PROIECT Baze de Date

Semestrul I +II

Proiectul se bazeaza pe baza de date a unei firme distribuitoare de masini la mana a doua. Aceasta include 6 tabele elementare si anume:

-Angajati, ce cuprinde numele angajatilor din cadrul companiei si informatii despre acestia.

-Client, aceasta reprezinta persoanele ce au achizitionat sau au cerut o oferta de vanzare a unei masini din cadrul firmei

-Masini, aceasta tabela fiind alcatuita din masinile disponibile

- Locatii, aici sunt stocate locatiile in care isi are firma sediile

-Departamente, categoriile in care sunt impartiti angajtii

-Comenzi masini, tranzactiile incheiate.

Aceasta stocheaza mult mai usor informatii atat despre angajatii companiei cat si despre tranzactiile pe care le realizeaza aceasta, eficientizand procesul de vanzare de masini.

CREATE TABLE CLIENT\_PROIECT (

id\_client NUMBER(3) PRIMARY KEY,

cnp CHAR(13) UNIQUE,

nume VARCHAR2(25) NOT NULL,

prenume VARCHAR2(25) NOT NULL,

telefon NUMBER(10) NOT NULL,

email VARCHAR(35) UNIQUE,

salariul NUMBER(8) NOT NULL

);

CREATE TABLE MASINI\_PROIECT (

id\_masina NUMBER(7) PRIMARY KEY,

pret NUMBER(10) NOT NULL,

kilometri NUMBER(9) NOT NULL,

id\_locatie NUMBER(4) NOT NULL,

SERIE NUMBER(15) UNIQUE,

MODEL VARCHAR(25)

);

CREATE TABLE LOCATII\_PROIECT (

id\_locatie NUMBER(4) PRIMARY KEY,

oras VARCHAR(20) NOT NULL,

strada VARCHAR(20) NOT NULL,

numar NUMBER(5) NOT NULL

);

CREATE TABLE ANGAJATI\_PROIECT (

id\_angajat NUMBER(6) PRIMARY KEY,

cnp CHAR(13) UNIQUE,

nume VARCHAR2(25) NOT NULL,

prenume VARCHAR2(25) NOT NULL,

salariul NUMBER(8) NOT NULL,

data\_anagajare DATE,

id\_locatie NUMBER(4),

CONSTRAINT FK\_angajati\_proiect FOREIGN KEY (id\_locatie) REFERENCES locatii\_proiect(id\_locatie));

);

CREATE TABLE COMENZI\_MASINI\_PROIECT (

id\_comanda NUMBER(7) PRIMARY KEY,

id\_masina NUMBER(7),

CONSTRAINT FK\_comenzi\_masini\_proiect FOREIGN KEY (id\_masina) REFERENCES masini\_proiect(id\_masina),

id\_client NUMBER(6),

CONSTRAINT FK2\_comenzi\_masini\_proiect FOREIGN KEY (id\_client) REFERENCES client\_proiect(id\_client),

data\_comanda DATE ,

id\_angajat NUMBER(6) ,

CONSTRAINT FK3\_comenzi\_masini\_proiect FOREIGN KEY (id\_angajat) REFERENCES angajati\_proiect(id\_angajat));

CREATE TABLE DEPARTAMENTE\_PROIECT (

id\_departament NUMBER(4) PRIMARY KEY,

denumire\_departament VARCHAR(15) NOT NULL,

id\_manager NUMBER(6),

CONSTRAINT FK\_departamente\_proiect FOREIGN KEY (id\_manager) REFERENCES angajati\_proiect(id\_angajat),

id\_locatie NUMBER(4),

CONSTRAINT FK2\_departamente\_proiect FOREIGN KEY (id\_locatie) REFERENCES locatii\_proiect(id\_locatie)

);

INSERT INTO masini\_proiect (id\_masina, pret, kilometri,serie, model) VALUES (1,20000,10000,75498,'BMW');

INSERT INTO masini\_proiect (id\_masina, pret, kilometri,serie, model) VALUES (2,25000,5000,75698,'Volkswagen');

INSERT INTO masini\_proiect (id\_masina, pret, kilometri,serie, model) VALUES (3,70000,2500,89745,'Mercedes');

INSERT INTO masini\_proiect (id\_masina, pret, kilometri,serie, model) VALUES (4,100000,5000,63214,'Porche');

INSERT INTO masini\_proiect (id\_masina, pret, kilometri,serie, model) VALUES (5,12500,50000,58695,'Skoda');

INSERT INTO masini\_proiect (id\_masina, pret, kilometri,serie, model) VALUES (6,40000,25000,45896,'Audi');

INSERT INTO masini\_proiect (id\_masina, pret, kilometri,serie, model) VALUES (7,30000,60000,75436,'Toyota');

INSERT INTO locatii\_proiect (id\_locatie, oras, strada,numar) VALUES (1,'Bucuresti','Dorobanti',25);

INSERT INTO locatii\_proiect (id\_locatie, oras, strada,numar) VALUES (2,'Bucuresti','Unirii',13);

INSERT INTO locatii\_proiect (id\_locatie, oras, strada,numar) VALUES (3,'Cluj','Iuliu Maniu',30);

INSERT INTO locatii\_proiect (id\_locatie, oras, strada,numar) VALUES (4,'Galati','Brailei',28);

INSERT INTO client\_proiect (id\_client, nume, prenume , cnp, telefon, email, salariu ) VALUES (2,'Neagu','Mihnea',5020810170018,0724587843,',mickertricker@yahoo.com',6000);

INSERT INTO client\_proiect (id\_client, nume, prenume , cnp, telefon, email, salariu ) VALUES (4,'Dinica','Mircea',5030240170018,0724587843,',miri@yahoo.com',8000);

INSERT INTO client\_proiect (id\_client, nume, prenume , cnp, telefon, email, salariu ) VALUES (5,'Stanciu','Sergiu',5030730170018,0724587843,',sergiu@yahoo.com',4000);

INSERT INTO client\_proiect (id\_client, nume, prenume , cnp, telefon, email, salariu ) VALUES (6,'Horceag','Andrei',5018130170018,0724582943,',andreihas@yahoo.com',9000);

INSERT INTO DEPARTAMENTE\_PROIECT (id\_departament, denumire\_departament, id\_locatie) VALUES(1,'HR',1);

INSERT INTO DEPARTAMENTE\_PROIECT (id\_departament, denumire\_departament, id\_locatie) VALUES(2,'Vanzari',1);

INSERT INTO DEPARTAMENTE\_PROIECT (id\_departament, denumire\_departament, id\_locatie) VALUES(3,'Marketing',1);

INSERT INTO DEPARTAMENTE\_PROIECT (id\_departament, denumire\_departament, id\_locatie) VALUES(4,'Service',4);

INSERT INTO DEPARTAMENTE\_PROIECT (id\_departament, denumire\_departament, id\_locatie) VALUES(5,'Investitii',3);

INSERT INTO Angajati\_PROIECT (id\_angajat, nume, prenume,cnp,salariul,id\_departament,id\_locatie, data\_angajare) VALUES( 2,'Stoica','Elias', 5030130170022, 20000,2,1, TO\_DATE('jan 28, 2006','mon dd, yyyy'));

INSERT INTO Angajati\_PROIECT (id\_angajat, nume, prenume,cnp,salariul,id\_departament,id\_locatie, data\_angajare) VALUES( 3,'Vlad','Tudor', 5020715170022, 20000,1,1, TO\_DATE('jan 25, 2007','mon dd, yyyy'));

INSERT INTO Angajati\_PROIECT (id\_angajat, nume, prenume,cnp,salariul,id\_departament,id\_locatie, data\_angajare) VALUES( 4,'Chiariac','Ella', 6030230170022, 20000,4,1, TO\_DATE('mar 25, 2006','mon dd, yyyy'));

INSERT INTO Angajati\_PROIECT (id\_angajat, nume, prenume,cnp,salariul,id\_departament,id\_locatie, data\_angajare) VALUES( 5,'Preda','Carla', 6030180170022, 20000,4,1, TO\_DATE('sep 15, 2010','mon dd, yyyy'));

INSERT INTO Angajati\_PROIECT (id\_angajat, nume, prenume,cnp,salariul,id\_departament,id\_locatie, data\_angajare) VALUES( 6,'Popa','Isabela', 6030230170022, 15000,5,3, TO\_DATE('jan 28, 2012','mon dd, yyyy'));

INSERT INTO Angajati\_PROIECT (id\_angajat, nume, prenume,cnp,salariul,id\_departament,id\_locatie, data\_angajare) VALUES( 7,'Popa','Emanuel', 5030090170022, 16000,4,4, TO\_DATE('jul 10, 2004','mon dd, yyyy'));

INSERT INTO Angajati\_PROIECT (id\_angajat, nume, prenume,cnp,salariul,id\_departament,id\_locatie, data\_angajare) VALUES( 8,'Grigoras','Tudor', 5020415170022, 15000,4,4, TO\_DATE('jan 28, 2012','mon dd, yyyy'))

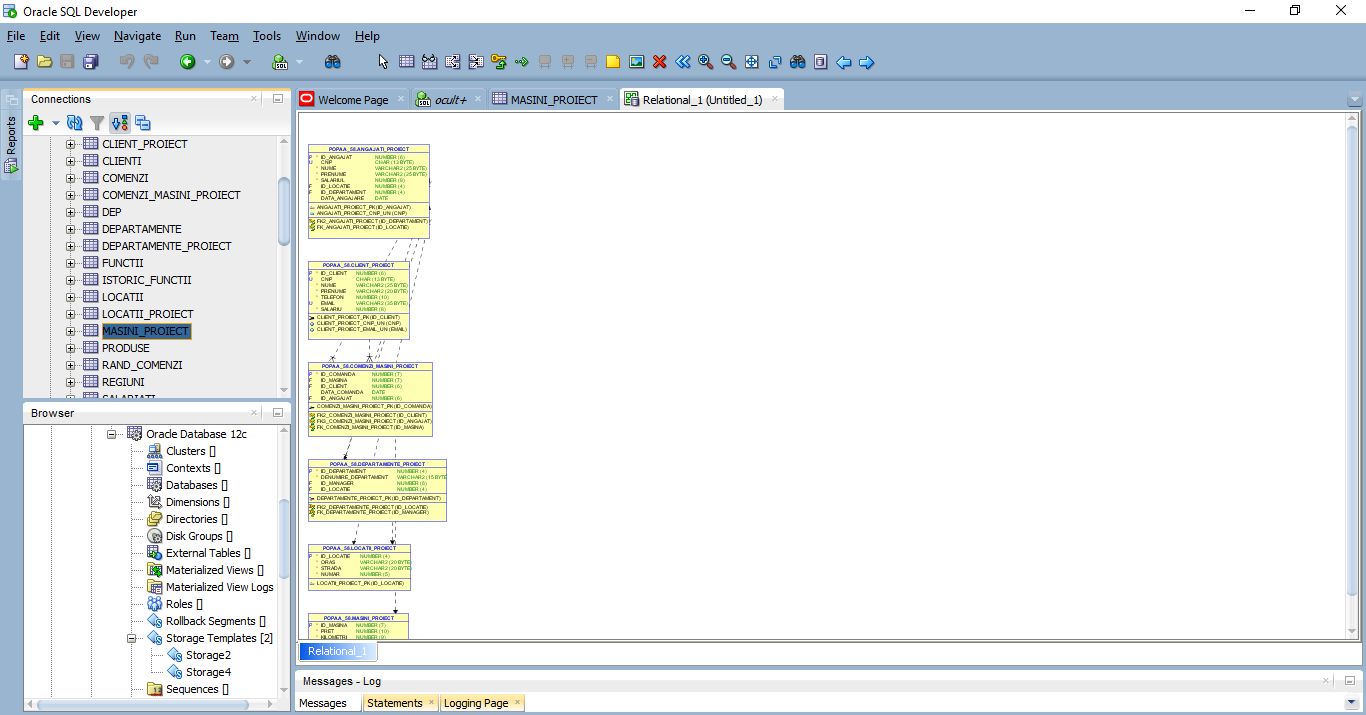
Delete from angajati\_proiect

where id\_angajat=11;

INSERT INTO COMENZI\_MASINI\_PROIECT (id\_comanda, id\_masina, id\_client,id\_angajat,data\_comanda) VALUES( 1,2,3,1, TO\_DATE('oct 28, 2022','mon dd, yyyy'));

INSERT INTO COMENZI\_MASINI\_PROIECT (id\_comanda, id\_masina, id\_client,id\_angajat,data\_comanda) VALUES( 2,1,1,2, TO\_DATE('nov 13, 2022','mon dd, yyyy'));

INSERT INTO COMENZI\_MASINI\_PROIECT (id\_comanda, id\_masina, id\_client,id\_angajat,data\_comanda) VALUES( 3,4,4,3, TO\_DATE('nov 25, 2022','mon dd, yyyy'));



Tema 4.12

UPDATE angajati\_proiect

SET salariul=salariul+1000

WHERE salariul<20000;

SELECT \* FROM angajati\_proiect;

UPDATE angajati\_proiect

SET id\_locatie=3

WHERE id\_locatie=4;

SELECT \* FROM angajati\_proiect;

UPDATE angajati\_proiect

SET id\_departament=4

WHERE id\_departament=3;

SELECT \* FROM angajati\_proiect;

UPDATE masini\_proiect

SET pret=pret-(10/100)\*pret

WHERE pret>50000;

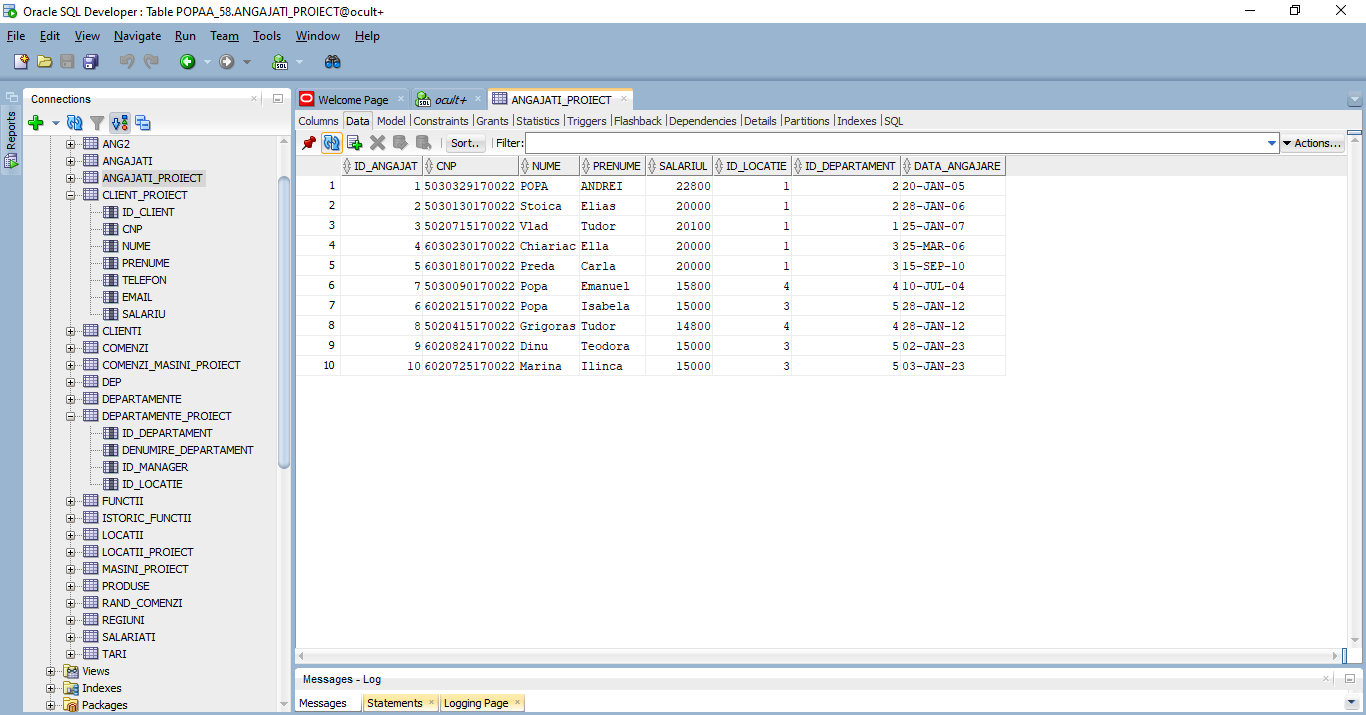
SELECT \* FROM masini\_proiect;

UPDATE masini\_proiect

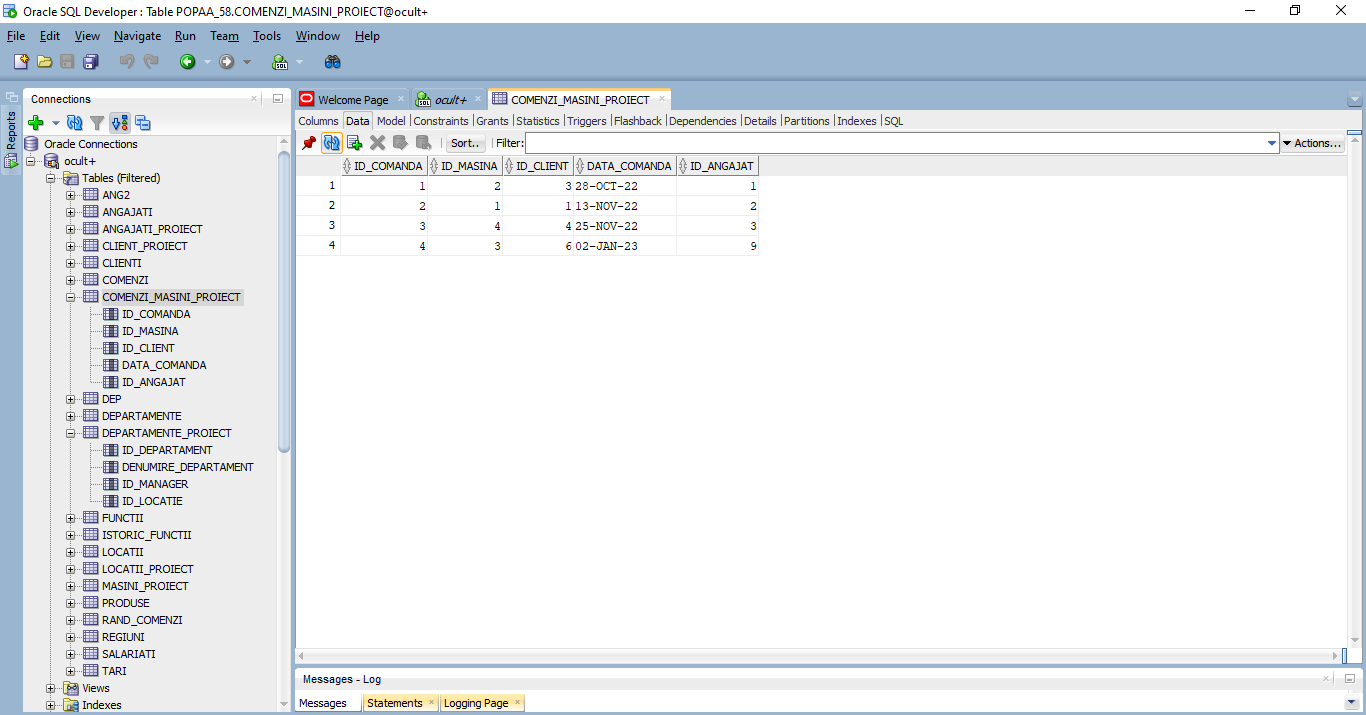
SET kilometri=kilometri-(10/100)\*kilometri

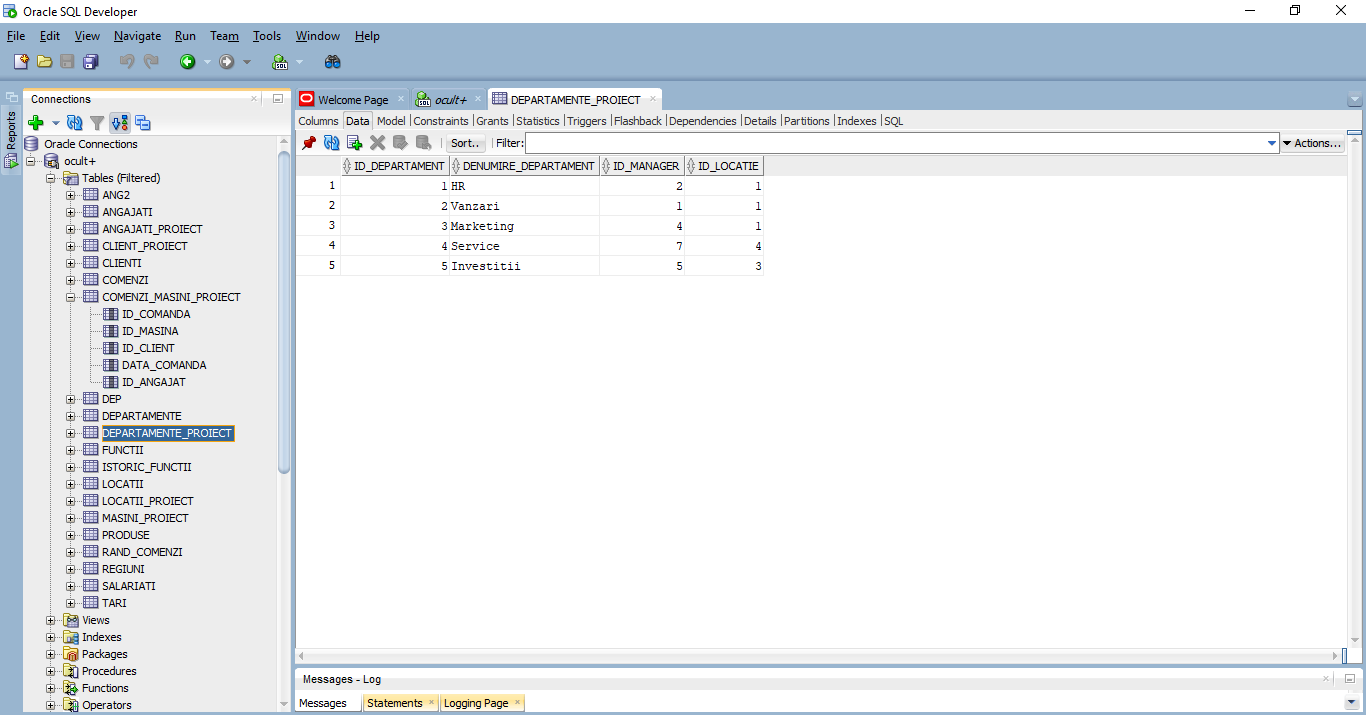
WHERE kilometri>50000;

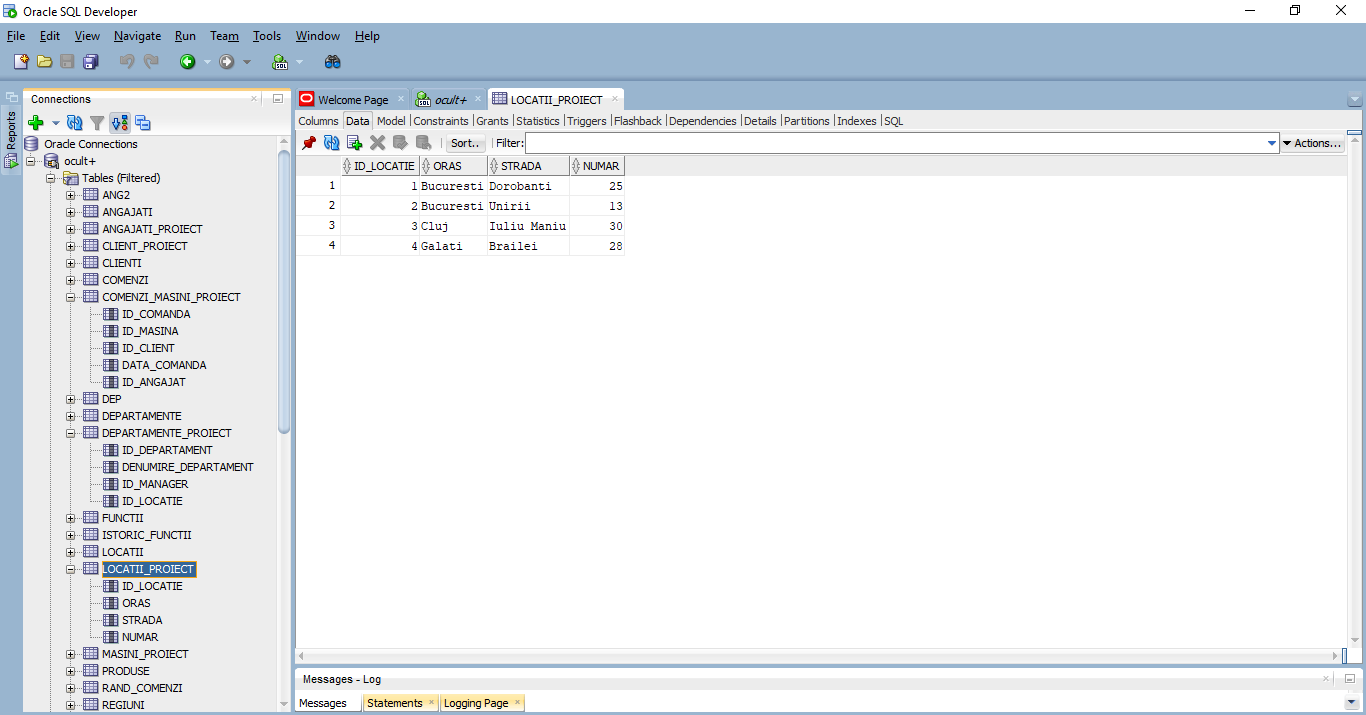
SELECT \* FROM masini\_proiect;

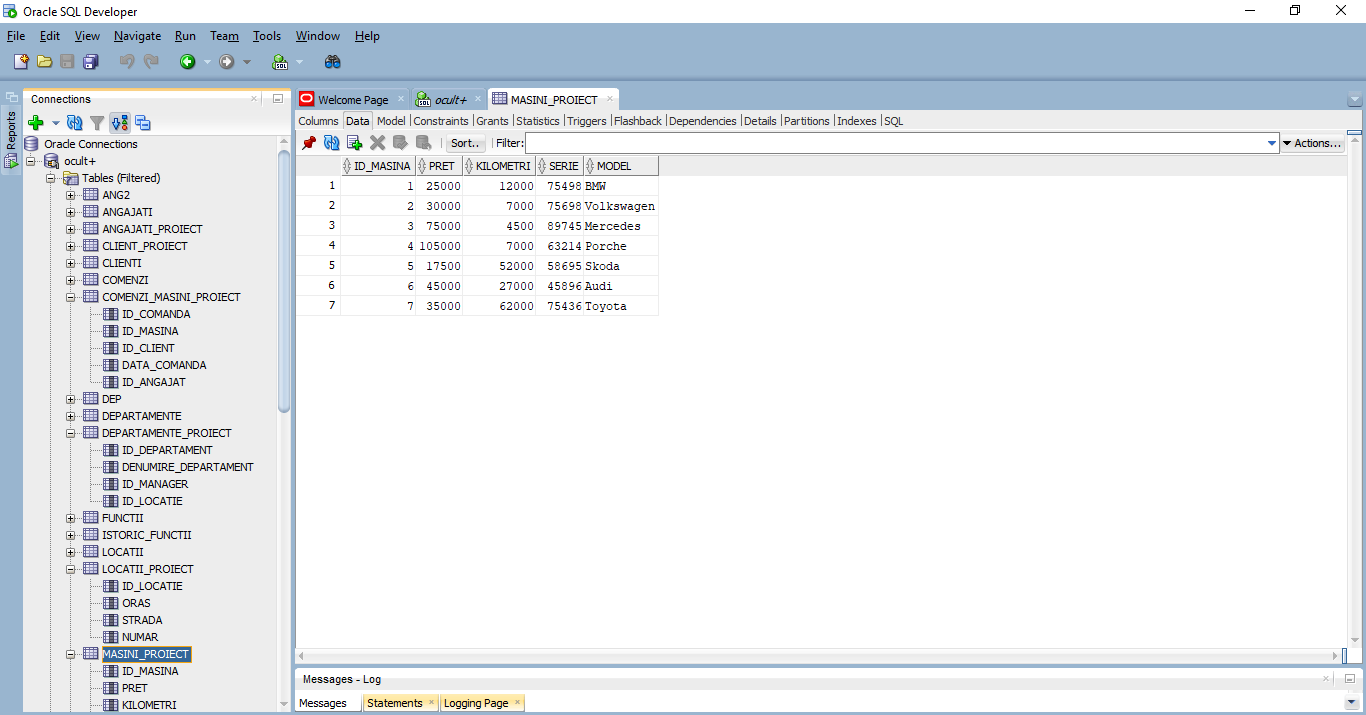












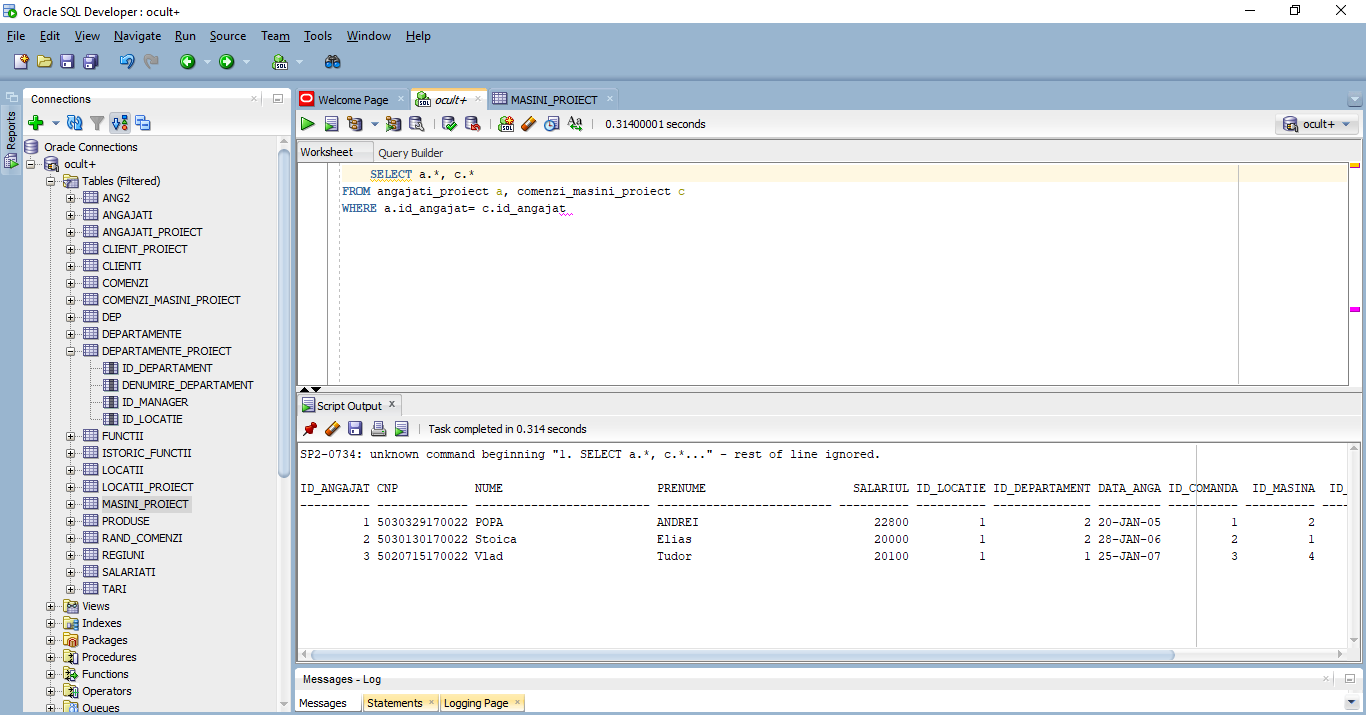
21.12

--1 Să se selecteze comenzile intermediate de angajați

SELECT a.\*, c.\*

FROM angajati\_proiect a, comenzi\_masini\_proiect c

WHERE a.id\_angajat= c.id\_angajat



--2 Să se selecteze comenzile intermediate de angajatul Vlad numai în luna noiembrie

SELECT angajati\_proiect.\*, comenzi\_masini\_proiect.\*

FROM angajati\_proiect, comenzi\_masini\_proiect

WHERE angajati\_proiect.id\_angajat=comenzi\_masini\_proiect.id\_angajat

AND lower(comenzi\_masini\_proiect.data\_comanda) like '%nov%'

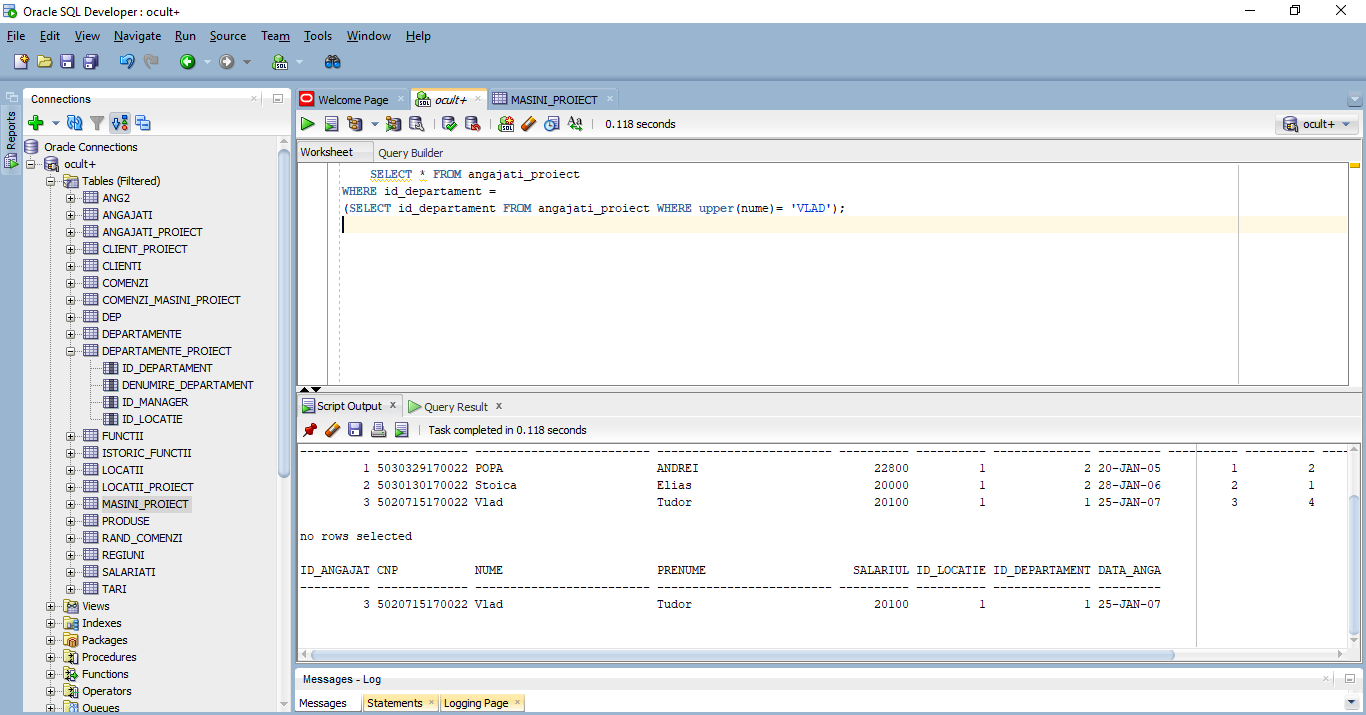
AND upper(angajati\_proiect.nume) = 'Vlad';

--3 Să se selecteze angajații care sunt în același departament cu angajatul Vlad

SELECT \* FROM angajati\_proiect

WHERE id\_departament =

(SELECT id\_departament FROM angajati\_proiect WHERE upper(nume)= 'VLAD');

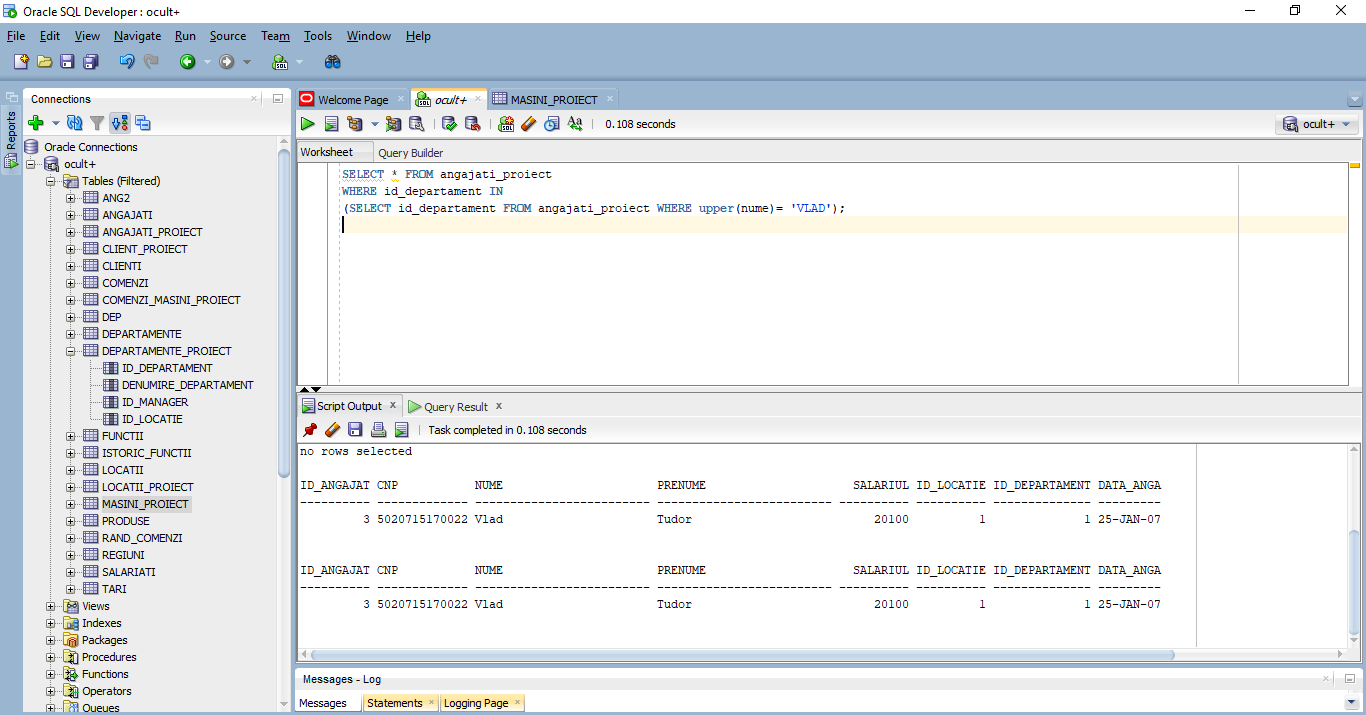


--4

SELECT \* FROM angajati\_proiect

WHERE id\_departament IN

(SELECT id\_departament FROM angajati\_proiect WHERE upper(nume)= 'VLAD');

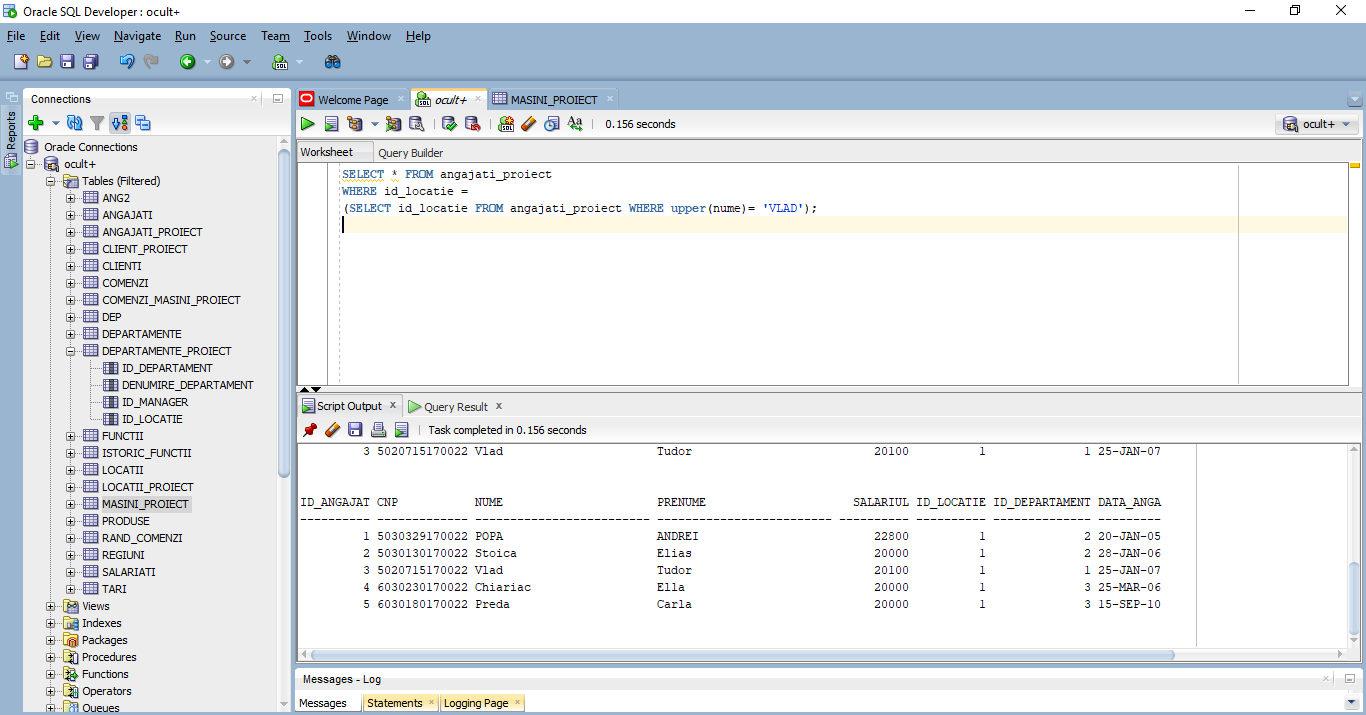


--5 Să se selecteze angajații care sunt în aceeași locatie cu angajatul Vlad

SELECT \* FROM angajati\_proiect

WHERE id\_locatie =

(SELECT id\_locatie FROM angajati\_proiect WHERE upper(nume)= 'VLAD');

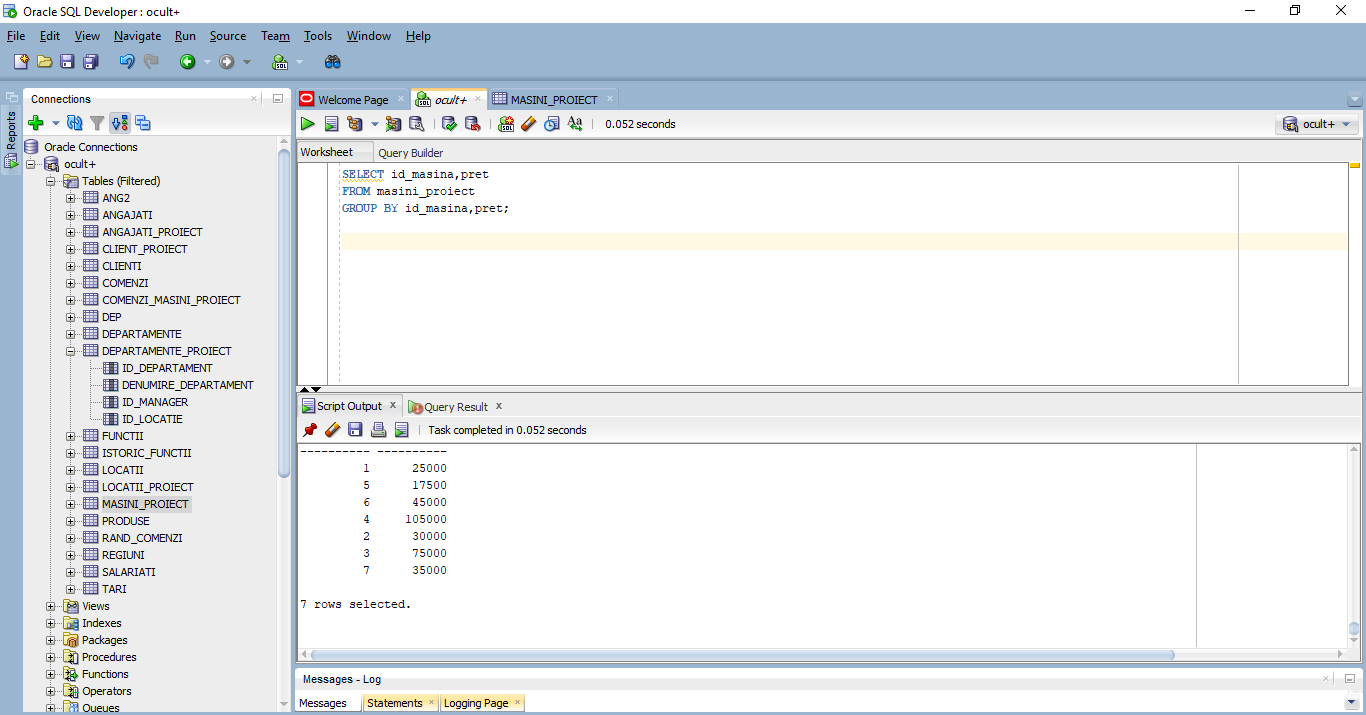


--1 Sa se afiseze preturile masinilor grupate dupa id

SELECT id\_masina,pret

FROM masini\_proiect

GROUP BY id\_masina,pret;



--2 Să se afişeze masinile și pretul celor cu pretul mediu mai mare de 2000

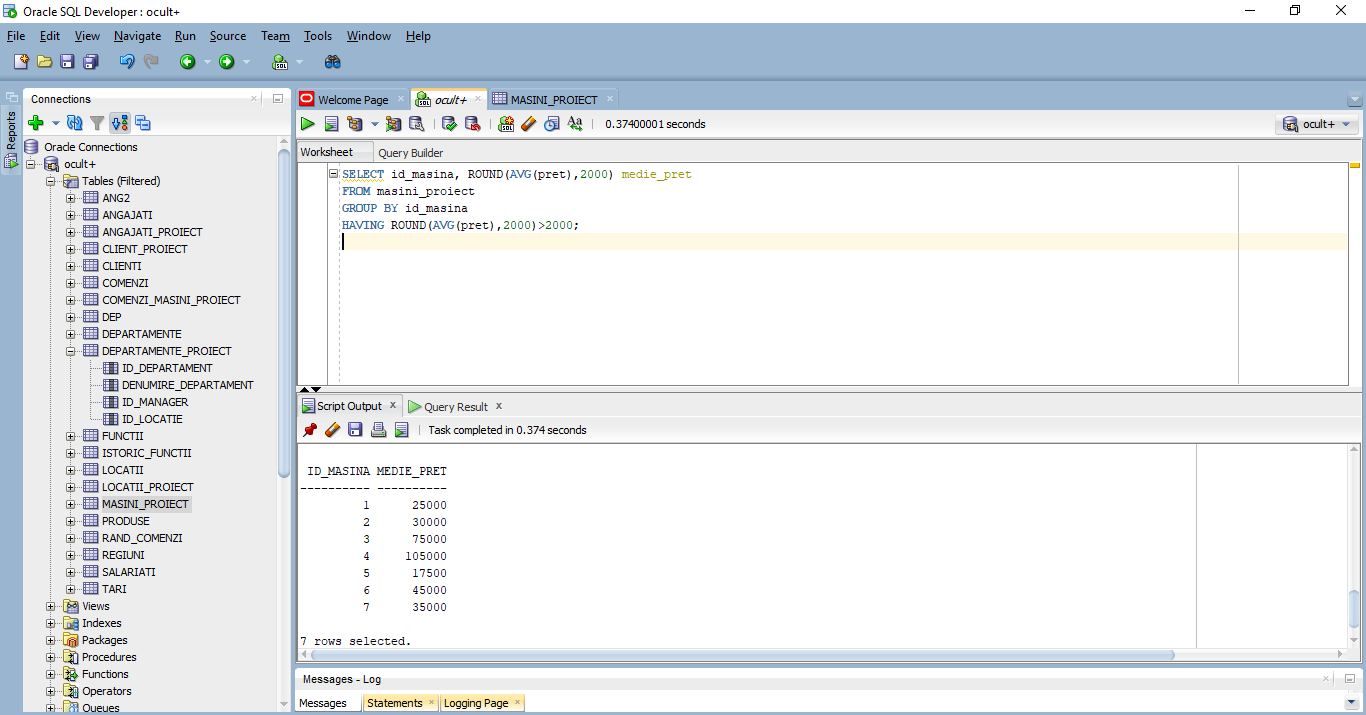
SELECT id\_masina,

ROUND(AVG(pret),2000) medie\_pret

FROM masini\_proiect

GROUP BY id\_masina

HAVING ROUND(AVG(pret),2000)>2000;



--3

SELECT \* FROM

(SELECT ROWNUM as rn, id\_angajat, nume, salariul

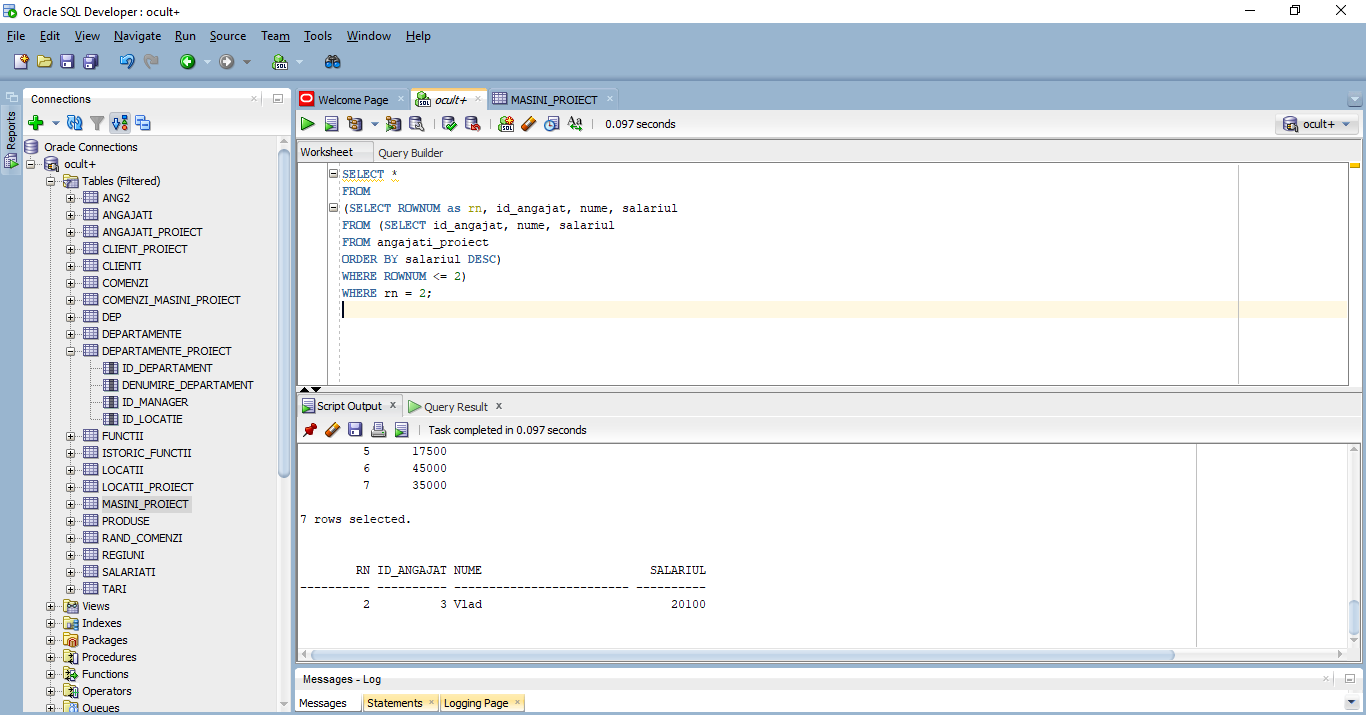
FROM (SELECT id\_angajat, nume, salariul

FROM angajati\_proiect

ORDER BY salariul DESC)

WHERE ROWNUM <= 2)

WHERE rn = 2;



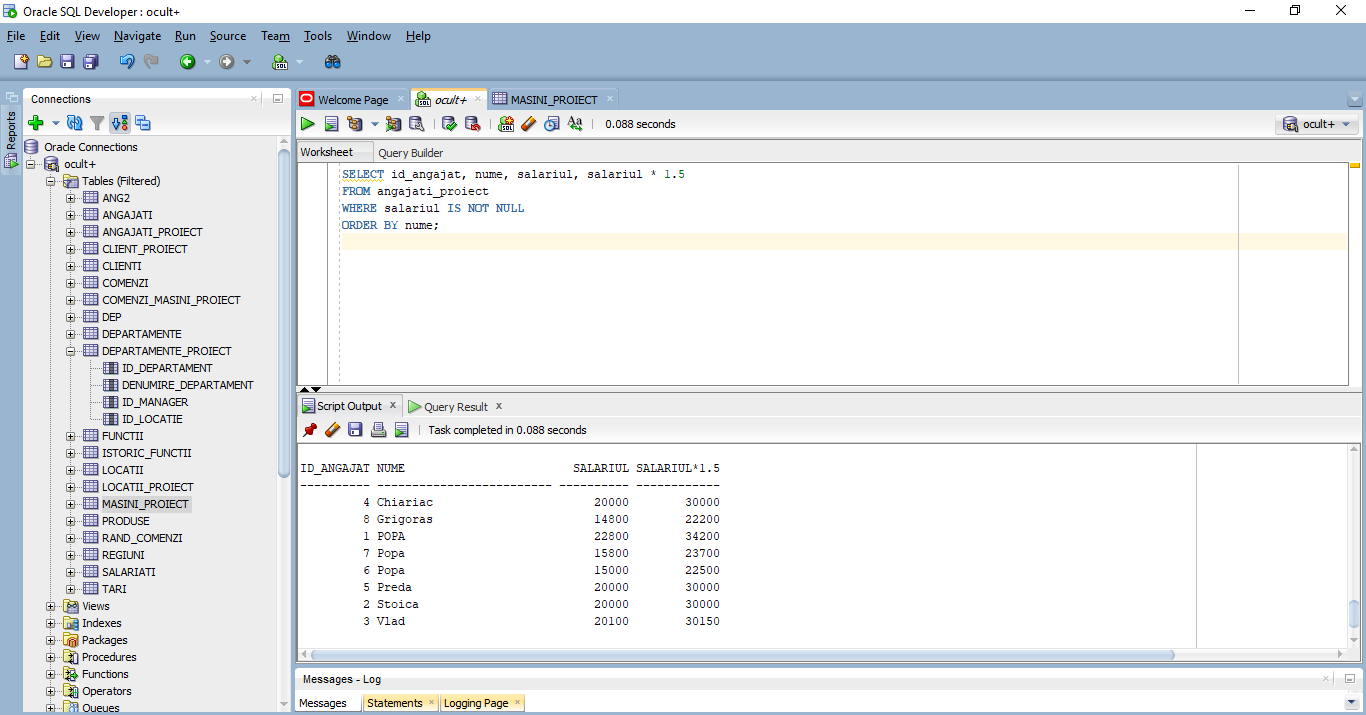
--4 Sa se afiseze angajatii si salariul acestora inainte si dupa marire salariala ordonati alfabetic

SELECT id\_angajat, nume, salariul, salariul \* 1.5

FROM angajati\_proiect

WHERE salariul IS NOT NULL

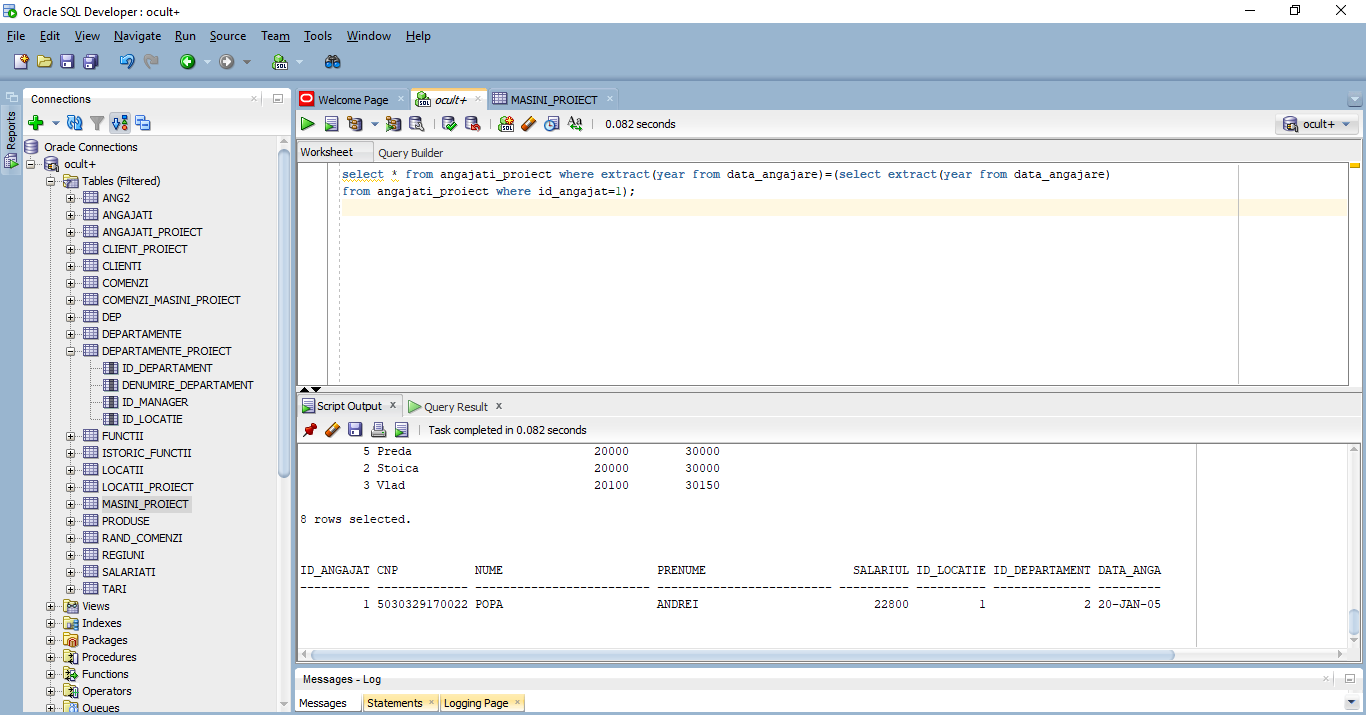
ORDER BY nume;



--1 Sa se afiseze angajatii care au fost angajati in acelasi an cu angajatul cu id-ul 1

select \* from angajati\_proiect where extract(year from data\_angajare)=(select extract(year from data\_angajare)

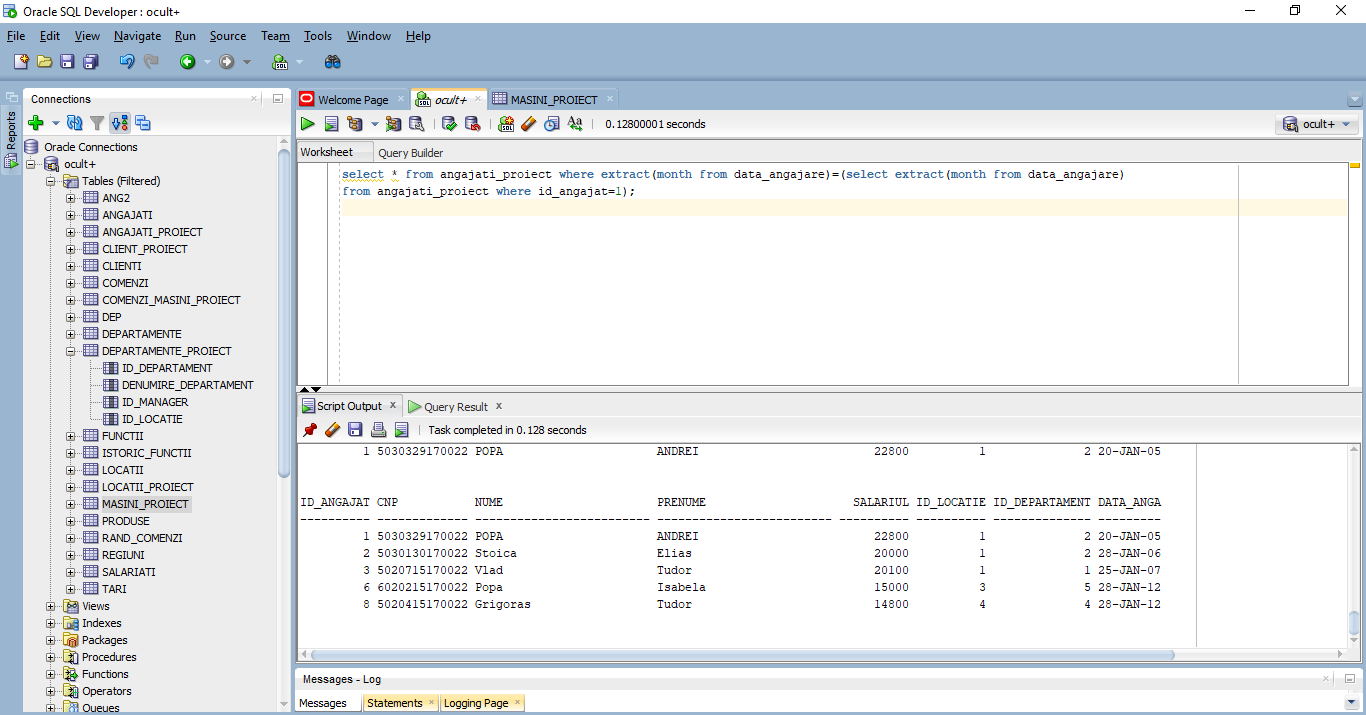
from angajati\_proiect where id\_angajat=1);



--2 Sa se afiseze angajatii care au fost angajati in aceeasi luna cu angajatul cu id-ul 1.

select \* from angajati\_proiect where extract(month from data\_angajare)=(select extract(month from data\_angajare)

from angajati\_proiect where id\_angajat=1);



--3 Sa se afiseze angajatii din aceeasi locatie cu angajatul Vlad diferiti de acesta

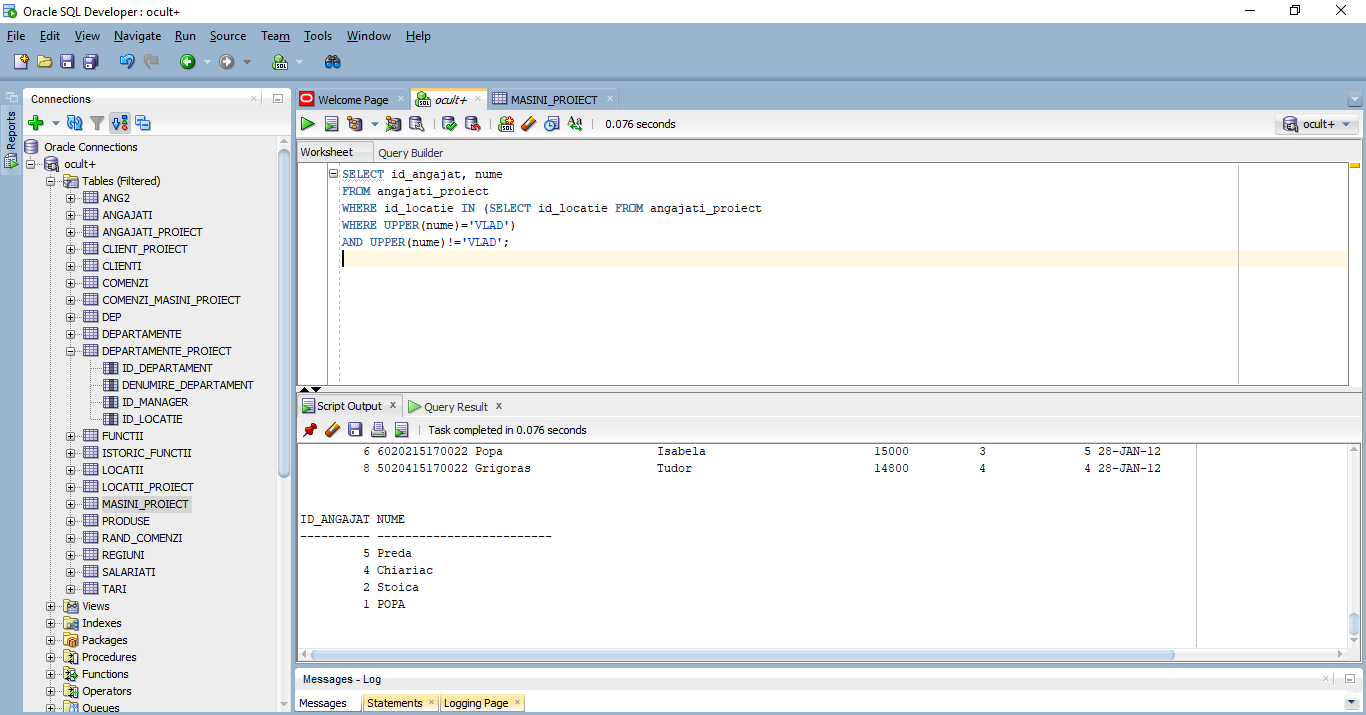
SELECT id\_angajat, nume

FROM angajati\_proiect

WHERE id\_locatie IN (SELECT id\_locatie FROM angajati\_proiect

WHERE UPPER(nume)='VLAD')

AND UPPER(nume)!='VLAD';



--4 Sa se mareasca salariul de 1,1 ori angajatilor din aceeasi locatie si department cu angajatul Vlad

UPDATE angajati\_proiect

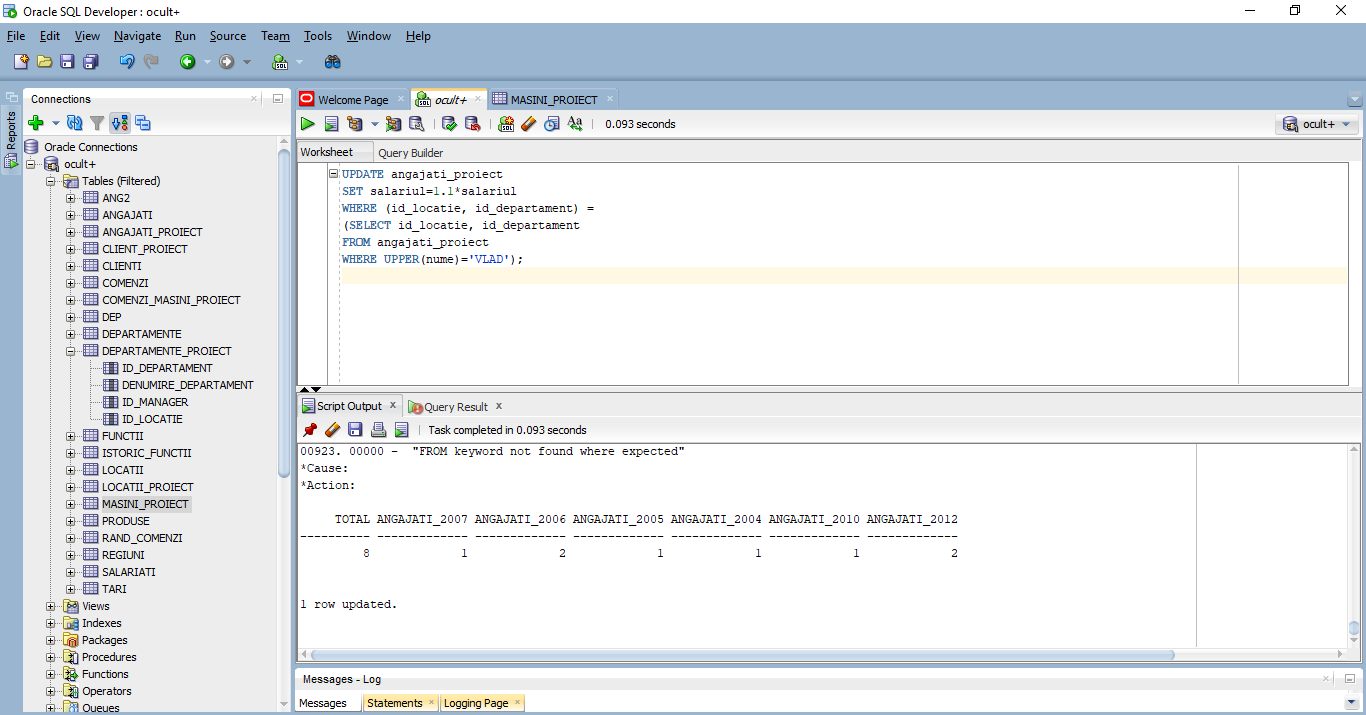
SET salariul=1.1\*salariul

WHERE (id\_locatie, id\_departament) =

(SELECT id\_locatie, id\_departament

FROM angajati\_proiect

WHERE UPPER(nume)='VLAD');



--5 Sa se afiseze totalul de angajati din fiecare dintre urmatorii ani:2004,2005,2006,2007,2010,2012

SELECT (SELECT COUNT(id\_angajat) FROM angajati\_proiect) "TOTAL",

(SELECT COUNT(id\_angajat) FROM angajati\_proiect WHERE EXTRACT(YEAR FROM

data\_angajare)=2007) "ANGAJATI\_2007",

(SELECT COUNT(id\_angajat) FROM angajati\_proiect WHERE EXTRACT(YEAR FROM

data\_angajare)=2006) "ANGAJATI\_2006",

(SELECT COUNT(id\_angajat) FROM angajati\_proiect WHERE EXTRACT(YEAR FROM

data\_angajare)=2005) "ANGAJATI\_2005",

(SELECT COUNT(id\_angajat) FROM angajati\_proiect WHERE EXTRACT(YEAR FROM

data\_angajare)=2004) "ANGAJATI\_2004",

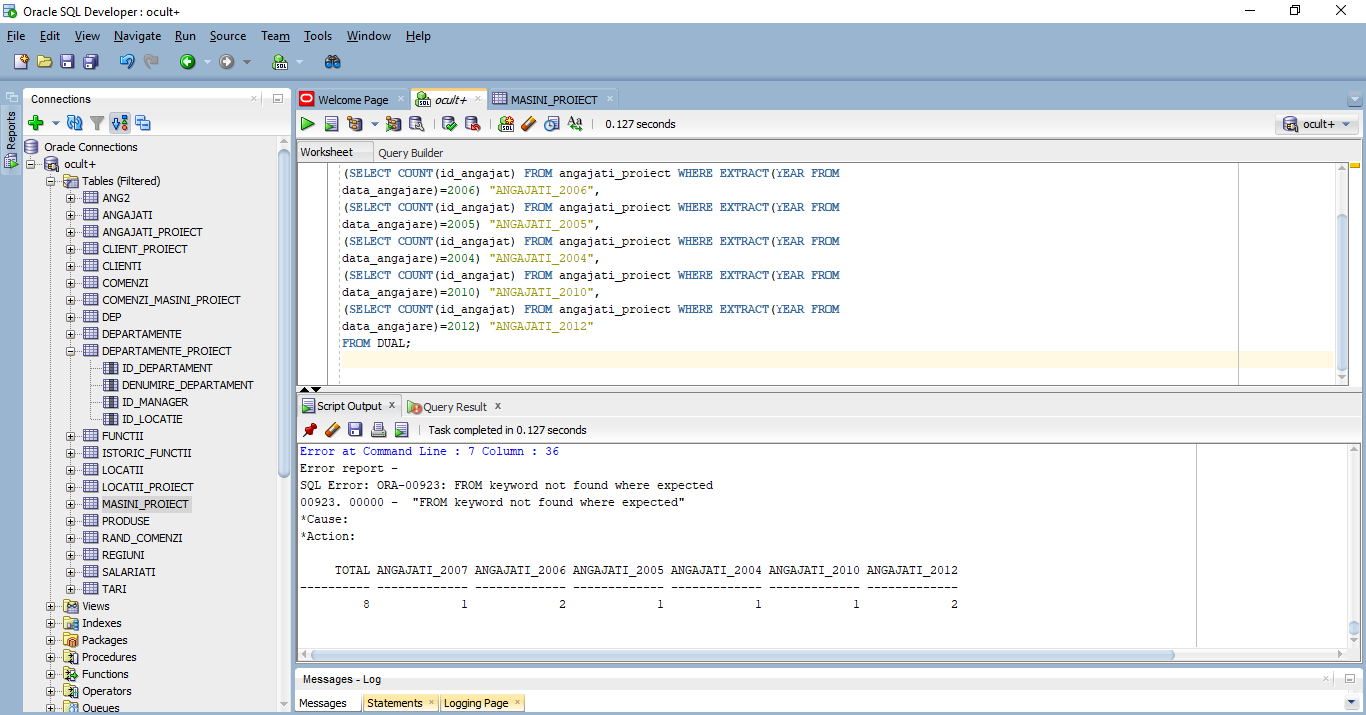
(SELECT COUNT(id\_angajat) FROM angajati\_proiect WHERE EXTRACT(YEAR FROM

data\_angajare)=2010) "ANGAJATI\_2010",

(SELECT COUNT(id\_angajat) FROM angajati\_proiect WHERE EXTRACT(YEAR FROM

data\_angajare)=2012) "ANGAJATI\_2012"

FROM DUAL;



4.01

--1. Sa se realizeze o tabela virtuala cu angajatii departamentului 1 si sa li se aplice o marime de salariu cu 100 de unitati

CREATE OR REPLACE VIEW v\_angajati\_1

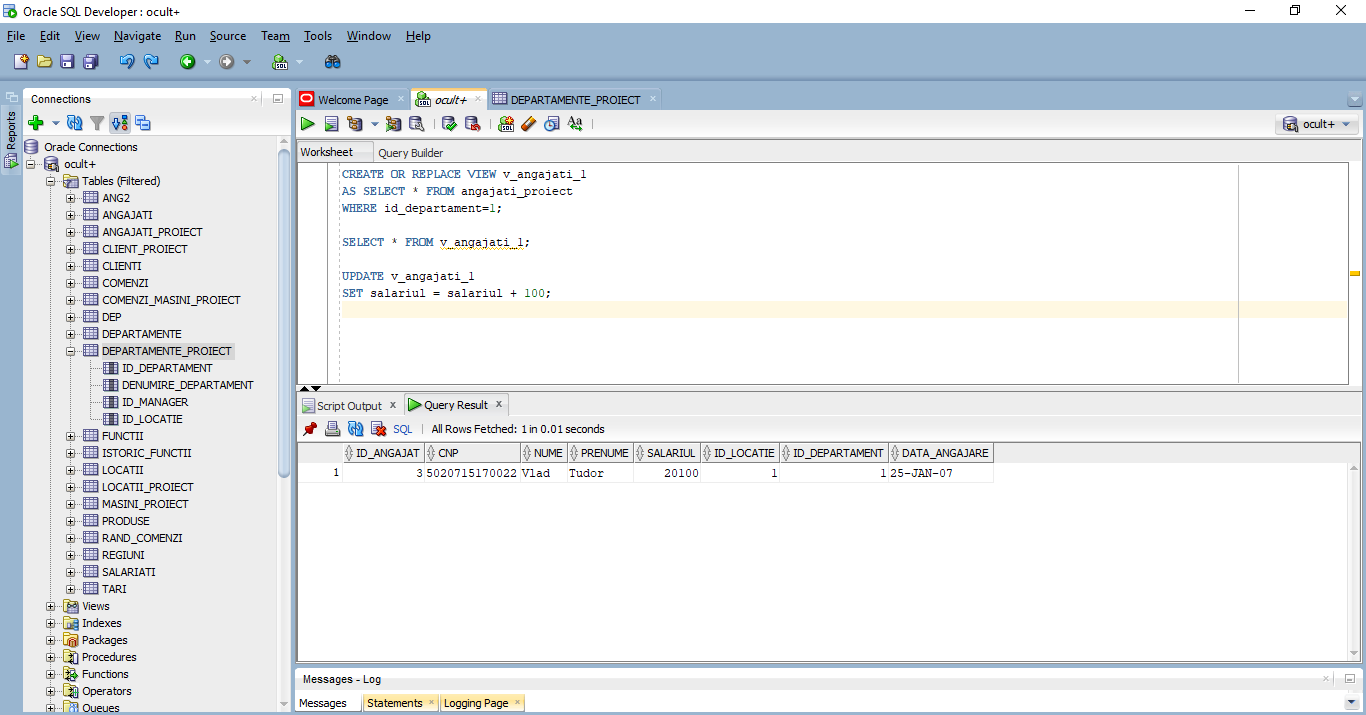
AS SELECT \* FROM angajati\_proiect

WHERE id\_departament=1;

SELECT \* FROM v\_angajati\_1;

UPDATE v\_angajati\_1

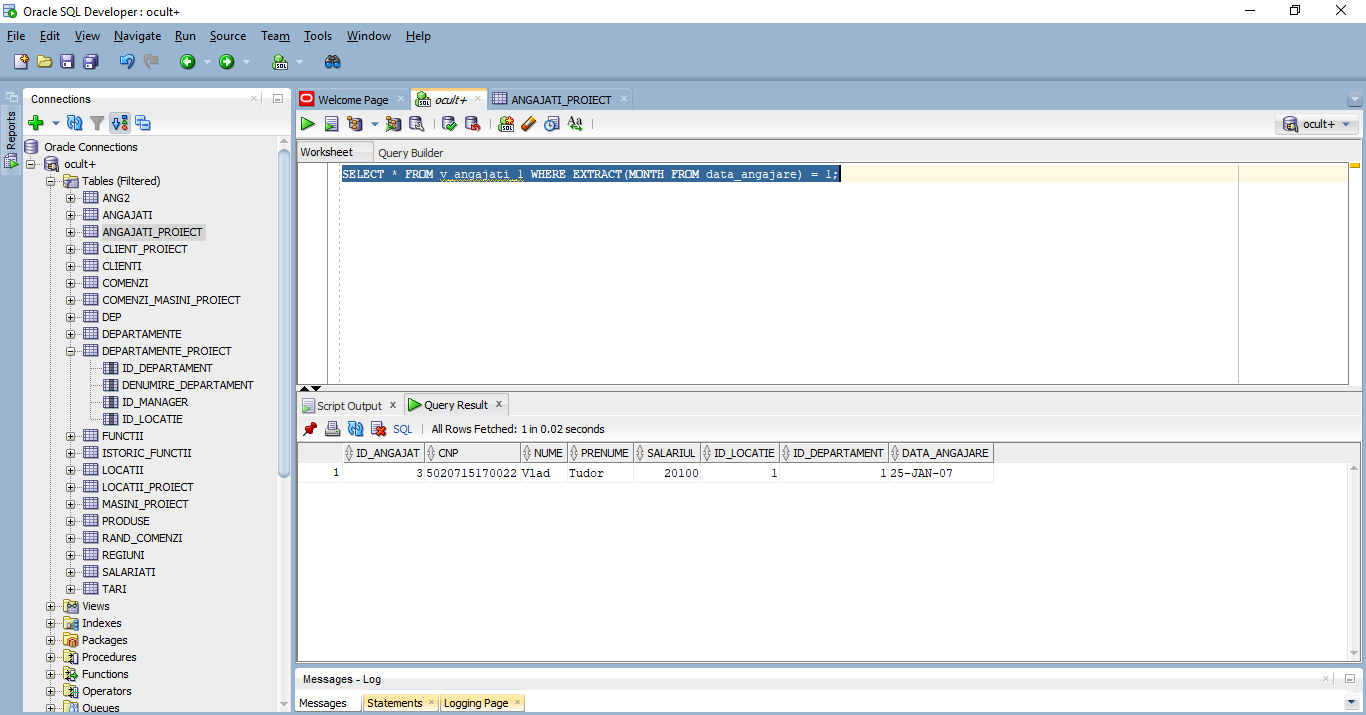
SET salariul = salariul + 100;



--2 . Sa se realizeze o tabela virtuala cu angajatii a caror data de angajare este ianuarie.

CREATE OR REPLACE VIEW v\_angajati\_1

AS SELECT \* FROM v\_angajati\_1 WHERE EXTRACT(MONTH FROM data\_angajare) = 1;



-- 3. . Sa se realizeze o tabela virtuala cu personalul ce a fost angajat in 2005 si sa li se creasca salariul acestora cu 1000 de unitati.

CREATE OR REPLACE VIEW v\_angajati

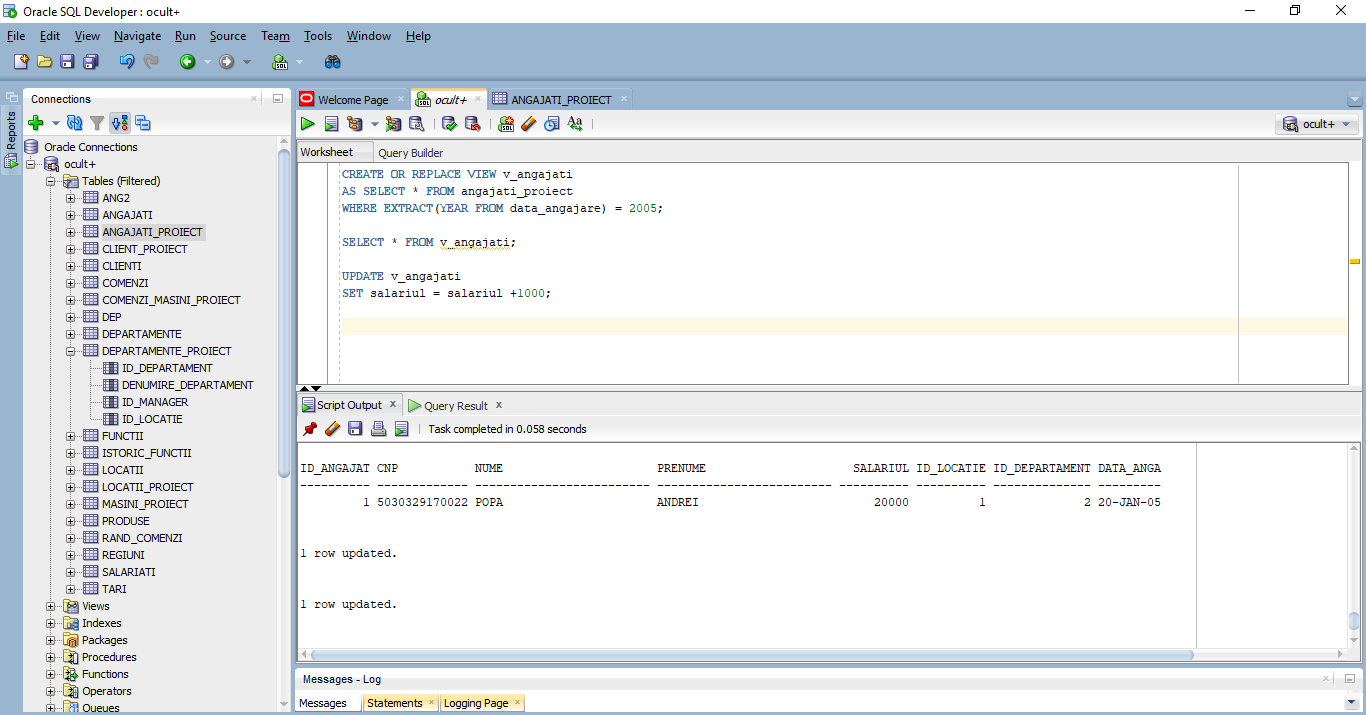
AS SELECT \* FROM angajati\_proiect

WHERE EXTRACT(YEAR FROM data\_angajare) = 2005;

SELECT \* FROM v\_angajati;

UPDATE v\_angajati

SET salariul = salariul +1000;



--4 Sa se realizeze o tabela virtuala cu angajatii din locatia 4 si sa li se micsoreze salariul cu 200 de unitati.

CREATE OR REPLACE VIEW v\_angajati

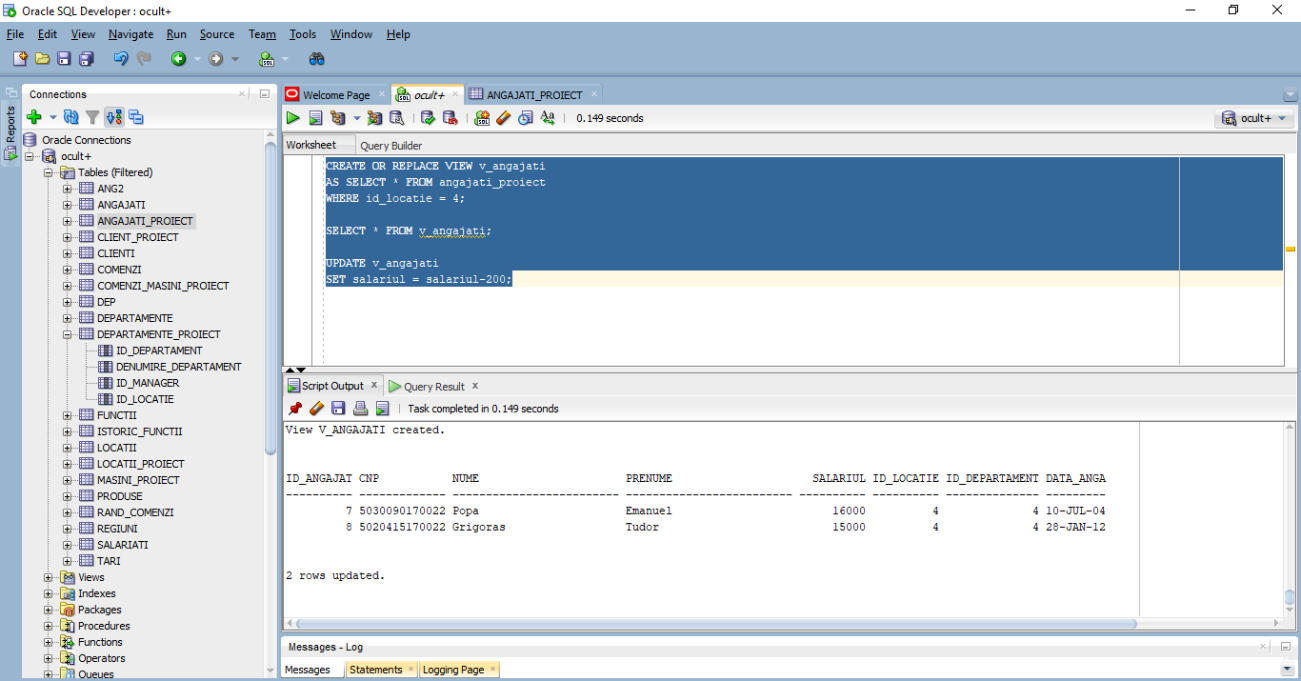
AS SELECT \* FROM angajati\_proiect

WHERE id\_locatie = 4;

SELECT \* FROM v\_angajati;

UPDATE v\_angajati

SET salariul = salariul-200;



--5. Sa se realizeze o tabela virtuala cu datele din comenzi\_masini\_proiect cu WITH READ ONLY

CREATE OR REPLACE VIEW v\_comenzi\_proiect

AS SELECT \* FROM comenzi\_masini\_proiect

WITH READ ONLY;

--6. Sa se realizeze o tabela virtuala cu masinile in care pretul creste cu 1000 de unitati

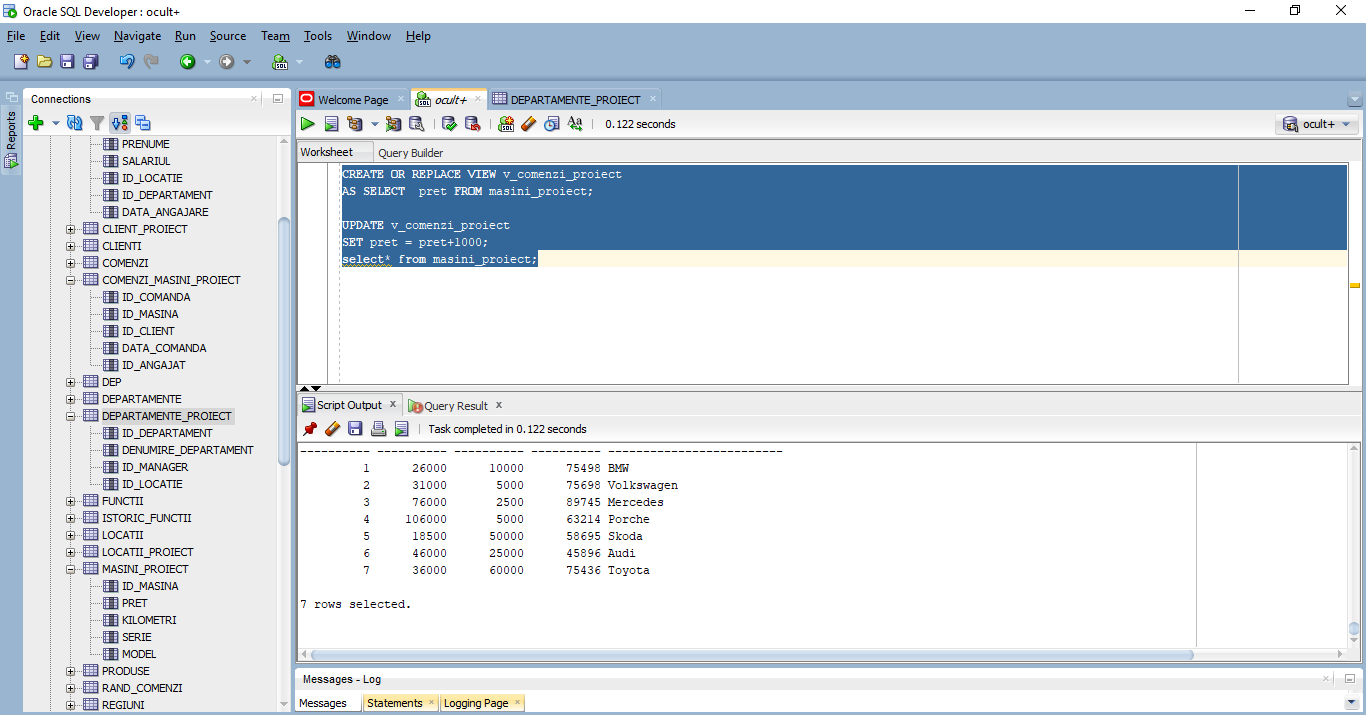
CREATE OR REPLACE VIEW v\_comenzi\_proiect

AS SELECT pret FROM masini\_proiect;

UPDATE v\_comenzi\_proiect

SET pret = pret+1000;

select\* from masini\_proiect;



--7. Sa se realizeze o tabela virtuala cu masinile in care actualizam kilometri

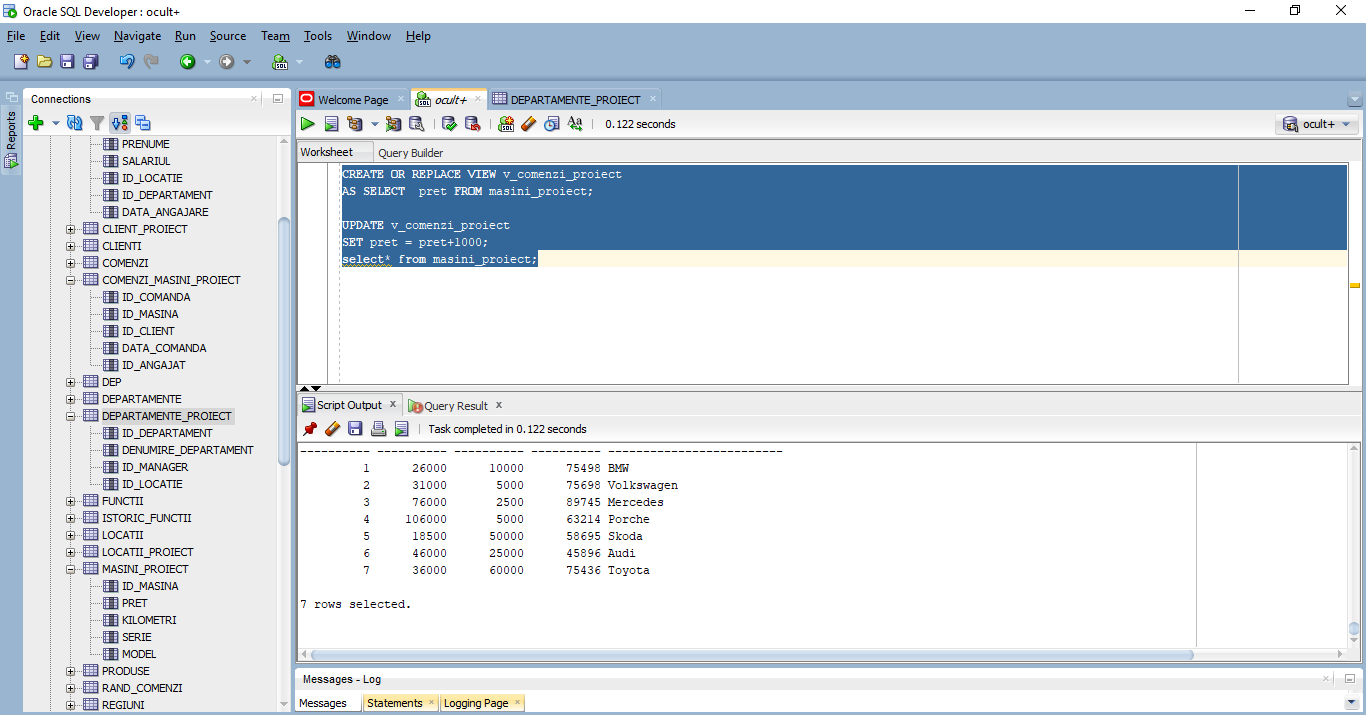
CREATE OR REPLACE VIEW v\_comenzi\_proiect

AS SELECT kilometri FROM masini\_proiect;

UPDATE v\_comenzi\_proiect

SET kilometri = kilometri+ 1000;

select\* from masini\_proiect;



--8 Sa se realizeze o tabela virtuala cu masinile unde aplicam o reducere cu 1000 de unitati masinilor cu peste 50000 de kilometri.

CREATE OR REPLACE VIEW v\_comenzi\_proiect

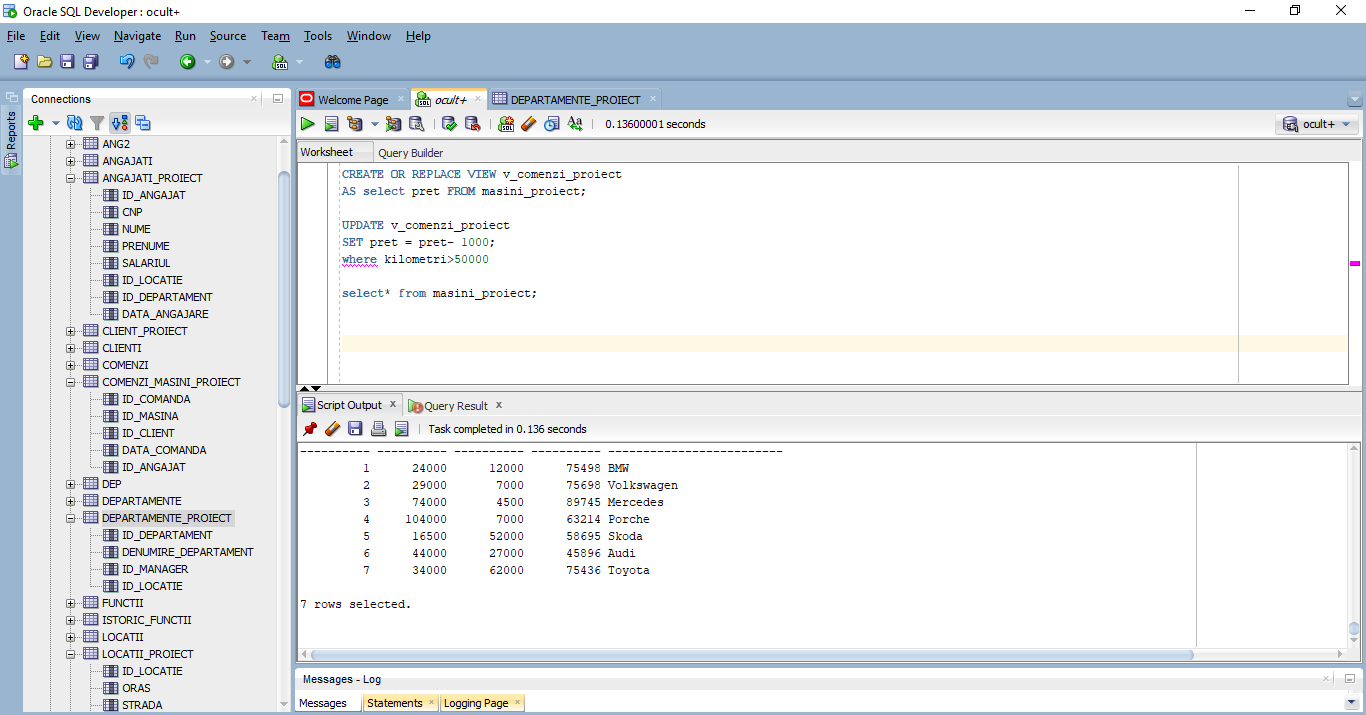
AS select pret FROM masini\_proiect;

UPDATE v\_comenzi\_proiect

SET pret = pret- 1000;

where kilometri>50000

select\* from masini\_proiect;

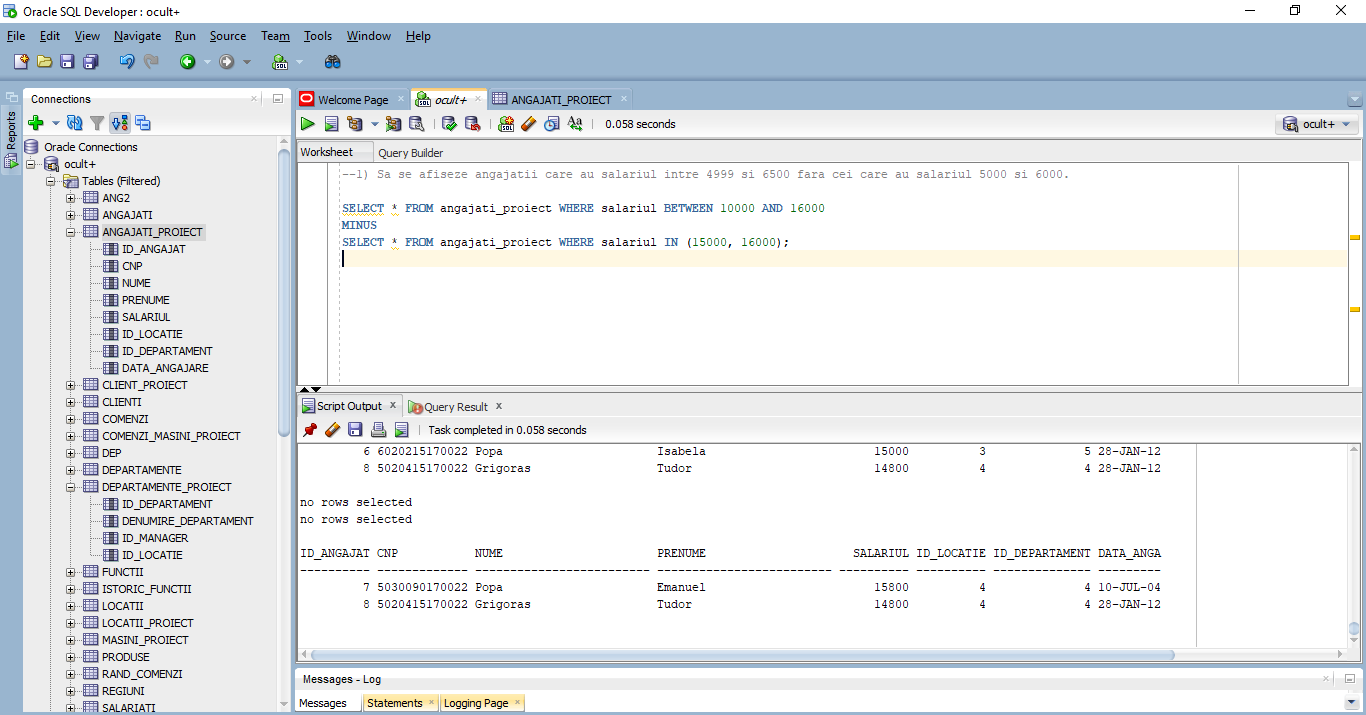


--1) Sa se afiseze angajatii care au salariul intre 4999 si 6500 fara cei care au salariul 5000 si 6000.

SELECT \* FROM angajati\_proiect WHERE salariul BETWEEN 10000 AND 16000

MINUS

SELECT \* FROM angajati\_proiect WHERE salariul IN (15000, 16000);



--Afisati salariul angajatilor si clientilor folosind UNION

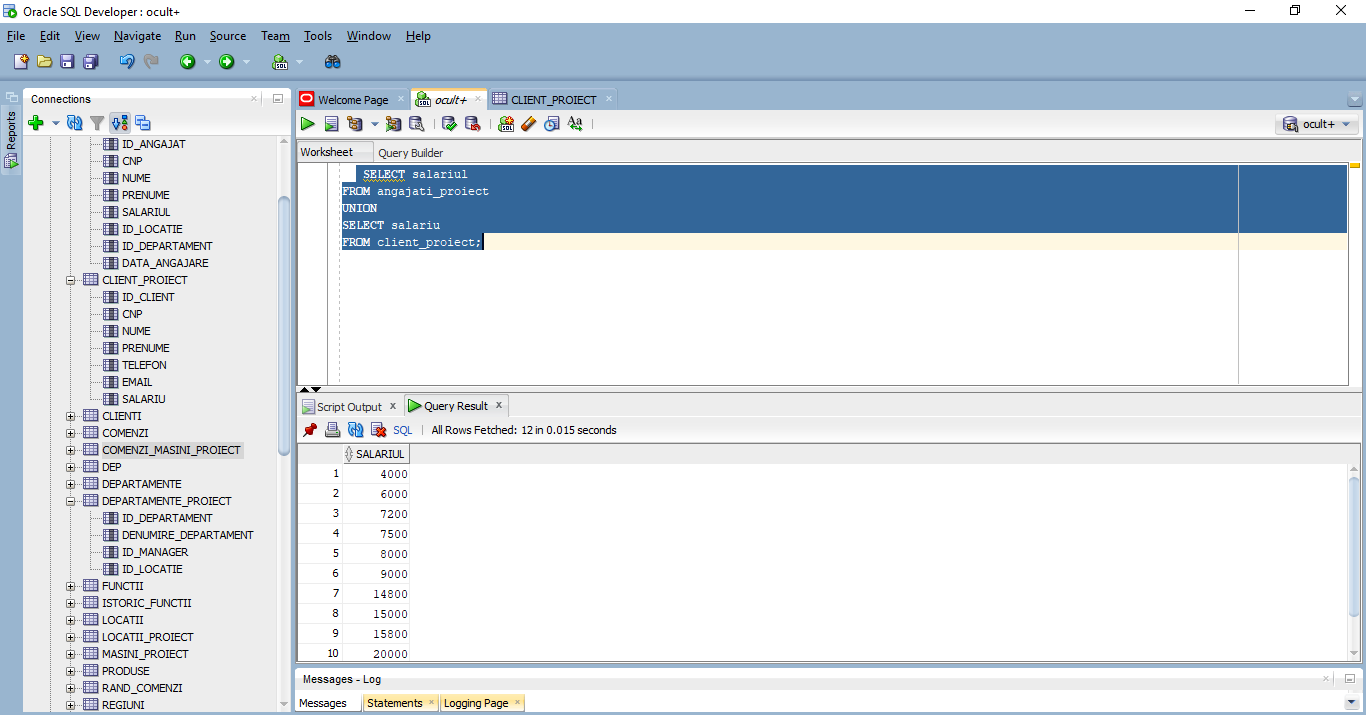
SELECT salariul

FROM angajati\_proiect

UNION

SELECT salariu

FROM client\_proiect;



--Afisati id urile angajatilor si id urile clientilor folosind interselect

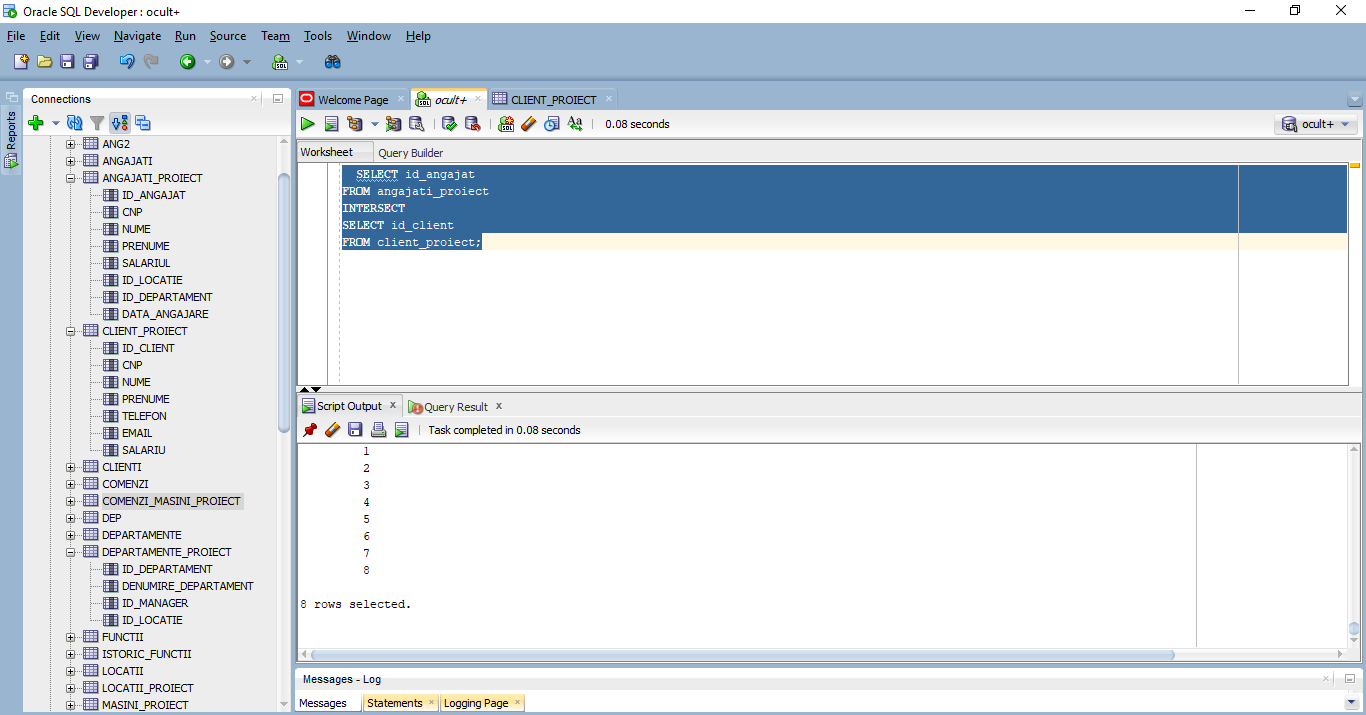
SELECT id\_angajat

FROM angajati\_proiect

INTERSECT

SELECT id\_client

FROM client\_proiect;



--Sa se afiseze angajatii eligibili pt bonus

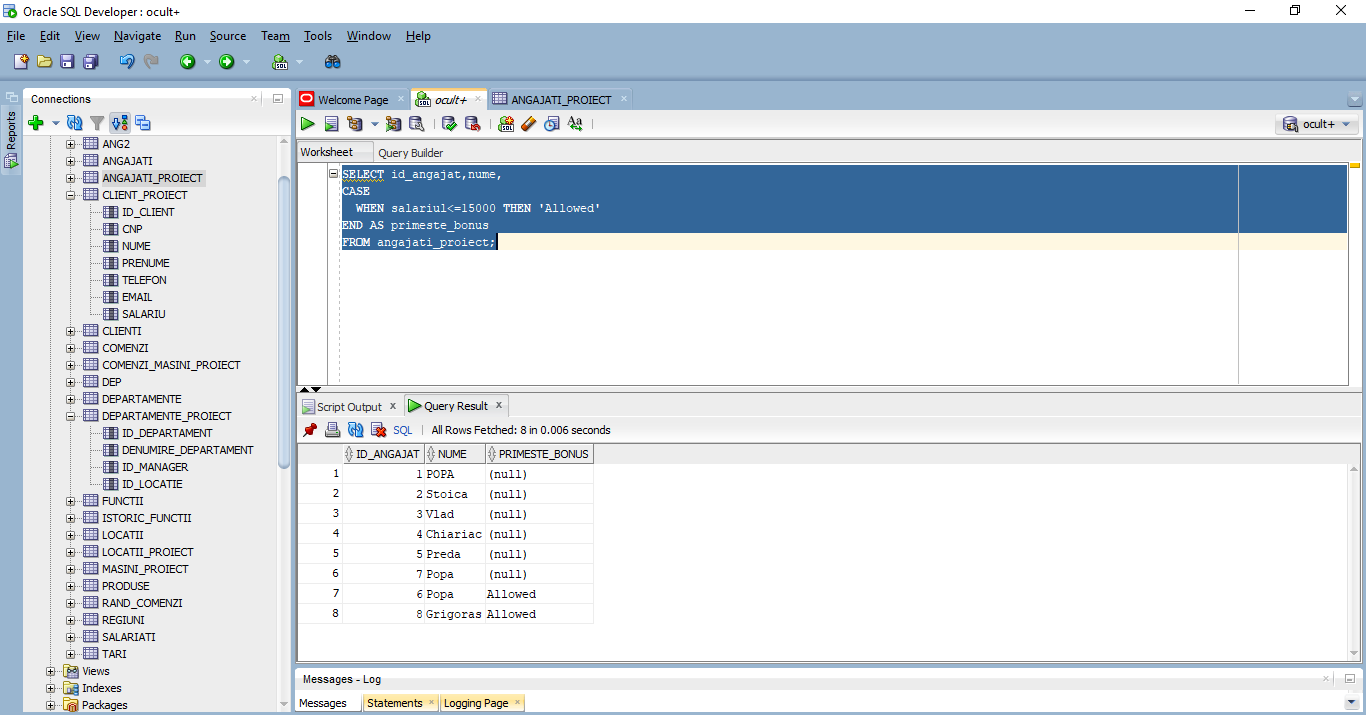
SELECT id\_angajat,nume,

CASE

WHEN salariul<=15000 THEN 'Allowed'

END AS primeste\_bonus

FROM angajati\_proiect;



Continuare proiect(SGBD)

I

* structuri de control
* cursori impliciți
* cursori expliciți

1. Sa se afiseze pretul masinii cu id-ul introdus de la tastatura sis a I se mareasca pretul dupa urmatoarele indicatii: se dubleza daca pretul este sub 15000, se inmulteste cu 1.5 daca este intre 15000 si 30000 si se inmulteste cu 1.25 daca este peste 30000.

SET SERVEROUTPUT ON

DECLARE

v\_lista masini\_proiect.pret%type;

BEGIN

SELECT pret into v\_lista from masini\_proiect where id\_masina=&p;

dbms\_output.put\_line ('Pretul de lista initial este: '||v\_lista);

IF v\_lista < 15000 THEN

v\_lista:=2\* v\_lista;

ELSIF v\_lista between 15000 and 30000 THEN

v\_lista:=1.5 \* v\_lista;

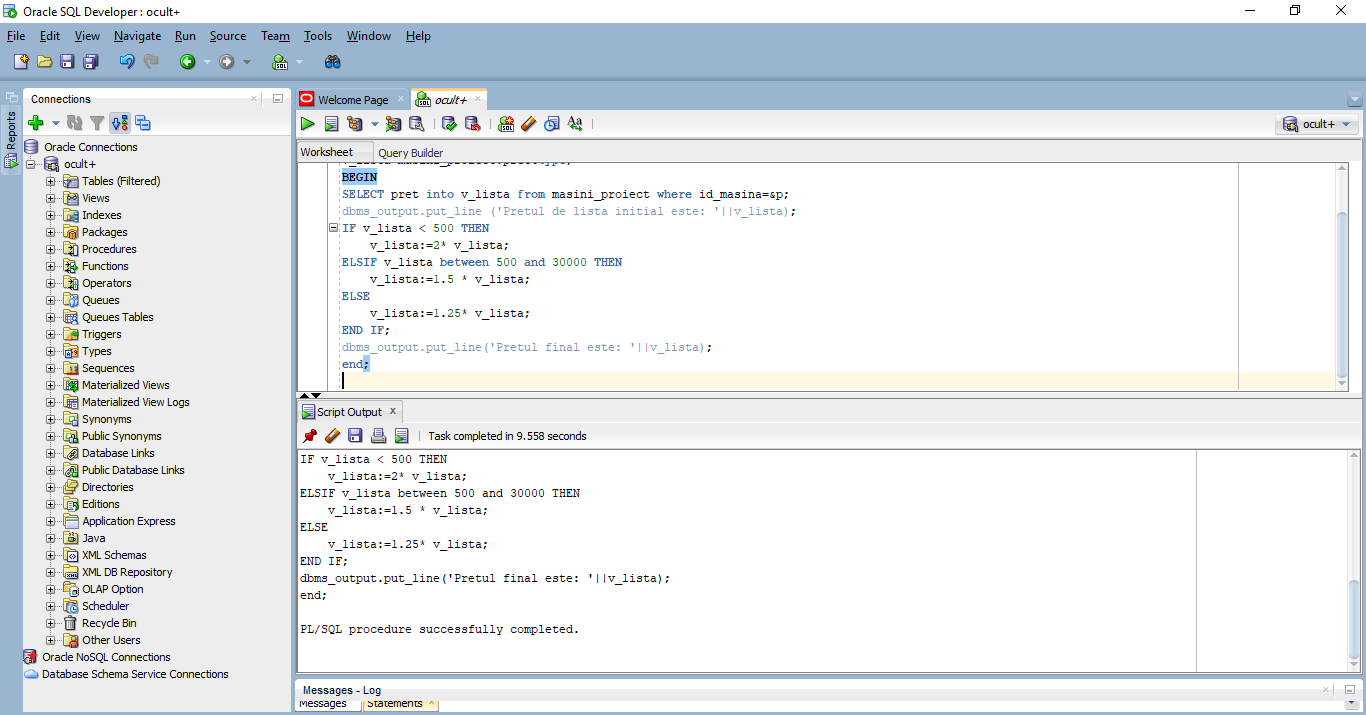
ELSE

v\_lista:=1.25\* v\_lista;

END IF;

dbms\_output.put\_line('Pretul final este: '||v\_lista);

end;



2.Sa se afiseze salariul mediu si toti angajatii consecutivi incepand cu primul angajat cu salariul mai mare decat cel mediu.

set serveroutput on;

DECLARE

v\_sal angajati\_proiect.salariul%type;

v\_salMediu v\_sal%type;

i number(4):=1;

BEGIN

SELECT avg(salariul) into v\_salmediu from angajati\_proiect;

dbms\_output.put\_line('Salariul mediu este: '||v\_salmediu);

loop

select salariul into v\_sal from angajati\_proiect where id\_angajat=i;

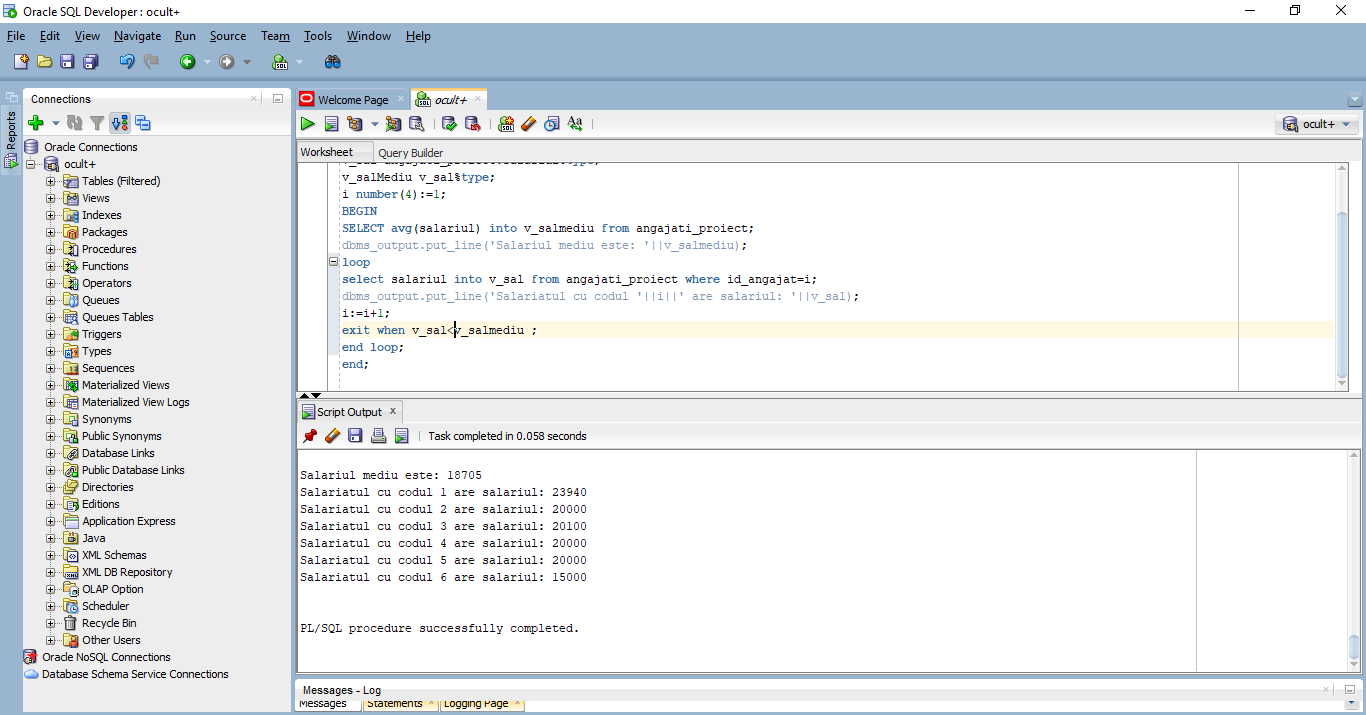
dbms\_output.put\_line('Salariatul cu codul '||i||' are salariul: '||v\_sal);

i:=i+1;

exit when v\_sal<v\_salmediu ;

end loop;

end;



3.

Sa se inlocuiasca numele masinii cu id-ul 3 si sa afiseze un mesaj in caz ca acesta nu exista.

BEGIN

UPDATE masini\_proiect

SET model='Tesla'

WHERE id\_masina=3;

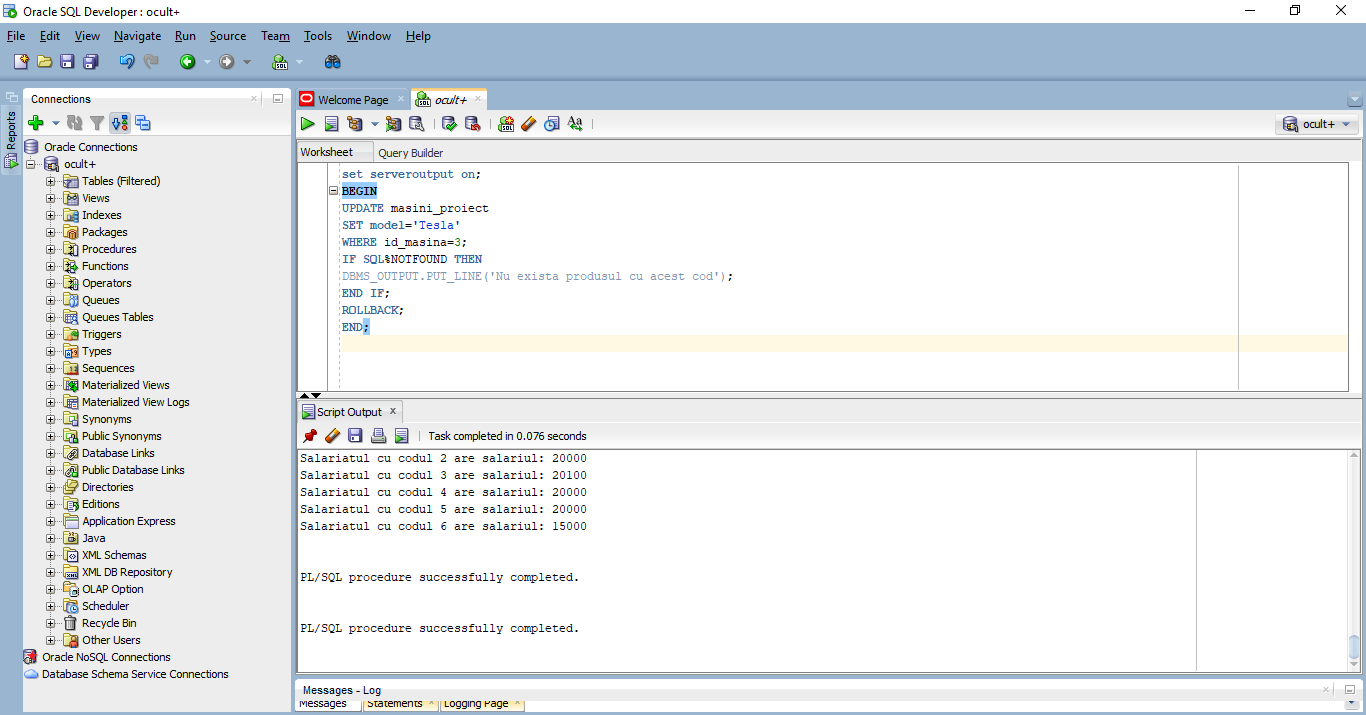
IF SQL%NOTFOUND THEN

DBMS\_OUTPUT.PUT\_LINE('Nu exista produsul cu acest cod');

END IF;

ROLLBACK;

END;



4. Sa se implementeze un cursor care afiseaza o lista cu angajatii departamentului cu id-ul 3.

set serveroutput on

DECLARE

cursor ang\_cursor is select id\_angajat, nume, salariul from angajati\_proiect where id\_departament=3;

ang\_id angajati\_proiect.id\_angajat%type;

ang\_nume angajati\_proiect.nume%type;

ang\_sal angajati\_proiect.salariul%type;

BEGIN

dbms\_output.put\_line('Lista cu salariariile angajatilor din departamentul 2');

open ang\_cursor;

loop

fetch ang\_cursor into ang\_id, ang\_nume, ang\_sal;

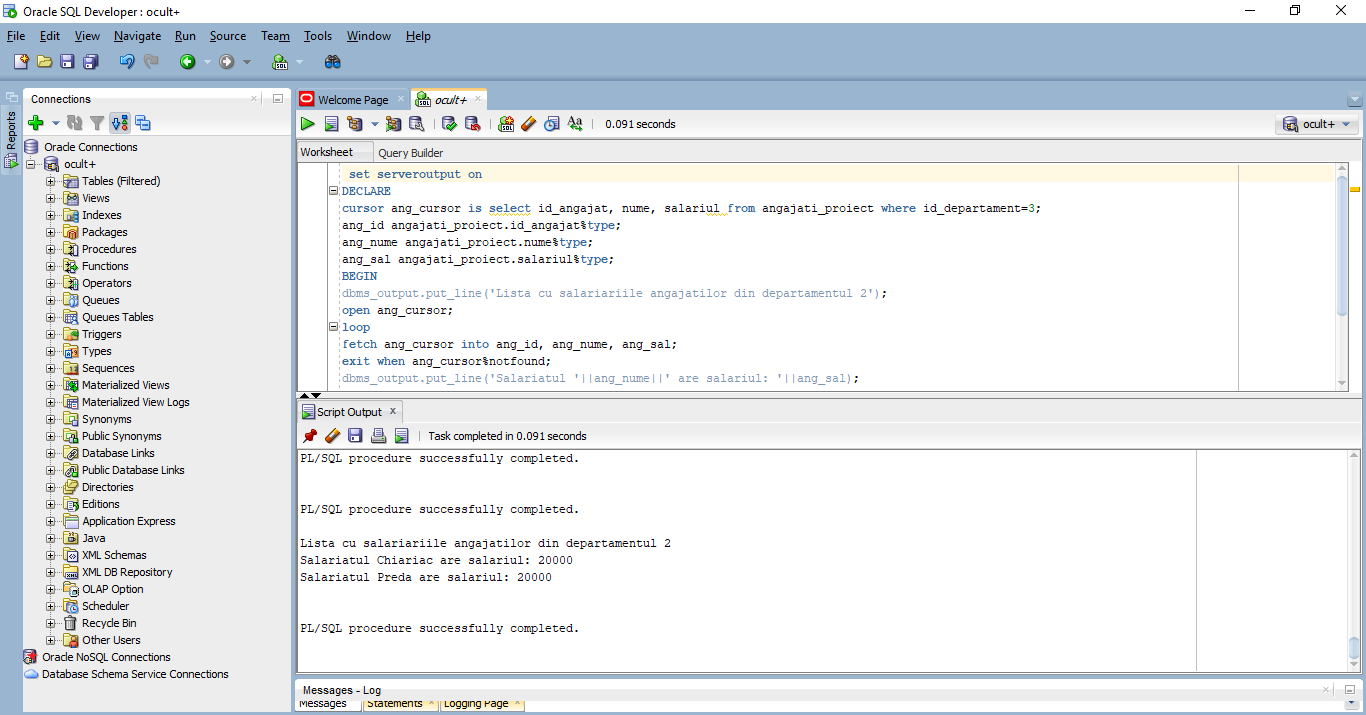
exit when ang\_cursor%notfound;

dbms\_output.put\_line('Salariatul '||ang\_nume||' are salariul: '||ang\_sal);

end loop;

close ang\_cursor;

end;



5. Sa se afiseze salariile angajatilor cu id\_ul intre 1 si 7 atat timp cat salariul acestora este mai mare decat salariul mediu.

DECLARE

v\_sal angajati\_proiect.salariul%type;

v\_salMediu v\_sal%type;

-- i nu mai trebuie declarat

BEGIN

SELECT avg(salariul) into v\_salmediu from angajati\_proiect;

dbms\_output.put\_line('Salariul mediu este: '||v\_salmediu);

for i in 1..6 loop

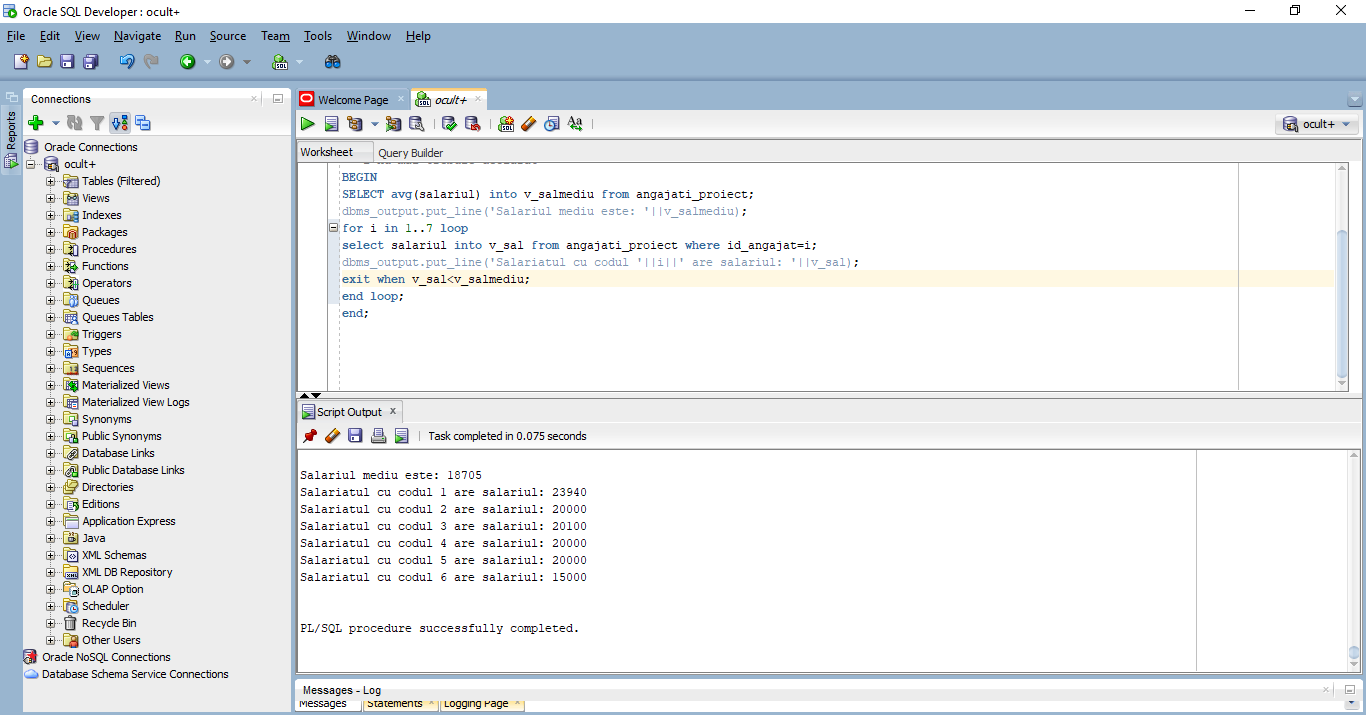
select salariul into v\_sal from angajati\_proiect where id\_angajat=i;

dbms\_output.put\_line('Salariatul cu codul '||i||' are salariul: '||v\_sal);

exit when v\_sal<v\_salmediu;

end loop;

end;



6. Se șterge din tabela masini\_proiect regiunea al cărei ID este introdus de utilizator prin intermediul variabilei de substituție *g\_rid*. Mesajul este afișat folosind variabila globală *nr\_sters*.

SET SERVEROUTPUT ON

ACCEPT g\_rid PROMPT 'Introduceti id-ul masinii:'

VARIABLE nr\_sters varchar2(100)

DECLARE

BEGIN

DELETE FROM masini\_proiect WHERE id\_masina=&g\_rid;

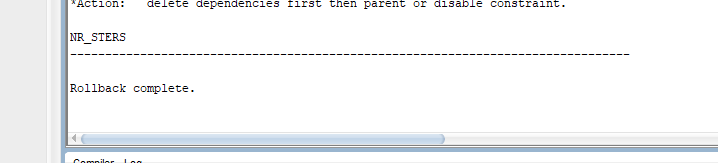
:nr\_sters:=TO\_CHAR(SQL%ROWCOUNT)||' INREGISTRARI STERSE';

END;

/

PRINT nr\_sters

ROLLBACK;



7. Sa se inlocuiasca numele masinii cu id-ul 3 daca are peste 20000km si sa afiseze un mesaj in caz ca acesta nu exista

BEGIN

UPDATE masini\_proiect

SET model='Toyota'

WHERE id\_masina=3 and kilometri>2000;

IF SQL%NOTFOUND THEN

DBMS\_OUTPUT.PUT\_LINE('Nu exista produsul cu acest cod');

END IF;

ROLLBACK;

END;

--8Sa se implementeze un cursor care afiseaza o lista cu angajatii cu primele 3 cele mai mari salarii

set serveroutput on

DECLARE

CURSOR c IS SELECT id\_angajat,salariul

FROM angajati\_proiect

ORDER BY salariul DESC

FETCH FIRST 3 ROWS ONLY;

v1 angajati.id\_angajat%TYPE;

v2 NUMBER;

BEGIN

OPEn c;

LOOP

FETCH c INTO v1, v2;

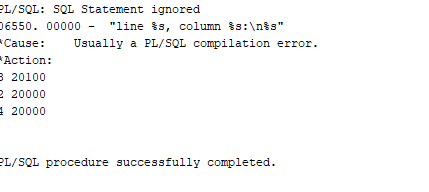
EXIT WHEN c%NOTFOUND;

DBMS\_OUTPUT.PUT\_LINE(v1||' '||v2);

END LOOP;

CLOSE c;

END;



--9Sa se implementeze un cursor care afiseaza o lista cu angajatii cu cei mai vechi 3 angajati

set serveroutput on

DECLARE

CURSOR c IS SELECT id\_angajat,salariul,(sysdate-data\_angajare)/365 as vechime

FROM angajati\_proiect

ORDER BY vechime DESC

FETCH FIRST 3 ROWS ONLY;

v1 angajati.id\_angajat%TYPE;

v2 NUMBER;

v3 number;

BEGIN

OPEn c;

LOOP

FETCH c INTO v1, v2,v3;

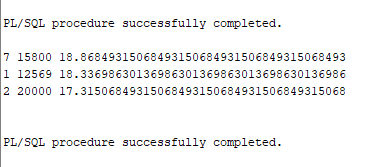
EXIT WHEN c%NOTFOUND;

DBMS\_OUTPUT.PUT\_LINE(v1||' '||v2||' '||v3);

END LOOP;

CLOSE c;

END;



II

Exceptii

* 1. Sa se afiseze in ordine descrescatoare salariile angajatilor. Sa se trateze exceptia in cazul in care cursorul a fost deja deschis.

DECLARE

cursor c is select nume,prenume,salariul from angajati\_proiect order by salariul desc;

BEGIN

open c;

for r in c loop

exit when c%rowcount>5;

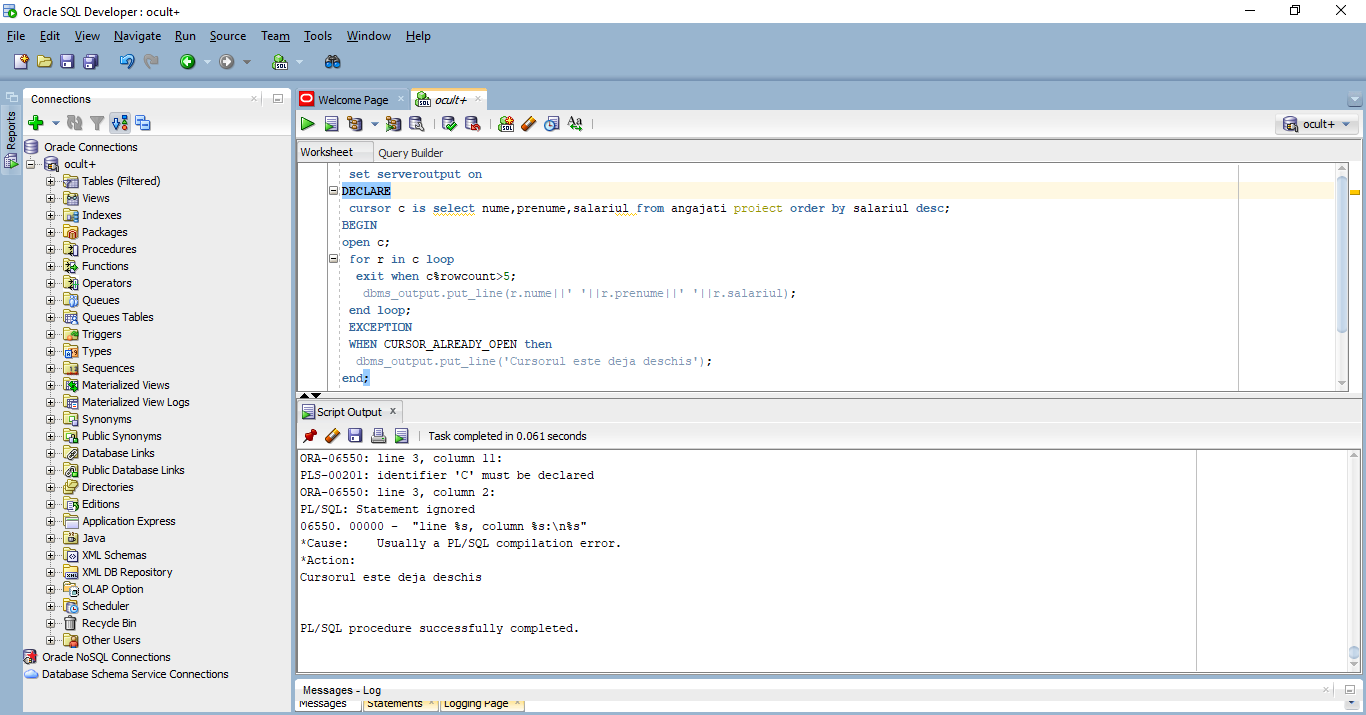
dbms\_output.put\_line(r.nume||' '||r.prenume||' '||r.salariul);

end loop;

EXCEPTION

WHEN CURSOR\_ALREADY\_OPEN then

dbms\_output.put\_line('Cursorul este deja deschis');

end; 

* 1. Sa se afiseze salariul angajatului cu numele Tudor. Sa se trateze exceptia cazului in care exista mai multi angajati Tudor.

SET SERVEROUTPUT ON

DECLARE

sal angajati\_proiect.salariul%type;

BEGIN

select salariul into sal from angajati\_proiect where prenume='Tudor';

DBMS\_OUTPUT.PUT\_LINE('Tudor are salariul de: '||sal);

EXCEPTION

WHEN TOO\_MANY\_ROWS THEN

DBMS\_OUTPUT.PUT\_LINE('Exista mai multi salariati cu numele Tudor! Utilizati un cursor pentru selectie!');

END;



3 Sa se afiseze salariul unui angajat introdus de la tastatura. Sa se trateze exceptia salariului peste medie.

SET SERVEROUTPUT ON

DECLARE

sal angajati\_proiect.salariul%type;

sal\_mediu angajati\_proiect.salariul%type;

sal\_peste\_medie exception;

BEGIN

select avg(salariul) into sal\_mediu from angajati\_proiect;

select salariul into sal from angajati\_proiect where id\_angajat=&id;

DBMS\_OUTPUT.PUT\_LINE('Angajatul are salariul de: '||sal);

if sal>sal\_mediu then raise sal\_peste\_medie;

end if;

EXCEPTION

WHEN sal\_peste\_medie THEN

DBMS\_OUTPUT.PUT\_LINE('Salariul este peste medie!');

END;

4 3 Sa se afiseze salariul unui angajat introdus de la tastatura. Sa se trateze exceptia salariului dublu peste medie.

SET SERVEROUTPUT ON

DECLARE

sal angajati\_proiect.salariul%type;

sal\_mediu angajati\_proiect.salariul%type;

sal\_prea\_mare exception;

BEGIN

select avg(salariul) into sal\_mediu from angajati\_proiect;

select salariul into sal from angajati\_proiect where id\_angajat=&id;

DBMS\_OUTPUT.PUT\_LINE('Angajatul are salariul de: '||sal);

if sal>=2\*sal\_mediu then raise sal\_prea\_mare;

else DBMS\_OUTPUT.PUT\_LINE('Salariul este unul normal');

end if;

EXCEPTION

WHEN sal\_prea\_mare THEN

DBMS\_OUTPUT.PUT\_LINE('Salariul este prea mare!');

END;



III Functii si Proceduri

1. Sa se creeze o procedura care modifica salariul unui angajat cu id\_ul si procentul primate ca parametri

Set serveroutput on

CREATE OR REPLACE

PROCEDURE modifica\_salariul\_procent

(p\_id\_angajat IN angajati\_proiect.id\_angajat%type, procent IN number)

IS

v\_salariul angajati\_proiect.salariul%type;

BEGIN

Select salariul into v\_salariul from angajati\_proiect where id\_angajat=p\_id\_angajat;

dbms\_output.put\_line('Angajatul are salariul de '||v\_salariul);

Update angajati\_proiect

Set salariul=salariul\*(1+procent/100)

Where id\_angajat=p\_id\_angajat;

Select salariul into v\_salariul from angajati\_proiect where id\_angajat=p\_id\_angajat;

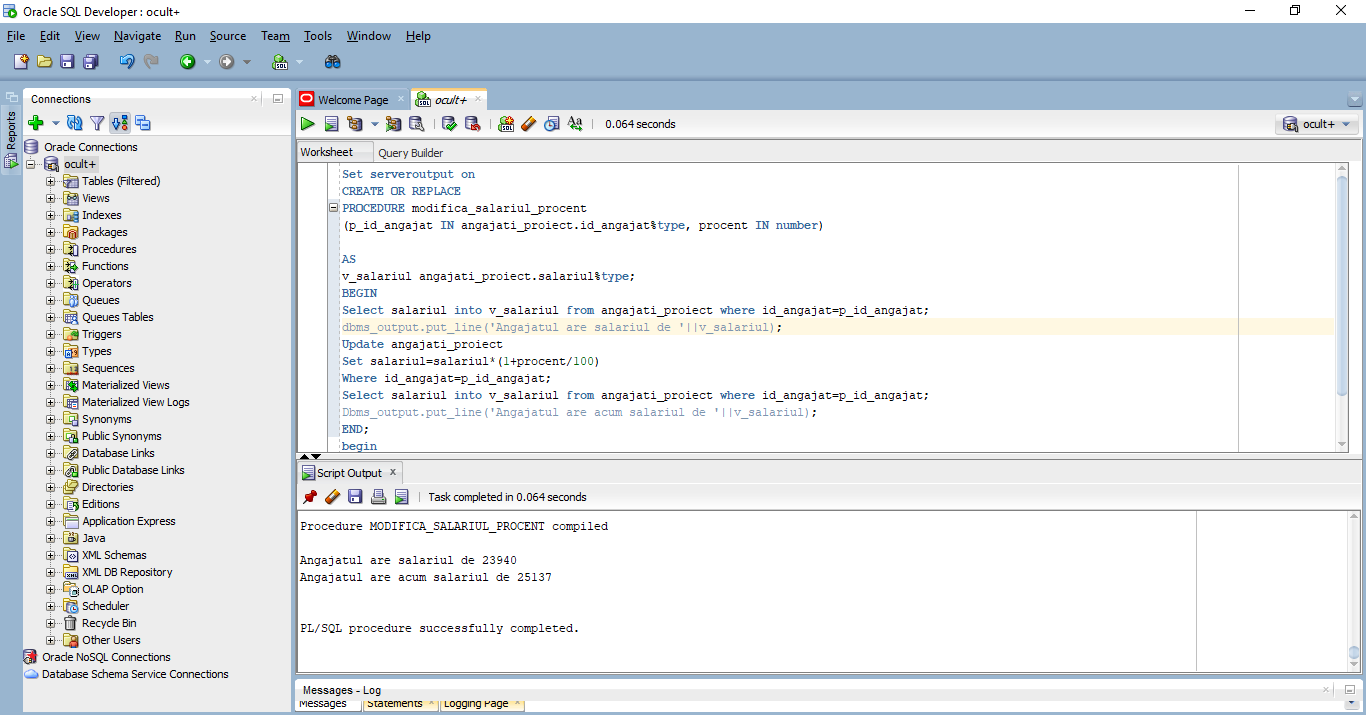
Dbms\_output.put\_line('Angajatul are acum salariul de '||v\_salariul);

END;

begin

modifica\_salariul\_procent(1,5);

end;



1. Sa se creeze o procedura care taie jumatate din salariul angajatului cu id\_ul dat ca parametru si cu conditia ca salariul sa fie mai mare decat marja setata.

Set serveroutput on

Create or Replace procedure taie\_salariul(p\_id in angajati\_proiect.id\_angajat%type,p\_sal in angajati\_proiect.salariul%type) as

begin

update angajati\_proiect

set salariul=salariul/2

where id\_angajat=p\_id and salariul>=p\_sal;

end;

begin

taie\_salariul(1,2000);

end;



3

set serveroutput on;

create or replace PROCEDURE top3\_angajati

AS

CURSOR c IS SELECT id\_angajat,nume,salariul

FROM angajati\_proiect

ORDER BY salariul DESC

FETCH FIRST 3 ROWS ONLY;

v1 angajati\_proiect.id\_angajat%TYPE;

v2 angajati\_proiect.nume%type;

v3 angajati\_proiect.salariul%type;

BEGIN

OPEN c;

LOOP

FETCH c INTO v1,v2,v3;

EXIT WHEN c%NOTFOUND;

DBMS\_OUTPUT.PUT\_LINE(v1||' '||v2||' '||v3);

END LOOP;

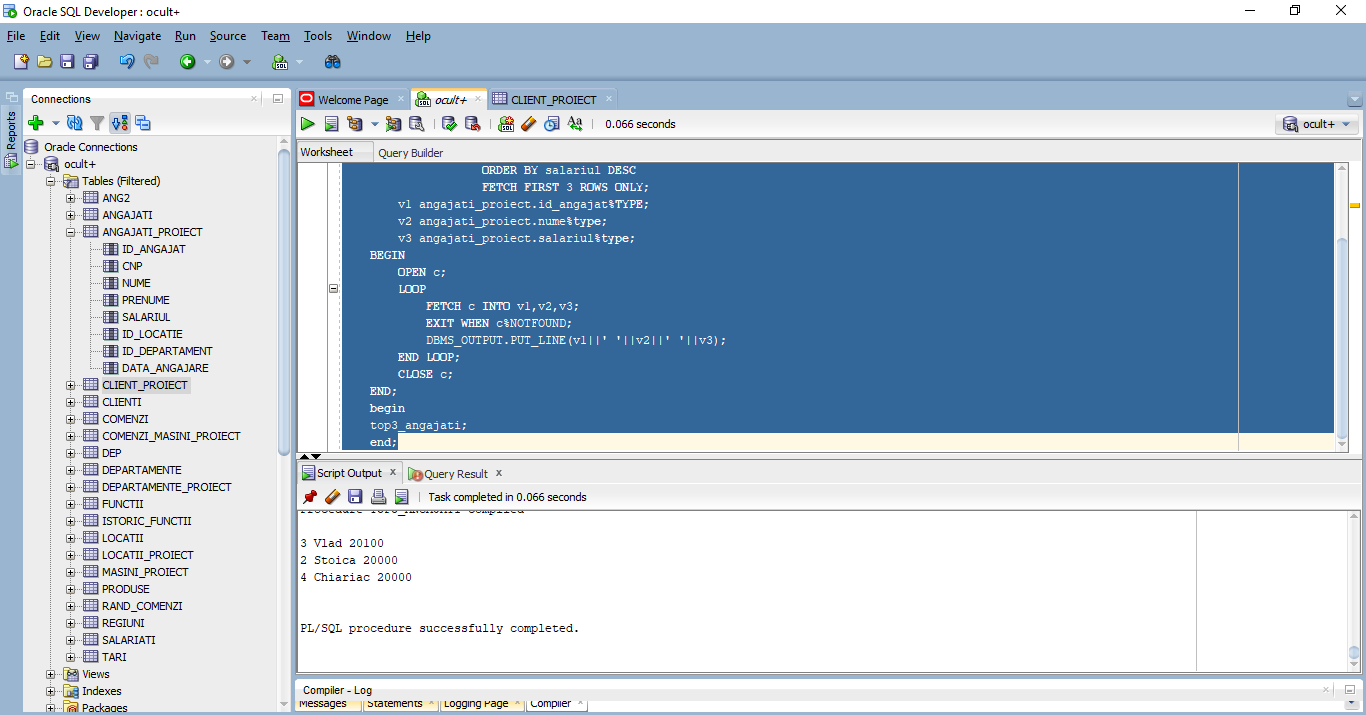
CLOSE c;

END;

begin

top3\_angajati;

end;



1

Sa se construiasca o functie care returneaza true in cazul in care salariul angajatului cu id\_ul dat ca parametru este mai mare decat salariul mediu si false in caz contrar.

set serveroutput on;

CREATE OR REPLACE FUNCTION verifica\_salariul

(p\_id\_angajat IN angajati\_proiect.id\_angajat%type)

RETURN Boolean

IS

v\_salariul angajati\_proiect.salariul%type;

v\_med angajati\_proiect.salariul%type;

BEGIN

SELECT salariul, avg(salariul) into v\_salariul, v\_med from angajati\_proiect where id\_angajat=p\_id\_angajat;

IF v\_salariul > v\_med then

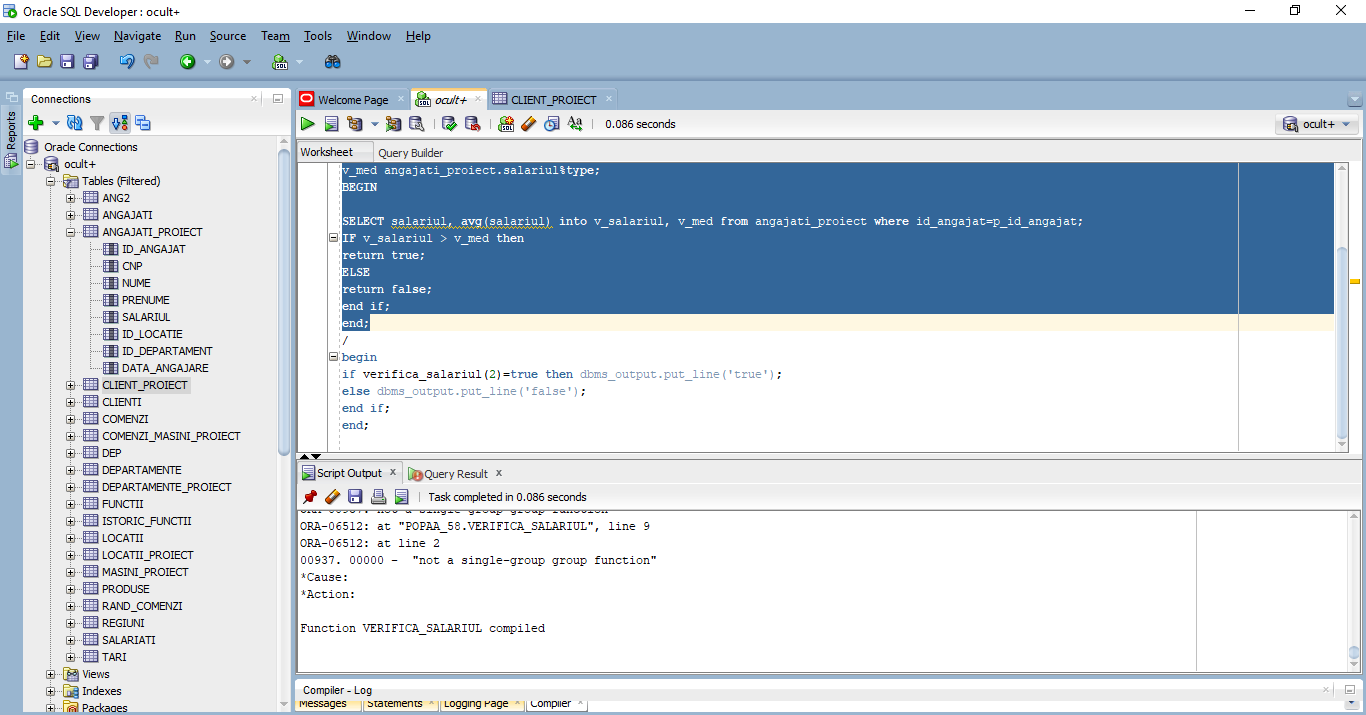
return true;

ELSE

return false;

end if;

end;





2.Sa se creeze functia care returneaza numele angajatului cu salariul cel mai mare

set serveroutput on;

CREATE OR REPLACE FUNCTION angajatul\_valoros

RETURN varchar

as

cursor c is select nume from angajati\_proiect

order by salariul desc

fetch first row only;

v1 angajati\_proiect.nume%type;

BEGIN

OPEN c;

LOOP

FETCH c INTO v1;

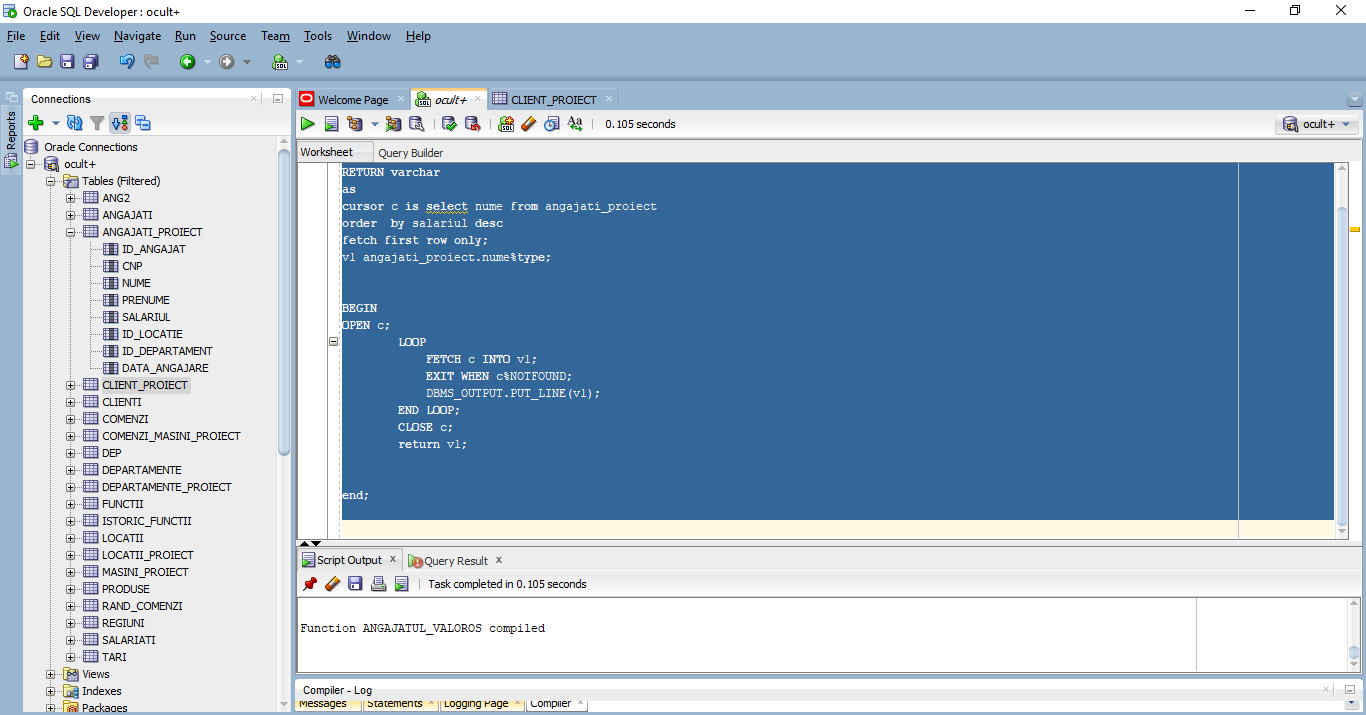
EXIT WHEN c%NOTFOUND;

DBMS\_OUTPUT.PUT\_LINE(v1);

END LOOP;

CLOSE c;

return v1;



end;

3. set serveroutput on;

CREATE OR REPLACE FUNCTION angajatul\_mai\_putin\_valoros

RETURN varchar

as

cursor c is select nume from angajati\_proiect

order by salariul

fetch first row only;

v1 angajati\_proiect.nume%type;

BEGIN

OPEN c;

LOOP

FETCH c INTO v1;

EXIT WHEN c%NOTFOUND;

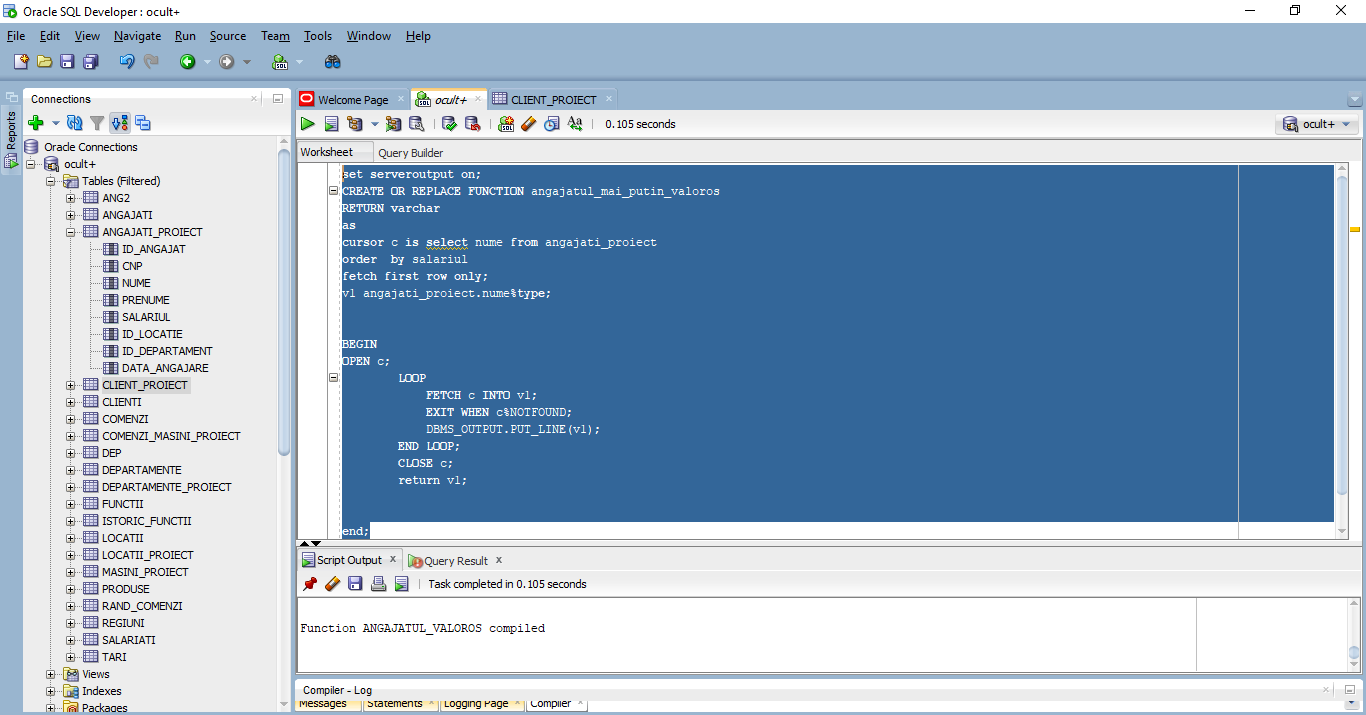
DBMS\_OUTPUT.PUT\_LINE(v1);

END LOOP;

CLOSE c;

return v1;

end;



Pachete

1. - Construiti un pachet care sa contina:

---- o procedura care returneaza numele, pentru angajatul al carui id este dat ca parametru;o procedura care mareste cu 2 salariul angajatului al carui id este dat ca parametru.În ambele proceduri de mai sus, sa se verifice situatia în care angajatul indicat nu exista (invocând o exceptie în acest caz) prin apelul unei functii private, creata în acest scop.

Sa se apeleze procedurile din cadrul pachetului.

Create or replace package pack as

procedure afiseaza(p\_id in angajati\_proiect.id\_angajat%type);

procedure mareste(p\_id in angajati\_proiect.id\_angajat%type);

end;

/create or replace package body pack as

function verifica\_angajat(p\_id in angajati\_proiect.id\_angajat%type) return number as

nr number:=0;

begin

select count(id\_angajat)into nr

from angajati\_proiect

where id\_angajat=p\_id;

return nr;

end;

procedure afiseaza(p\_id in angajati\_proiect.id\_angajat%type)as

v1 angajati\_proiect.nume%type;

v2 number;

nu\_exista exception;

begin

if verifica\_angajat(p\_id)=0 then raise nu\_exista;

else select nume,round(sysdate-data\_angajare) into v1,v2 from angajati\_proiect

where id\_angajat=p\_id;

dbms\_output.put\_line(v1||' '||v2);

end if;

exception

when nu\_exista then return -1;

end;

procedure mareste(p\_id in angajati\_proiect.id\_angajat%type)as

nu exista exception;

begin

if verifica\_angajat(p\_id)=0 then raise nu\_exista;

else update angajati\_proiect

set salariul=salariul+2

where id\_angajat=p\_id;

end if;

exception

when nu\_exista then return -1;

end;

end;

execute pack.mareste(2);



2.Realizati un pachet de subprograme care sa contina:

- o procedura care sa adauge o înregistrare noua în tabela Departamente\_proiect. Informatiile ce trebuie adaugate sunt furnizate drept parametri procedurii. Se trateaza cazul în care exista deja un departament cu codul introdus.

-o procedura care sa modifice denumirea unui departament. IDul departamente pentru care se face modificarea si noua denumire a departamentului sunt parametrii procedurii. Se trateaza cazul în care modificarea nu are loc din cauza precizarii unui cod care nu se regaseste în tabela.

- o procedura care sa stearga un departament pe baza codului primit drept parametru. Se trateaza cazul în care codul furnizat nu exista.

SET SERVEROUTPUT ON

CREATE OR REPLACE PACKAGE pac2

AS

PROCEDURE insert\_departament (p\_id departamente\_proiect.id\_departament%TYPE, p\_den departamente\_proiect.denumire\_departament%type, p\_man departamente\_proiect.id\_manager%TYPE,p\_loc departamente\_proiect.id\_locatie%TYPE);

PROCEDURE modifica\_departament (p\_id departamente\_proiect.id\_departament%TYPE, p\_den departamente\_proiect.denumire\_departament%type);

PROCEDURE delete\_departament (p\_id departamente\_proiect.id\_departament%TYPE);

END;

/

CREATE OR REPLACE PACKAGE BODY pac2

AS

FUNCTION verifica\_existenta (p\_id departamente\_proiect.id\_departament%TYPE)RETURN NUMBER

AS

nr NUMBER;

BEGIN

SELECT COUNT(id\_departament) INTO nr

FROM departamente\_proiect

WHERE id\_departament=p\_id;

RETURN nr;

END;

PROCEDURE insert\_departament (p\_id departamente\_proiect.id\_departament%TYPE, p\_den departamente\_proiect.denumire\_departament%type, p\_man departamente\_proiect.id\_manager%TYPE,p\_loc departamente\_proiect.id\_locatie%TYPE)

AS

BEGIN

IF verifica\_existenta(p\_id)=0 THEN

INSERT INTO departamente\_proeict VALUES (p\_id, p\_den,p\_man, p\_loc);

ELSE

DBMS\_OUTPUT.PUT\_LINE('Deja exista id');

END IF;

END;

PROCEDURE modifica\_departament (p\_id departamente\_proiect.id\_departament%TYPE, p\_den departamente\_proiect.denumire\_departament%type)

AS

BEGIN

IF verifica\_existenta(p\_id)=1 THEN

UPDATE departamente\_proiect

SET denumire\_departament = p\_den

WHERE id\_departament = p\_id;

ELSE

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

END IF;

END;

PROCEDURE delete\_departament (p\_id departamente\_proiect.id\_departament%TYPE)

AS

BEGIN

IF verifica\_existenta(p\_id)=0 THEN

DELETE FROM departamente\_proiect

WHERE id\_departament=p\_id;

ELSE

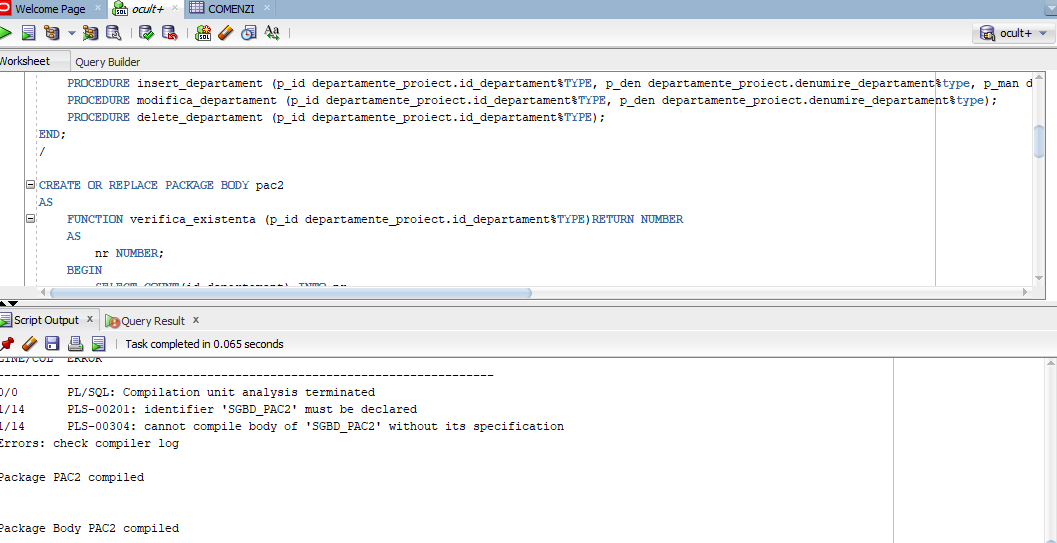
DBMS\_OUTPUT.PUT\_LINE('Deja exista id');

END IF;

END;

END;

/

EXECUTE pac2.modifica\_departament('2','Relatii Clienti');

Triggeri

1. Sa se creeze un declanşator, trg\_stop\_marire, care să împiedice mărirea salariului pentru angajații cu vechimea mai mare de 2 de ani. Testați declanșarea trigger-ului.

CREATE OR REPLACE TRIGGER trg\_stop\_marire

BEFORE UPDATE OF SALARIUL ON ANGAJATI\_PROIECT

FOR EACH ROW

BEGIN

IF :NEW.SALARIUL > :OLD.SALARIUL AND (SYSDATE-:OLD.DATA\_ANGAJARE)/365>2 THEN

RAISE\_APPLICATION\_ERROR(-20000, 'Nu se poate mari salariul!');

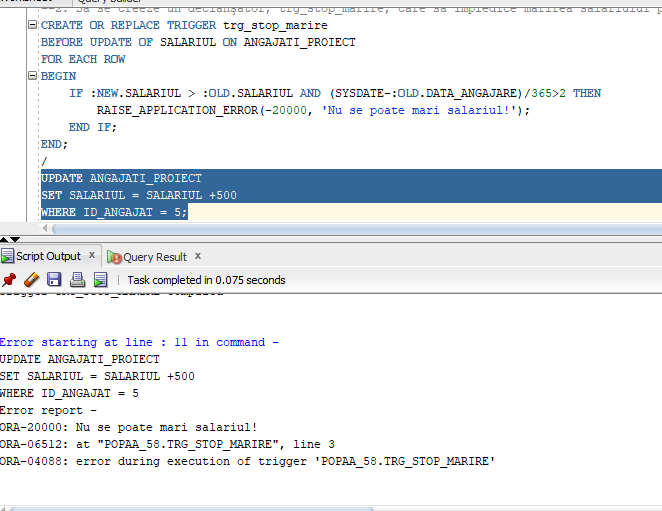
END IF;

END;

/

UPDATE ANGAJATI\_PROIECT

SET SALARIUL = SALARIUL +500

WHERE ID\_ANGAJAT = 5;

--2. Sa se creeze un declanşator, trg\_stop\_micsorare, care să împiedice micsorarea salariului pentru angajații mai noi de 1 an. Testați declanșarea trigger-ului.

CREATE OR REPLACE TRIGGER trg\_stop\_micsorare

BEFORE UPDATE OF SALARIUL ON ANGAJATI\_PROIECT

FOR EACH ROW

BEGIN

IF :NEW.SALARIUL > :OLD.SALARIUL AND (SYSDATE-:OLD.DATA\_ANGAJARE)/365<1 THEN

RAISE\_APPLICATION\_ERROR(-20000, 'Nu se poate micsora salariul!');

END IF;

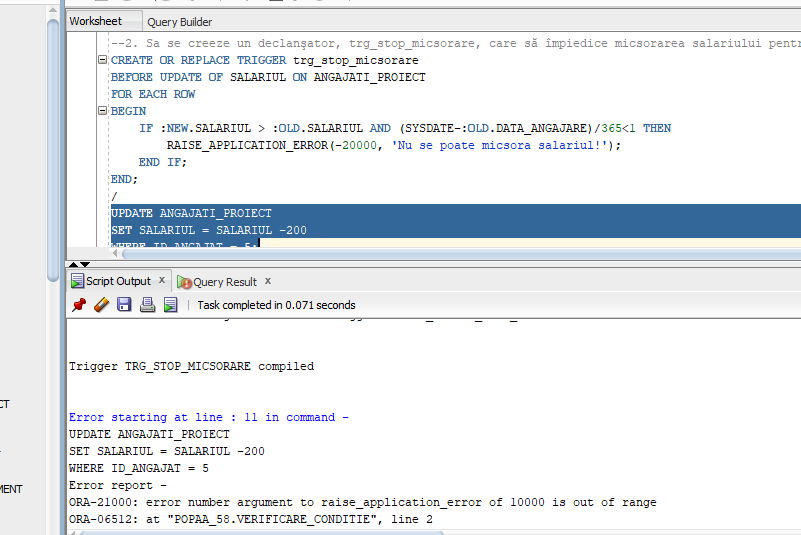
END;

/

UPDATE ANGAJATI\_PROIECT

SET SALARIUL = SALARIUL -200

WHERE ID\_ANGAJAT = 5;



--3. Sa se creeze un declanşator, trg\_stop\_dublare, care să împiedice dublarea salariului pentru angajații din departamente cu id\_ul peste5. Testați declanșarea trigger-ului.

CREATE OR REPLACE TRIGGER trg\_stop\_dublare

BEFORE UPDATE OF SALARIUL ON ANGAJATI\_PROIECT

FOR EACH ROW

BEGIN

IF :NEW.SALARIUL > :OLD.SALARIUL AND :OLD.id\_departament>5 THEN

RAISE\_APPLICATION\_ERROR(-20000, 'Nu se poate dubla salariul!');

END IF;

END;

/

UPDATE ANGAJATI\_PROIECT

SET SALARIUL = SALARIUL \*2

WHERE ID\_ANGAJAT = 6;

