Proiect SGBD

-Șerban Andrei, 242-

Proiectul ales de mine este o bază de date pentru o firmă de agricultură, proiectat pentru a eficientiza operațiunile afacerilor agricole moderne. Acesta pune împreună datele legate de managementul departamentelor, activitățile angajaților, interacțiuni cu clienți, procesare comenzi, inventarul produselor si echipamentelor agricole. Baza de date oferă perspective in timp real asupra fiecărui aspect al operațiunilor firmei, ajutând la procesul decizional, eficienta operațională si satisfacția clienților.

Următoarele diagrame prezinta funcționalitatea bazei de date.

Diagrama entitate-relație:

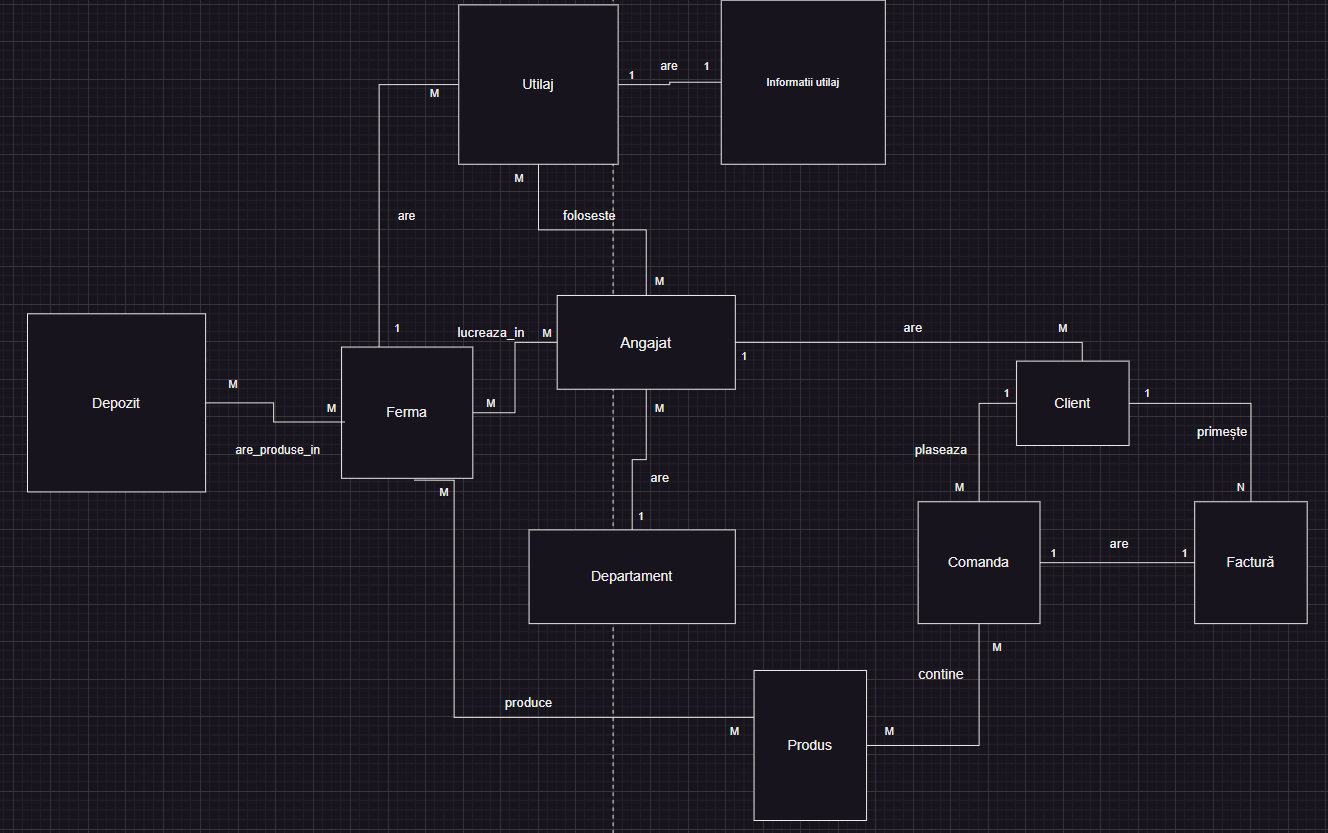
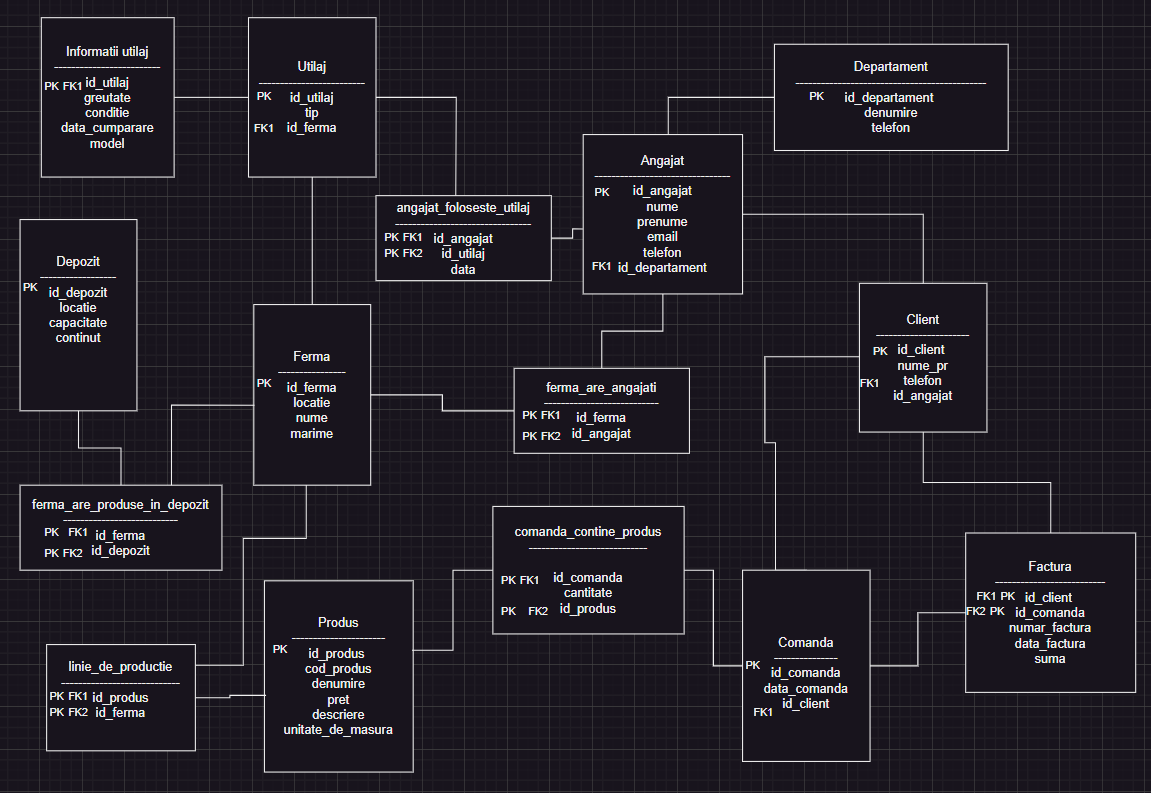


Diagrama conceptuală:



Codul folosit pentru crearea bazei de date:

CREATE TABLE Departament (

id\_departament NUMBER PRIMARY KEY,

denumire VARCHAR2(100) NOT NULL,

telefon VARCHAR2(15)

);

CREATE TABLE Angajat (

id\_angajat NUMBER PRIMARY KEY,

nume VARCHAR2(50) NOT NULL,

prenume VARCHAR2(50) NOT NULL,

email VARCHAR2(100),

telefon VARCHAR2(15),

id\_departament NUMBER,

CONSTRAINT fk\_angajat\_departament FOREIGN KEY (id\_departament)

REFERENCES Departament (id\_departament)

);

CREATE TABLE Client (

id\_client NUMBER PRIMARY KEY,

nume\_pr VARCHAR2(100) NOT NULL,

telefon VARCHAR2(15),

id\_angajat NUMBER,

CONSTRAINT fk\_client\_angajat FOREIGN KEY (id\_angajat)

REFERENCES Angajat (id\_angajat)

);

CREATE TABLE Comanda (

id\_comanda NUMBER PRIMARY KEY,

cantitate NUMBER NOT NULL,

data\_comanda DATE NOT NULL,

id\_client NUMBER,

CONSTRAINT fk\_comanda\_client FOREIGN KEY (id\_client)

REFERENCES Client (id\_client)

);

CREATE TABLE Factura (

id\_comanda NUMBER ,

id\_client NUMBER ,

PRIMARY KEY (id\_comanda,id\_client),

numar\_factura VARCHAR2(50) NOT NULL,

data\_factura DATE NOT NULL,

suma NUMBER NOT NULL,

CONSTRAINT fk\_factura\_client FOREIGN KEY (id\_client)

REFERENCES Client (id\_client),

CONSTRAINT fk\_factura\_client\_2 FOREIGN KEY (id\_comanda)

REFERENCES Comanda (id\_comanda)

);

CREATE TABLE Produs (

id\_produs NUMBER PRIMARY KEY,

cod\_produs VARCHAR2(50) NOT NULL,

denumire VARCHAR2(100) NOT NULL,

pret NUMBER NOT NULL,

descriere VARCHAR2(255),

unitate\_de\_masura VARCHAR2(50)

);

CREATE TABLE Depozit (

id\_depozit NUMBER PRIMARY KEY,

locatie VARCHAR2(100) NOT NULL,

capacitate NUMBER NOT NULL,

continut VARCHAR2(255)

);

CREATE TABLE Ferma (

id\_ferma NUMBER PRIMARY KEY,

locatie VARCHAR2(100) NOT NULL,

nume VARCHAR2(100) NOT NULL,

marime NUMBER NOT NULL

);

CREATE TABLE Utilaj (

id\_utilaj NUMBER PRIMARY KEY,

tip VARCHAR2(100) NOT NULL,

id\_ferma NUMBER,

CONSTRAINT fk\_utilaj\_ferma FOREIGN KEY (id\_ferma)

REFERENCES Ferma (id\_ferma)

);

CREATE TABLE Informatii\_Utilaj (

id\_utilaj NUMBER PRIMARY KEY,

greutate NUMBER NOT NULL,

conditie VARCHAR2(100),

data\_cumparare DATE,

model VARCHAR2(100),

CONSTRAINT fk\_utilaj\_informatii FOREIGN KEY (id\_utilaj)

REFERENCES Utilaj (id\_utilaj)

);

CREATE TABLE Angajat\_Foloseste\_Utilaj (

id\_angajat NUMBER,

id\_utilaj NUMBER,

data DATE NOT NULL,

PRIMARY KEY (id\_angajat, id\_utilaj),

CONSTRAINT fk\_afu\_angajat FOREIGN KEY (id\_angajat)

REFERENCES Angajat (id\_angajat),

CONSTRAINT fk\_afu\_utilaj FOREIGN KEY (id\_utilaj)

REFERENCES Utilaj (id\_utilaj)

);

CREATE TABLE Ferma\_Are\_Angajati (

id\_ferma NUMBER,

id\_angajat NUMBER,

PRIMARY KEY (id\_ferma, id\_angajat),

CONSTRAINT fk\_faa\_ferma FOREIGN KEY (id\_ferma)

REFERENCES Ferma (id\_ferma),

CONSTRAINT fk\_faa\_angajat FOREIGN KEY (id\_angajat)

REFERENCES Angajat (id\_angajat)

);

CREATE TABLE Ferma\_Are\_Produse\_In\_Depozit (

id\_ferma NUMBER,

id\_depozit NUMBER,

PRIMARY KEY (id\_ferma, id\_depozit),

CONSTRAINT fk\_fapd\_ferma FOREIGN KEY (id\_ferma)

REFERENCES Ferma (id\_ferma),

CONSTRAINT fk\_fapd\_depozit FOREIGN KEY (id\_depozit)

REFERENCES Depozit (id\_depozit)

);

CREATE TABLE Linie\_De\_Productie (

id\_produs NUMBER,

id\_ferma NUMBER,

PRIMARY KEY (id\_produs, id\_ferma),

CONSTRAINT fk\_ldp\_produs FOREIGN KEY (id\_produs)

REFERENCES Produs (id\_produs),

CONSTRAINT fk\_ldp\_ferma FOREIGN KEY (id\_ferma)

REFERENCES Ferma (id\_ferma)

);

CREATE TABLE Comanda\_Contine\_Produs (

id\_comanda NUMBER,

id\_produs NUMBER,

cantitate NUMBER NOT NULL,

PRIMARY KEY (id\_comanda, id\_produs),

CONSTRAINT fk\_ccp\_comanda FOREIGN KEY (id\_comanda)

REFERENCES Comanda (id\_comanda),

CONSTRAINT fk\_ccp\_produs FOREIGN KEY (id\_produs)

REFERENCES Produs (id\_produs)

);

commit;

Datele introduse la inceput in baza de date sunt:

INSERT INTO Departament VALUES (1, 'Contabilitate', '0312345679');

INSERT INTO Departament VALUES (2, 'Vanzari', '0312345681');

INSERT INTO Departament VALUES (3, 'IT', '0312345682');

INSERT INTO Departament VALUES (4, 'Intretinere si Reparatii', '0312345683');

INSERT INTO Departament VALUES (5, 'Intretinere Culturi Agricole', '0312345684');

INSERT INTO Angajat VALUES (1, 'Popescu', 'Ion', 'ion.popescu@example.com', '0722345678', 5);

INSERT INTO Angajat VALUES (2, 'Ionescu', 'Maria', 'maria.ionescu@example.com', '0722345679', 1);

INSERT INTO Angajat VALUES (3, 'Georgescu', 'Andrei', 'andrei.georgescu@example.com', '0722345680', 4);

INSERT INTO Angajat VALUES (4, 'Dumitrescu', 'Elena', 'elena.dumitrescu@example.com', '0722345681', 1);

INSERT INTO Angajat VALUES (5, 'Stanescu', 'Mihai', 'mihai.stanescu@example.com', '0722345682', 5);

INSERT INTO Angajat VALUES (6, 'Ilie', 'Andrei', 'andrei.ilie@example.com', '0723000011', 2);

INSERT INTO Angajat VALUES (7, 'Popa', 'Ioana', 'ioana.popa@example.com', '0723000012', 4);

INSERT INTO Angajat VALUES (8, 'Nicolae', 'Raluca', 'raluca.nicolae@example.com', '0723000013', 3);

INSERT INTO Angajat VALUES (9, 'Marin', 'Dan', 'dan.marin@example.com', '0723000014', 2);

INSERT INTO Angajat VALUES (10, 'Dobre', 'Elena', 'elena.dobre@example.com', '0723000015', 3);

INSERT INTO Ferma VALUES (1, 'Craiova', 'Ferma legumicola din Craiova', '3');

INSERT INTO Ferma VALUES (2, 'Iasi', 'Ferma viticola din Iasi', '150');

INSERT INTO Ferma VALUES (3, 'Vaslui', 'Ferma de cereale de la Vaslui', '500');

INSERT INTO Ferma VALUES (4, 'Sibiu', 'Ferma pomicola din Sibiu', '60');

INSERT INTO Ferma VALUES (5, 'Galati', 'Ferma de cereale din Galati ', '1000');

INSERT INTO Produs VALUES (1, 'P001', 'Struguri', 2.99, 'Struguri rosii', 'kg');

INSERT INTO Produs VALUES (2, 'P002', 'Castraveti', 2.83, 'Castraveti cornison', 'kg');

INSERT INTO Produs VALUES (3, 'P003', 'Porumb', 850.00, 'Porumb', 't');

INSERT INTO Produs VALUES (4, 'P004', 'Orz', 800.00, 'Orz', 't');

INSERT INTO Produs VALUES (5, 'P005', 'Mere', 30.00, 'Mere verzi', 'lada');

INSERT INTO Produs VALUES (6, 'I001', 'Ingrasamant NPK', 17.60, 'Ingrasamant lichid', 'l');

INSERT INTO Produs VALUES (7, 'P001', 'Struguri', 3.50, 'Struguri albi', 'kg');

INSERT INTO Produs VALUES (8, 'P006', 'Pere', 35.99, 'Pere', 'lada');

INSERT INTO Produs VALUES (9, 'P007', 'Rosii', 3.50, 'Rosii Roma', 'kg');

INSERT INTO linie\_de\_productie VALUES (2, 1);

INSERT INTO linie\_de\_productie VALUES (1, 2);

INSERT INTO linie\_de\_productie VALUES (7, 2);

INSERT INTO linie\_de\_productie VALUES (3, 3);

INSERT INTO linie\_de\_productie VALUES (4, 3);

INSERT INTO linie\_de\_productie VALUES (5, 4);

INSERT INTO linie\_de\_productie VALUES (3, 5);

INSERT INTO linie\_de\_productie VALUES (4, 5);

INSERT INTO linie\_de\_productie VALUES (8, 4);

INSERT INTO linie\_de\_productie VALUES (9, 1);

INSERT INTO Client VALUES (1, 'Ion Ionescu', '0722000001', 1);

INSERT INTO Client VALUES (2, 'Maria Popescu', '0722000002', 2);

INSERT INTO Client VALUES (3, 'George Vasile', '0722000003', 3);

INSERT INTO Client VALUES (4, 'Elena Radu', '0722000004', 4);

INSERT INTO Client VALUES (5, 'Mihai Zaharia', '0722000005', 5);

INSERT INTO Client VALUES (6, 'Ion Ionescu', '0722002001', 3);

INSERT INTO Utilaj VALUES (1, 'Tractor', 1);

INSERT INTO Utilaj VALUES (6, 'Tractor', 2);

INSERT INTO Utilaj VALUES (10, 'Tractor', 4);

INSERT INTO Utilaj VALUES (7, 'Tractor', 3);

INSERT INTO Utilaj VALUES (8, 'Tractor', 5);

INSERT INTO Utilaj VALUES (9, 'Tractor', 5);

INSERT INTO Utilaj VALUES (2, 'Combinator', 1);

INSERT INTO Utilaj VALUES (11, 'Semantoare', 3);

INSERT INTO Utilaj VALUES (3, 'Semantoare', 5);

INSERT INTO Utilaj VALUES (4, 'Cultivator', 2);

INSERT INTO Utilaj VALUES (5, 'Irigator', 1);

INSERT INTO angajat\_foloseste\_utilaj VALUES (1, 1, TO\_DATE('2022-04-24', 'YYYY-MM-DD'));

INSERT INTO angajat\_foloseste\_utilaj VALUES (3, 2, TO\_DATE('2023-08-16', 'YYYY-MM-DD'));

INSERT INTO angajat\_foloseste\_utilaj VALUES (5, 3, TO\_DATE('2023-05-08', 'YYYY-MM-DD'));

INSERT INTO angajat\_foloseste\_utilaj VALUES (1, 4, TO\_DATE('2022-08-26', 'YYYY-MM-DD'));

INSERT INTO angajat\_foloseste\_utilaj VALUES (3, 5, TO\_DATE('2021-06-25', 'YYYY-MM-DD'));

INSERT INTO angajat\_foloseste\_utilaj VALUES (5, 2, TO\_DATE('2021-10-31', 'YYYY-MM-DD'));

INSERT INTO angajat\_foloseste\_utilaj VALUES (7, 3, TO\_DATE('2022-11-18', 'YYYY-MM-DD'));

INSERT INTO angajat\_foloseste\_utilaj VALUES (7, 4, TO\_DATE('2023-04-04', 'YYYY-MM-DD'));

INSERT INTO angajat\_foloseste\_utilaj VALUES (1, 5, TO\_DATE('2023-11-01', 'YYYY-MM-DD'));

INSERT INTO angajat\_foloseste\_utilaj VALUES (5, 1, TO\_DATE('2022-04-21', 'YYYY-MM-DD'));

INSERT INTO angajat\_foloseste\_utilaj VALUES (5, 6, TO\_DATE('2023-04-21', 'YYYY-MM-DD'));

INSERT INTO angajat\_foloseste\_utilaj VALUES (7, 6, TO\_DATE('2023-05-21', 'YYYY-MM-DD'));

INSERT INTO ferma\_are\_angajati VALUES (1, 1);

INSERT INTO ferma\_are\_angajati VALUES (2, 3);

INSERT INTO ferma\_are\_angajati VALUES (3, 5);

INSERT INTO ferma\_are\_angajati VALUES (4, 7);

INSERT INTO ferma\_are\_angajati VALUES (5, 1);

INSERT INTO ferma\_are\_angajati VALUES (1, 2);

INSERT INTO ferma\_are\_angajati VALUES (2, 3);

INSERT INTO ferma\_are\_angajati VALUES (3, 4);

INSERT INTO ferma\_are\_angajati VALUES (4, 5);

INSERT INTO ferma\_are\_angajati VALUES (5, 2);

INSERT INTO Depozit VALUES (1, 'Onesti', 100, 'Depozit de cereale');

INSERT INTO Depozit VALUES (2, 'Adjud', 200, 'Depozit de legume');

INSERT INTO Depozit VALUES (3, 'Ploiesti', 150, 'Depozit de fructe');

INSERT INTO Depozit VALUES (4, 'Focsani', 250, 'Depozit universal');

INSERT INTO Depozit VALUES (5, 'Bacau', 300, 'Depozit de cereale');

INSERT INTO ferma\_are\_produse\_in\_depozit VALUES (3, 2);

INSERT INTO ferma\_are\_produse\_in\_depozit VALUES (5, 2);

INSERT INTO ferma\_are\_produse\_in\_depozit VALUES (3, 3);

INSERT INTO ferma\_are\_produse\_in\_depozit VALUES (4, 4);

INSERT INTO ferma\_are\_produse\_in\_depozit VALUES (5, 5);

INSERT INTO ferma\_are\_produse\_in\_depozit VALUES (1, 2);

INSERT INTO ferma\_are\_produse\_in\_depozit VALUES (2, 3);

INSERT INTO ferma\_are\_produse\_in\_depozit VALUES (3, 4);

INSERT INTO ferma\_are\_produse\_in\_depozit VALUES (4, 5);

INSERT INTO ferma\_are\_produse\_in\_depozit VALUES (5, 1);

INSERT INTO comanda VALUES (1, TO\_DATE('2021-01-23', 'YYYY-MM-DD'), 1);

INSERT INTO comanda VALUES (2, TO\_DATE('2021-04-01', 'YYYY-MM-DD'), 2);

INSERT INTO comanda VALUES (3, TO\_DATE('2023-03-24', 'YYYY-MM-DD'), 3);

INSERT INTO comanda VALUES (4, TO\_DATE('2021-11-30', 'YYYY-MM-DD'), 4);

INSERT INTO comanda VALUES (5, TO\_DATE('2023-10-06', 'YYYY-MM-DD'), 5);

INSERT INTO comanda VALUES (6, TO\_DATE('2023-05-06', 'YYYY-MM-DD'), 5);

INSERT INTO comanda\_contine\_produs VALUES (1, 1, 6);

INSERT INTO comanda\_contine\_produs VALUES (2, 2, 7);

INSERT INTO comanda\_contine\_produs VALUES (3, 3, 1);

INSERT INTO comanda\_contine\_produs VALUES (4, 4, 2);

INSERT INTO comanda\_contine\_produs VALUES (5, 5, 3);

INSERT INTO comanda\_contine\_produs VALUES (1, 2, 6);

INSERT INTO comanda\_contine\_produs VALUES (2, 3, 7);

INSERT INTO comanda\_contine\_produs VALUES (3, 4, 8);

INSERT INTO comanda\_contine\_produs VALUES (4, 5, 2);

INSERT INTO comanda\_contine\_produs VALUES (5, 1, 3);

INSERT INTO factura VALUES (1, 1, 'FAC001', TO\_DATE('2021-01-30', 'YYYY-MM-DD'), 123.45);

INSERT INTO factura VALUES (2, 2, 'FAC002', TO\_DATE('2021-04-02', 'YYYY-MM-DD'), 678.90);

INSERT INTO factura VALUES (3, 3, 'FAC003', TO\_DATE('2023-03-28', 'YYYY-MM-DD'), 234.56);

INSERT INTO factura VALUES (4, 4, 'FAC004', TO\_DATE('2021-10-10', 'YYYY-MM-DD'), 789.01);

INSERT INTO factura VALUES (5, 5, 'FAC005', TO\_DATE('2023-10-30', 'YYYY-MM-DD'), 345.67);

INSERT INTO factura VALUES (6, 5, 'FAC006', TO\_DATE('2023-05-23', 'YYYY-MM-DD'), 143.47);

INSERT INTO Informatii\_utilaj VALUES (1, 1500, 'Excelenta', TO\_DATE('2019-04-08', 'YYYY-MM-DD'), 'John Deere 5075E');

INSERT INTO Informatii\_utilaj VALUES (2, 1800, 'Buna', TO\_DATE('2019-12-19', 'YYYY-MM-DD'), 'Case IH Maxxum 110');

INSERT INTO Informatii\_utilaj VALUES (3, 900, 'Nou', TO\_DATE('2019-03-07', 'YYYY-MM-DD'), 'Kverneland Miniair Nova');

INSERT INTO Informatii\_utilaj VALUES (4, 1200, 'Uzura Medie', TO\_DATE('2019-08-16', 'YYYY-MM-DD'), 'Amazone KE 3000 Super');

INSERT INTO Informatii\_utilaj VALUES (5, 2000, 'Buna', TO\_DATE('2019-02-22', 'YYYY-MM-DD'), 'Valley 8000 series');

INSERT INTO Informatii\_utilaj VALUES (6, 1600, 'Foarte Buna', TO\_DATE('2019-11-15', 'YYYY-MM-DD'), 'Massey Ferguson 6713');

INSERT INTO Informatii\_utilaj VALUES (7, 1400, 'Excelenta', TO\_DATE('2019-07-21', 'YYYY-MM-DD'), 'New Holland T5.105');

INSERT INTO Informatii\_utilaj VALUES (8, 1550, 'Nou', TO\_DATE('2019-06-30', 'YYYY-MM-DD'), 'Fendt 500 Vario');

INSERT INTO Informatii\_utilaj VALUES (9, 1700, 'Uzura Redusa', TO\_DATE('2019-10-12', 'YYYY-MM-DD'), 'CLAAS Arion 630');

INSERT INTO Informatii\_utilaj VALUES (10, 1300, 'Buna', TO\_DATE('2019-05-24', 'YYYY-MM-DD'), 'Kubota M5-091');

INSERT INTO Informatii\_utilaj VALUES (11, 950, 'Uzura Medie', TO\_DATE('2019-09-17', 'YYYY-MM-DD'), 'Accord Optima');

commit;

6.Sa se creeze o procedura RaportFerma care va primi prin id o ferma si va retine in 3 colectii diferite utilajele si modelele utilajelor prezente in ferma, angajatii care lucreaza in ferma si produsele cultivate/facute in ferma.

CREATE OR REPLACE PROCEDURE RaportFerma(p\_id\_ferma IN number) IS

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

t\_angajati\_id tablou\_indexat;

TYPE tablou\_imbricat IS TABLE OF NUMBER;

t\_produse\_id tablou\_imbricat := tablou\_imbricat();

v\_index PLS\_INTEGER := 1;

TYPE vector IS VARRAY(105) OF VARCHAR2(255);

t\_utilaje vector := vector();

BEGIN

FOR rec IN (SELECT id\_angajat FROM ferma\_are\_angajati WHERE id\_ferma = p\_id\_ferma) LOOP

t\_angajati\_id(v\_index) := rec.id\_angajat;

v\_index := v\_index + 1;

END LOOP;

v\_index := 1;

FOR rec IN (SELECT id\_produs FROM linie\_de\_productie WHERE id\_ferma = p\_id\_ferma) LOOP

t\_produse\_id.EXTEND;

t\_produse\_id(v\_index) := rec.id\_produs;

v\_index := v\_index + 1;

END LOOP;

FOR rec IN (SELECT u.tip, iu.model

FROM Utilaj u

JOIN Informatii\_utilaj iu ON u.id\_utilaj = iu.id\_utilaj

WHERE u.id\_ferma = p\_id\_ferma) LOOP

t\_utilaje.EXTEND;

t\_utilaje(t\_utilaje.LAST) := 'Tip: ' || rec.tip || ', Model: ' || rec.model;

END LOOP;

FOR i IN 1..t\_angajati\_id.COUNT LOOP

FOR rec IN (SELECT nume, prenume, email, telefon FROM Angajat WHERE id\_angajat = t\_angajati\_id(i)) LOOP

DBMS\_OUTPUT.PUT\_LINE('Nume: ' || rec.nume || ' Prenume: ' || rec.prenume ||

' Email: ' || rec.email || ' Telefon: ' || rec.telefon);

END LOOP;

END LOOP;

FOR i IN 1..t\_produse\_id.COUNT LOOP

FOR rec IN (SELECT cod\_produs, denumire, pret, descriere FROM Produs WHERE id\_produs = t\_produse\_id(i)) LOOP

DBMS\_OUTPUT.PUT\_LINE('Produs - Cod: ' || rec.cod\_produs || ', Denumire: ' || rec.denumire ||

', Pret: ' || rec.pret || ', Descriere: ' || rec.descriere);

END LOOP;

END LOOP;

FOR i IN 1..t\_utilaje.COUNT LOOP

DBMS\_OUTPUT.PUT\_LINE('Utilaj ' || i || ': ' || t\_utilaje(i));

END LOOP;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

DBMS\_OUTPUT.PUT\_LINE('Nu exista date pentru ferma introdusa');

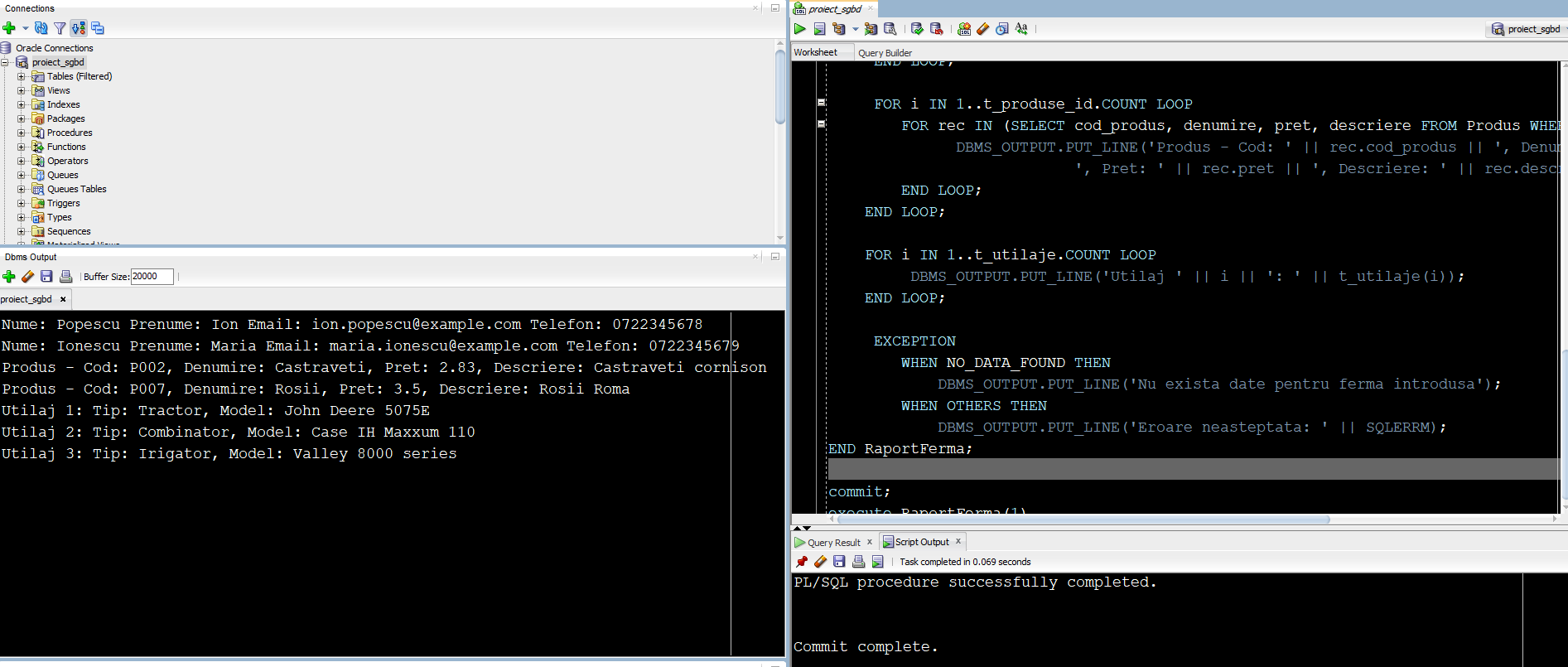
WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE('Eroare neasteptata: ' || SQLERRM);

END RaportFerma;

commit;

execute RaportFerma(1)



7. Calculati si afisati suma totala a comenzilor pentru fiecare angajat din companie.

CREATE OR REPLACE PROCEDURE AfisareSumaComenzi is

CURSOR c\_angajati IS

SELECT id\_angajat,nume,prenume,email,telefon FROM Angajat;

CURSOR c\_clienti (p\_id\_angajat NUMBER) IS

SELECT id\_client FROM Client WHERE id\_angajat = p\_id\_angajat;

CURSOR c\_facturi (p\_id\_client NUMBER) IS

SELECT suma FROM Factura WHERE id\_client = p\_id\_client;

v\_suma NUMBER;

BEGIN

FOR ang IN c\_angajati LOOP

v\_suma := 0;

FOR cli IN c\_clienti(ang.id\_angajat) LOOP

FOR fac IN c\_facturi(cli.id\_client) LOOP

v\_suma := v\_suma + fac.suma;

END LOOP;

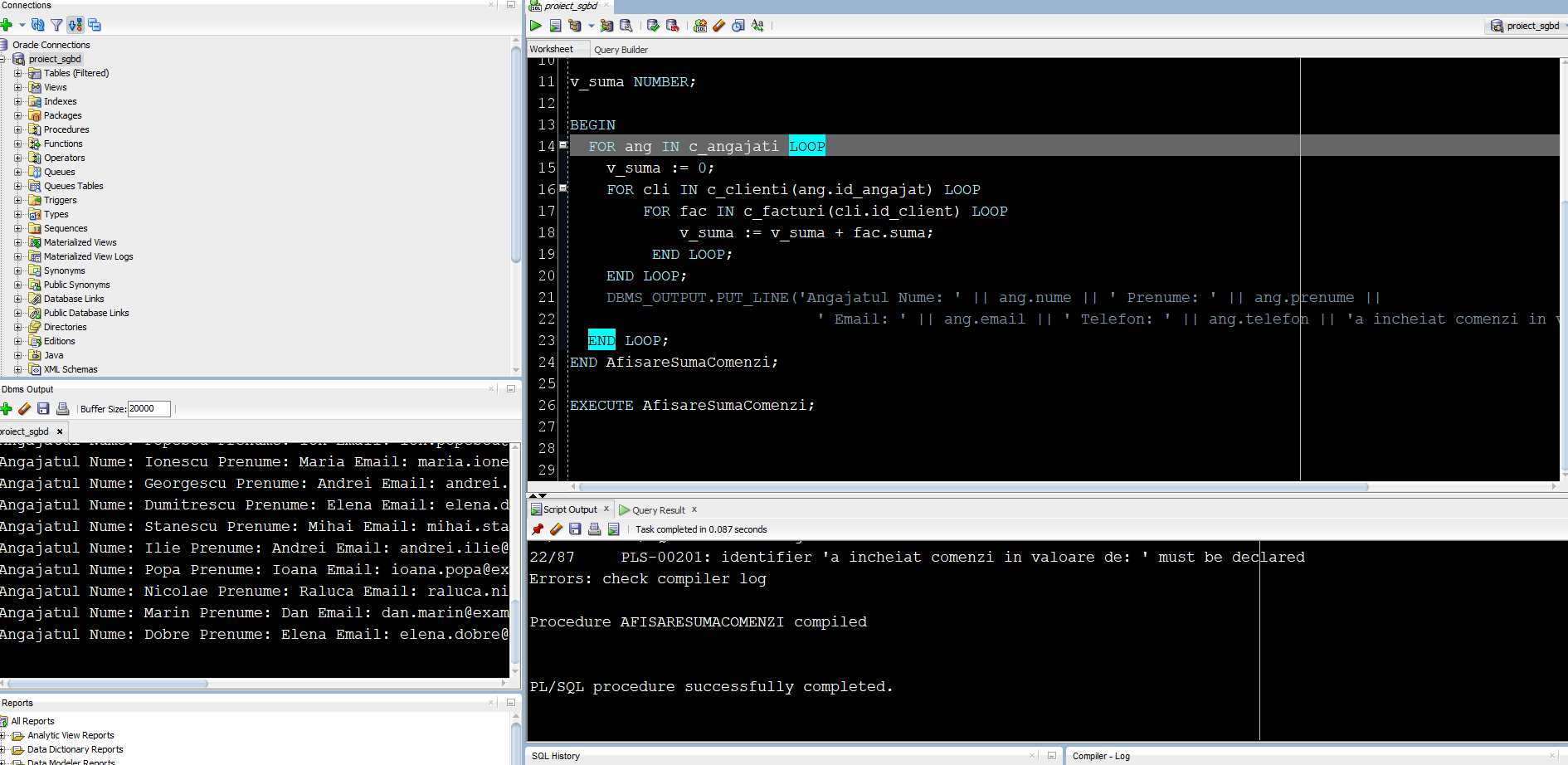
END LOOP;

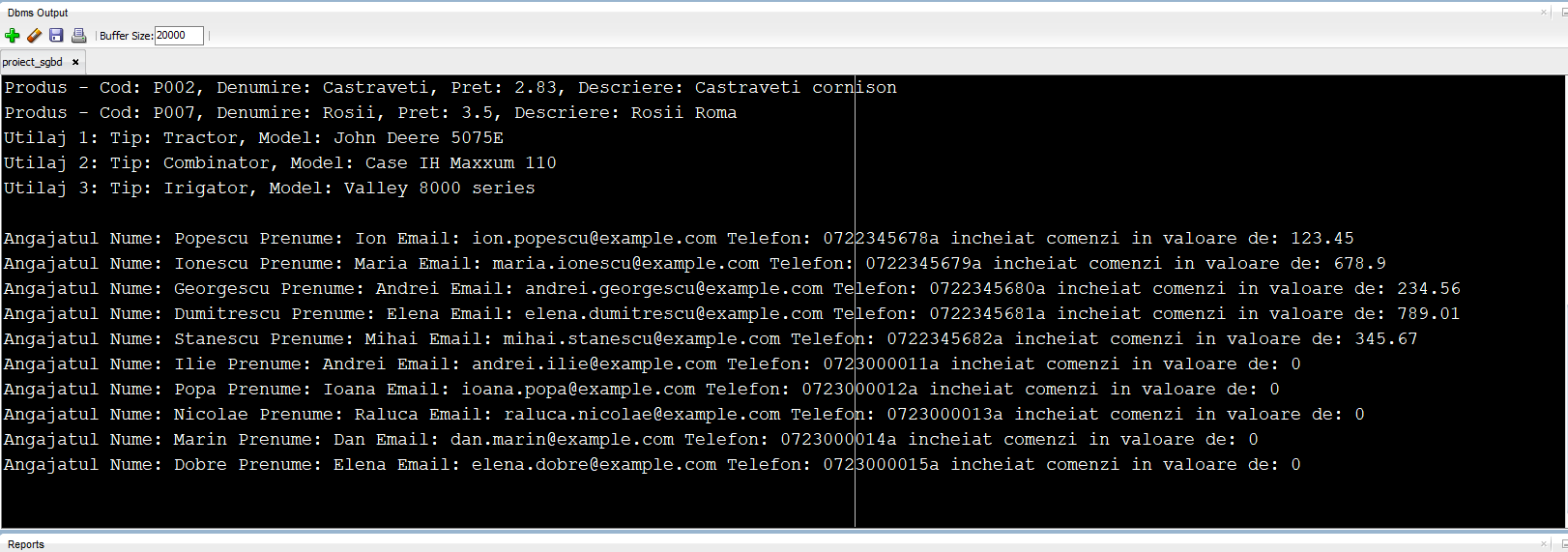
DBMS\_OUTPUT.PUT\_LINE('Angajatul Nume: ' || ang.nume || ' Prenume: ' || ang.prenume ||

' Email: ' || ang.email || ' Telefon: ' || ang.telefon || 'a incheiat comenzi in valoare de: ' || v\_suma);

END LOOP;

END AfisareSumaComenzi;





8.Sa se afiseze de cate ori a fost folosit un tip de utilaj intr-un anumit an. Tipul de utilaj si si anul va fi dat de utilizator. Daca utilajul nu a fost folosit in acel an se va afisa un mesaj corespunzator, iar daca utilajul a fost utilizat de mai mult de 24 de ori se va afisa un mesaj de suprafolosire a utilajului.

CREATE OR REPLACE FUNCTION UsageChart(p\_tip\_utilaj VARCHAR2, p\_an NUMBER)

RETURN VARCHAR2 IS

v\_nr\_angajati NUMBER;

e\_input\_data exception;

e\_no\_use exception;

e\_over\_use exception;

BEGIN

--daca nu au fost introduse suficiente date ridicam eroarea de input\_data

IF p\_tip\_utilaj IS NULL OR p\_an IS NULL THEN

RAISE e\_input\_data;

END IF;

--calculam numarul de angajati care au folosit utilajul introdus

SELECT COUNT(\*)

INTO v\_nr\_angajati

FROM Angajat a

JOIN angajat\_foloseste\_utilaj afu ON a.id\_angajat = afu.id\_angajat

JOIN Utilaj u ON afu.id\_utilaj = u.id\_utilaj

WHERE lower(u.tip) = lower(p\_tip\_utilaj)and EXTRACT(YEAR FROM TO\_DATE(afu.data, 'DD-MON-YY')) = p\_an;

--ridicam si erorile de suprasolicitare si nefolosire

IF v\_nr\_angajati = 0 THEN

RAISE e\_no\_use;

ELSIF v\_nr\_angajati > 24 THEN

RAISE e\_over\_use;

END IF;

--afisam mesajul dorit

RETURN 'Utilajul ' || p\_tip\_utilaj || ' in anul ' || p\_an|| ' a fost folosit de ' || v\_nr\_angajati || ' ori';

--tratam erorile

EXCEPTION

WHEN e\_input\_data THEN

RETURN 'Nu au fost introduse suficiente date!';

WHEN e\_no\_use THEN

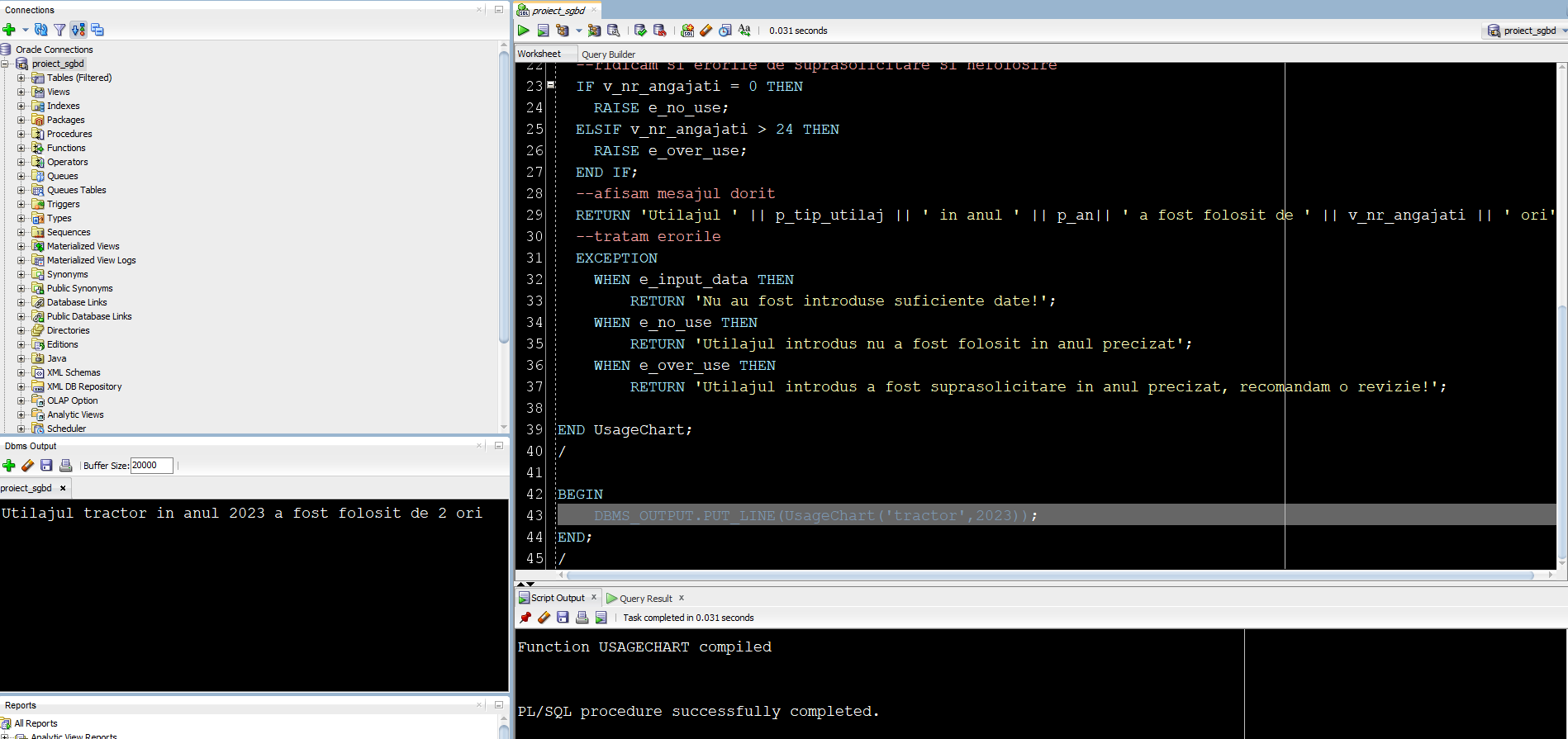
RETURN 'Utilajul introdus nu a fost folosit in anul precizat';

WHEN e\_over\_use THEN

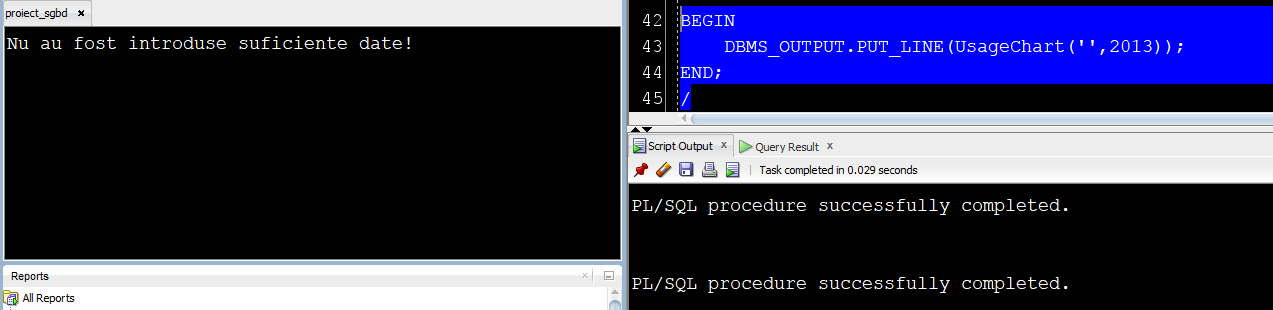
RETURN 'Utilajul introdus a fost suprasolicitare in anul precizat, recomandam o revizie!';

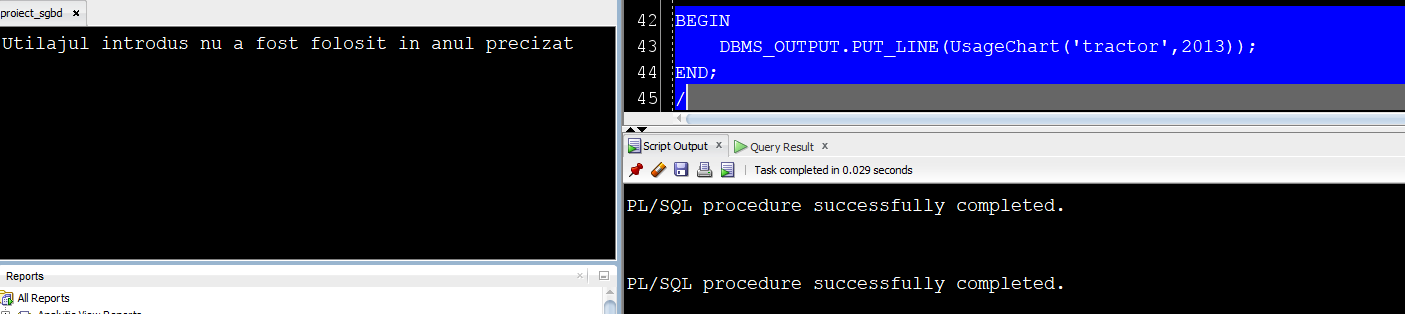
END UsageChart;

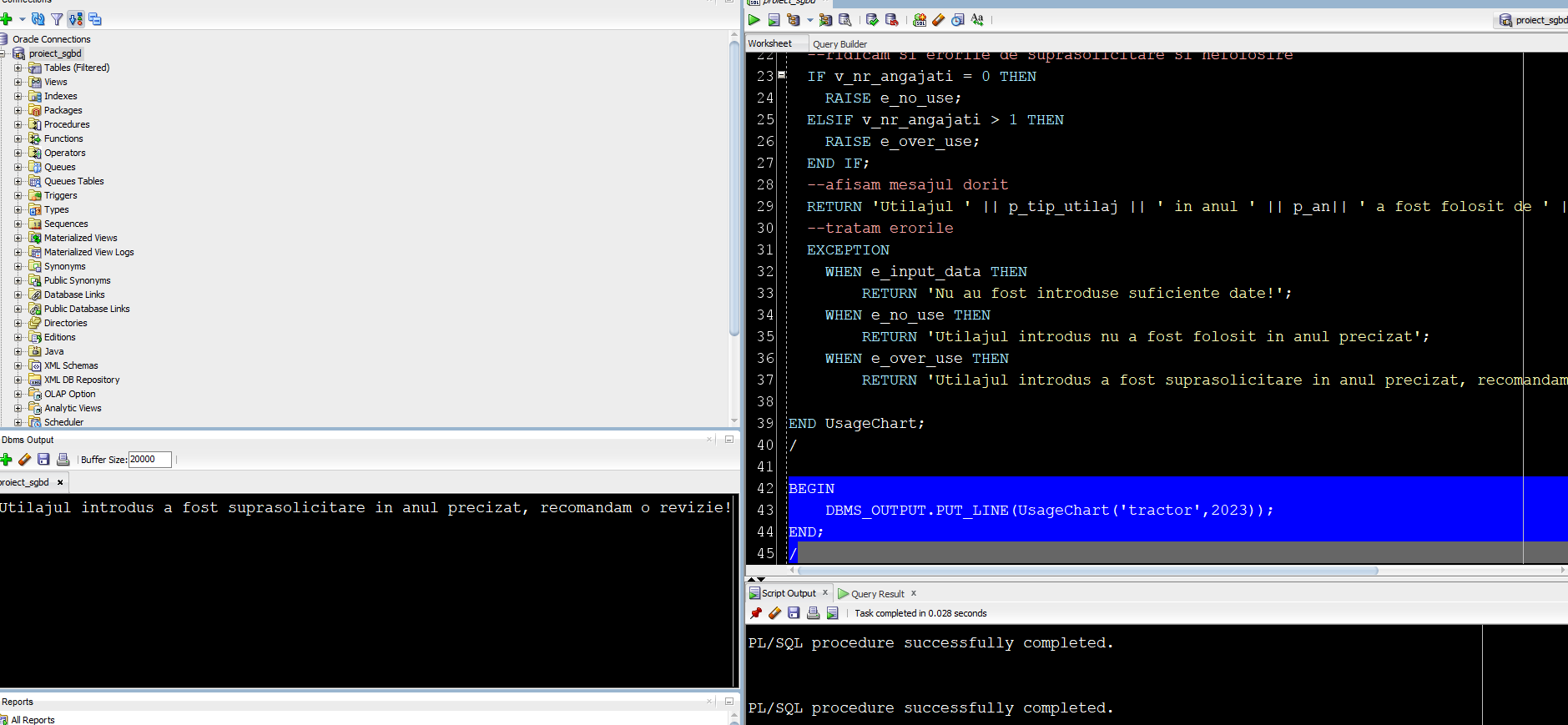
/



Exceptiile: (pentru exceptia de suprasolicitare, am modificat temporar conditia la mai mult de o utilizare pentru a evidentia eroarea, deoarece nu am 25 de date introduse in tabela.)







9.Pornind de la numele si prenumele unui client sa se afiseze: numele si departamentul angajatului responsabil de acest client

CREATE OR REPLACE PROCEDURE AfisareComenziClient(nume\_client VARCHAR2) IS

--ne folosim de un cursor sa iteram prin detaliile cerute

CURSOR comenzi\_cursor IS

SELECT c.id\_comanda, f.numar\_factura, a.nume || ' ' || a.prenume AS nume\_angajat, d.denumire AS nume\_departament

FROM Client cl

JOIN Comanda c ON cl.id\_client = c.id\_client

JOIN Factura f ON c.id\_comanda = f.id\_comanda

JOIN Angajat a ON cl.id\_angajat = a.id\_angajat

JOIN Departament d ON a.id\_departament = d.id\_departament

WHERE lower(cl.nume\_pr) = lower(nume\_client);

v\_id\_comanda Comanda.id\_comanda%TYPE;

v\_numar\_factura Factura.numar\_factura%TYPE;

v\_nume\_angajat VARCHAR2(100);

v\_nume\_departament Departament.denumire%TYPE;

v\_nr\_clienti INT;

e\_input EXCEPTION;

BEGIN

--verificam daca au fost introduse date

IF nume\_client IS NULL THEN

RAISE e\_input;

END IF;

--calculam numarul de clienti cu acest nume si prenume pentru a ne asigura ca exista doar un client

SELECT COUNT(\*)

INTO v\_nr\_clienti

FROM Client

WHERE lower(nume\_pr) = lower(nume\_client);

--daca nu exista un client cu acest nume sau daca exista mai multi vom ridica erorile corespondente

IF v\_nr\_clienti = 0 THEN

RAISE NO\_DATA\_FOUND;

ELSIF v\_nr\_clienti > 1 THEN

RAISE TOO\_MANY\_ROWS;

END IF;

--deschidem cursorul si iteram prin toate datele, pentru a afisa toate comenzile si facturile clientului

OPEN comenzi\_cursor;

LOOP

FETCH comenzi\_cursor INTO v\_id\_comanda, v\_numar\_factura, v\_nume\_angajat, v\_nume\_departament;

EXIT WHEN comenzi\_cursor%NOTFOUND;

DBMS\_OUTPUT.PUT\_LINE('Comanda ID: ' || v\_id\_comanda ||

', Numar Factura: ' || v\_numar\_factura ||

', Angajat: ' || v\_nume\_angajat ||

', Departament: ' || v\_nume\_departament);

END LOOP;

CLOSE comenzi\_cursor;

--tratam exceptiile

EXCEPTION

WHEN e\_input THEN

DBMS\_OUTPUT.PUT\_LINE('Nu au fost introduse numele si prenumele clientului');

RETURN;

WHEN NO\_DATA\_FOUND THEN

DBMS\_OUTPUT.PUT\_LINE('Nu exista date pentru clientul specificat');

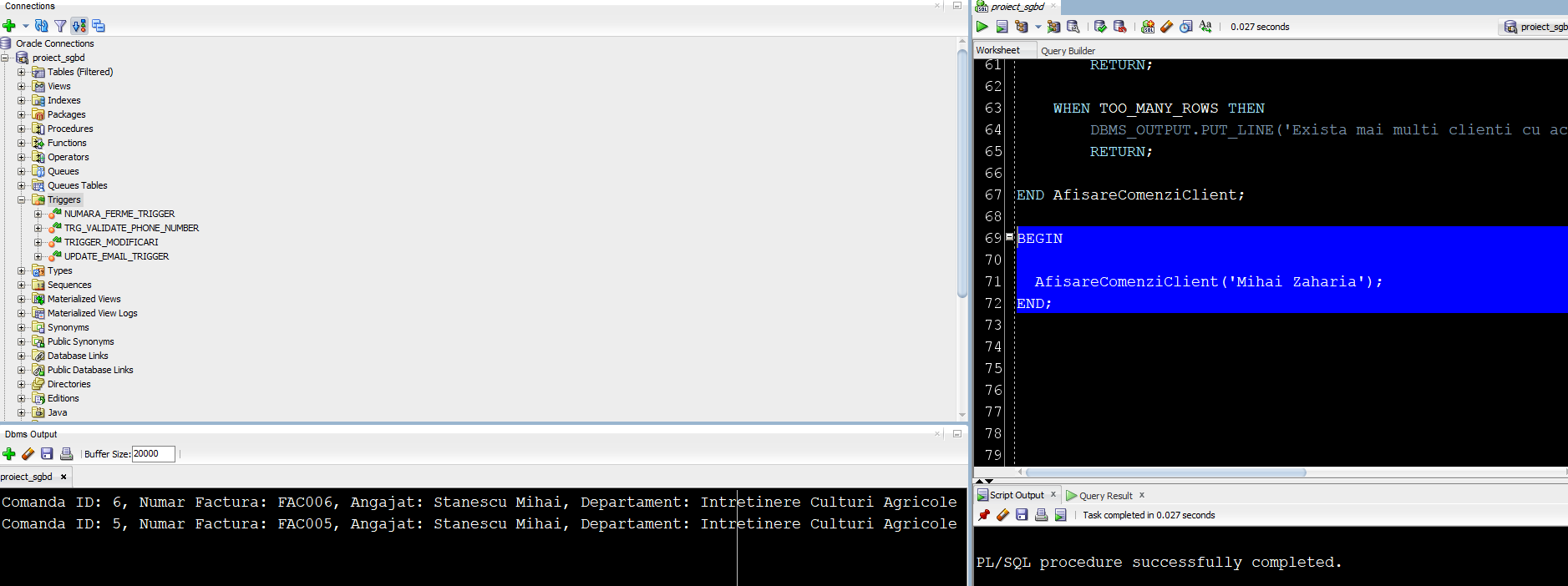
RETURN;

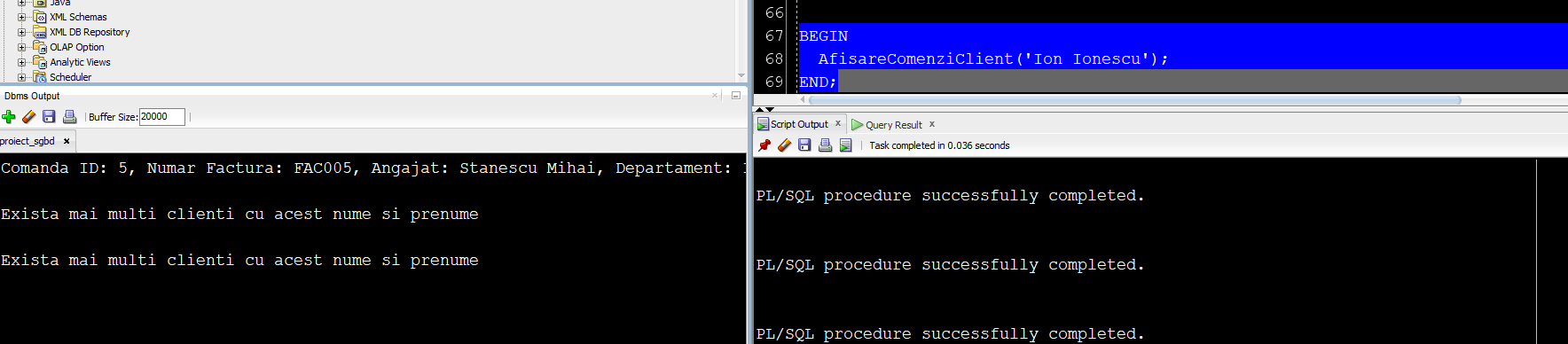
WHEN TOO\_MANY\_ROWS THEN

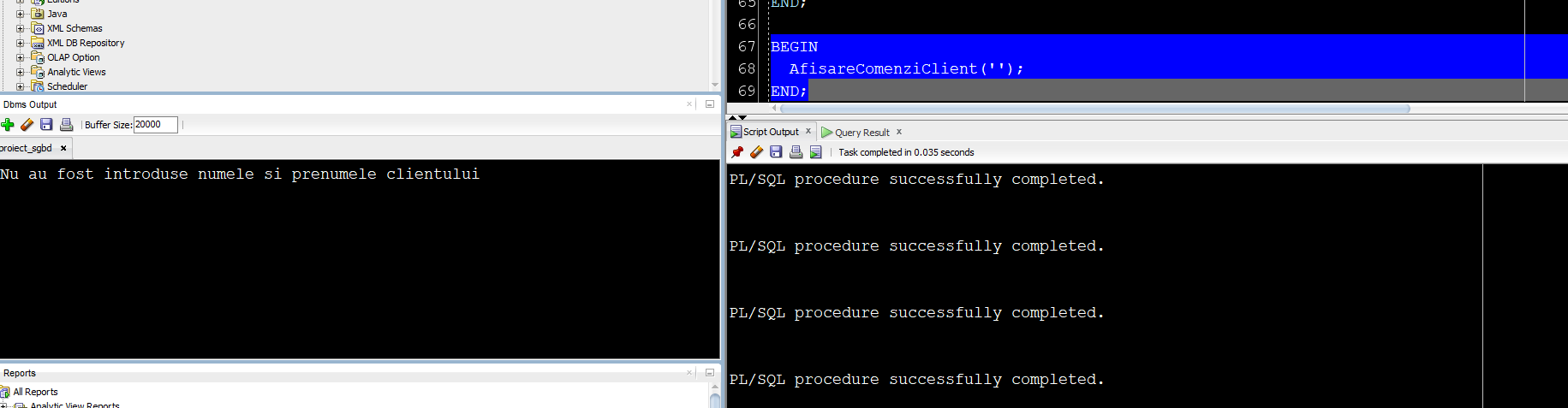
DBMS\_OUTPUT.PUT\_LINE('Exista mai multi clienti cu acest nume si prenume');

RETURN;

END AfisareComenziClient;







10.

CREATE OR REPLACE TRIGGER trg\_depozit\_operations

BEFORE DELETE OR INSERT ON Depozit

DECLARE

v\_depozit\_count NUMBER := 0;

BEGIN

SELECT COUNT(id\_depozit) INTO v\_depozit\_count FROM Depozit;

IF DELETING AND v\_depozit\_count = 1 THEN

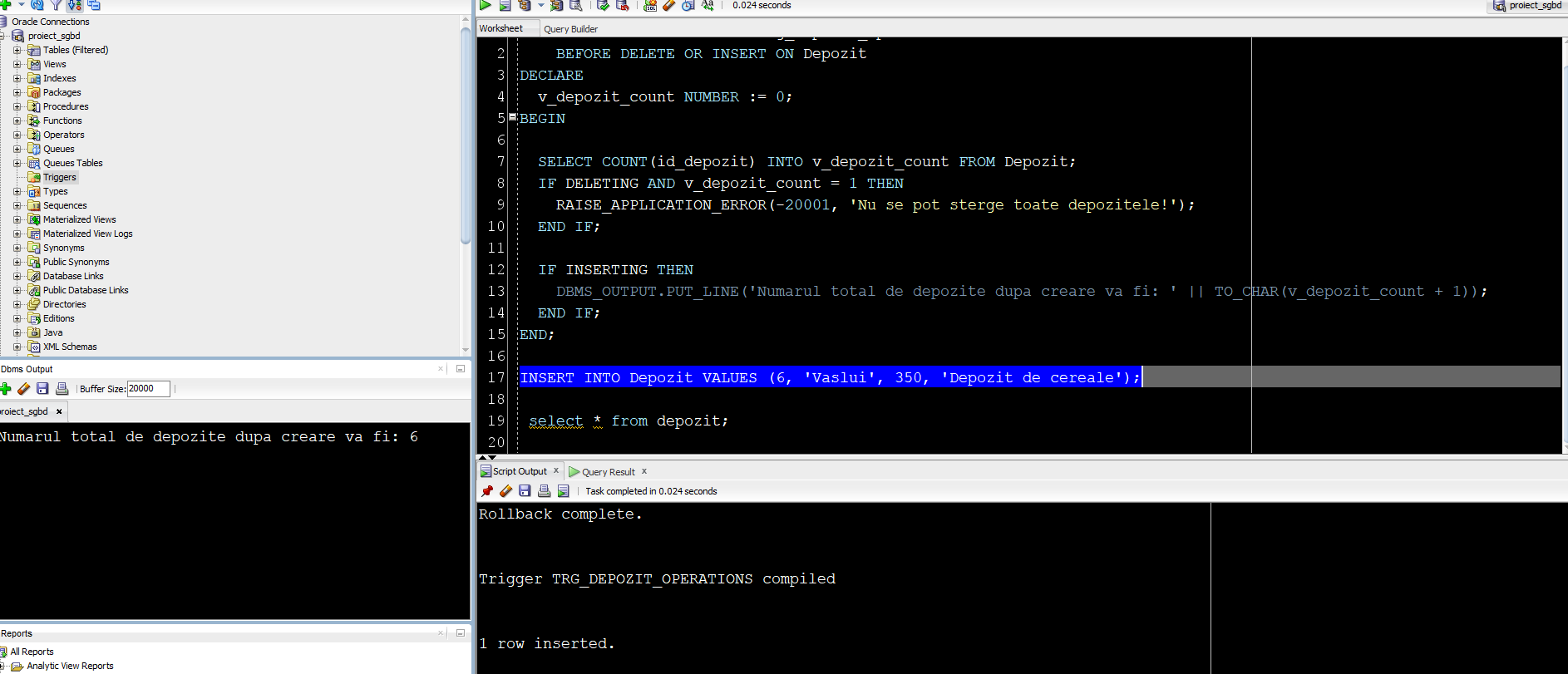
RAISE\_APPLICATION\_ERROR(-20001, 'Nu se pot sterge toate depozitele!');

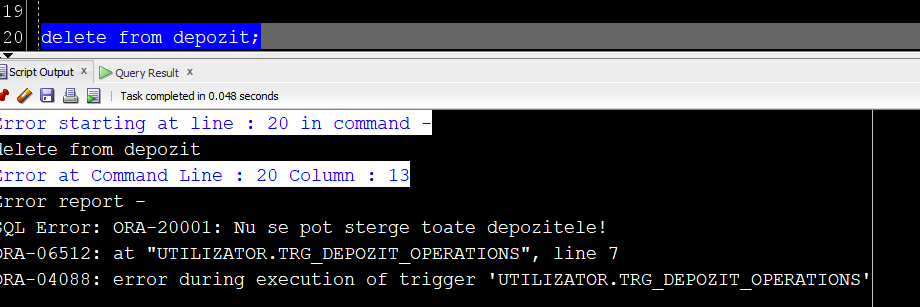
END IF;

IF INSERTING THEN

DBMS\_OUTPUT.PUT\_LINE('Numarul total de depozite dupa creare va fi: ' || TO\_CHAR(v\_depozit\_count + 1));

END IF;

END;



11. Se va crea un trigger LMD de linie care va actualiza automat suma din factura la modificarea cantitatii din comanda.

CREATE OR REPLACE TRIGGER actualizeaza\_suma\_factura

AFTER INSERT OR UPDATE ON Comanda\_Contine\_Produs

FOR EACH ROW

DECLARE

pret\_unitar NUMBER;

suma\_noua NUMBER;

BEGIN

SELECT pret INTO pret\_unitar FROM Produs WHERE id\_produs = :NEW.id\_produs;

-- calculam suma noua

suma\_noua := pret\_unitar \* :NEW.cantitate;

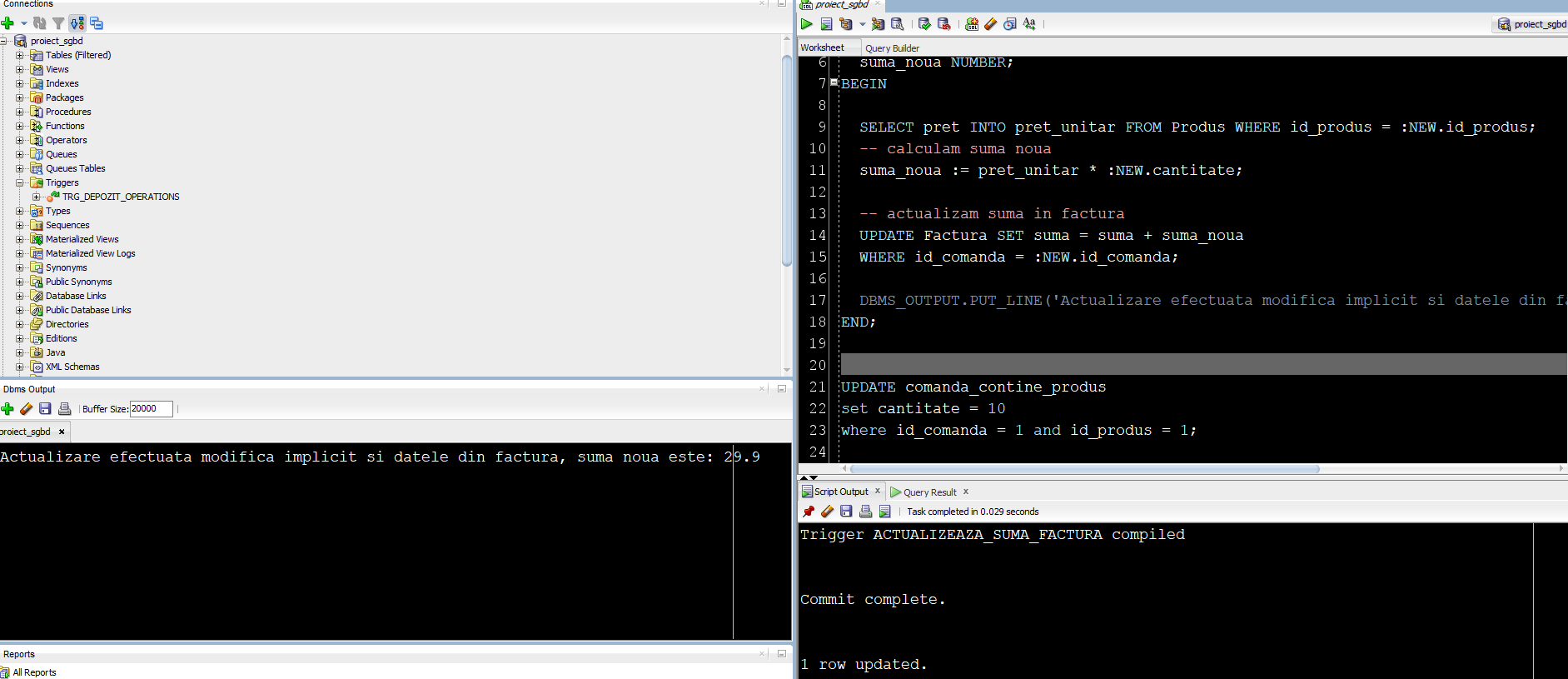
-- actualizam suma in factura

UPDATE Factura SET suma = suma + suma\_noua

WHERE id\_comanda = :NEW.id\_comanda;

DBMS\_OUTPUT.PUT\_LINE('Actualizare efectuata modifica implicit si datele din factura, suma noua este: ' || suma\_noua);

END;



12.Se va crea un trigger LDD care va adauga intr-o tabela auxiliara detalii despre modificarile facute pe baza de date, daca user-ul curent este „UTILIZATOR”.

CREATE TABLE audit\_firma (

id\_modificare NUMBER PRIMARY KEY,

tip\_modificare VARCHAR2(50),

tip\_obiect VARCHAR2(50),

nume\_obiect VARCHAR2(50),

data\_modficare DATE

);

CREATE SEQUENCE audit\_firma\_seq START WITH 1 INCREMENT BY 1;

CREATE OR REPLACE TRIGGER audit\_firma\_agricultura

AFTER CREATE OR ALTER OR DROP ON SCHEMA

BEGIN

-- verificarea pentru utilizator

IF lower(SYS\_CONTEXT('USERENV', 'SESSION\_USER')) != 'utilizator' THEN

--ridicam eroare in caz contrar

RAISE\_APPLICATION\_ERROR(-20001, 'Nu aveti autorizatie sa modificati baza de date!!');

ELSE

--daca se trece de verificare inseram detatliile modificarilor

INSERT INTO audit\_firma (id\_modificare,tip\_modificare, tip\_obiect, nume\_obiect, data\_modficare)

VALUES (audit\_firma\_seq.NEXTVAL,ora\_sysevent, ora\_dict\_obj\_type, ora\_dict\_obj\_name, SYSDATE);

END IF;

END;