAR(20) NOT NULL,

data\_inceput DATE NOT NULL,

data\_final DATE NOT NULL,

sponsor VARCHAR(20)

);

ALTER TABLE COMPETITII

ADD nume VARCHAR(20) NOT NULL;

CREATE TABLE ETAPE(

etapa\_id NUMBER(10) PRIMARY KEY,

competitie\_id NUMBER(10) NOT NULL

);

CREATE TABLE ANTRENEAZA(

echipa\_id NUMBER(10),

antrenor\_id NUMBER(10),

jucator\_id NUMBER(10),

data\_antrenament DATE NOT NULL,

tip\_antrenament VARCHAR(20) NOT NULL,

CONSTRAINT pk\_antr PRIMARY KEY(echipa\_id, antrenor\_id, jucator\_id)

);

CREATE TABLE TRANSFERA(

echipa\_id NUMBER(10),

patron\_id NUMBER(10),

jucator\_id NUMBER(10),

suma\_transfer FLOAT(10),

nume\_agent VARCHAR(20) NOT NULL

);

ALTER TABLE TRANSFERA

ADD CONSTRAINT pk\_transf PRIMARY KEY(echipa\_id, jucator\_id, patron\_id);

CREATE TABLE JOACA\_MECIURI(

etapa\_id NUMBER(10),

arbitru\_id NUMBER(10),

gazda NUMBER(10) NOT NULL,

oaspete NUMBER(10) NOT NULL,

data\_meci DATE NOT NULL,

stadion VARCHAR(20) NOT NULL,

CONSTRAINT pk\_jm PRIMARY KEY(etapa\_id, arbitru\_id)

);

CREATE TABLE LEGITIMEAZA(

echipa\_id NUMBER(10),

jucator\_id NUMBER(10),

salariu FLOAT(10),

ani\_contract NUMBER(10) NOT NULL,

CONSTRAINT pk\_legit PRIMARY KEY(echipa\_id, jucator\_id),

CHECK (ani\_contract >=1 AND ani\_contract <= 6)

);

CREATE TABLE PARTICIPA(

echipa\_id NUMBER(10),

competitie\_id NUMBER(10),

poz\_clasament NUMBER(10) NOT NULL,

puncte NUMBER(10),

goluri\_marcate NUMBER(10),

goluri\_primite NUMBER(10),

CONSTRAINT pk\_participa PRIMARY KEY(echipa\_id, competitie\_id),

CHECK (poz\_clasament >=1 AND poz\_clasament <=20)

);

CREATE TABLE ORGANIZEAZA(

federatie\_id NUMBER(10),

competitie\_id NUMBER(10),

CONSTRAINT pk\_org PRIMARY KEY(federatie\_id, competitie\_id)

);

CREATE TABLE ANGAJEAZA(

federatie\_id NUMBER(10),

arbitru\_id NUMBER(10),

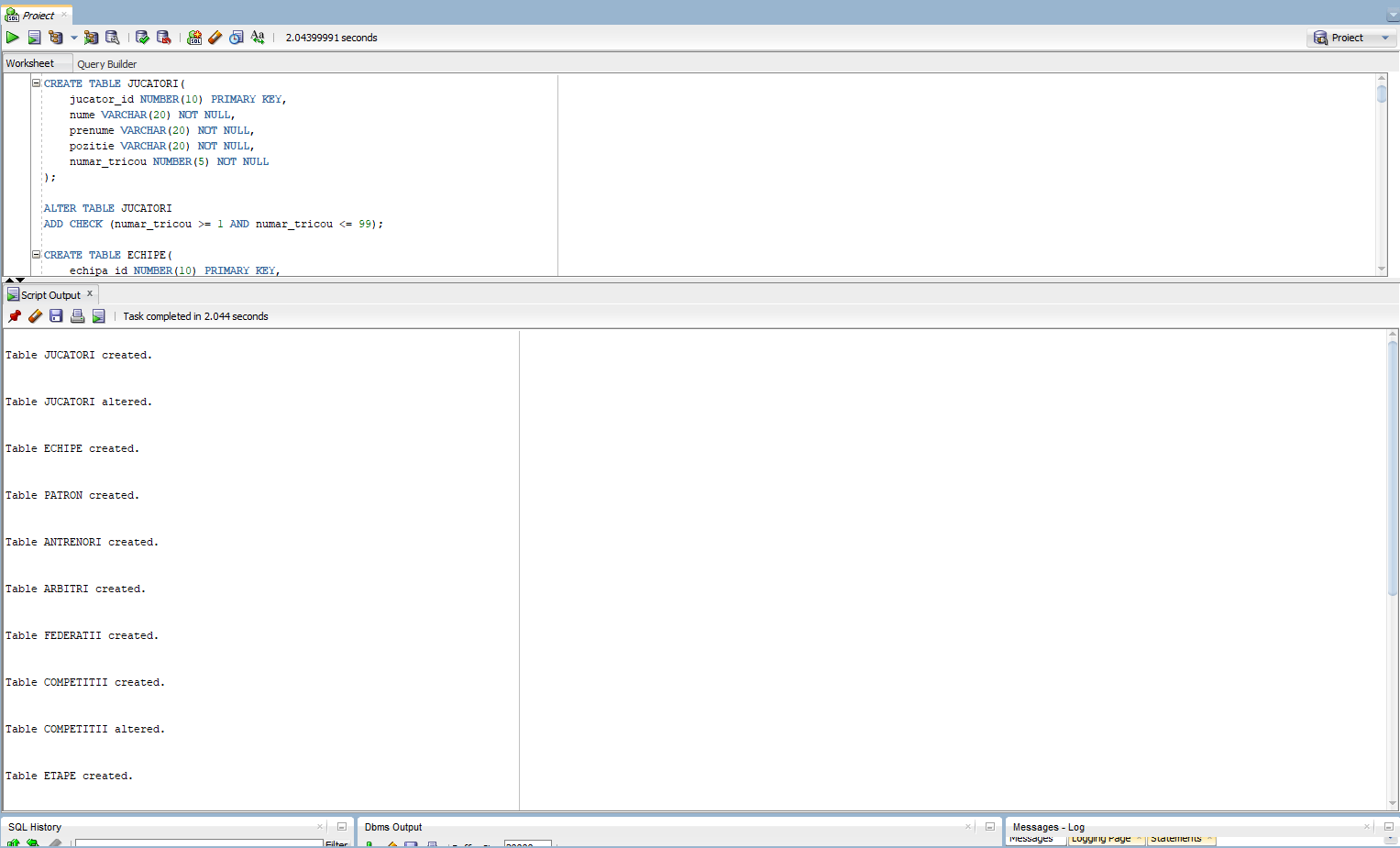
salariu FLOAT(10),

ani\_contract NUMBER(10) NOT NULL,

CONSTRAINT pk\_ang PRIMARY KEY(federatie\_id, arbitru\_id),

CHECK(ani\_contract >= 1 AND ani\_contract <= 6)

);



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

INSERT INTO JUCATORI

values(1, 'BACIU','RARES','ATACANT','10');

INSERT INTO JUCATORI

values(2, 'BOT','GEORGE','ATACANT','9');

INSERT INTO JUCATORI

values(3, 'DAN','DOD','MIJLOCAS','7');

INSERT INTO JUCATORI

values(4, 'DONIU','CRISTIAN','PORTAR','1');

INSERT INTO JUCATORI

values(5, 'MERCU','ADELIN','MIJLOCAS','18');

INSERT INTO JUCATORI

values(6, 'ADONIS','TIK','FUNDAS','4');

INSERT INTO JUCATORI

values(7, 'OBREZ','MERCAN','ATACANT','9');

INSERT INTO JUCATORI

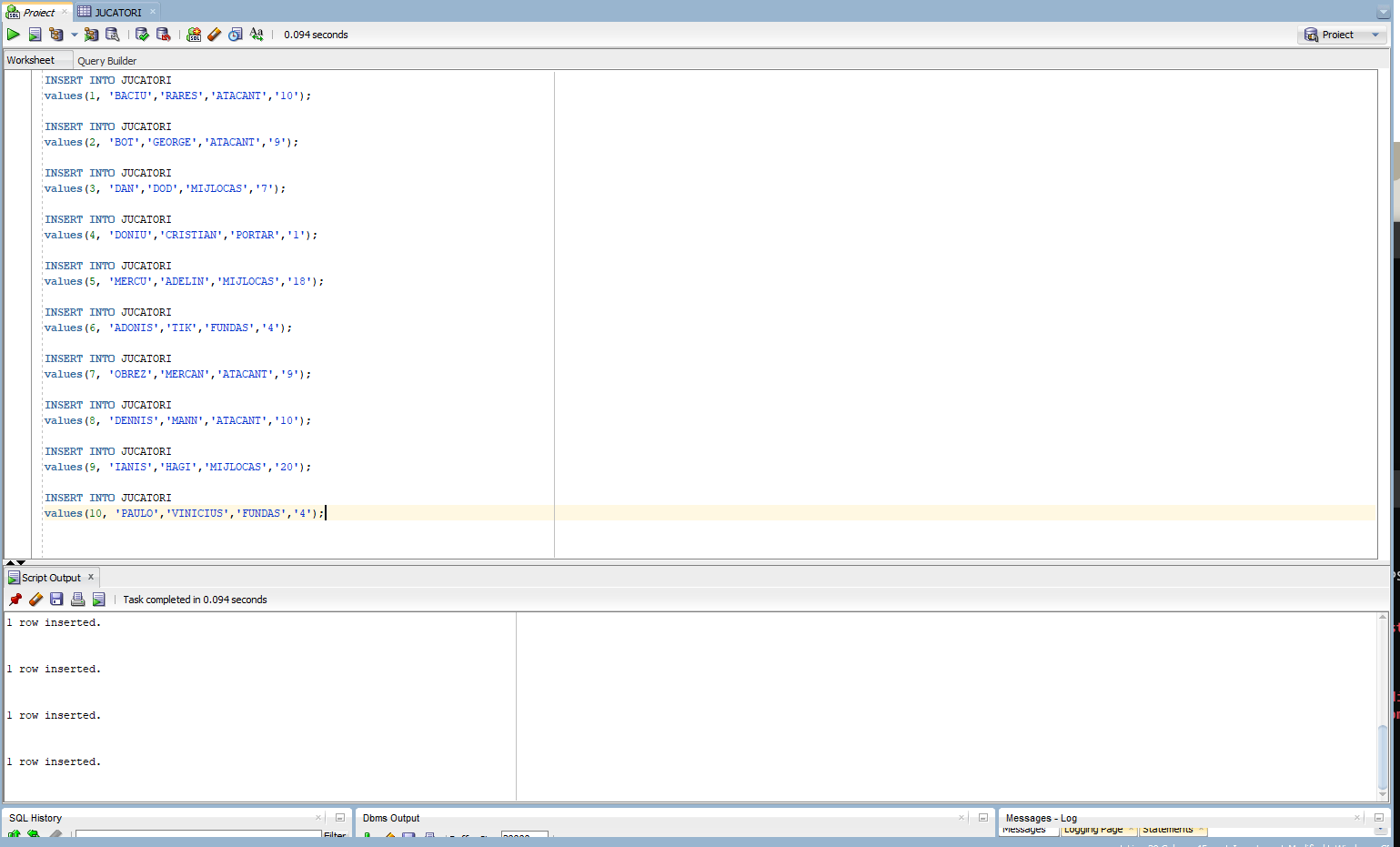
values(8, 'DENNIS','MANN','ATACANT','10');

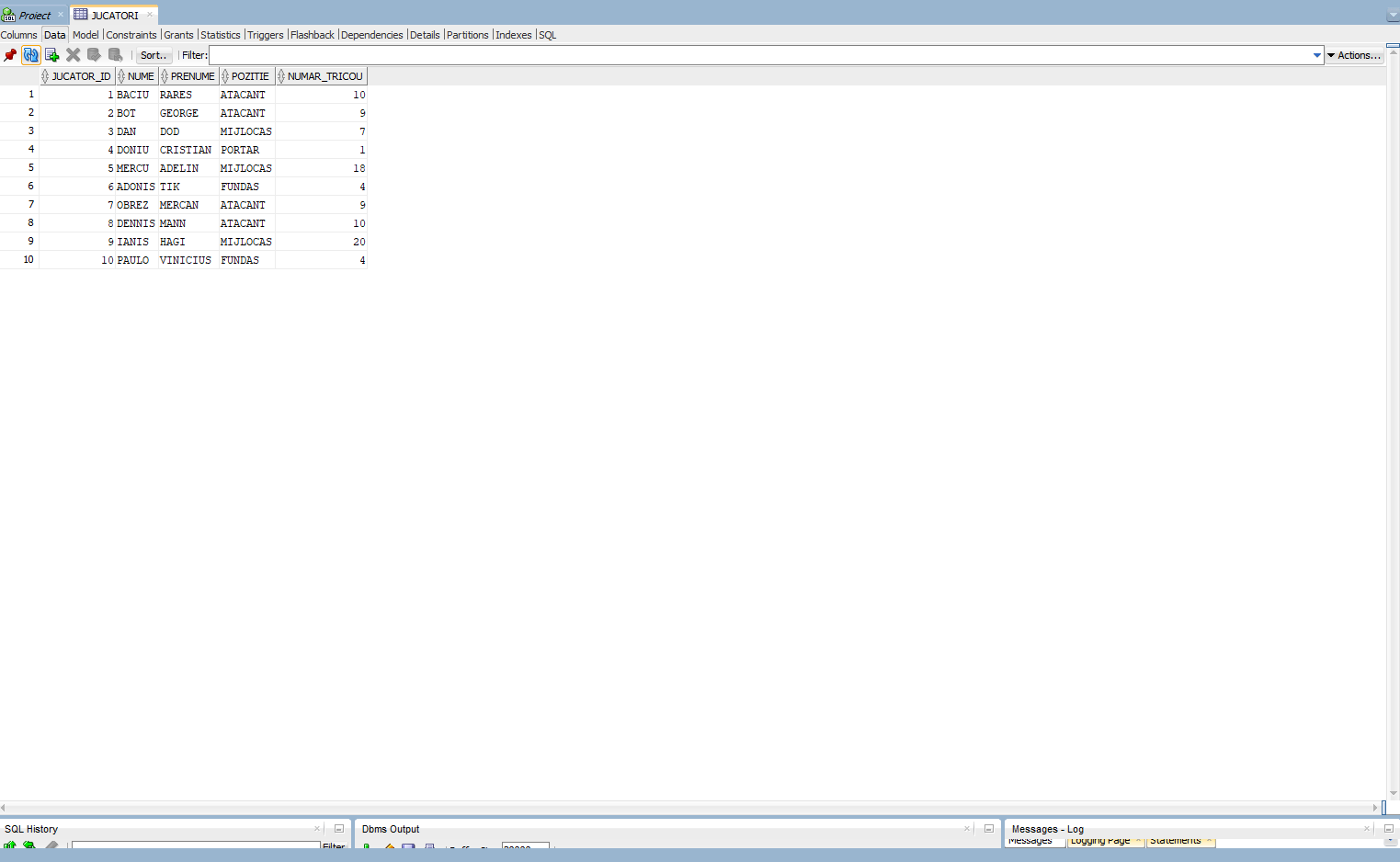
INSERT INTO JUCATORI

values(9, 'IANIS','HAGI','MIJLOCAS','20');

INSERT INTO JUCATORI

values(10, 'PAULO','VINICIUS','FUNDAS','4');





INSERT INTO ECHIPE

values(1,'DINAMO BUCURESTI','STEFAN CEL MARE',18,1);

INSERT INTO ECHIPE

values(2,'FC VASLUI','MUNICIPAL',100,2);

INSERT INTO ECHIPE

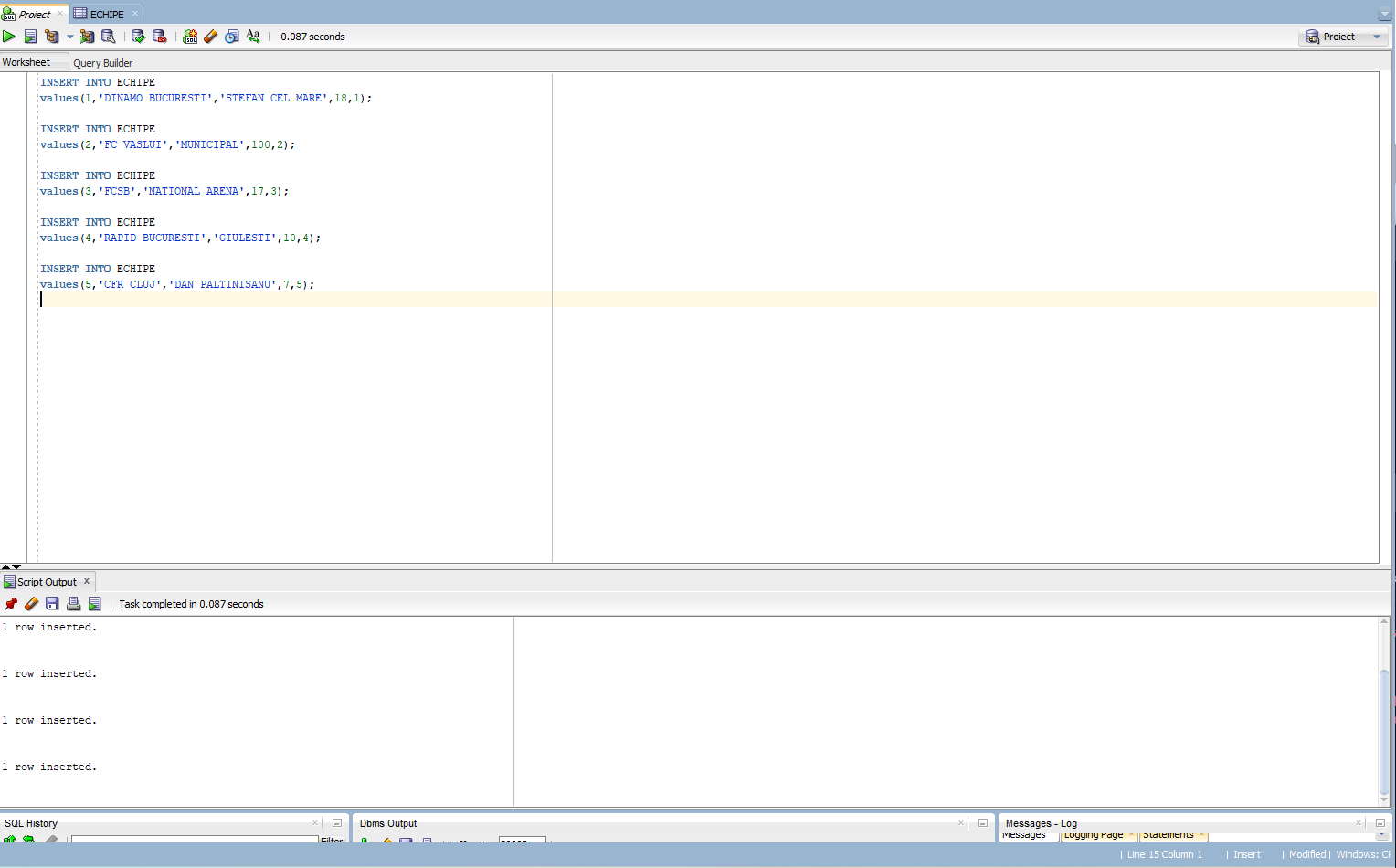
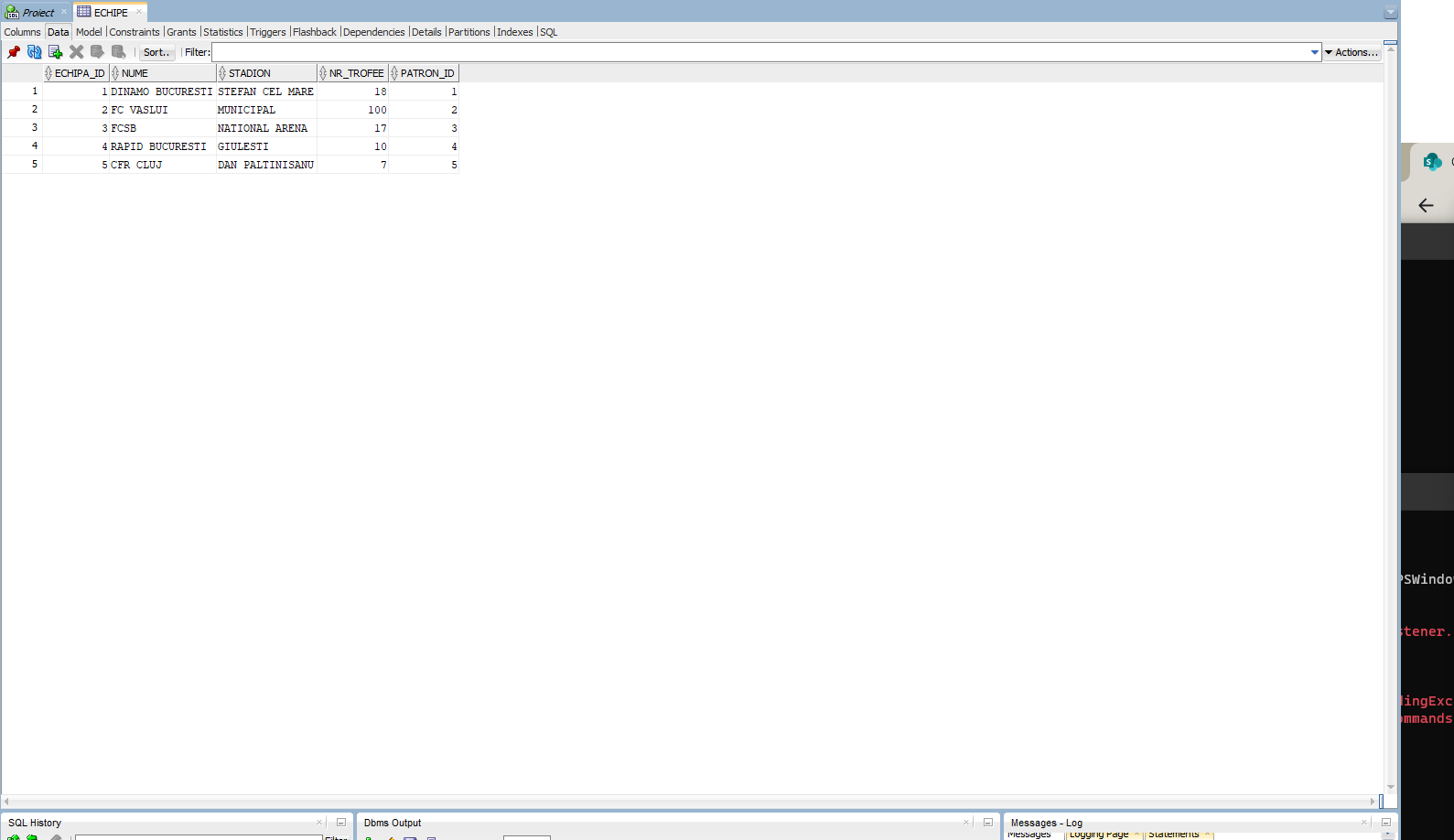
values(3,'FCSB','NATIONAL ARENA',17,3);

INSERT INTO ECHIPE

values(4,'RAPID BUCURESTI','GIULESTI',10,4);

INSERT INTO ECHIPE

values(5,'CFR CLUJ','DAN PALTINISANU',7,5);



INSERT INTO ARBITRI

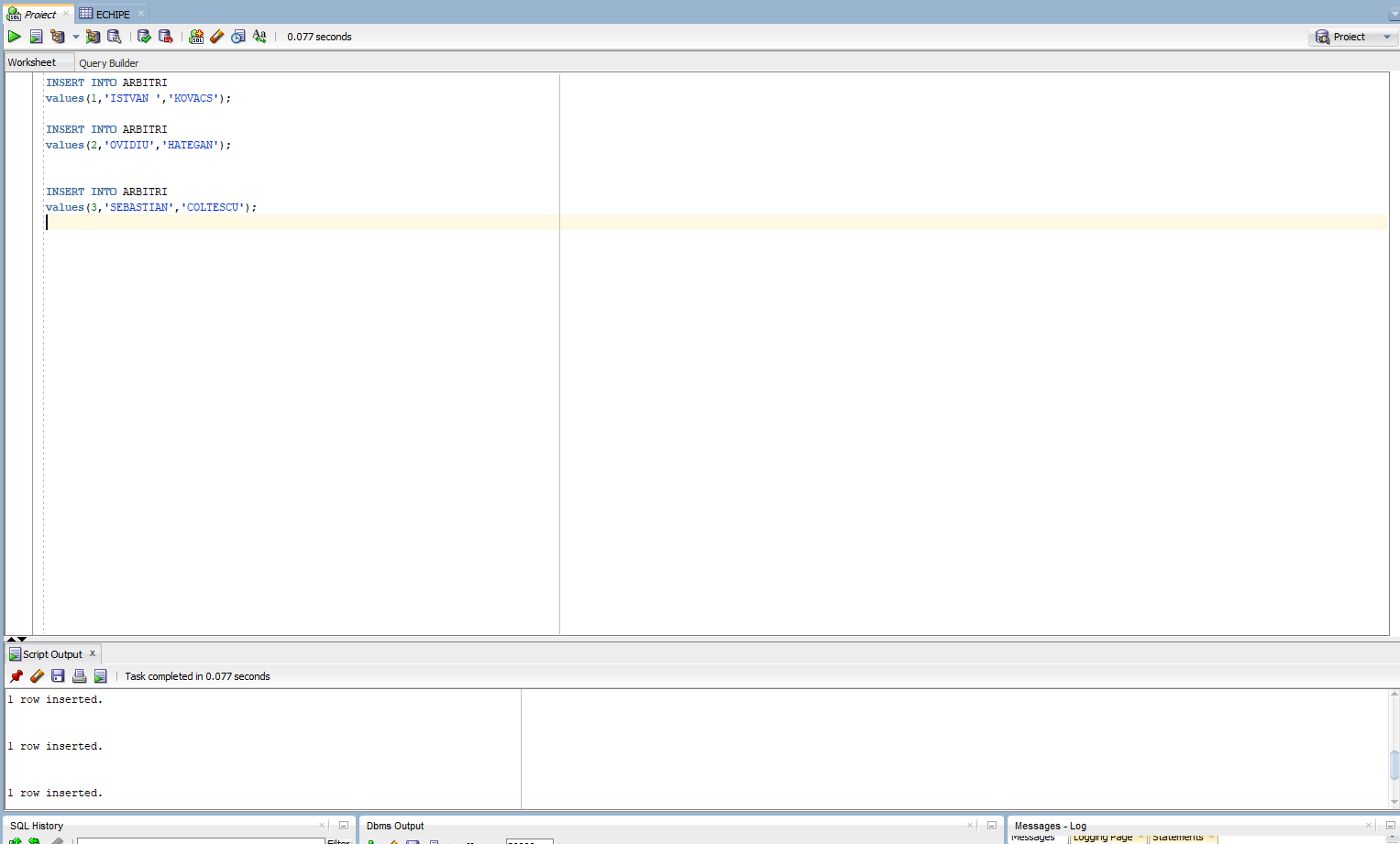
values(1,'ISTVAN ','KOVACS');

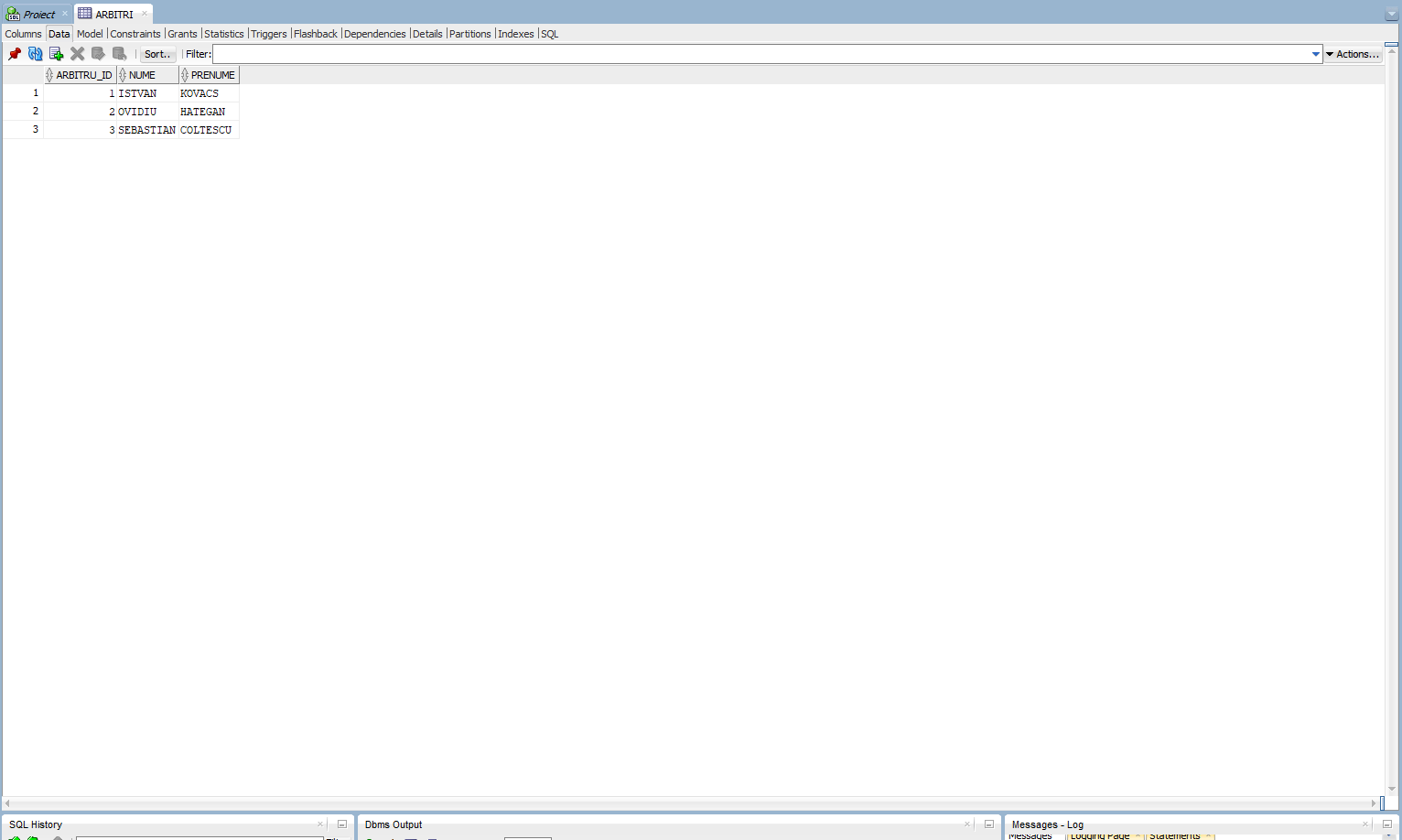
INSERT INTO ARBITRI

values(2,'OVIDIU','HATEGAN');

INSERT INTO ARBITRI

values(3,'SEBASTIAN','COLTESCU');





INSERT INTO COMPETITII

values(1,'ROMANIA','10-AUG-22','10-JUN-22','BERGENBIER','LIGA 1');

INSERT INTO COMPETITII

values(2,'ANGLIA','11-AUG-22','10-MAY-22','BARCLAYS','PREMIER LEAGUE');

INSERT INTO COMPETITII

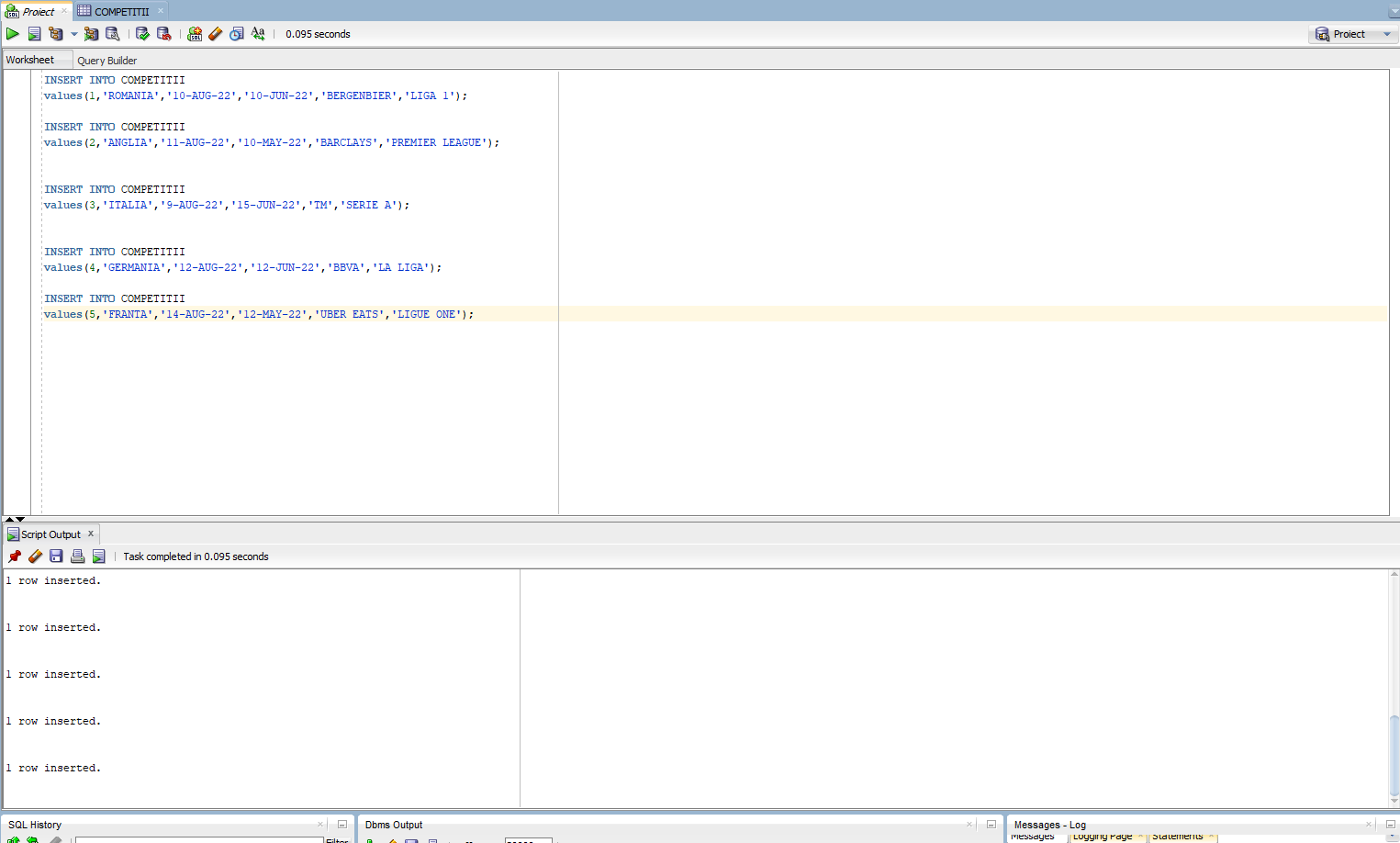
values(3,'ITALIA','9-AUG-22','15-JUN-22','TM','SERIE A');

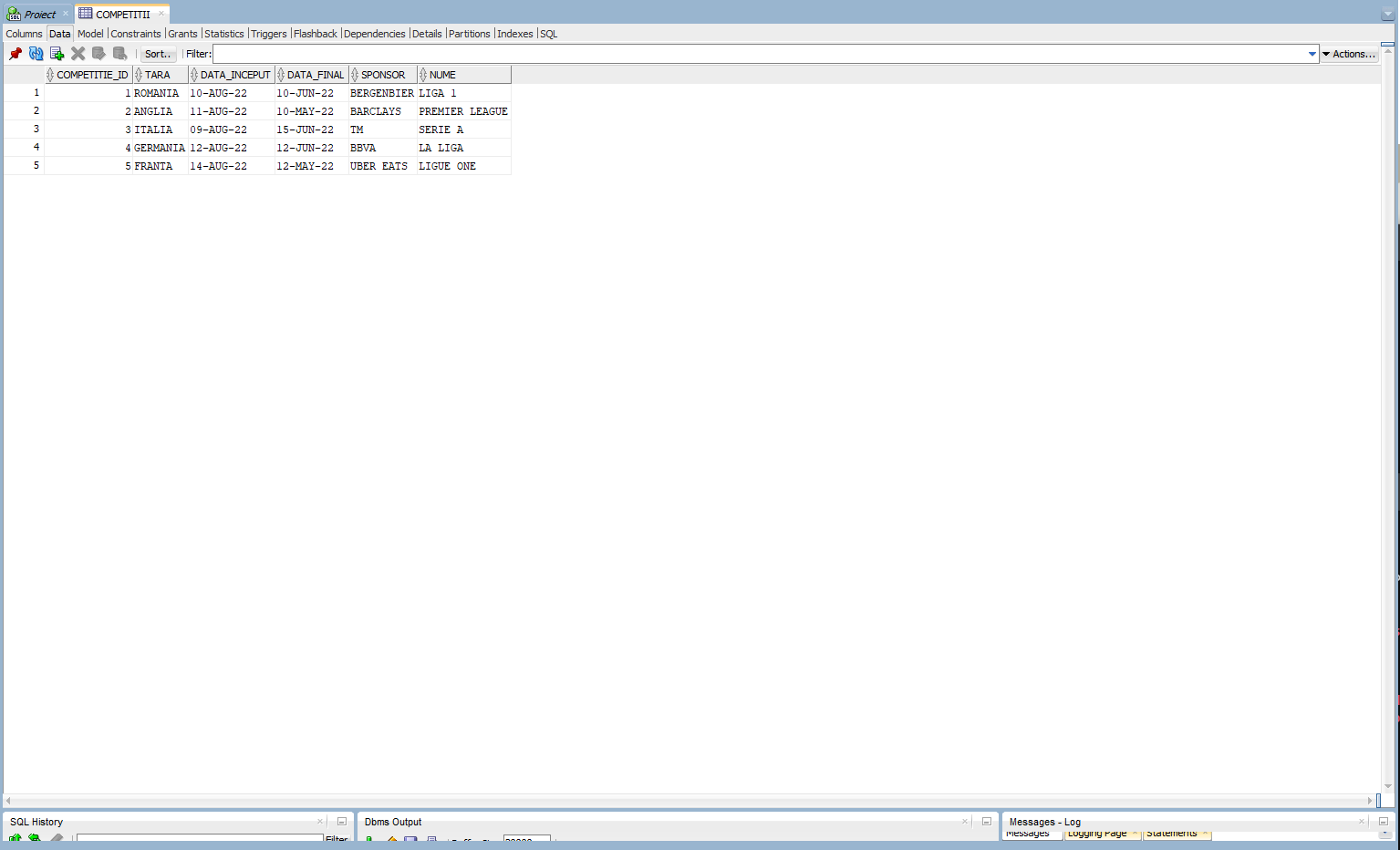
INSERT INTO COMPETITII

values(4,'GERMANIA','12-AUG-22','12-JUN-22','BBVA','LA LIGA');

INSERT INTO COMPETITII

values(5,'FRANTA','14-AUG-22','12-MAY-22','UBER EATS','LIGUE ONE');





INSERT INTO ANTRENORI

values(1,'DAN','PETRESCU','PRINCIPAL');

INSERT INTO ANTRENORI

values(2,'MIRCEA','LUCESCU','PRINCIPAL');

INSERT INTO ANTRENORI

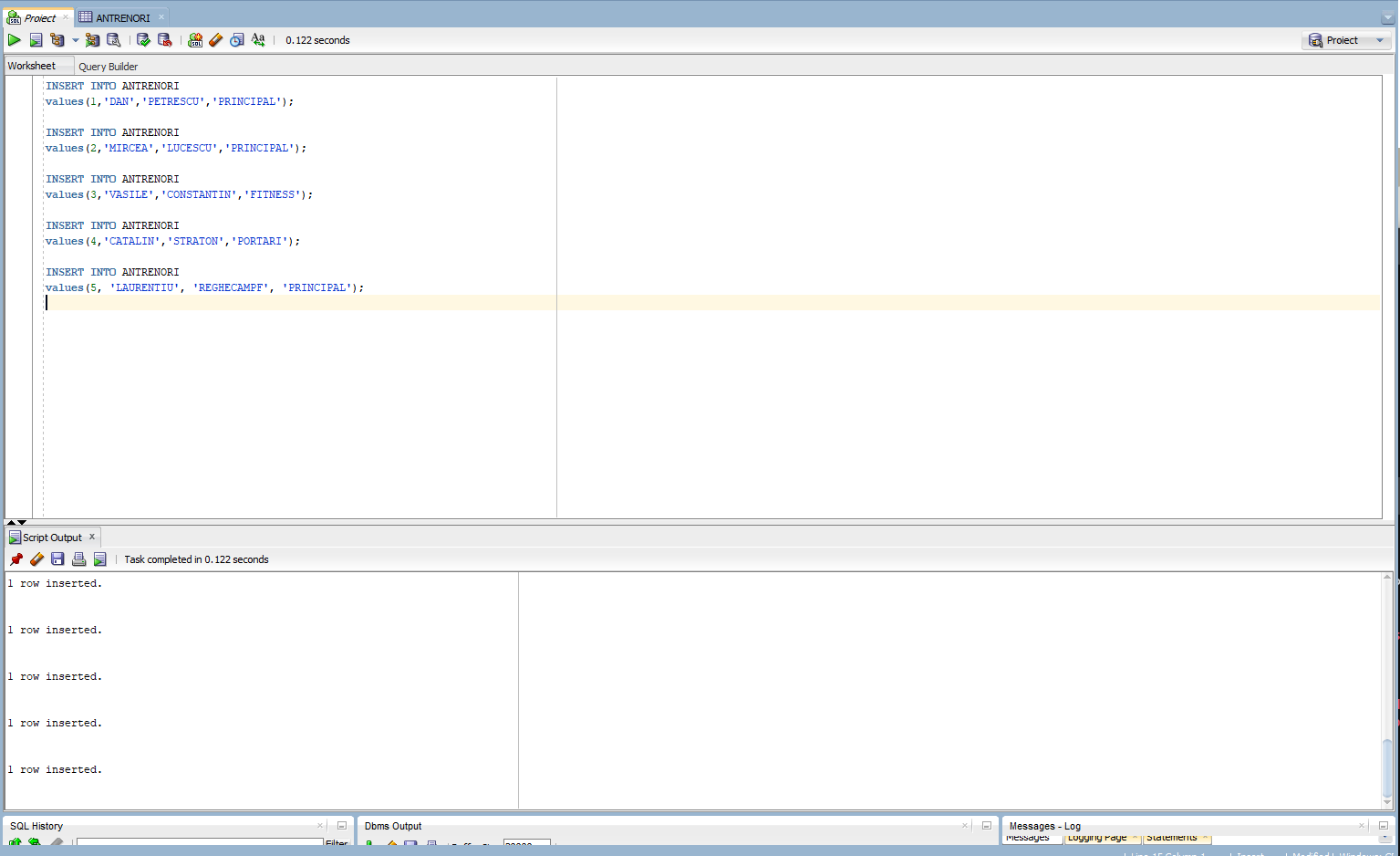
values(3,'VASILE','CONSTANTIN','FITNESS');

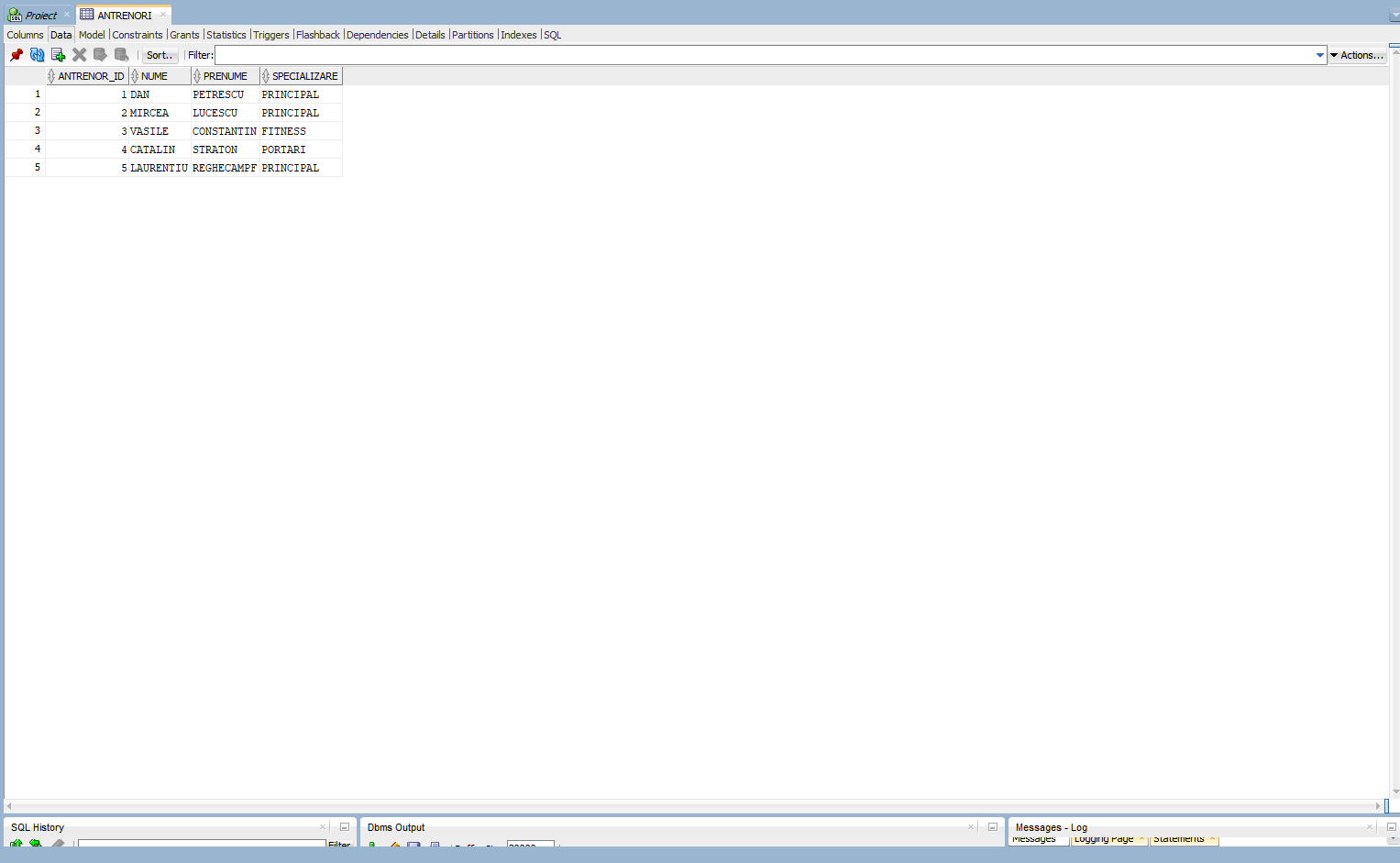
INSERT INTO ANTRENORI

values(4,'CATALIN','STRATON','PORTARI');

INSERT INTO ANTRENORI

values(5, 'LAURENTIU', 'REGHECAMPF', 'PRINCIPAL');



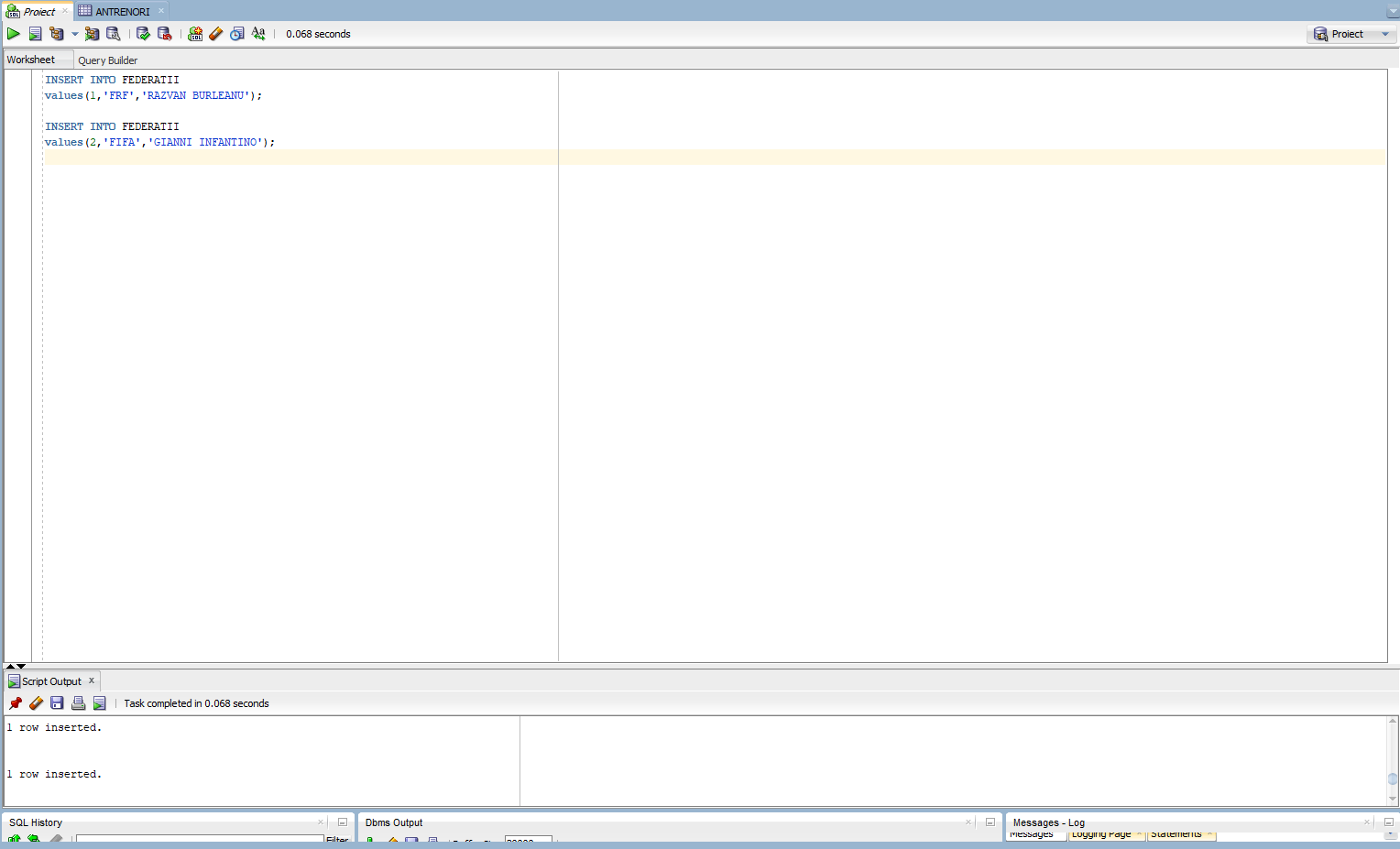


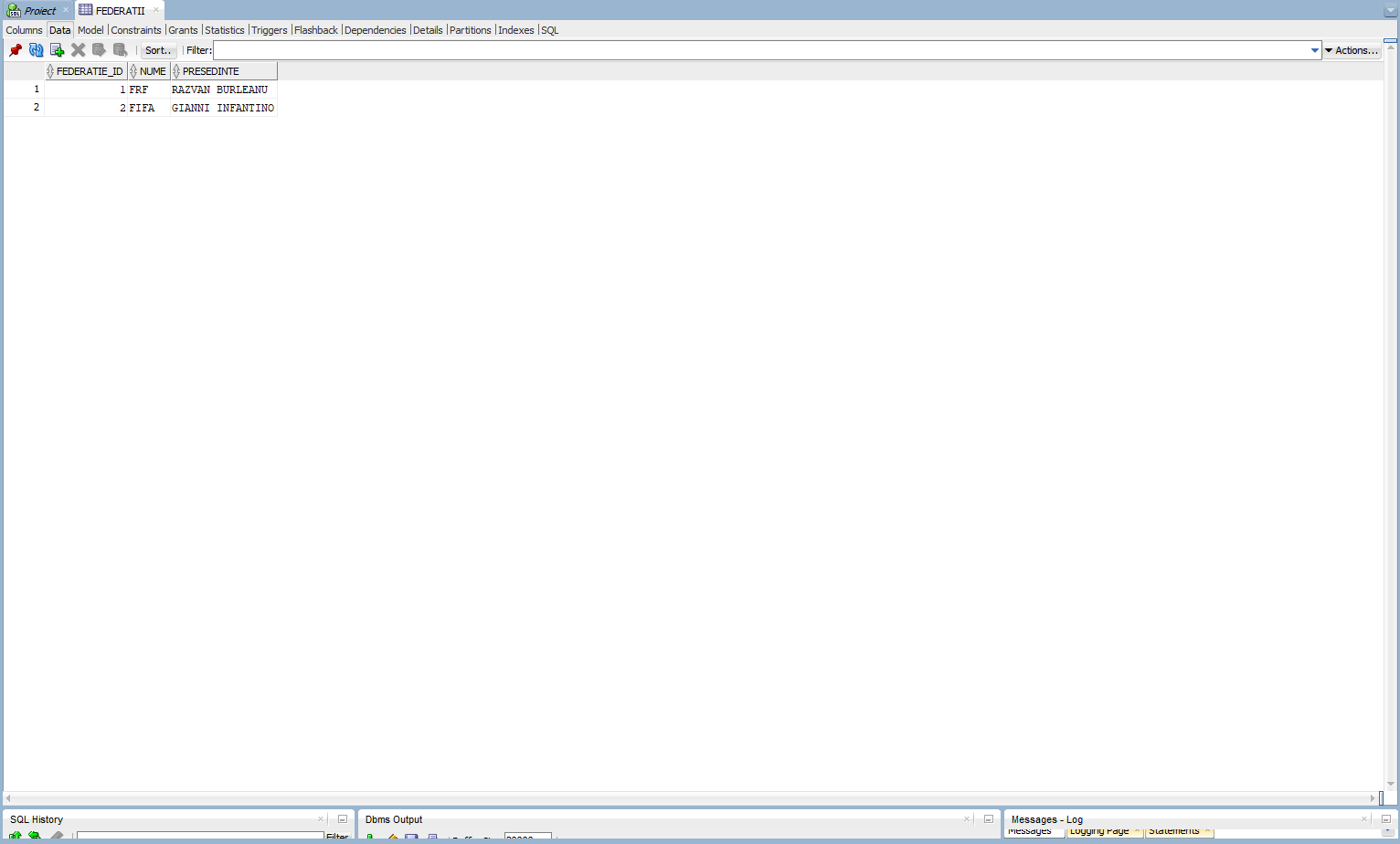
INSERT INTO FEDERATII

values(1,'FRF','RAZVAN BURLEANU');

INSERT INTO FEDERATII

values(2,'FIFA','GIANNI INFANTINO');





INSERT INTO PATRON

values(1,'BORCEA','CRISTI');

INSERT INTO PATRON

values(2,'PORUMBOIU','ADRIAN');

INSERT INTO PATRON

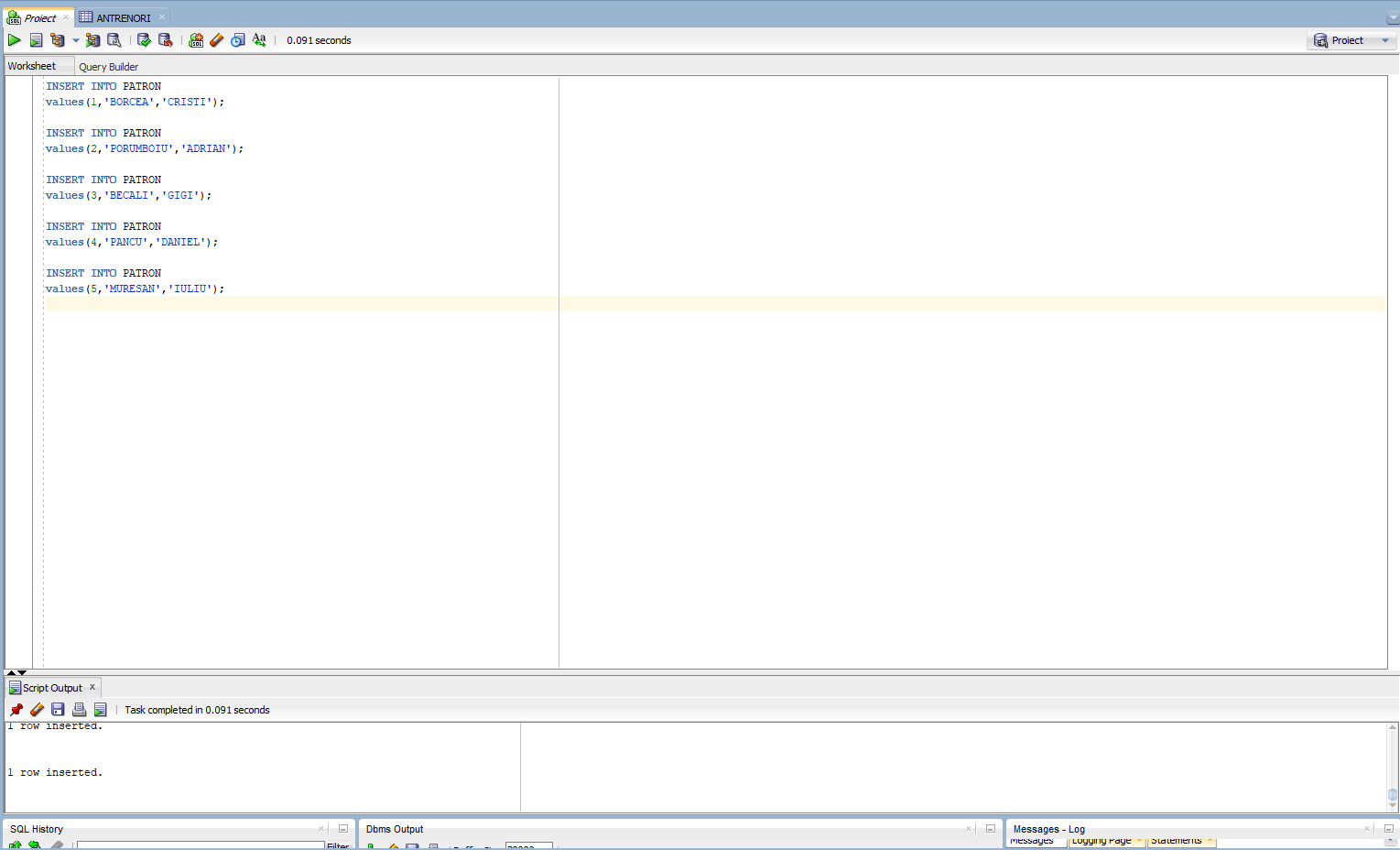
values(3,'BECALI','GIGI');

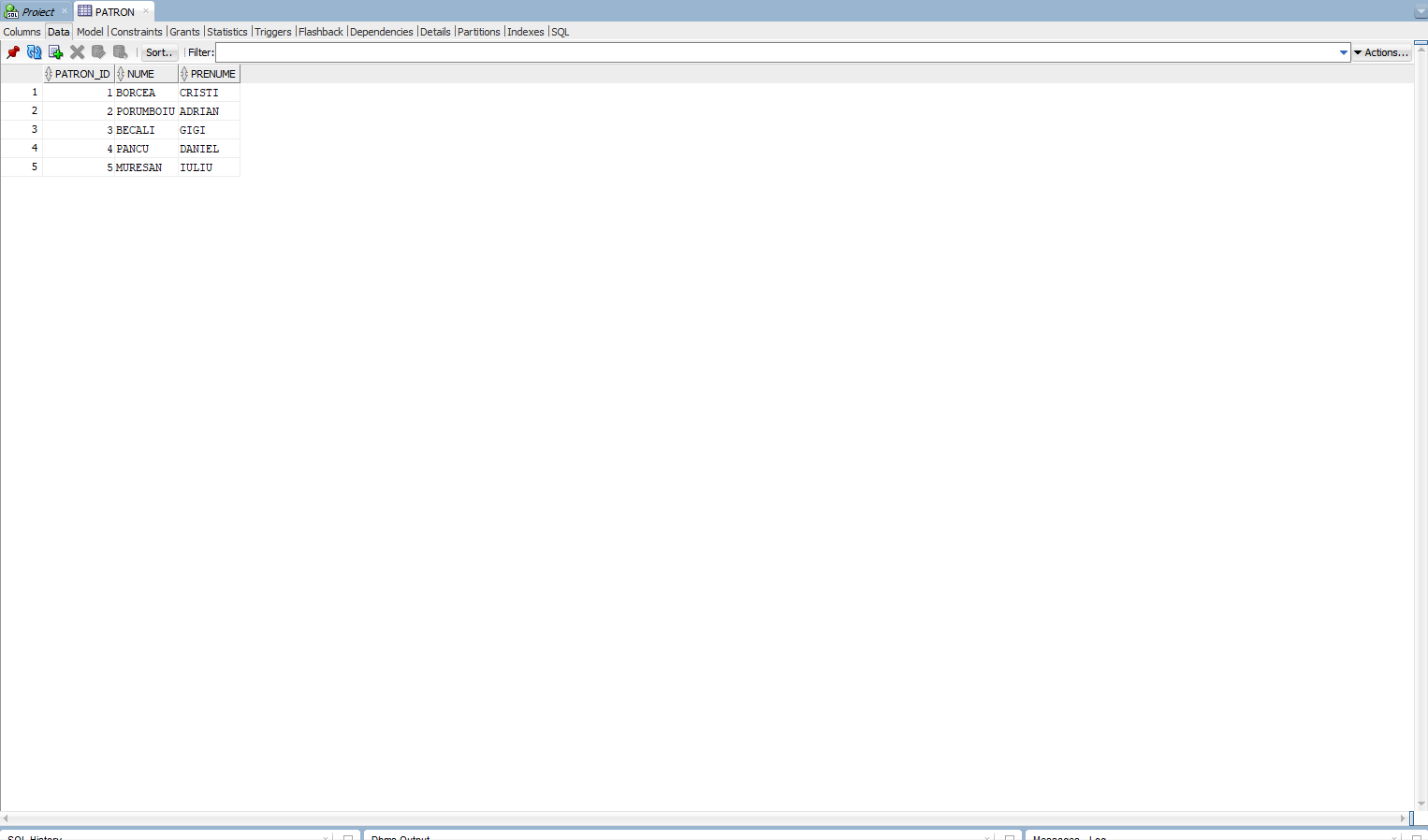
INSERT INTO PATRON

values(4,'PANCU','DANIEL');

INSERT INTO PATRON

values(5,'MURESAN','IULIU');





INSERT INTO ETAPE

values(1,1);

INSERT INTO ETAPE

values(2,1);

INSERT INTO ETAPE

values(3,1);

INSERT INTO ETAPE

values(4,1);

INSERT INTO ETAPE

values(5,1);

INSERT INTO ETAPE

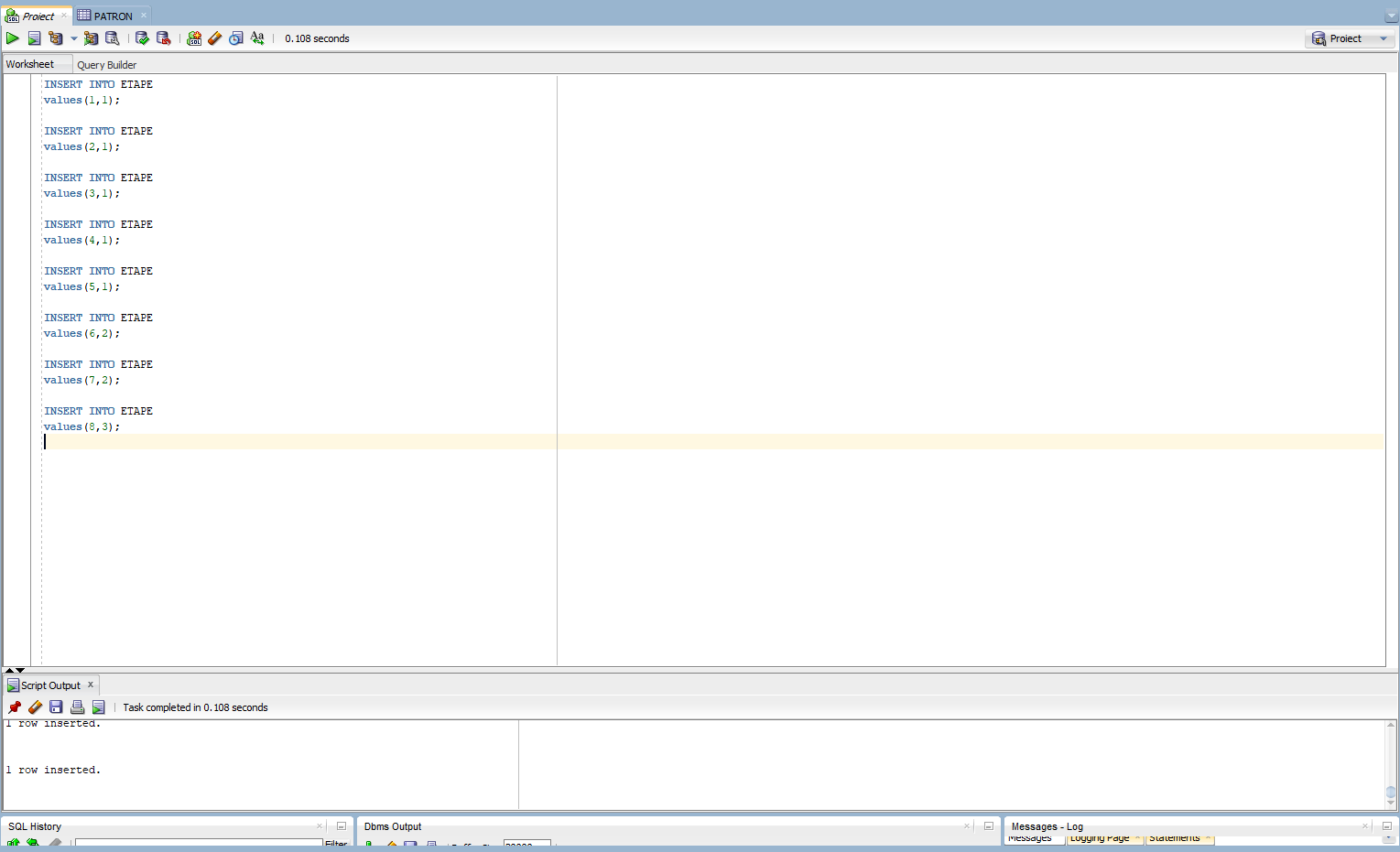
values(6,2);

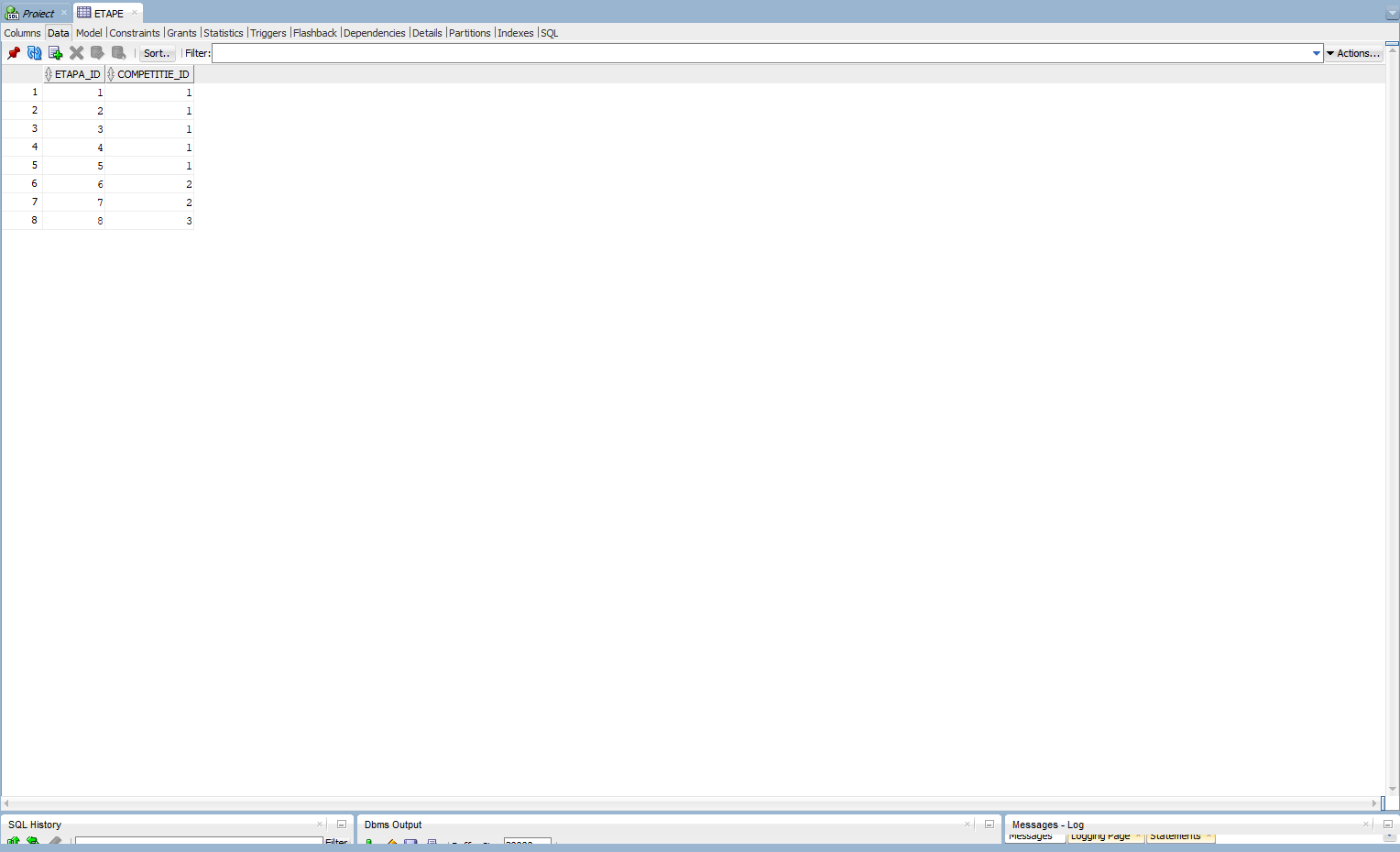
INSERT INTO ETAPE

values(7,2);

INSERT INTO ETAPE

values(8,3);





INSERT INTO ANGAJEAZA

values(1,1,3000,4);

INSERT INTO ANGAJEAZA

values(1,2,2000,5);

INSERT INTO ANGAJEAZA

values(1,3,3500,6);

INSERT INTO ANTRENEAZA

values(1,4,4,'10-AUG-2022','fitness');

INSERT INTO ANTRENEAZA

values(1,3,3,'10-AUG-2022','rezistenta');

INSERT INTO ANTRENEAZA

values(1,3,1,'10-AUG-2022','shooting');

INSERT INTO ANTRENEAZA

values(1,2,2,'10-AUG-2022','passing');

INSERT INTO ANTRENEAZA

values(1,2,5,'10-AUG-2022','rezistenta');

INSERT INTO ANTRENEAZA

values(1,2,6,'10-AUG-2022','defending');

INSERT INTO ANTRENEAZA

values(1,4,4,'10-AUG-2022','fitness');

INSERT INTO ANTRENEAZA

values(3,5,9,'10-AUG-2022','passing');

INSERT INTO ANTRENEAZA

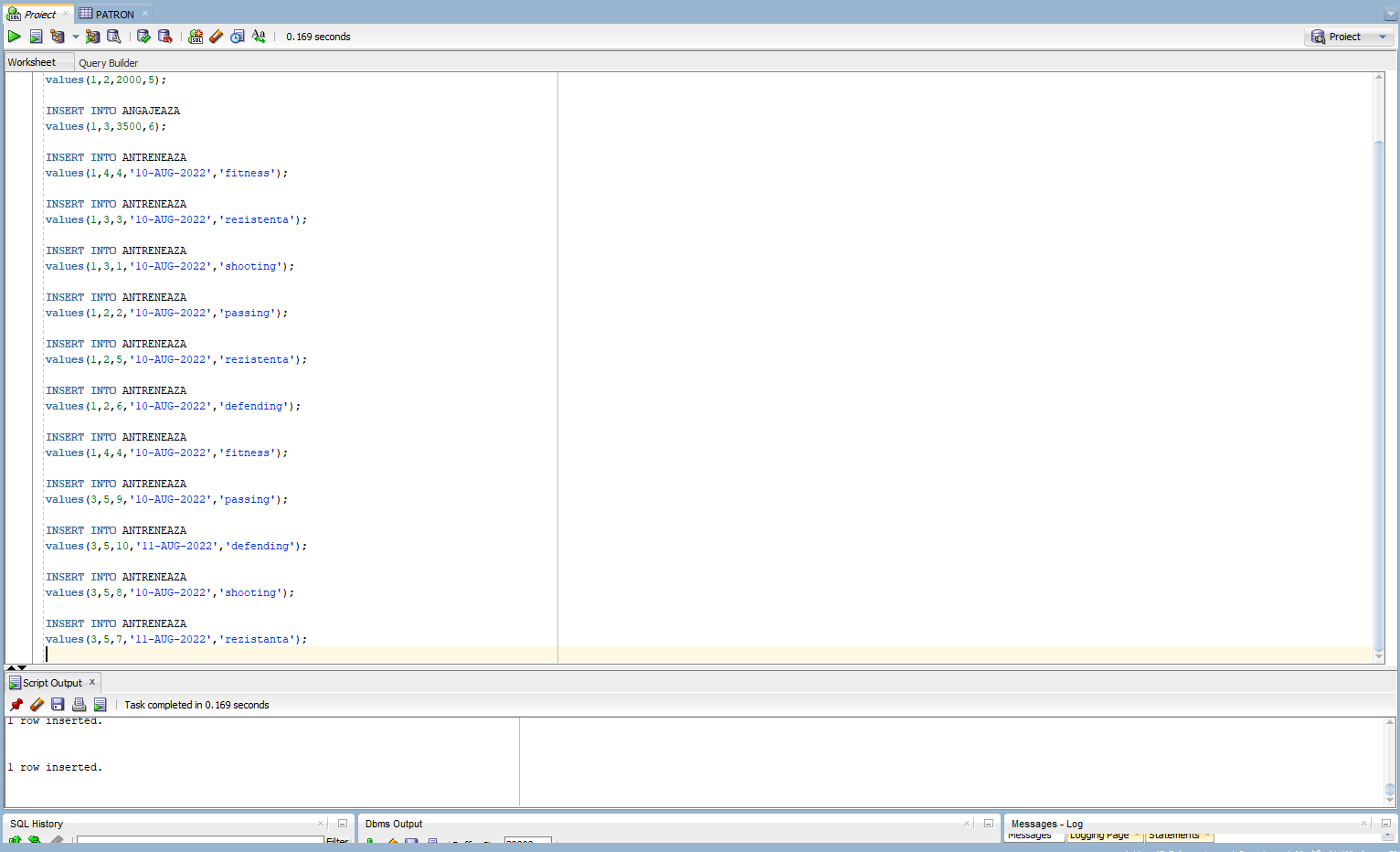
values(3,5,10,'11-AUG-2022','defending');

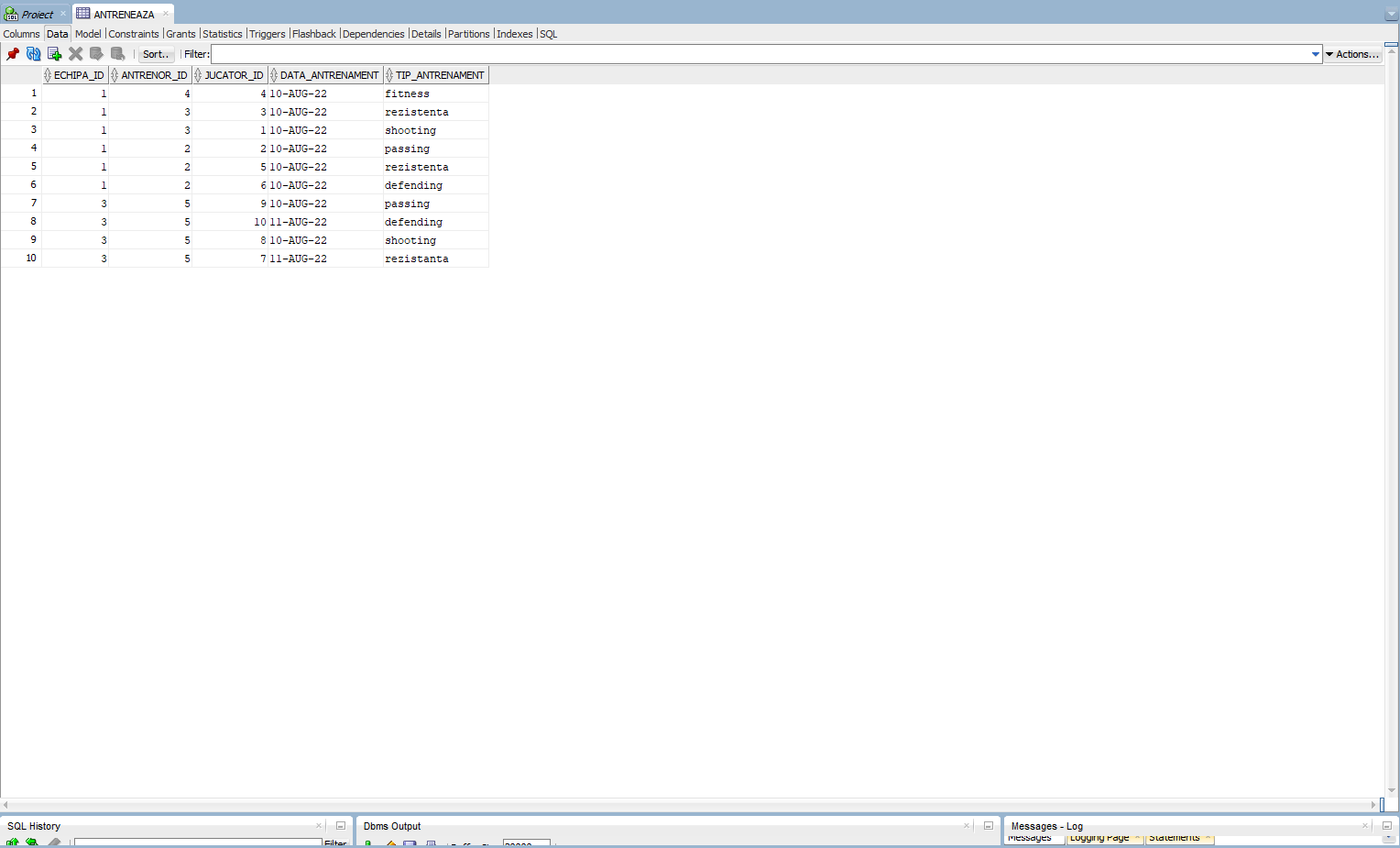
INSERT INTO ANTRENEAZA

values(3,5,8,'10-AUG-2022','shooting');

INSERT INTO ANTRENEAZA

values(3,5,7,'11-AUG-2022','rezistanta');





INSERT INTO LEGITIMEAZA

values(1,1,999999.9,5);

INSERT INTO LEGITIMEAZA

values(1,2,999.9,3);

INSERT INTO LEGITIMEAZA

values(1,3,2000,2);

INSERT INTO LEGITIMEAZA

values(1,4,1300.50,4);

INSERT INTO LEGITIMEAZA

values(1,5,1500,4);

INSERT INTO LEGITIMEAZA

values(1,6,3000,3);

INSERT INTO LEGITIMEAZA

values(3,7,2000,3);

INSERT INTO LEGITIMEAZA

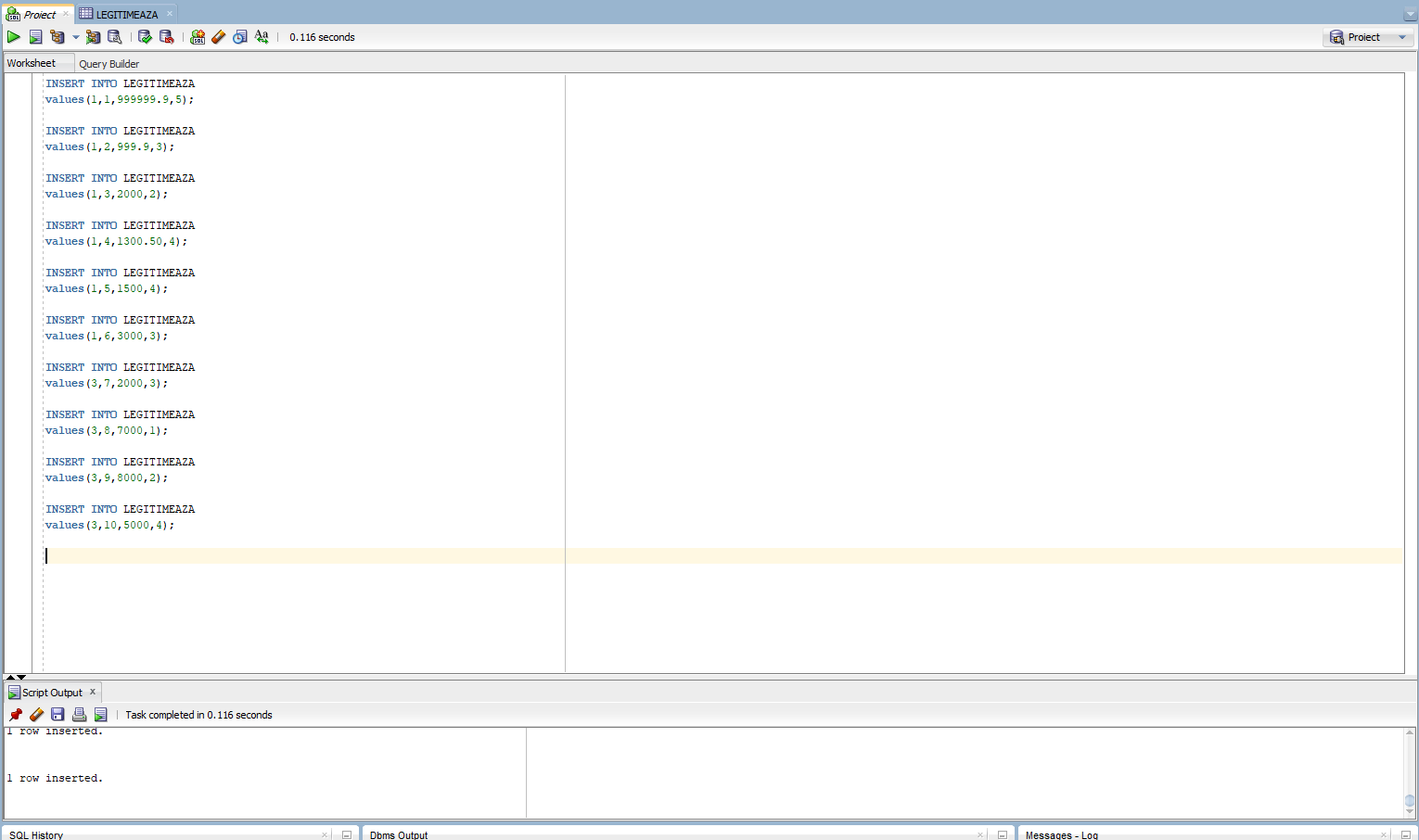
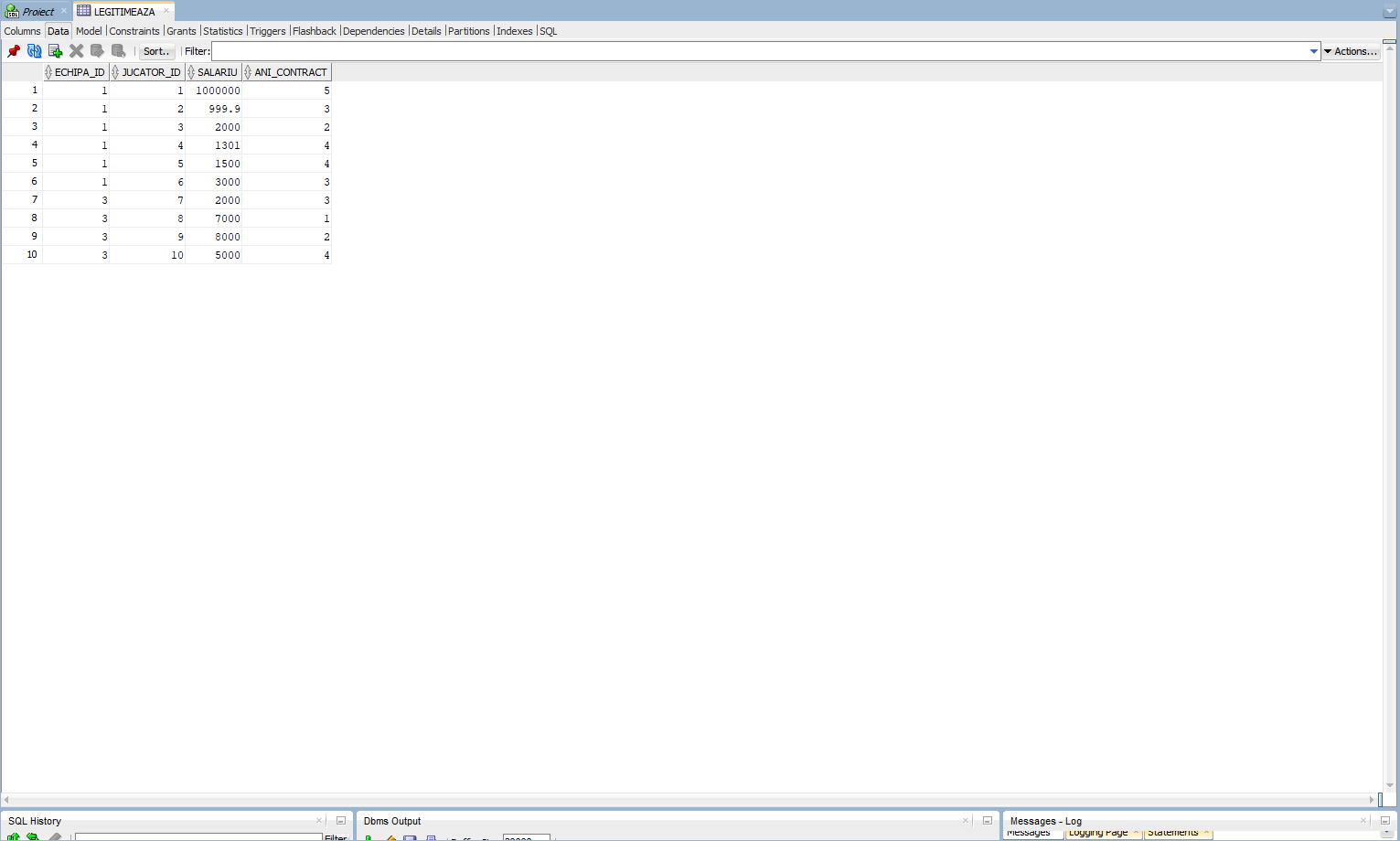
values(3,8,7000,1);

INSERT INTO LEGITIMEAZA

values(3,9,8000,2);

INSERT INTO LEGITIMEAZA

values(3,10,5000,4);

INSERT INTO PARTICIPA

values(1,1,1,90,50,0);

INSERT INTO PARTICIPA

values(2,1,2,89,49,1);

INSERT INTO PARTICIPA

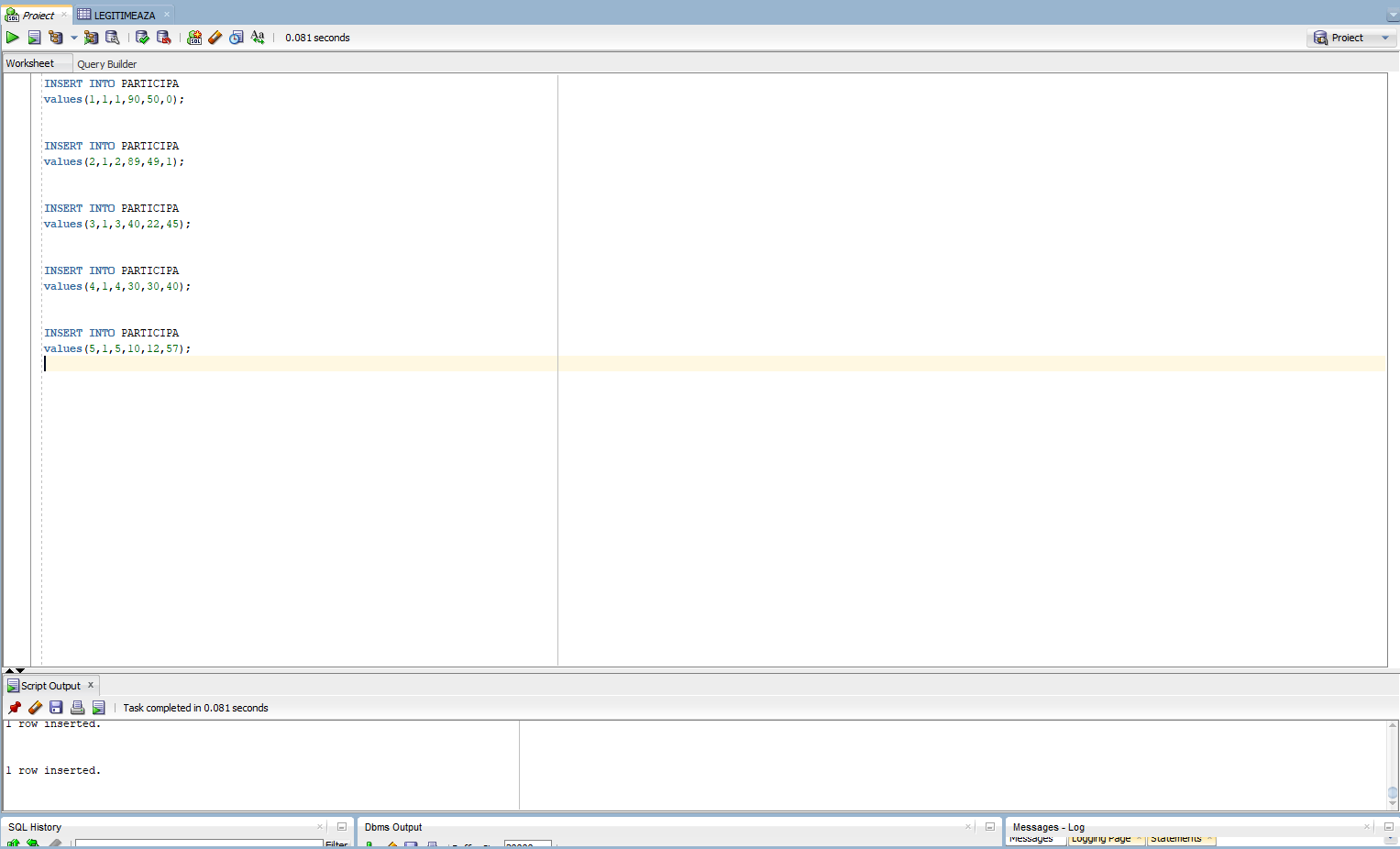
values(3,1,3,40,22,45);

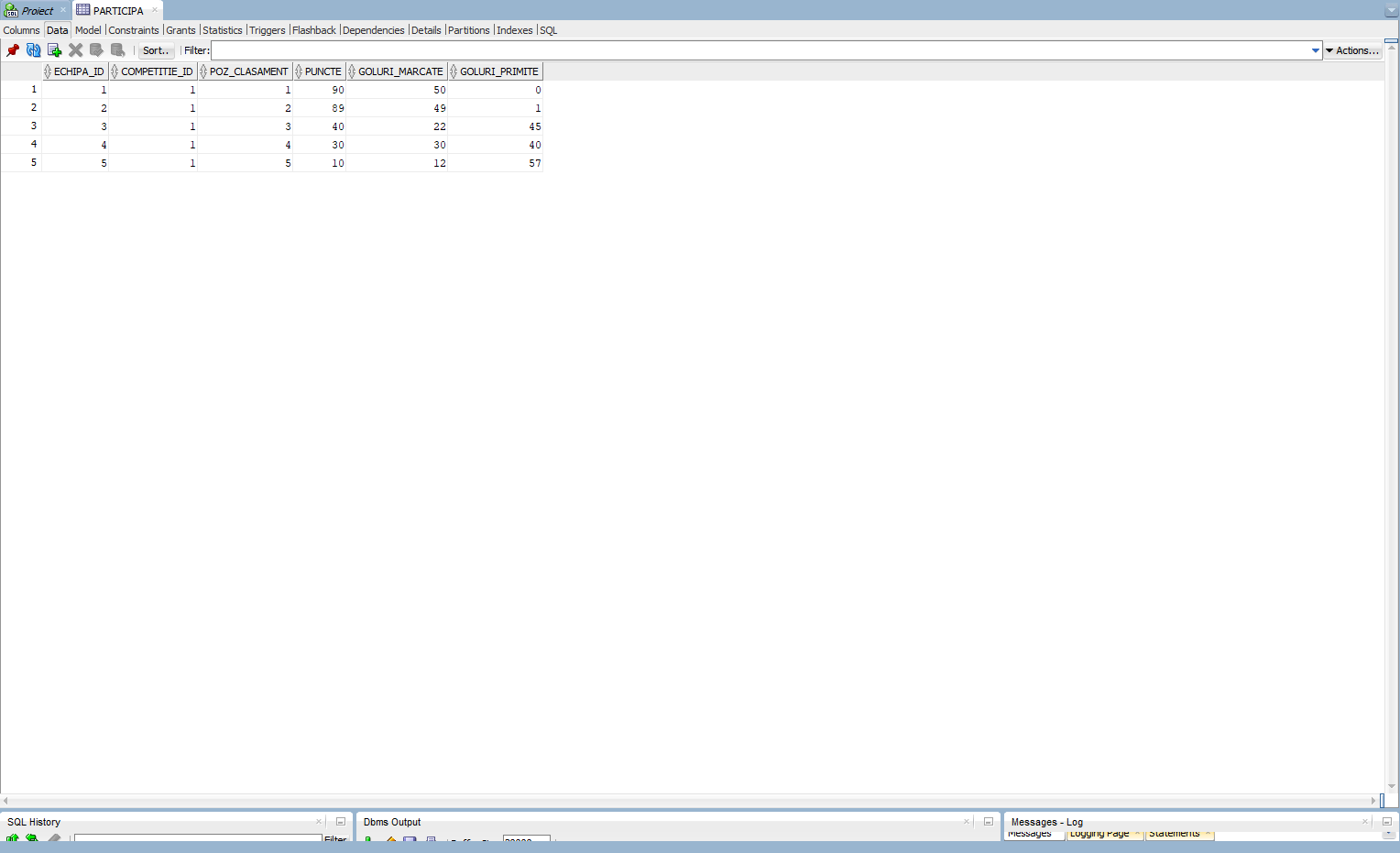
INSERT INTO PARTICIPA

values(4,1,4,30,30,40);

INSERT INTO PARTICIPA

values(5,1,5,10,12,57);





INSERT INTO ORGANIZEAZA

values(1,1);

INSERT INTO ORGANIZEAZA

values(2,2);

INSERT INTO ORGANIZEAZA

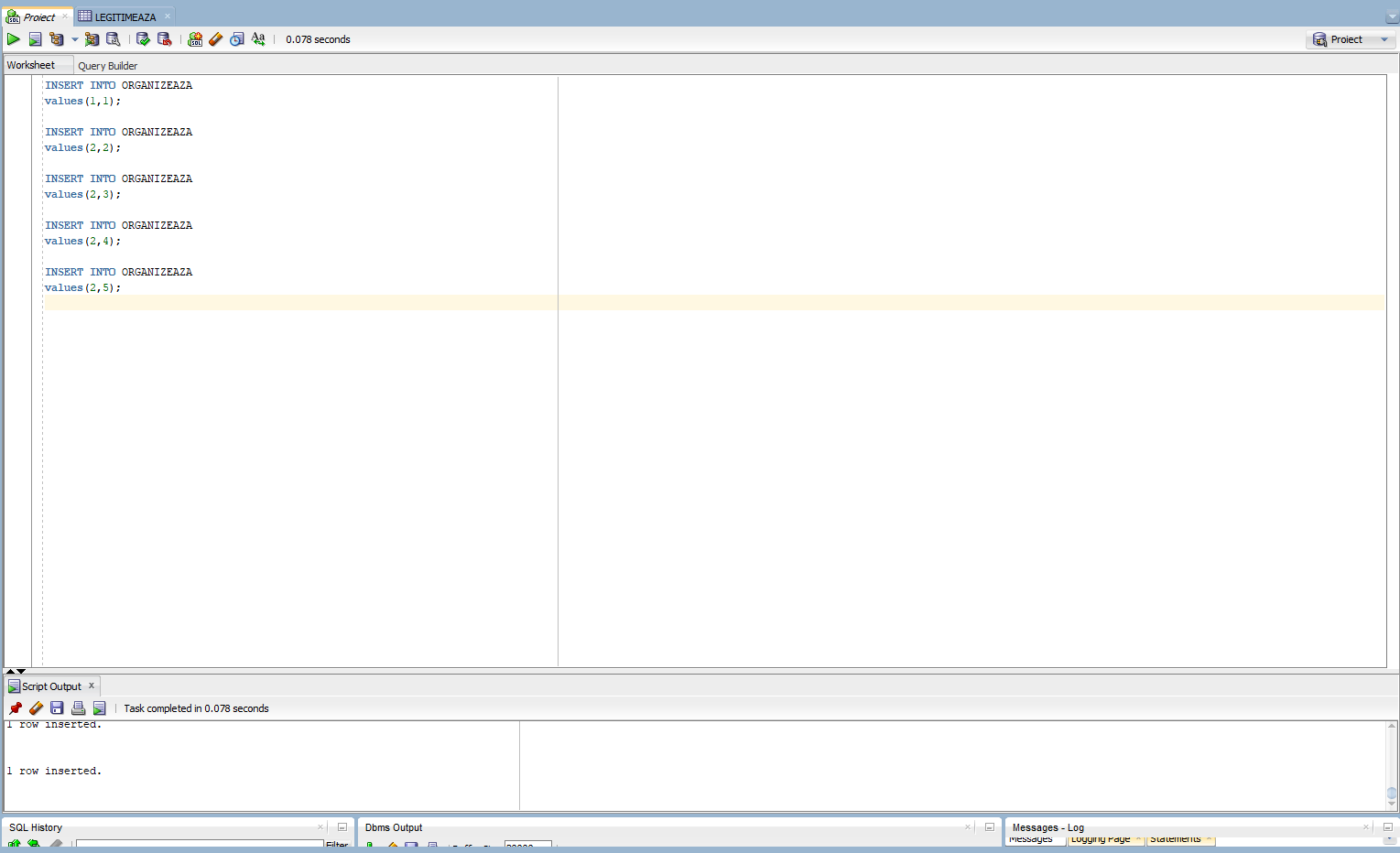
values(2,3);

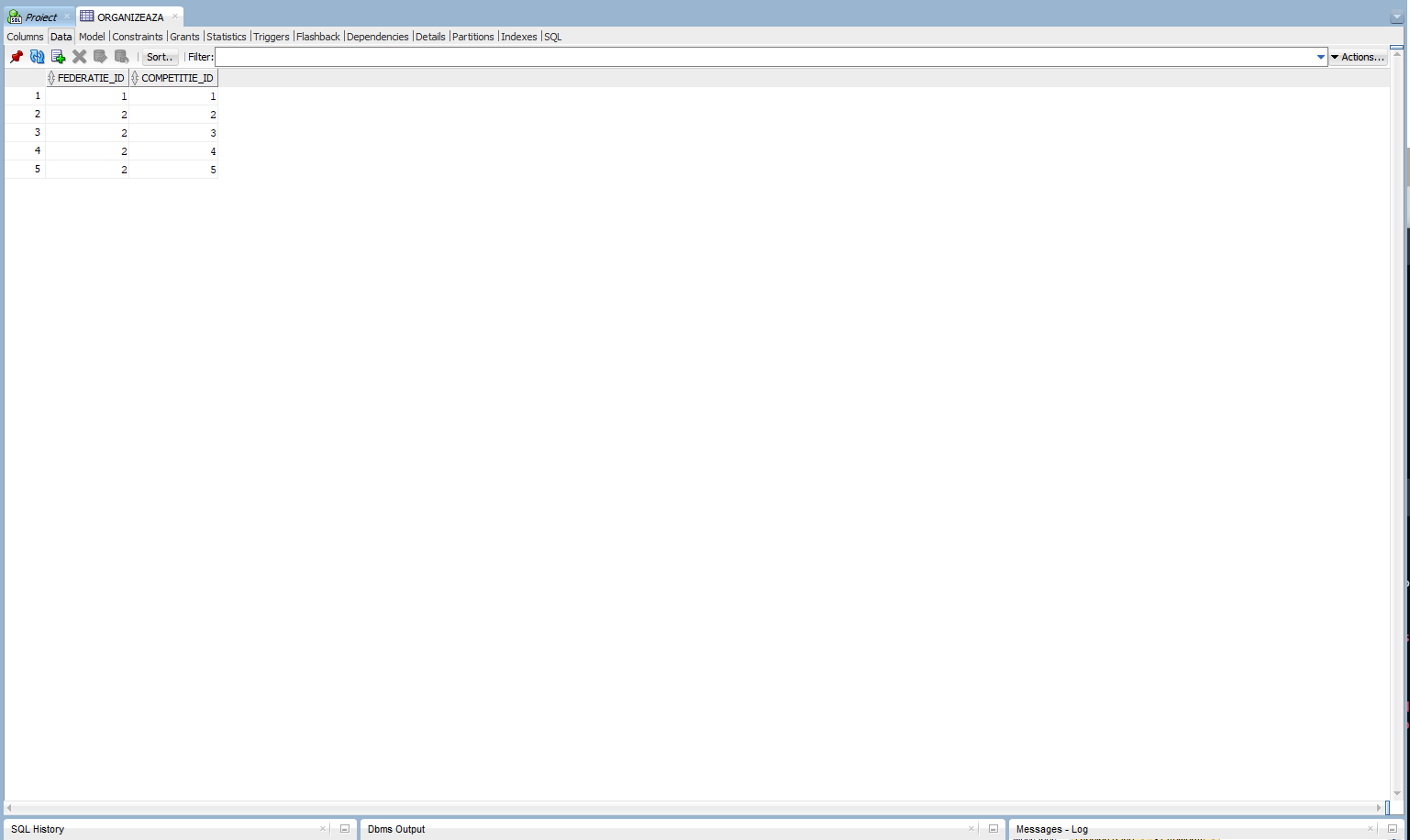
INSERT INTO ORGANIZEAZA

values(2,4);

INSERT INTO ORGANIZEAZA

values(2,5);





INSERT INTO TRANSFERA

values(1,2,1,500000,'FABRIZIO ROMANO');

INSERT INTO TRANSFERA

values(1,2,2,2200,'AGENT 007');

INSERT INTO TRANSFERA

values(1,2,3,45000,'FABRIZIO ROMANO');

INSERT INTO TRANSFERA

values(1,2,4,0,'AGENT 001');

INSERT INTO TRANSFERA

values(1,2,5,3000,'JUSTITIARU');

INSERT INTO TRANSFERA

values(3,3,7,3200,'AGENT 007');

INSERT INTO TRANSFERA

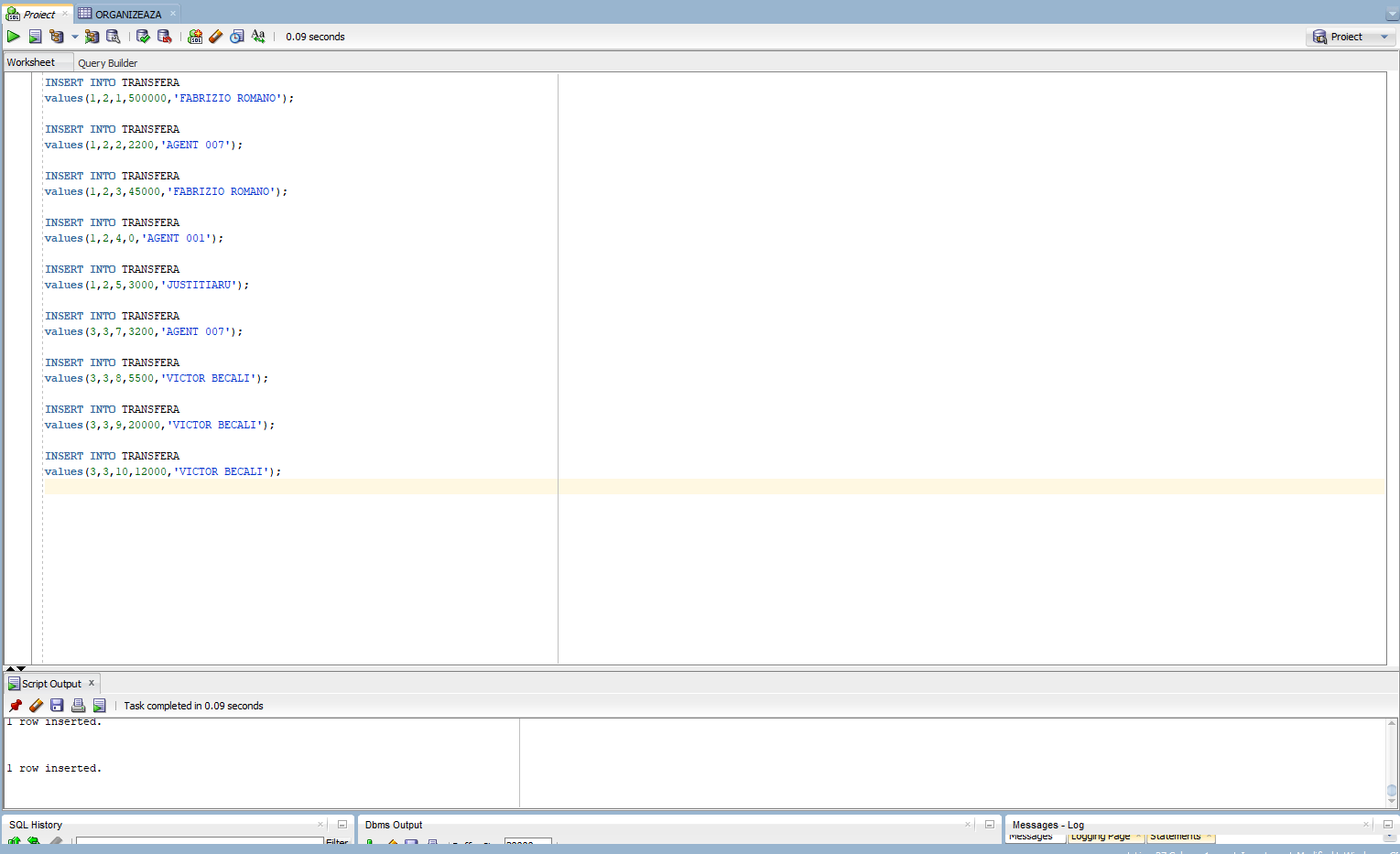
values(3,3,8,5500,'VICTOR BECALI');

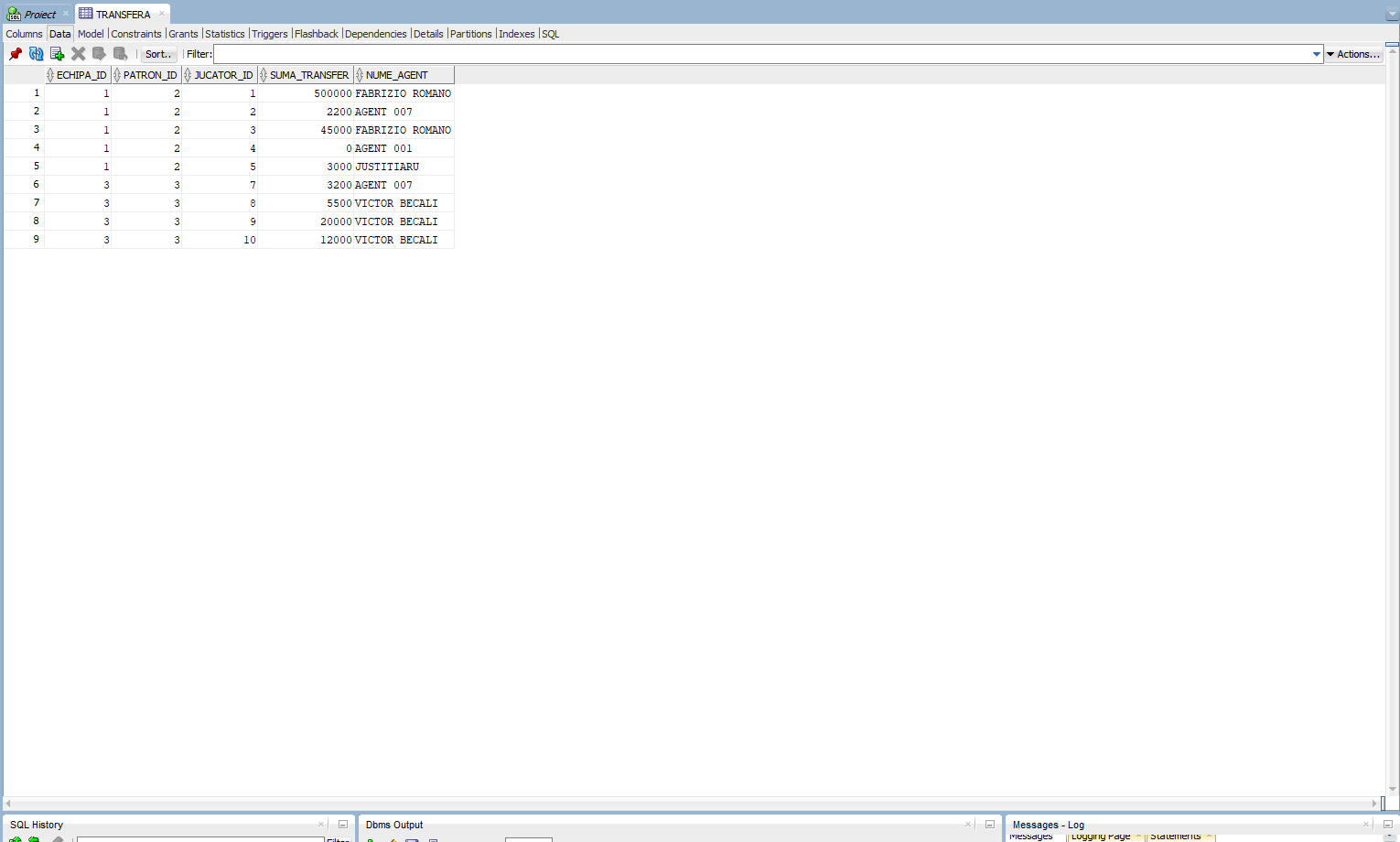
INSERT INTO TRANSFERA

values(3,3,9,20000,'VICTOR BECALI');

INSERT INTO TRANSFERA

values(3,3,10,12000,'VICTOR BECALI');





INSERT INTO JOACA\_MECIURI

values(1,2,1,3,'11-APR-2002','STEFAN CEL MARE');

INSERT INTO JOACA\_MECIURI

values(2,1,1,2,'17-APR-2002','STEFAN CEL MARE');

INSERT INTO JOACA\_MECIURI

values(3,3,1,4,'20-APR-2002','STEFAN CEL MARE');

INSERT INTO JOACA\_MECIURI

values(4,1,2,1,'30-APR-2002','MUNICIPAL');

INSERT INTO JOACA\_MECIURI

values(5,2,2,3,'11-MAY-2002','MUNICIPAL');

INSERT INTO JOACA\_MECIURI

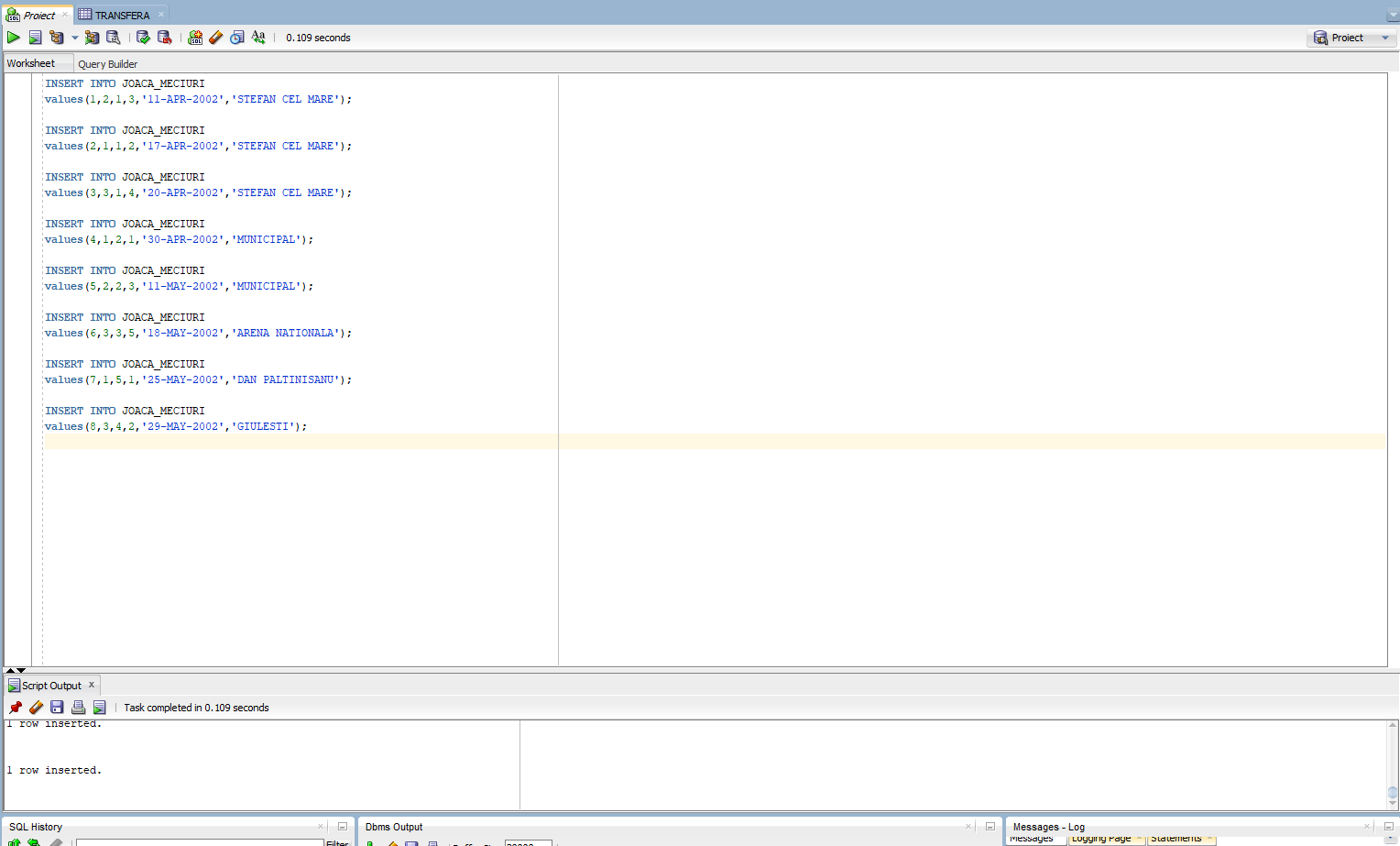
values(6,3,3,5,'18-MAY-2002','ARENA NATIONALA');

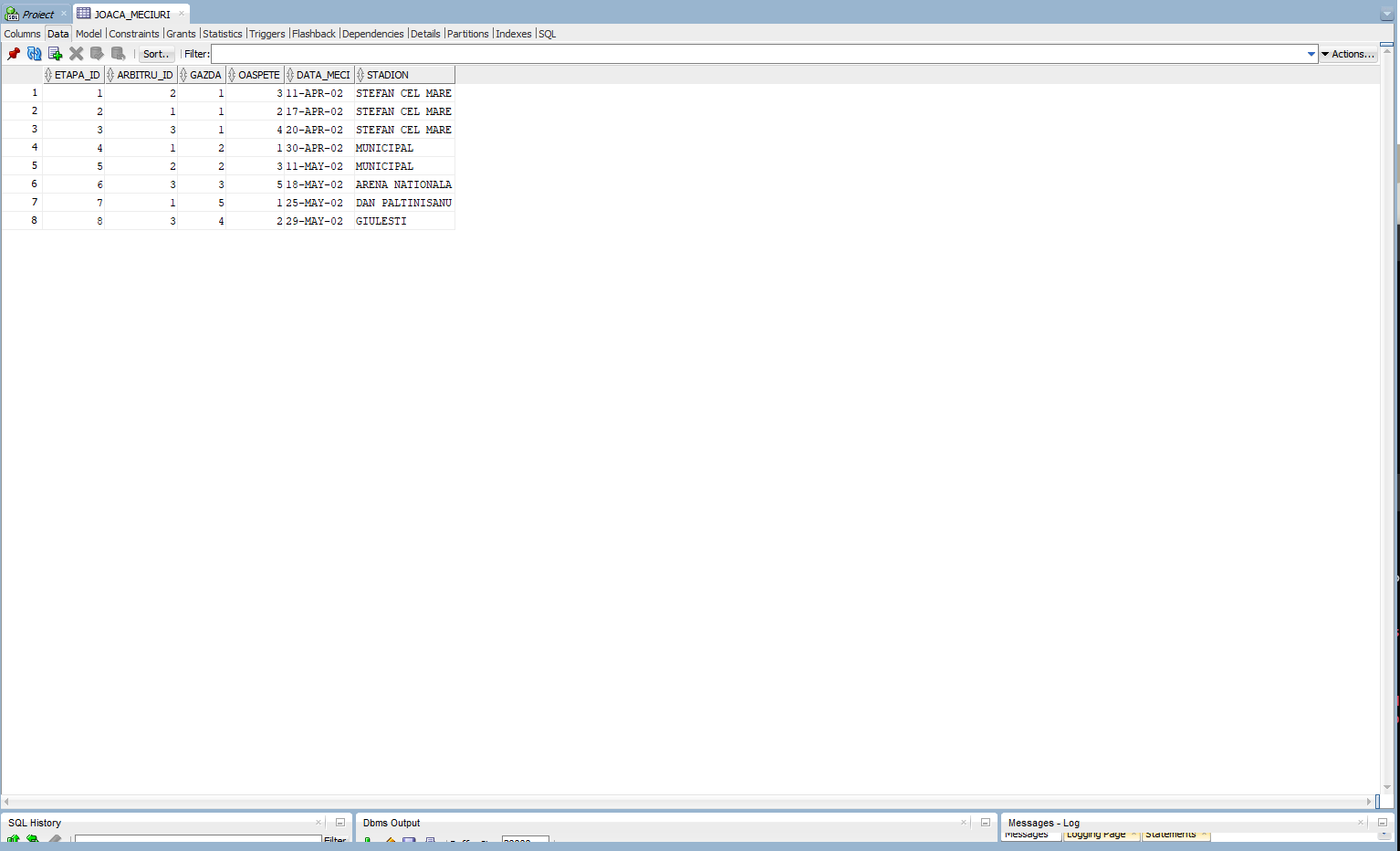
INSERT INTO JOACA\_MECIURI

values(7,1,5,1,'25-MAY-2002','DAN PALTINISANU');

INSERT INTO JOACA\_MECIURI

values(8,3,4,2,'29-MAY-2002','GIULESTI');



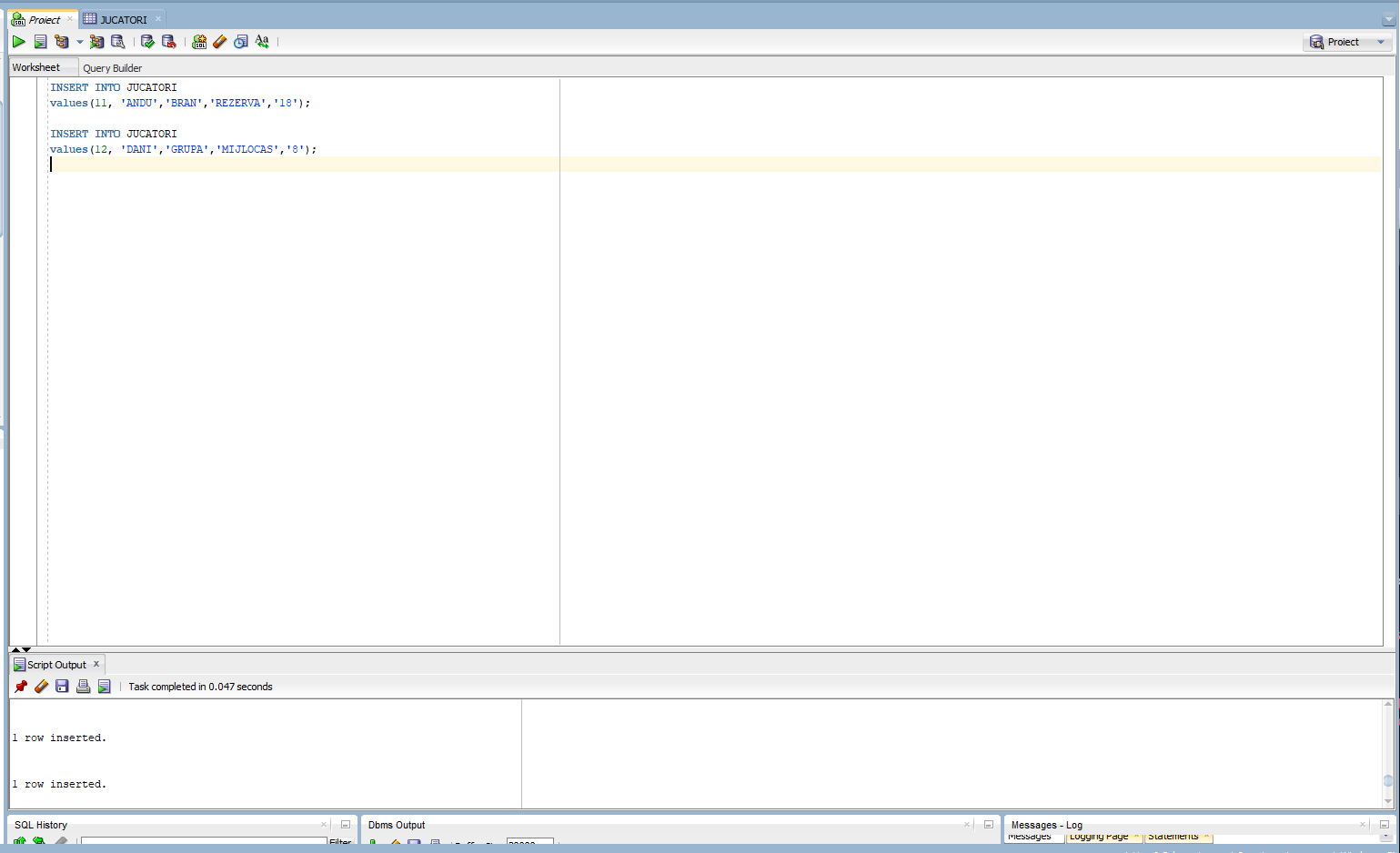


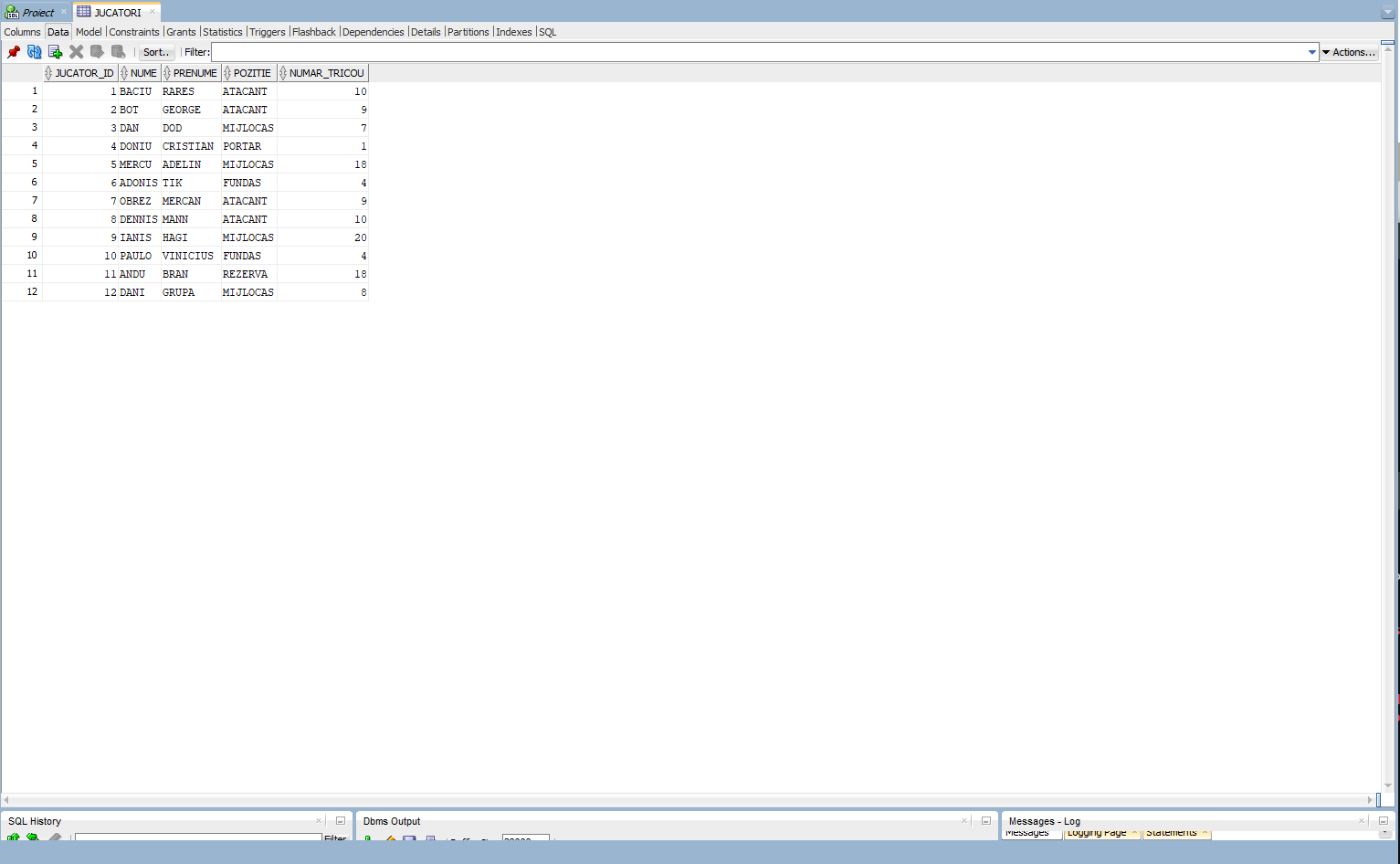
INSERT INTO JUCATORI

values(11, 'ANDU','BRAN','REZERVA','18');

INSERT INTO JUCATORI

values(12, 'DANI','GRUPA','MIJLOCAS','8');



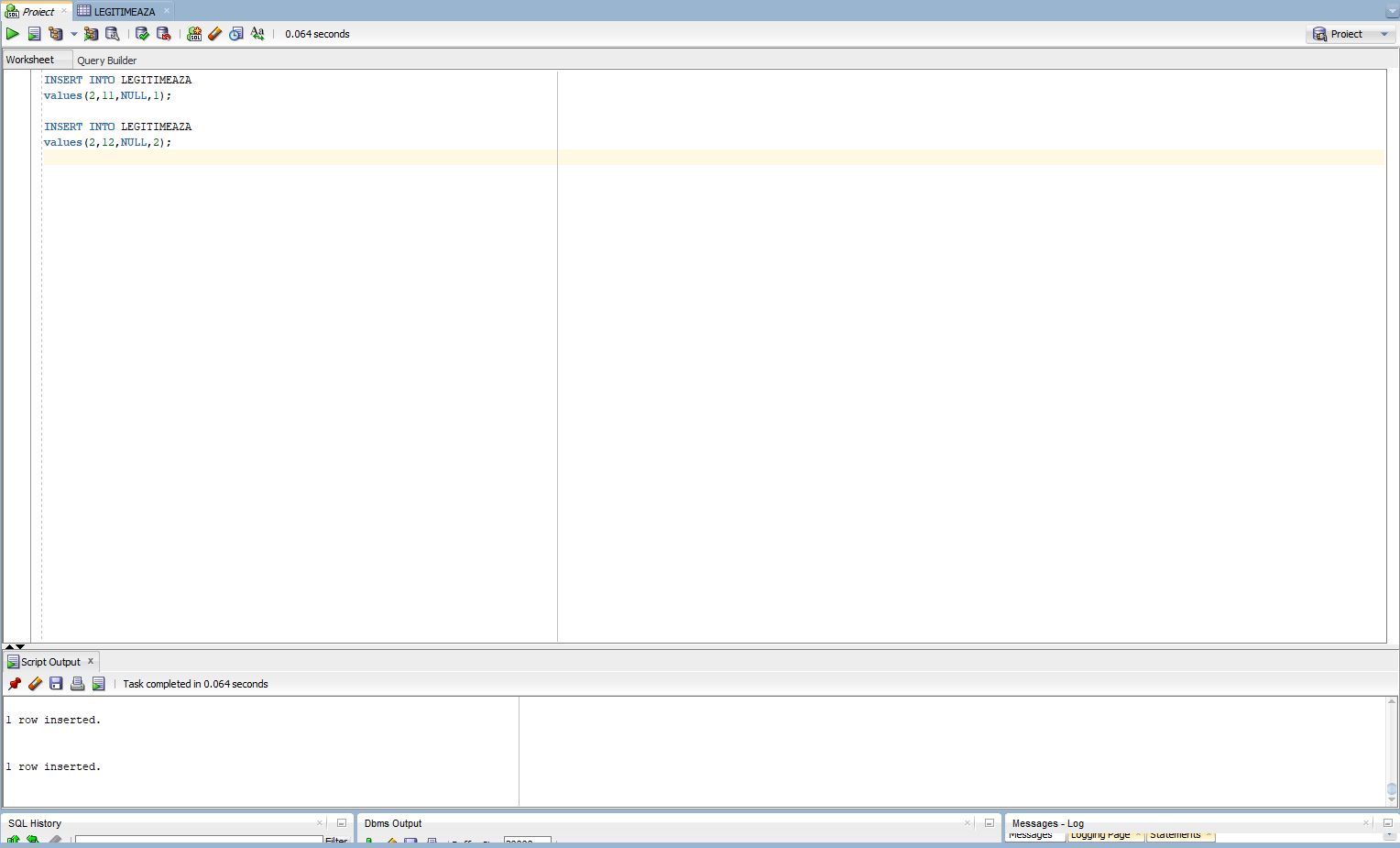


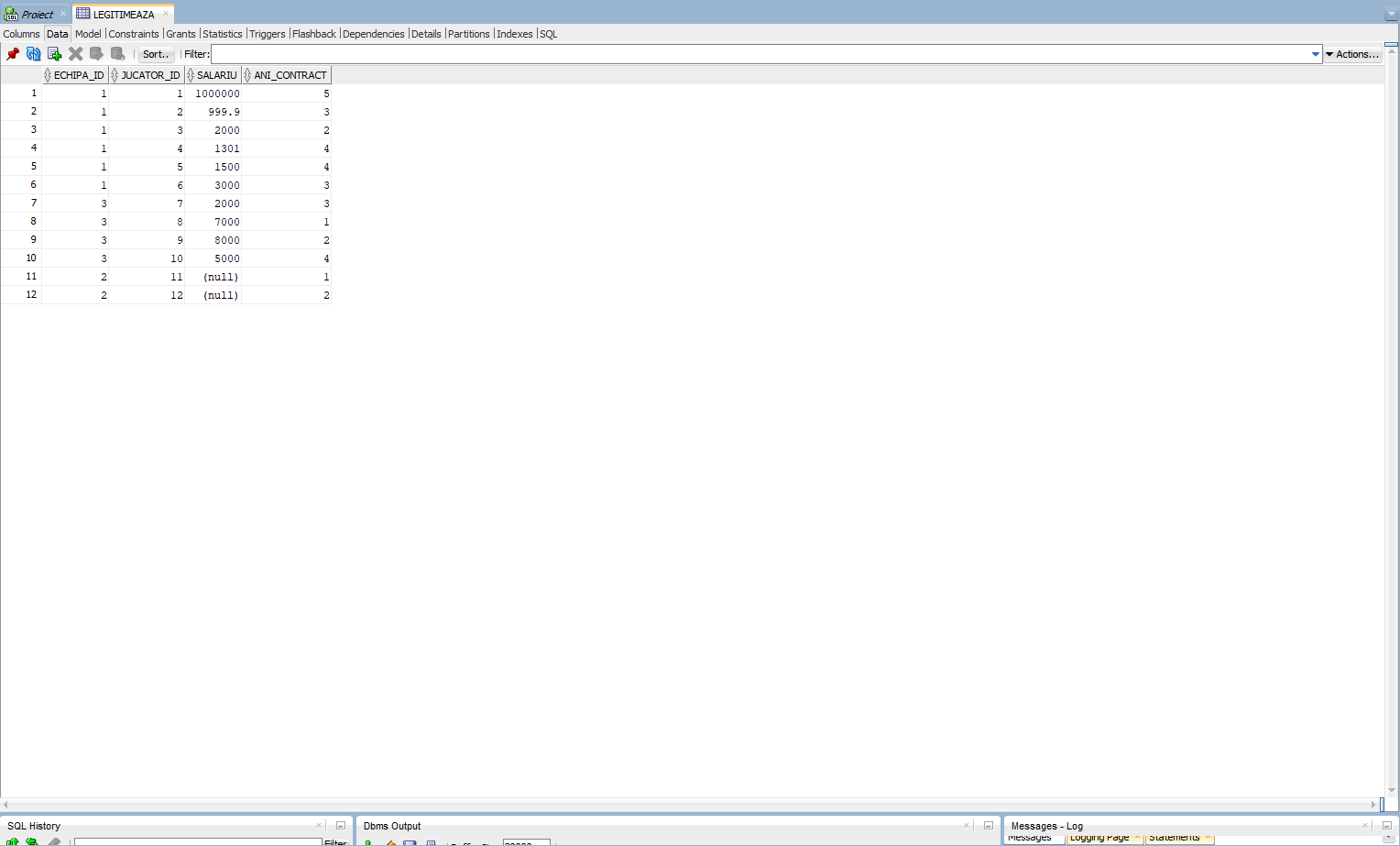
INSERT INTO LEGITIMEAZA

values(2,11,NULL,1);

INSERT INTO LEGITIMEAZA

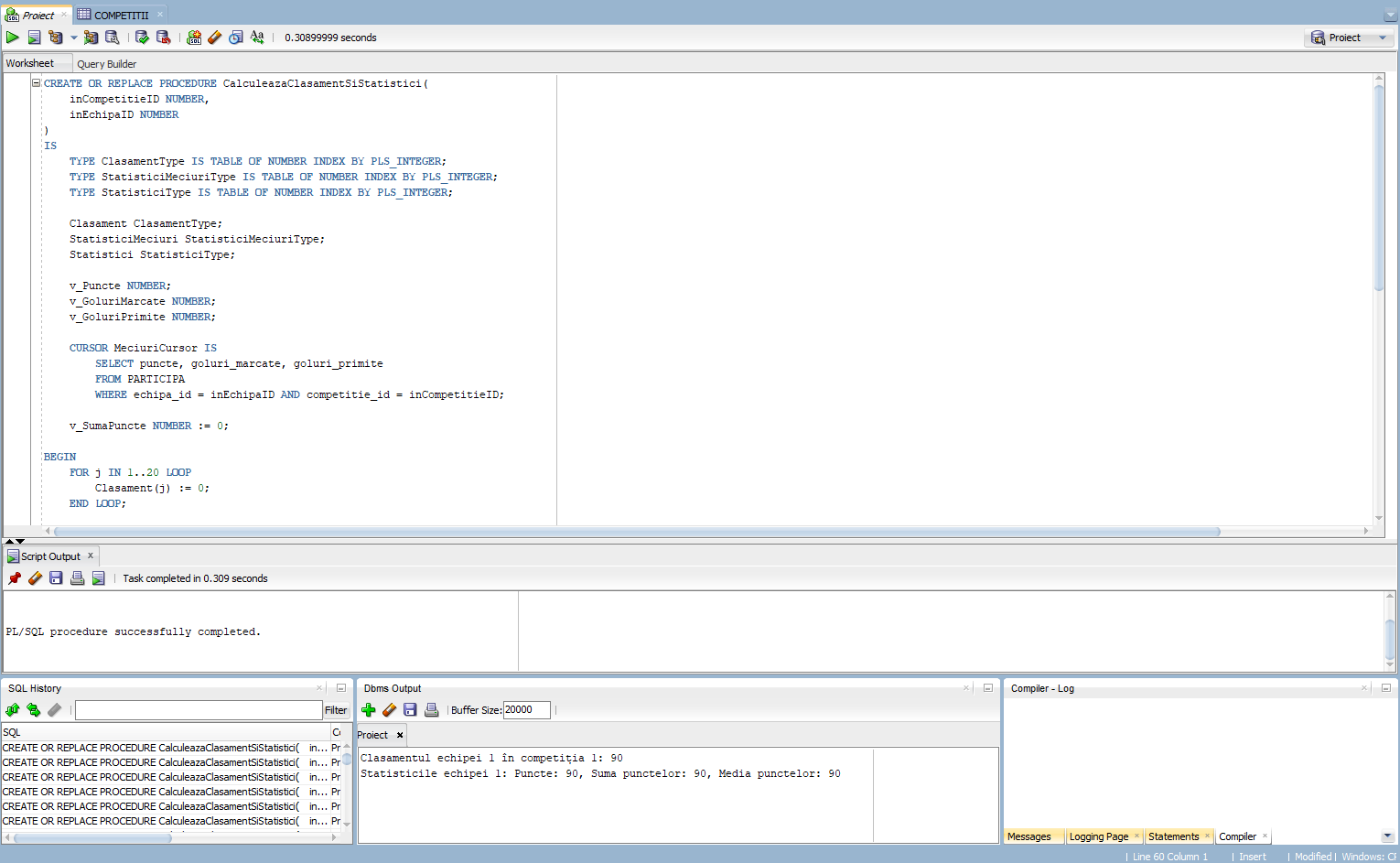
values(2,12,NULL,2);





1. Formulați în limbaj natural o problemă pe care să o rezolvați folosind un subprogram stocat independent care să utilizeze toate cele 3 tipuri de colecții studiate. Apelați subprogramul.

Subprogramul este utilizat pentru a obține informații despre performanța echipei într-o competiție, calculând clasamentul și statistici precum suma punctelor și media punctelor obținute în meciurile respective.

Am folosit index-by tables pentru colecția Clasament, nested tables pentru colecția StatisticiMeciuri, iar associative arrays pentru colecția Statistici, care este asemanatoare cu Clasament doar ca tine statisticile globale ale unei echipe.

CREATE OR REPLACE PROCEDURE CalculeazaClasamentSiStatistici(

inCompetitieID NUMBER,

inEchipaID NUMBER

)

IS

TYPE ClasamentType IS TABLE OF NUMBER INDEX BY PLS\_INTEGER;

TYPE StatisticiMeciuriType IS TABLE OF NUMBER INDEX BY PLS\_INTEGER;

TYPE StatisticiType IS TABLE OF NUMBER INDEX BY PLS\_INTEGER;

Clasament ClasamentType;

StatisticiMeciuri StatisticiMeciuriType;

Statistici StatisticiType;

v\_Puncte NUMBER;

v\_GoluriMarcate NUMBER;

v\_GoluriPrimite NUMBER;

CURSOR MeciuriCursor IS

SELECT puncte, goluri\_marcate, goluri\_primite

FROM PARTICIPA

WHERE echipa\_id = inEchipaID AND competitie\_id = inCompetitieID;

v\_SumaPuncte NUMBER := 0;

BEGIN

FOR j IN 1..20 LOOP

Clasament(j) := 0;

END LOOP;

StatisticiMeciuri := StatisticiMeciuriType();

FOR meci IN MeciuriCursor LOOP

v\_Puncte := meci.puncte;

v\_GoluriMarcate := meci.goluri\_marcate;

v\_GoluriPrimite := meci.goluri\_primite;

Clasament(inEchipaID) := Clasament(inEchipaID) + v\_Puncte;

StatisticiMeciuri(StatisticiMeciuri.COUNT + 1) := v\_Puncte;

END LOOP;

FOR i IN 1..StatisticiMeciuri.COUNT LOOP

v\_SumaPuncte := v\_SumaPuncte + StatisticiMeciuri(i);

END LOOP;

Statistici(inEchipaID) := v\_SumaPuncte;

DBMS\_OUTPUT.PUT\_LINE('Clasamentul echipei ' || inEchipaID || ' în competiția ' || inCompetitieID || ': ' || Clasament(inEchipaID));

DBMS\_OUTPUT.PUT\_LINE('Statisticile echipei ' || inEchipaID || ': Puncte: ' || Clasament(inEchipaID) || ', Suma punctelor: ' || Statistici(inEchipaID) || ', Media punctelor: ' || Statistici(inEchipaID) / StatisticiMeciuri.COUNT);

END CalculeazaClasamentSiStatistici;

/

DECLARE

CompetitieID NUMBER := 1;

EchipaID NUMBER := 1;

BEGIN

CalculeazaClasamentSiStatistici(CompetitieID, EchipaID);

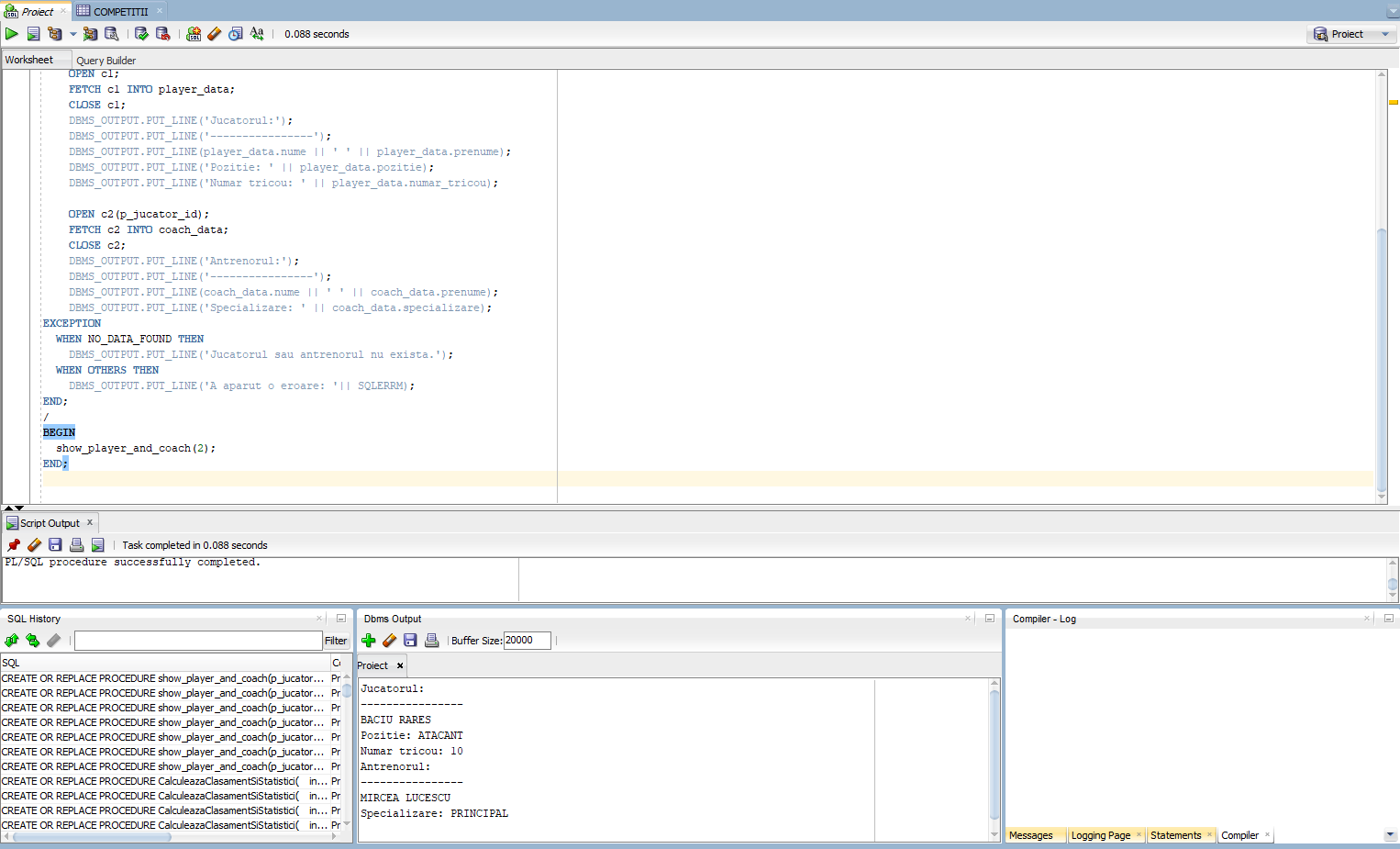
END;

/

1. 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, dependent de celălalt cursor. Apelați subprogramul.

Afisam jucatorul si antrenorul cu care lucreaza dupa id-ul jucatorului.

Cursorul c2 este implicit si depinde de cursorul parametrizat 1.



CREATE OR REPLACE PROCEDURE show\_player\_and\_coach(p\_jucator\_id IN NUMBER)

IS

CURSOR c1 IS

SELECT jucator\_id, nume, prenume, pozitie, numar\_tricou

FROM Jucatori;

CURSOR c2 (p\_jucator\_id\_in NUMBER) IS

SELECT nume, prenume, specializare

FROM Antrenori

JOIN Antreneaza ON (Antreneaza.antrenor\_id = Antrenori.antrenor\_id)

WHERE jucator\_id = p\_jucator\_id\_in;

player\_data Jucatori%ROWTYPE;

TYPE coach\_data\_type IS RECORD(nume VARCHAR(20), prenume VARCHAR(20), specializare VARCHAR(20));

coach\_data coach\_data\_type;

BEGIN

OPEN c1;

FETCH c1 INTO player\_data;

CLOSE c1;

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

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

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

DBMS\_OUTPUT.PUT\_LINE('Pozitie: ' || player\_data.pozitie);

DBMS\_OUTPUT.PUT\_LINE('Numar tricou: ' || player\_data.numar\_tricou);

OPEN c2(p\_jucator\_id);

FETCH c2 INTO coach\_data;

CLOSE c2;

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

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

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

DBMS\_OUTPUT.PUT\_LINE('Specializare: ' || coach\_data.specializare);

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

DBMS\_OUTPUT.PUT\_LINE('Jucatorul sau antrenorul nu exista.');

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE('A aparut o eroare: '|| SQLERRM);

END;

/

BEGIN

show\_player\_and\_coach(2);

END;

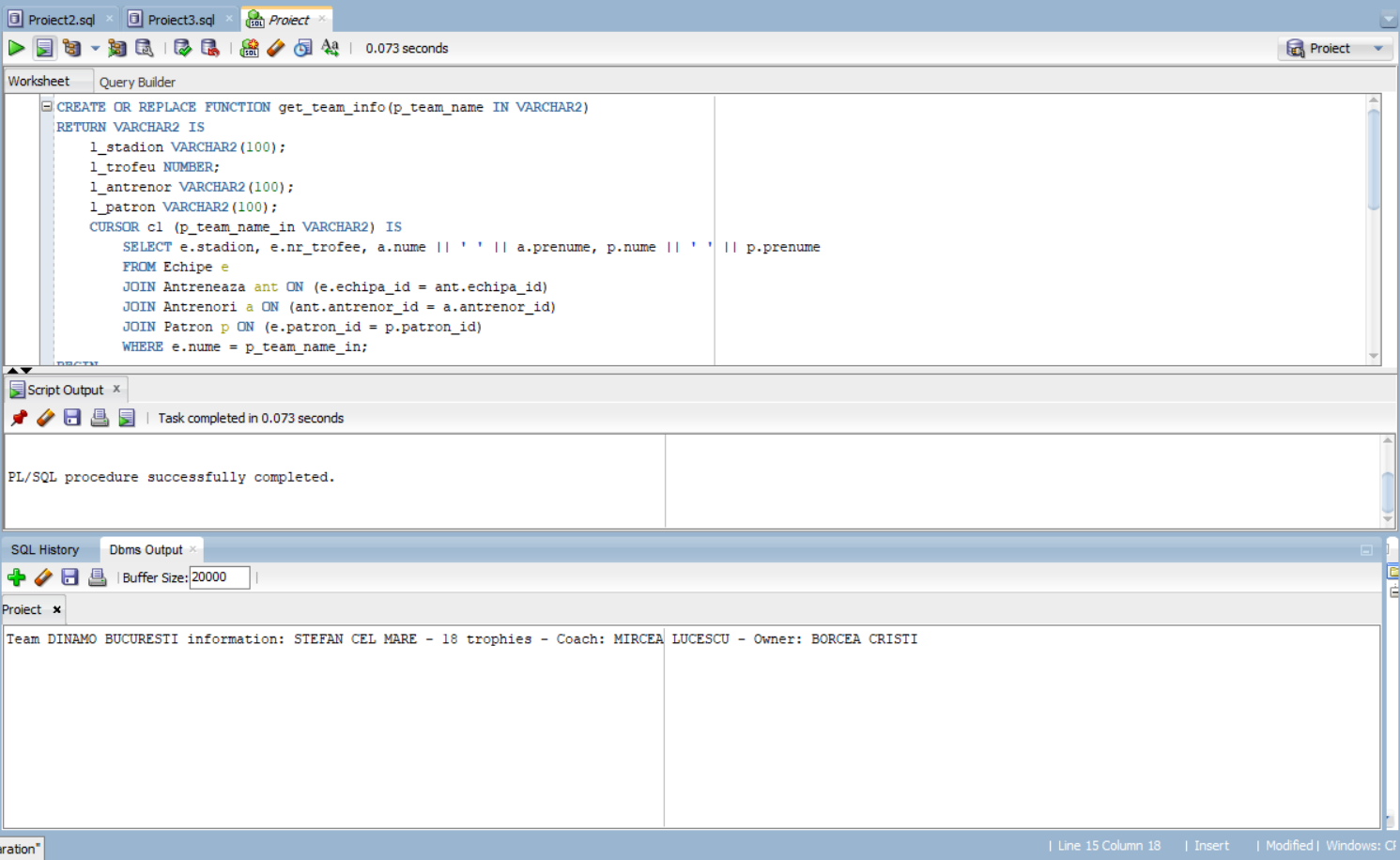
1. Formulați în limbaj natural o problemă pe care să o rezolvați folosind un subprogram stocat independent de tip funcție care să utilizeze într-o singură comandă SQL 3 dintre tabelele definite. Definiți minim 2 excepții proprii. Apelați subprogramul astfel încât să evidențiați toate cazurile definite și tratate.

Afisam mai multe informatii despre o echipa dupa numele ei.

-- afisam mai multe informatii despre o echipa: nume stadion, nr trofee, antrenor, patron

-- imbinam 4 tabele, (echipe, antreneaza, antrenori, patron)

-- avem 2 exceptii (nu gasim numele echipei, mai multe echipe cu acelasi nume)



CREATE OR REPLACE FUNCTION get\_team\_info(p\_team\_name IN VARCHAR2)

RETURN VARCHAR2 IS

l\_stadion VARCHAR2(100);

l\_trofeu NUMBER;

l\_antrenor VARCHAR2(100);

l\_patron VARCHAR2(100);

CURSOR c1 (p\_team\_name\_in VARCHAR2) IS

SELECT e.stadion, e.nr\_trofee, a.nume || ' ' || a.prenume, p.nume || ' ' || p.prenume

FROM Echipe e

JOIN Antreneaza ant ON (e.echipa\_id = ant.echipa\_id)

JOIN Antrenori a ON (ant.antrenor\_id = a.antrenor\_id)

JOIN Patron p ON (e.patron\_id = p.patron\_id)

WHERE e.nume = p\_team\_name\_in;

BEGIN

OPEN c1(p\_team\_name);

FETCH c1 INTO l\_stadion, l\_trofeu, l\_antrenor, l\_patron;

CLOSE c1;

RETURN 'Team '||p\_team\_name||' information: '||l\_stadion||' - '||l\_trofeu||' trophies - Coach: '||l\_antrenor||' - Owner: '||l\_patron;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RETURN 'Team not found';

WHEN TOO\_MANY\_ROWS THEN

RETURN 'Multiple teams found with the same name';

END;

/

BEGIN

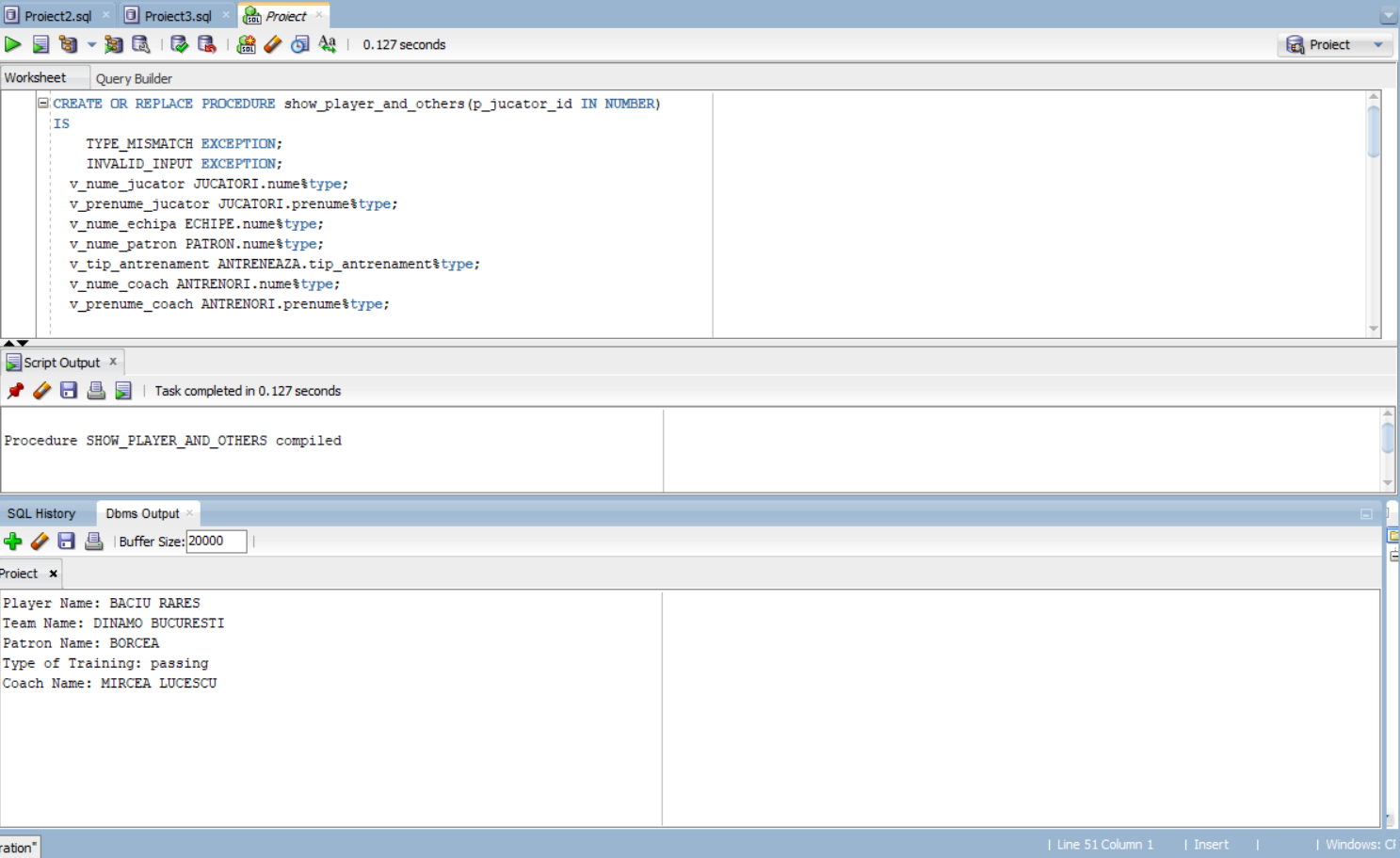
DBMS\_OUTPUT.PUT\_LINE(get\_team\_info('DINAMO BUCURESTI'));

END;

/

1. 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.

Afisam numele jucatorului, echipa, numele patronului, tipul antrenamentului si antrenorul unui jucator dupa id-ul sau, unind 5 tabele si de asemenea adaugat si diverse exceptii.



CREATE OR REPLACE PROCEDURE show\_player\_and\_others(p\_jucator\_id IN NUMBER)

IS

TYPE\_MISMATCH EXCEPTION;

INVALID\_INPUT EXCEPTION;

v\_nume\_jucator JUCATORI.nume%type;

v\_prenume\_jucator JUCATORI.prenume%type;

v\_nume\_echipa ECHIPE.nume%type;

v\_nume\_patron PATRON.nume%type;

v\_tip\_antrenament ANTRENEAZA.tip\_antrenament%type;

v\_nume\_coach ANTRENORI.nume%type;

v\_prenume\_coach ANTRENORI.prenume%type;

CURSOR c1(p\_jucator\_id IN NUMBER) IS

SELECT j.nume, j.prenume, e.nume, p.nume, a.tip\_antrenament, coach.prenume, coach.nume FROM JUCATORI j

JOIN LEGITIMEAZA l ON j.jucator\_id = l.jucator\_id

JOIN ECHIPE e ON l.echipa\_id = e.echipa\_id

JOIN PATRON p ON p.patron\_id = e.patron\_id

JOIN ANTRENEAZA a ON a.echipa\_id = e.echipa\_id

JOIN ANTRENORI coach ON coach.antrenor\_id = a.antrenor\_id

WHERE j.jucator\_id = p\_jucator\_id;

BEGIN

OPEN c1(p\_jucator\_id);

FETCH c1 INTO v\_nume\_jucator, v\_prenume\_jucator, v\_nume\_echipa, v\_nume\_patron, v\_tip\_antrenament, v\_nume\_coach, v\_prenume\_coach;

DBMS\_OUTPUT.PUT\_LINE('Player Name: ' || v\_nume\_jucator || ' '|| v\_prenume\_jucator);

DBMS\_OUTPUT.PUT\_LINE('Team Name: ' || v\_nume\_echipa);

DBMS\_OUTPUT.PUT\_LINE('Patron Name: ' || v\_nume\_patron);

DBMS\_OUTPUT.PUT\_LINE('Type of Training: ' || v\_tip\_antrenament);

DBMS\_OUTPUT.PUT\_LINE('Coach Name: ' || v\_prenume\_coach || ' '|| v\_nume\_coach);

CLOSE c1;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

DBMS\_OUTPUT.PUT\_LINE('No player with the ID ' || p\_jucator\_id || ' found.');

WHEN TOO\_MANY\_ROWS THEN

DBMS\_OUTPUT.PUT\_LINE('More than one player with the ID ' || p\_jucator\_id || ' found.');

WHEN TYPE\_MISMATCH THEN

DBMS\_OUTPUT.PUT\_LINE('Ati gresit un tip de date!');

RETURN;

WHEN INVALID\_INPUT THEN

DBMS\_OUTPUT.PUT\_LINE('Id-ul jucatorului trebuie sa fie de tip int si nu poate fi NULL!');

RETURN;

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE('An error occurred: ' || SQLERRM);

END;

/

BEGIN

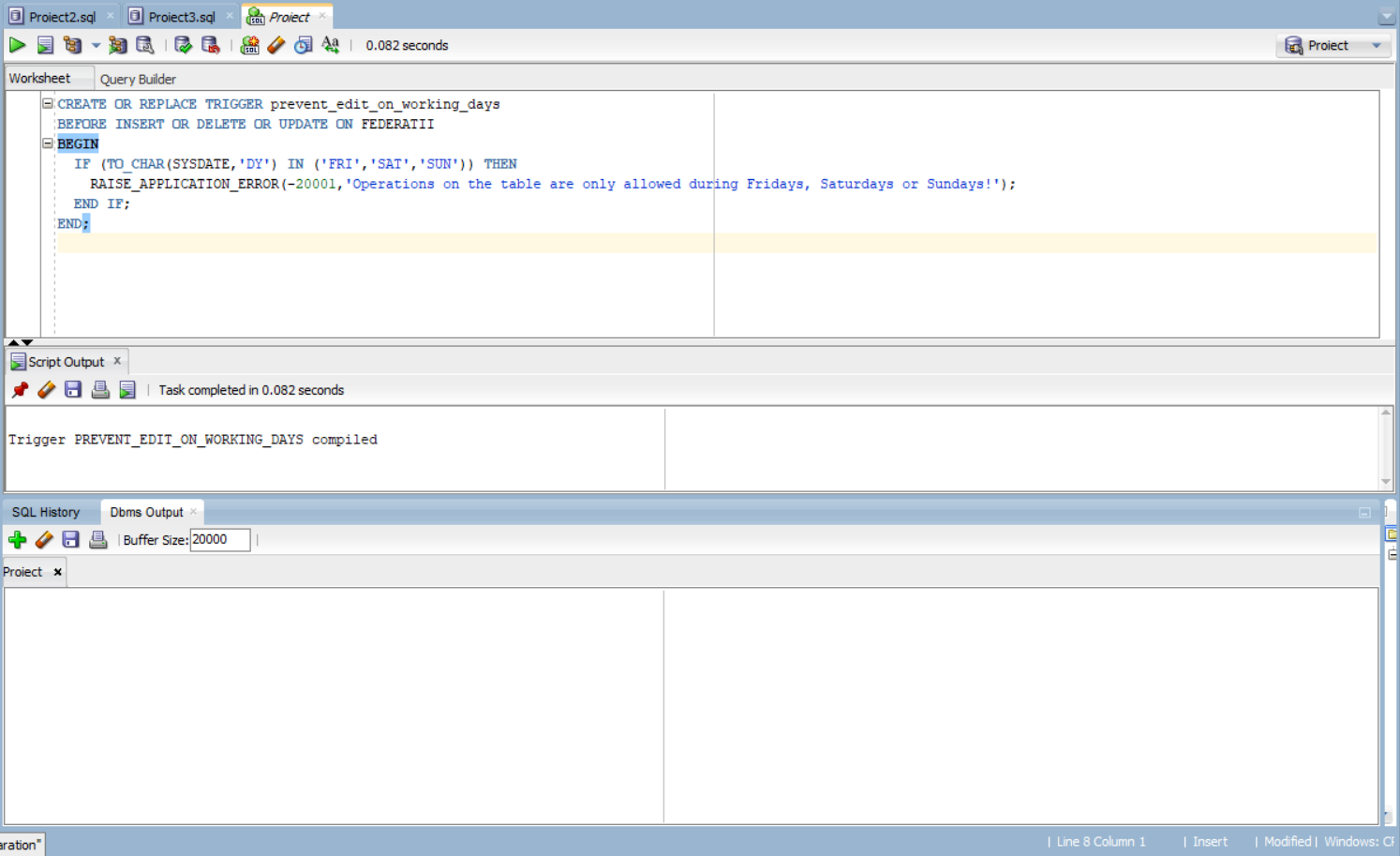
show\_player\_and\_others(1);

END;

/

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

Acest trigger ii interzice utilizatorului sa faca Insert Delete sau Update pe tabelul Federatii in orice din zilele vineri, sambata, duminica.



CREATE OR REPLACE TRIGGER prevent\_edit\_on\_working\_days

BEFORE INSERT OR DELETE OR UPDATE ON FEDERATII

BEGIN

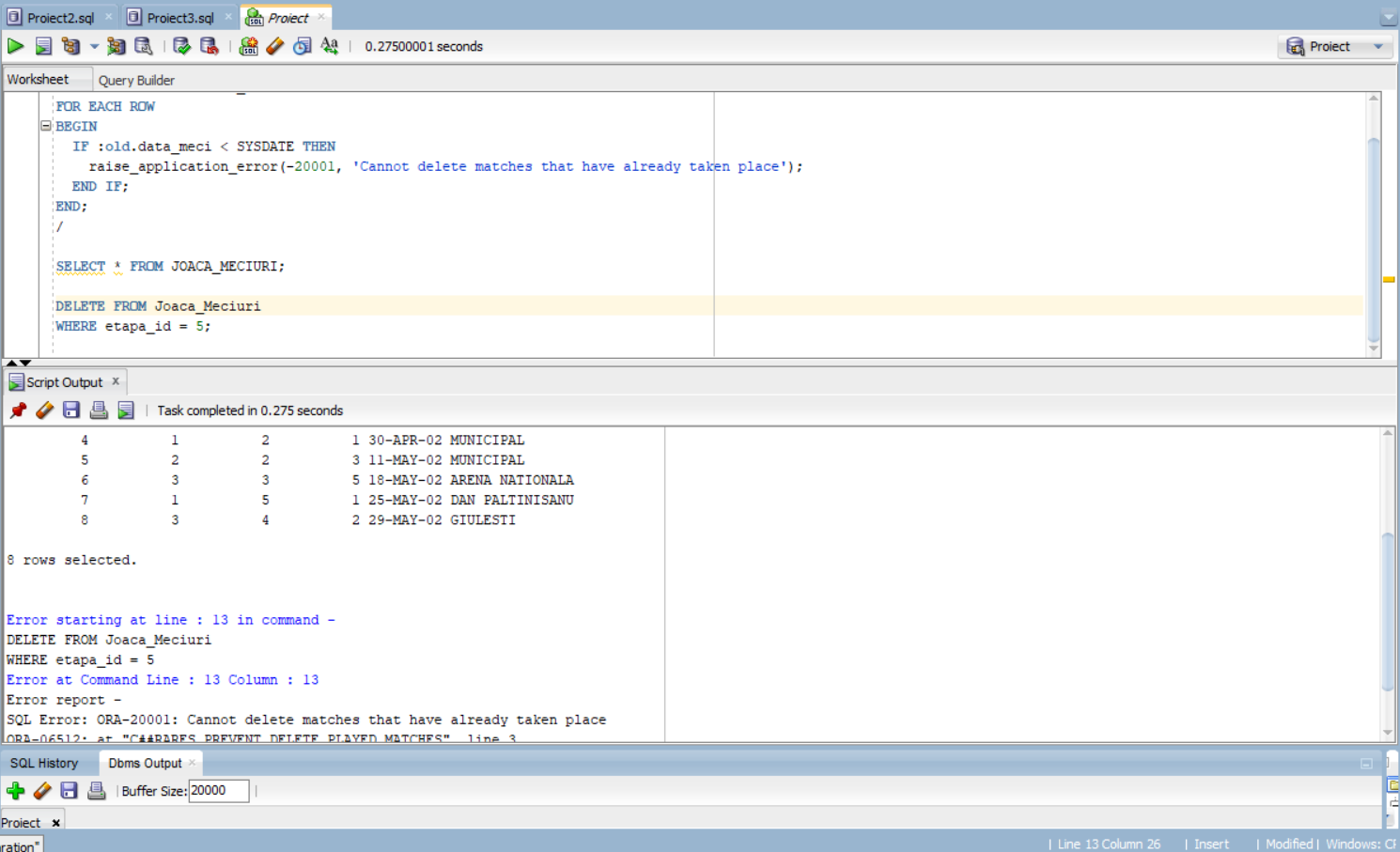
IF (TO\_CHAR(SYSDATE,'DY') IN ('FRI','SAT','SUN')) THEN

RAISE\_APPLICATION\_ERROR(-20001,'Operations on the table are only allowed during Fridays, Saturdays or Sundays!');

END IF;

END;

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



Interzicem utilizatorului sa stearga meciuri care au avut deja loc.

CREATE OR REPLACE TRIGGER prevent\_delete\_played\_matches

BEFORE DELETE ON Joaca\_Meciuri

FOR EACH ROW

BEGIN

IF :old.data\_meci < SYSDATE THEN

raise\_application\_error(-20001, 'Cannot delete matches that have already taken place');

END IF;

END;

/

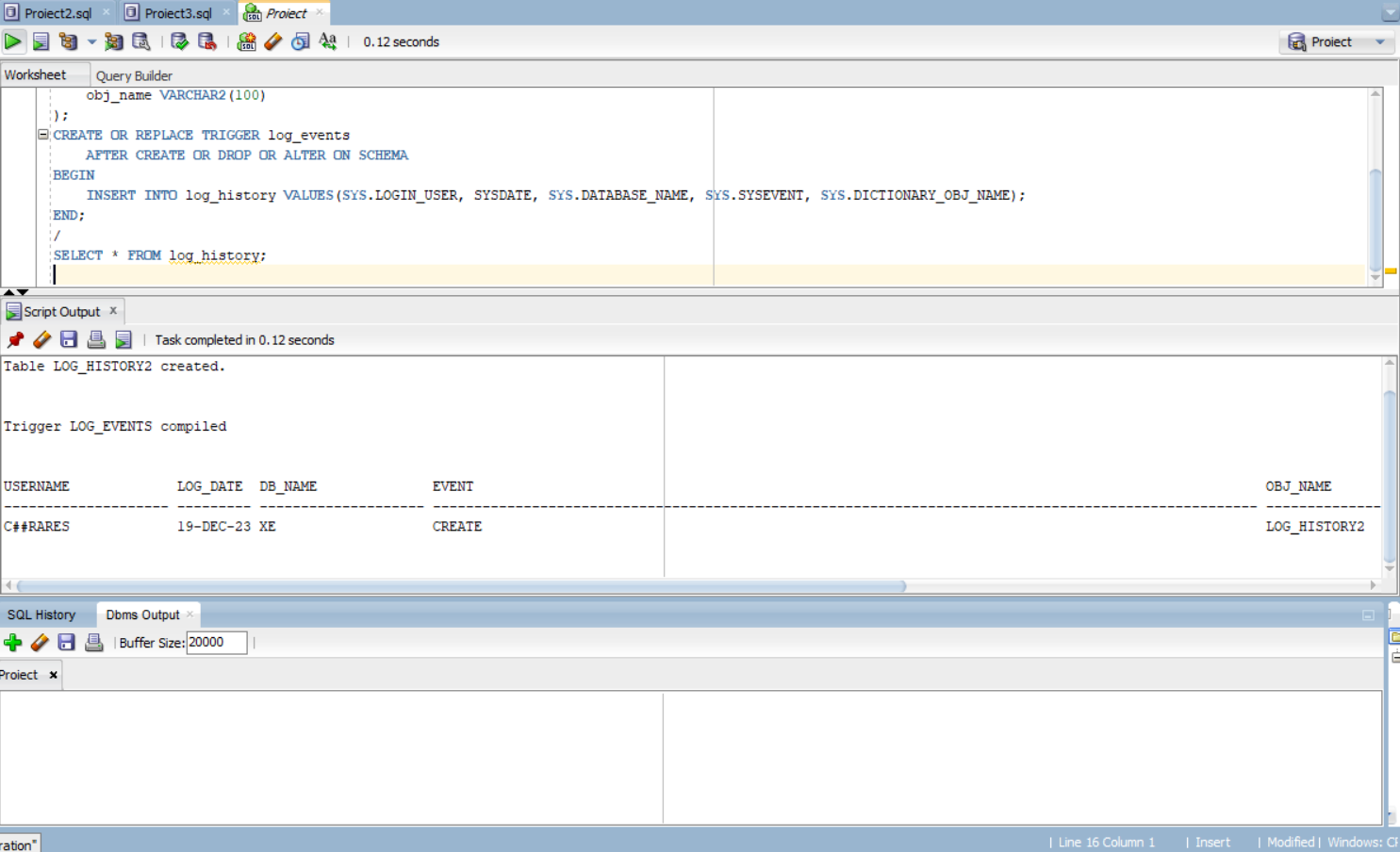
SELECT \* FROM JOACA\_MECIURI;

DELETE FROM Joaca\_Meciuri

WHERE etapa\_id = 5;

1. Definiți un trigger de tip LDD. Declanșați trigger-ul.

Am creat un nou tabel in care retinem orice modificare se intampla in baza noastra de date. Spre exemplu, am oprit si repornit un Trigger si acesta este outputul din tabel.



CREATE TABLE log\_history2

(

username VARCHAR2(20),

log\_date DATE,

db\_name VARCHAR2(20),

event VARCHAR2(100),

obj\_name VARCHAR2(100)

);

CREATE OR REPLACE TRIGGER log\_events

AFTER CREATE OR DROP OR ALTER ON SCHEMA

BEGIN

INSERT INTO log\_history VALUES(SYS.LOGIN\_USER, SYSDATE, SYS.DATABASE\_NAME, SYS.SYSEVENT, SYS.DICTIONARY\_OBJ\_NAME);

END;

/

SELECT \* FROM log\_history;