*Magazin GSM* (**Sistem Global pentru Comunicații Mobile**)

*Dumitrescu Marius Cristian grupa 242*

**REGULI DE FUNCTIONARE:**

1. Mai multe piese pot apartine unui depozit, iar o piesa poate apartine unui singur depozit.
2. Produsele reprezinta obiecte(folosite sau noi) comercializate de magazin. Un posibil produs ar fi de exemplu un telefon mobil reconditionat.
3. Un depozit poate contine mai multe produse, iar un produs poate apartine mai multor depozite.
4. Un depozit poate fi aprovizionat de mai multi distribuitori, cel putin unul, iar un distribuitor poate aproviziona mai multe depozite.
5. Un magazin poate avea mai multe depozite, iar un depozit poate poate asigura necesarul mai multor magazine.
6. Un produs poate apartine mai multor depozite.
7. Se numeste client, o persoana care achizitioneaza cel putin un produs sau isi inregistreaza cel putin un device pentru a fi reparat.
8. Un client poate cumpara mai multe produse, iar un produs poate apartine unui singur client.
9. Un client poate aduce mai multe device-uri, insa un device poate fi adus de catre un singur client. Device-ul defect reprezinta un obiect, de obicei telefon, adus de catre client magazinului/service-ului in scopul remedierii acestuia.
10. Fiecare magazin are cate un istoric pe luna, unde sunt trecute informatiile legate de incasari,taxe,obiecte etc. Un magazin poate avea mai multe istorice, insa fiecare istoric corespunde unui singur magazin.
11. Un magazin are cel putin un angajat, iar un angajat lucreaza intr-un singur magazin.
12. Angajatii pot face parte dintr-o singura categorie (casier,tehnician,paznic) sau din niciuna.

**ENTITATI:**

Pentru modelul de date referitor la un un service specializat de telefoane (GSM) structurile PIESA,PRODUS, CLIENT, MAGAZIN, DEVICE\_DEFECT, DEPOZIT, ISTORIC, DISTRIBUITOR, ANGAJAT, PAZNIC, CASIER, TEHNICIAN reprezinta entitatile.

MAGAZIN = entitate ce contine informatii cu privire la locatia, programul de lucru, numarul de angajati si nu numai din cadrul magazinului. Cheia primara a acestei entitati este cod\_magazin.

PIESA = entitate ce contine informatii cu privire la piesele disponibile impreuna cu denumirea acestora. Cheia primara a acestei entitati este cod\_piesa#.

CLIENT = persoana fizica, independenta, care a achizitionat cel putin un produs sau a inregistrat un device defect. Cheia primara a acestei entitati este cod\_client.

PRODUS = entitate care contine informatii referitoare la produsele disponibile pentru a fi cumparate. Cheia primara a acestei entitati este cod\_produs.

DEVICE\_DEFECT = entitate ce contine informatii cu privire la device-urile inregistarte intr-un program de depanare. Cheia primara a acestei entitati este cod\_device\_defect.

ISTORIC = entitate dependenta de MAGAZIN ce contie informatii despre situatia magazinelor. Cheia primara a acestei entitati este cod\_istoric.

DEPOZIT = entitate independenta ce contine informatii cu privire la depozitele magazinelor. Cheia primara a acestei entitati este cod\_depozit.

DISTRIBUITOR = persoana juridica ce asigura necesarul de piese, accesorii depozitului. Cheia primara a acestei entitati este cod\_distribuitor.

ANGAJAT = persoana fizica care presteaza munca in cadrul magazinului. Cheia primara a acestei entitati este cod\_angajat.

PAZNIC = subentitate din ANGAJAT ce contine detalii despre fiecare persoana fizica care presteaza munca in cadrul magazinului sub denumirea de “pazinc”. Paznicul are rolul de a mentine un standard inalt de siguranta pentru magazin. Cheia primara a acestei entitati este cod\_paznic.

CASIER = subentitate din ANGAJAT ce contine detalii despre fiecare persoana fizica care presteaza munca in cadrul magazinului sub denumirea de “casier”. Casierul are rolul de a incasa banii clientilor, folosindu-se de casa de marcat. Cheia primara a acestei entitati este cod\_casier.

TEHNICIAN = subentitate din ANGAJAT ce contine detalii despre fiecare persoana fizica care presteaza munca in cadrul magazinului sub denumirea de “tehnician”. Tehnicianul are rolul de a pune in functiune device-urile defecte. Cheia primara a acestei entitati este cod\_tehnician.

**ATRIBUTE:**

*Entitatea MAGAZIN are ca atribute:*

locatie = variabila de tip caracter, de lungime maxima 20, care reprezinta locatia magazinului

program = variabila de tip text care contine informatii cu privire la programul de lucru.

nr\_angajati = variabila de tip intreg, de lungime maxima 2 care reprezinta numarul de angajati din magazin.

Cod\_magazin = variabila de tip number, de lungime maxima 4 ce reprezinta codul magazinului

Entitatea PIESA are ca atribute:

Denumire = variabila de tip caracter, de lungime maxima 20 ce reprezinta denumirea piesei.

Cod\_piesa = variabila de tip caracter, de lungime maxima 3 ce reprezinta codul piesei

*Entitatea independenta CLIENT are ca atribute:*

Nume = variabila de tip caracter, de lungime maxima 20 ce reprezinta numele clientului.

Prenume = variabila de tip caracter, de lungime maxima 20 ce reprezinta prenumele clientului.

Device = variabila de tip caracter, de lungime maxima 20 ce reprezinta modelul device-ului.

e-mail = variabila de tip caracter, de lungime maxima 20 ce reprezinta adresa de e-mail a clientului.

Adresa= variabila de tip caracter, de lungime maxima 40 ce reprezinta adresa clientului.

Cod\_client = variabila de tip number, de lungime maxima 4 ce reprezinta codul clientului.

*Entitatea PRODUS are ca atribute:*

Model = variabila de tip caracter, de lungime maxima 20 ce reprezinta denumirea produsului

Pret = variabila de tip intreg ce reprezinta pretul produsului in magazin

Cantitate = variabila de tip intreg reprezentand numarul de produse de acest tip

Cod\_produs= variabila de tip number, de lungime maxima 4 ce reprezinta codul produsului

Cod\_client = variabila de tip number, de lungime maxima 4 ce reprezinta codul clientului; aceasta este o cheie externa catre coloana cod\_client din tabelul CLIENT

*Entitatea DEVICE\_DEFECT are ca atribute:*

Model = variabila de tip caracter, de lungime maxima 20 ce reprezinta denumirea device-ului

Cost = variabila de tip intreg ce reprezinta costul reparatiei

Data\_primire = variabila calendaristica ce indica data primirii

Data\_predare = variabila calendaristica ce indica data preluarii din service de catre client

Cod\_device\_defect = variabila de tip number, de lungime maxima 4 ce reprezinta codul device-ului

Cod\_client = variabila de tip number, de lungime maxima 4 ce reprezinta codul clientului; aceasta este o cheie externa catre coloana cod\_client din tabelul CLIENT

Cod\_magazin = variabila de tip number, de lungime maxima 4 ce reprezinta codul magazinului; aceasta este o cheie externa catre coloana cod\_magazin din tabelul MAGAZIN

*Entitate ISTORIC are ca atribute:*

Data = variabila de tip caracter, de lungime maxima 20 ce reprezinta data

Incasari = variabila de tip intreg asociata tuturor incasarilor magazinului in luna respectiva

Taxe = variabila de tip intreg ce reprezinta valoarea totala a taxelor achitate in luna respectiva

Salarii = variabila de tip intreg ce reprezinta valoarea totala insumata din salariile angajatiilor din luna respectiva

Cod\_istoric = variabila de tip number, de lungime maxima 4 ce reprezinta codul istoricului

Cod\_magazin = variabila de tip number, de lungime maxima 4 ce reprezinta codul magazinului; aceasta este o cheie externa catre coloana cod\_magazin din tabelul MAGAZIN

*Entitate DEPOZIT are ca atribute:*

Denumire = variabila de tip caracter, de lungime maxima 20 ce reprezinta denumirea produsului

Cantitate = variabila de tip intreg, de lungime maxima 4 ce reprezinta stocul de produse de acelasi fel

Data\_primire = variabila calendaristica ce indica data primirii ultimului produs din categoria sa

Cod\_depozit = variabila de tip number, de lungime maxima 4 ce reprezinta codul depozitului

*Entitate DISTRIBUITOR are ca atribute:*

Denumire = variabila de tip caracter, de lungime maxima 20 ce reprezinta numele distribuitorului

Adresa = variabila de tip caracter, de lungime maxima 40 ce reprezinta adresa distribuitorului

Produs = variabila de tip caracter, de lungime maxima 20 ce reprezinta denumirea produsului distribuit

Cantitate = variabila de tip intreg, de lungime maxima 4 ce reprezinta cantitatea de produse distribuite

Cod\_distribuitor = variabila de tip number, de lungime maxima 4 ce reprezinta codul distribuitorului

*Entitate ANGAJAT are ca atribute:*

Nume= variabila de tip caracter, de lungime maxima 20 ce reprezinta numele angajatului

Prenume = variabila de tip caracter, de lungime maxima 20 ce reprezinta prenumele angajatului

Salariu = variabila de tip intreg, de lungime maxima 3 ce reprezinta salariu lunar al angajatului

Cod\_magazin = variabila de tip number, de lungime maxima 4 ce reprezinta codul magazinului; aceasta este o cheie externa catre coloana cod\_magazin din tabelul MAGAZIN

Cod\_angajat = variabila de tip number, de lungime maxima 4 ce reprezinta codul angajatului

*Subentitate CASIER are ca atribute:*

Uniforma = variabila de tip caracter, de lungime maxima 3 ce reprezinta codul specific tipului de uniforma

produse\_vandute = variabila de tip intreg, de lungime maxima 3 ce reprezinta numarul total de produse vandute

cod\_casier = variabila de tip number, de lungime maxima 4 ce reprezinta codul casierului

Cod\_angajat = variabila de tip number, de lungime maxima 4 ce reprezinta codul angajatului; aceasta este o cheie externa catre coloana cod\_angajat din tabelul ANGAJAT

*Subentitate PAZNIC are ca atribute:*

Denumire\_firma = variabila de tip caracter, de lungime maxima 20 ce reprezinta denumirea firmei de paza

Cod\_paza = variabila de tip number, de lungime maxima 4 ce reprezinta codul paznicului

Cod\_angajat = variabila de tip number, de lungime maxima 4 ce reprezinta codul angajatului; aceasta este o cheie externa catre coloana cod\_angajat din tabelul ANGAJAT

*Subentitate TEHNICIAN are ca atribute:*

Vechime = variabila de tip intreg, de lungime maxima 2 ce reprezinta numarul de ani in care a profesat pe post de tehnician

Cod\_tehnician = variabila de tip number, de lungime maxima 4 ce reprezinta codul tehnicianului

Cod\_angajat = variabila de tip number, de lungime maxima 4 ce reprezinta codul angajatului; aceasta este o cheie externa catre coloana cod\_angajat din tabelul ANGAJAT

**Relatii:**

PIESA\_aprtine\_DEPOZIT = relatie care leaga entitatile PIESA si DEPOZIT (ce piese apartin unui depozit). Relatia are cardinalitate minima 1:1 si cardinalitate maxima M:1.

DISTRIBUITOR\_aprovizioneaza\_DEPOZIT = relatie care leaga entitatile DISTRIBUITOR si DEPOZIT (ce distribuitori aprovizioneaza depozitul). Relatia are cardinalitate minima 1:1 si cardinalitate maxima M:M.

CLIENT\_cumpara\_PRODUS = relatie care leaga entitatile CLIENT si PRODUS (ce produse poate cumpara un client). Relatia are cardinalitate minima 1:0 si cardinalitate maxima 1:M.

ANGAJAT\_lucreaza\_la\_MAGAZIN = relatie care leaga ANGAJAT si MAGAZIN (cine lucreaza in acel magazin).Relatia are cardinalitate minima 1:1 si cardinalitate maxima M:1.

CLIENT\_aduce\_DEVICE\_DEFECT = relatie care leaga CLIENT si DEVICE\_DEFECT (device-urile aduse de catre client). Relatia are cardinalitate minima 0:0 si cardinalitate maxima1:M.

MAGAZIN\_are\_ISTORIC = relatie dintre MAGAZIN si ISTORIC (ce istoric/situatie are magazinul).

Relatia are cardinalitate minima 1:1 si cardinalitate maxima 1:M.

PODUS\_exista\_in\_DEPOZIT\_al\_MAGAZIN = relatie de tip 3 ce leaga entitatile PRODUS, DEPOZIT si MAGAZIN(un produs poate exista intr-un depozit destinat unui magazin). Relatia are cardinalitate minima 1:1 si cardinalitate maxima M:M.

ANGAJAT\_is\_a\_CASIER = relatie de tip IS A care are cardinalitatea maximă 1:1 şi minimă 1:0. Un angajat trebuie sa fie casier

ANGAJAT\_is\_a\_TEHNICIAN = relatie de tip IS A care are cardinalitatea maximă 1:1 şi minimă 1:0. Un angajat trebuie sa fie tehnician

ANGAJAT\_is\_a\_PAZNIC = relatie de tip IS A care are cardinalitatea maximă 1:1 şi minimă 1:0. Un angajat trebuie sa fie paznic

**SCHEME RELATIONALE**

PIESA(cod\_piesa#, cod\_depozit#,denumire)

DEPOZIT(cod\_depozit#, denumire)

DISTRIBUITOR(cod\_distribuitor#,denumire,adresa,produs,cantitate)

PRODUS(cod\_produs#, cod\_client#, model,pret,cantitate)

MAGAZIN(cod\_magazin#,locatie,program,nr\_angajati)

ANGAJAT(cod\_angajat#, cod\_magazin#,nume,prenume,salariu)

CLIENT(cod\_client#, nume,prenume,device,email,adresa)

DEVICE\_DEFECT(cod\_device\_defect#, cod\_client#, cod\_magazin#, model,cost,data\_primire,data\_predare)

ISTORIC(cod\_istoric#, cod\_magazin#,data,incasari,taxe,salarii)

TEHNICIAN(cod\_tehnician#,cod\_angajat#,vechime)

CASIER(cod\_casier#,cod\_angajat#,uniforma,produse\_vandute)

PAZNIC(cod\_paznic#,cod\_angajat#,denumire\_firma)

APROVIZIONEAZA(cod\_depozit#,cod\_distribuitor#)

EXISTA(cod\_depozit#,cod\_magazin#,cod\_piesa#)

Diagrama E/R

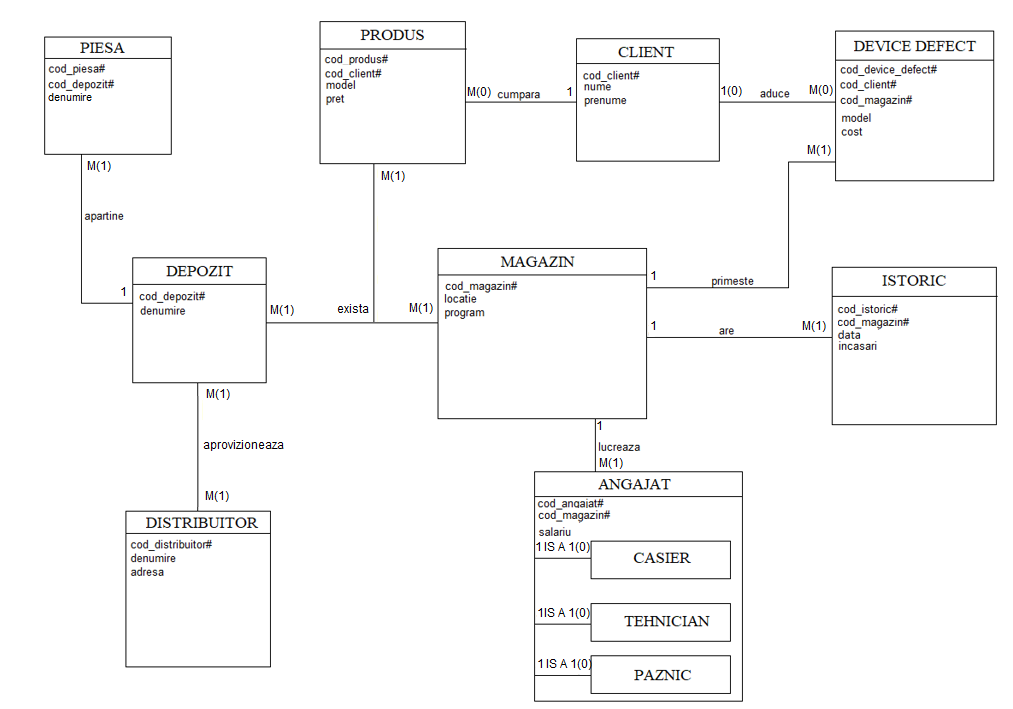
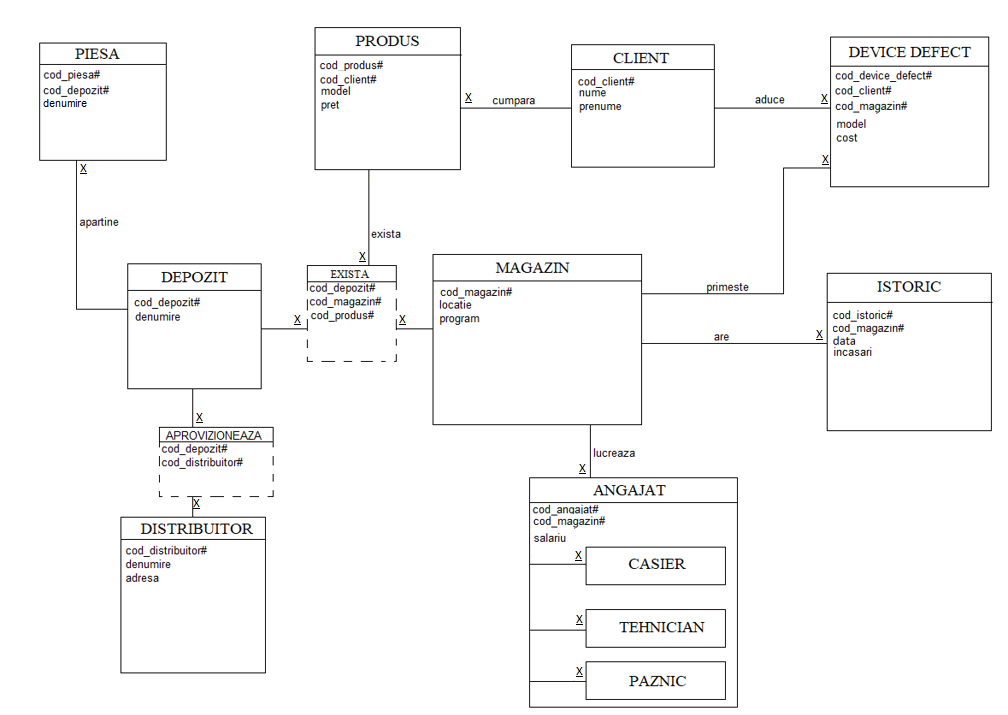


Diagrama Conceptuala



**Creare tabele si inserari date:**

CREATE TABLE CLIENTT

(nume VARCHAR(20),

prenume VARCHAR(20),

device VARCHAR(20),

email VARCHAR(20),

adresa VARCHAR(20),

cod\_client number(4) constraint pkey\_client primary key

);

CREATE TABLE PRODUS

(modell VARCHAR(20),

pret number(4),

cantitate number(4),

cod\_produs number(4) UNIQUE,

cod\_client number(4),

CONSTRAINT fkey\_produs\_client FOREIGN KEY(cod\_client) REFERENCES CLIENTT(cod\_client),

CONSTRAINT pkey\_produs PRIMARY KEY(cod\_produs,cod\_client)

);

CREATE TABLE DEPOZIT

(denumire VARCHAR(20),

cantitate number(4),

data\_primire date,

cod\_depozit number(4) constraint pkey\_depozit primary key

);

CREATE TABLE PIESA

(denumire VARCHAR(20),

cod\_piesa number(4),

cod\_depozit number(4),

CONSTRAINT fkey\_piesa\_depozit FOREIGN KEY(cod\_depozit) REFERENCES DEPOZIT(cod\_depozit),

CONSTRAINT pkey\_piesa PRIMARY KEY(cod\_piesa,cod\_depozit)

);

CREATE TABLE DISTRIBUITOR

(

denumire VARCHAR(20),

adresa VARCHAR(20),

produs VARCHAR(20),

cantitate number(4),

cod\_distribuitor number(4) constraint pkey\_distribuitor primary key

);

CREATE TABLE MAGAZIN

(cod\_magazin number(4) constraint pkey\_comp1 primary key,

locatie varchar(40),

programm VARCHAR(40),

nr\_angajati number(2)

);

CREATE TABLE ANGAJAT

(

nume VARCHAR(20),

prenume VARCHAR(20),

salariu number(4),

cod\_angajat number(4) UNIQUE,

cod\_magazin number(4),

CONSTRAINT fkey\_angajat\_magazin FOREIGN KEY(cod\_magazin) REFERENCES MAGAZIN(cod\_magazin),

CONSTRAINT pkey\_angajat PRIMARY KEY(cod\_angajat,cod\_magazin)

);

select \* from angajat;

CREATE TABLE CASIER

(

uniforma VARCHAR(20),

cod\_angajat number(4),

produse\_vandute NUMBER(4),

cod\_casier number(4),

CONSTRAINT fkey\_casier\_ang FOREIGN KEY(cod\_angajat) REFERENCES ANGAJAT(cod\_angajat),

CONSTRAINT pkey\_casier PRIMARY KEY(cod\_casier,cod\_angajat)

);

CREATE TABLE PAZNIC

(

denumire\_firma VARCHAR(20),

cod\_angajat number(4),

cod\_paza number(4),

CONSTRAINT fkey\_paznic\_ang FOREIGN KEY(cod\_angajat) REFERENCES ANGAJAT(cod\_angajat),

CONSTRAINT pkey\_paznic PRIMARY KEY(cod\_paza,cod\_angajat)

);

CREATE TABLE TEHNICIAN

(

vechime number(4),

cod\_angajat number(4),

cod\_tehnician number(4),

CONSTRAINT fkey\_tehn\_ang FOREIGN KEY(cod\_angajat) REFERENCES ANGAJAT(cod\_angajat),

CONSTRAINT pkey\_tehnician PRIMARY KEY(cod\_tehnician,cod\_angajat)

);

CREATE TABLE ISTORIC

(

dataa date,

incasari number(4),

taxe number(4),

salarii number(4),

cod\_istoric number(4),

cod\_magazin number(4),

CONSTRAINT fkey\_istoric\_magazin FOREIGN KEY(cod\_magazin) REFERENCES MAGAZIN(cod\_magazin),

CONSTRAINT pkey\_istoric PRIMARY KEY(cod\_istoric,cod\_magazin)

);

CREATE TABLE DEVICE\_DEFECT

(modell VARCHAR(20),

costt NUMBER(4),

data\_primire date,

data\_predare date,

cod\_device\_defect number(4),

cod\_client number(4),

cod\_magazin number(4),

CONSTRAINT fkey\_device\_defect\_client FOREIGN KEY(cod\_client) REFERENCES CLIENTT(cod\_client),

CONSTRAINT fkey\_device\_defect\_magazin FOREIGN KEY(cod\_magazin) REFERENCES MAGAZIN(cod\_magazin),

CONSTRAINT pkey\_device\_defect PRIMARY KEY(cod\_device\_defect,cod\_client,cod\_magazin)

);

CREATE TABLE APROVIZIONEAZA

(cod\_depozit number(4),

cod\_distribuitor number(4),

CONSTRAINT fkey\_apro\_dep FOREIGN KEY(cod\_depozit) REFERENCES DEPOZIT(cod\_depozit),

CONSTRAINT fkey\_apro\_dis FOREIGN KEY(cod\_distribuitor) REFERENCES DISTRIBUITOR(cod\_distribuitor),

CONSTRAINT pkey\_apro PRIMARY KEY(cod\_depozit,cod\_distribuitor)

);

CREATE TABLE EXISTA

(cod\_depozit number(4),

cod\_magazin number(4),

cod\_produs number(4),

CONSTRAINT fkey\_exista\_dep FOREIGN KEY(cod\_depozit) REFERENCES DEPOZIT(cod\_depozit),

CONSTRAINT fkey\_exista\_mag FOREIGN KEY(cod\_magazin) REFERENCES MAGAZIN(cod\_magazin),

CONSTRAINT fkey\_exista\_pro FOREIGN KEY(cod\_produs) REFERENCES PRODUS(cod\_produs),

CONSTRAINT pkey\_exista PRIMARY KEY(cod\_depozit,cod\_magazin,cod\_produs)

);

CREATE SEQUENCE SEQV

INCREMENT BY 1

START WITH 0

MINVALUE 0

MAXVALUE 100;

INSERT INTO MAGAZIN VALUES(1,'SOS Alexandria 344','L-V:7:00-22:00',5);

INSERT INTO MAGAZIN VALUES(2,'SOS Iuliu Maniu 3','L-V:8:00-22:00',6);

INSERT INTO MAGAZIN VALUES(3,'Strada Alexandru Vlahuta 99','L-V:7:00-22:00',2);

INSERT INTO MAGAZIN VALUES(4,'SOS Ion Mihalache 23A','L-V:7:00-22:00',3);

INSERT INTO MAGAZIN VALUES(5,'Strada Paris 10','L-V:8:00-22:00',3);

INSERT INTO PIESA VALUES('display amoled 9inch',1,2);

INSERT INTO PIESA VALUES('difuzor SAMSUNG A51',2,3);

INSERT INTO PIESA VALUES('RAMA NOKIA 1200',3,4);

INSERT INTO PIESA VALUES('Geam Galaxy A71',4,1);

INSERT INTO PIESA VALUES('Conector Audio',5,5);

INSERT INTO CLIENTT VALUES('Ion','Marian','Samsung Galaxy A71','ion.pop91@gmail.com','Sos.Giurgiu 7, bl5',SEQV.nextval);

INSERT INTO CLIENTT VALUES('Iliescu','Maria','Samsung Galaxy A51','maria2000@gmail.com','Sos.Alexandru Voda 1',SEQV.nextval);

INSERT INTO CLIENTT VALUES('Dragomir','Florin','Nokia 7 PLUS','dragomir@gmail.com','strada Orizont, bl4 ',SEQV.nextval);

INSERT INTO CLIENTT VALUES('Hahaianu','Lorena','Iphone SE','lorena@gmail.com','Calea Rahovei 7',SEQV.nextval);

INSERT INTO CLIENTT VALUES('Cartan','Robert','Xiaomi Redmi 9A','robert@gmail.com','Strada Valcea 17',SEQV.nextval);

INSERT INTO PRODUS VALUES('Xiaomi Redmi 9A',1299,23,1,2);

INSERT INTO PRODUS VALUES('Samsung Galaxy A51',1599,13,2,2);

INSERT INTO PRODUS VALUES('Telefon Apple 12',3299,3,3,1);

INSERT INTO PRODUS VALUES('Telefon Nokia 216',900,43,4,3);

INSERT INTO PRODUS VALUES('Telefon mobil Huawei',2299,2,5,4);

INSERT INTO DEVICE\_DEFECT VALUES('Samsung Galaxy A71',200,TO\_DATE('20-05-2021','DD-MM-YYYY'),TO\_DATE('23-05-2021','DD-MM-YYYY'),1,2,2);

INSERT INTO DEVICE\_DEFECT VALUES('Xiaomi Redmi 9A',100,TO\_DATE('10-05-2021','DD-MM-YYYY'),TO\_DATE('13-05-2021','DD-MM-YYYY'),2,3,4);

INSERT INTO DEVICE\_DEFECT VALUES('Iphone SE',300,TO\_DATE('5-05-2021','DD-MM-YYYY'),TO\_DATE('13-05-2021','DD-MM-YYYY'),3,5,2);

INSERT INTO DEVICE\_DEFECT VALUES('Iphone SE',600,TO\_DATE('1-05-2021','DD-MM-YYYY'),TO\_DATE('8-05-2021','DD-MM-YYYY'),4,1,4);

INSERT INTO DEVICE\_DEFECT VALUES('Nokia 7 PLUS',50,TO\_DATE('13-05-2021','DD-MM-YYYY'),TO\_DATE('13-05-2021','DD-MM-YYYY'),5,5,5);

INSERT INTO ISTORIC VALUES (TO\_DATE('10-01-2021','DD-MM-YYYY'),8900,800,4000,1,2);

INSERT INTO ISTORIC VALUES (TO\_DATE('10-02-2021','DD-MM-YYYY'),8700,800,4000,2,2);

INSERT INTO ISTORIC VALUES (TO\_DATE('10-03-2021','DD-MM-YYYY'),9900,980,4200,3,3);

INSERT INTO ISTORIC VALUES (TO\_DATE('10-04-2021','DD-MM-YYYY'),9900,980,4200,4,4);

INSERT INTO ISTORIC VALUES (TO\_DATE('10-05-2021','DD-MM-YYYY'),9990,1100,4500,5,4);

INSERT INTO DEPOZIT VALUES ('RAMA NOKIA 1200',400,TO\_DATE('12-06-2018','DD-MM-YYYY'),1);

INSERT INTO DEPOZIT VALUES ('difuzor SAMSUNG ',30,TO\_DATE('12-09-2018','DD-MM-YYYY'),2);

INSERT INTO DEPOZIT VALUES ('Geam Samsung Galaxy',450,TO\_DATE('22-06-2018','DD-MM-YYYY'),3);

INSERT INTO DEPOZIT VALUES ('Conector Audio ',300,TO\_DATE('12-02-2018','DD-MM-YYYY'),4);

INSERT INTO DEPOZIT VALUES ('display 9inch',40,TO\_DATE('22-07-2019','DD-MM-YYYY'),5);

INSERT INTO DISTRIBUITOR VALUES('ROMPRIX','SOS Independentei 7','Geam Camera Spate',100,1);

INSERT INTO DISTRIBUITOR VALUES('protel','SOS Independentei 17','display amoled 9in',100,2);

INSERT INTO DISTRIBUITOR VALUES('GSMGX','SOS Colentina 12A','difuzor SAMSUNG A51',100,3);

INSERT INTO DISTRIBUITOR VALUES('APPLE','Sos Raului 10','display Iphone SE',100,4);

INSERT INTO DISTRIBUITOR VALUES('NOKIA','SOS Floreasca 7','RAMA NOKIA 1200',100,5);

INSERT INTO ANGAJAT VALUES ('Ion','Marian',1200,1,2);

INSERT INTO ANGAJAT VALUES ('Popescu','Marian',1800,2,2);

INSERT INTO ANGAJAT VALUES ('Flavius','Maria',2000,3,3);

INSERT INTO ANGAJAT VALUES ('Chitaru','Roxana',1200,4,3);

INSERT INTO ANGAJAT VALUES ('Berescu','Andi',2200,5,1);

INSERT INTO ANGAJAT VALUES ('Georgescu','Serban',2200,6,1);

INSERT INTO ANGAJAT VALUES ('Iliescu','Alexandru',2200,7,4);

INSERT INTO ANGAJAT VALUES ('Onicescu','Bogdan',2200,8,4);

INSERT INTO ANGAJAT VALUES ('Minoda','Victor',2200,9,5);

INSERT INTO ANGAJAT VALUES ('Nita','George',2200,10,5);

INSERT INTO ANGAJAT VALUES ('Anton','Cristina',2200,11,1);

INSERT INTO ANGAJAT VALUES ('Jelea','Mihai',2200,12,2);

INSERT INTO ANGAJAT VALUES ('Turcu','Vanessa',2200,13,1);

INSERT INTO ANGAJAT VALUES ('Diaconescu','Cristian',2200,14,4);

INSERT INTO ANGAJAT VALUES ('Popa','Robert',2200,15,5);

INSERT INTO CASIER VALUES('TIP-2',1,230,1);

INSERT INTO CASIER VALUES('TIP-1',2,0,2);

INSERT INTO CASIER VALUES('TIP-1',3,0,3);

INSERT INTO CASIER VALUES('TIP-3',4,330,4);

INSERT INTO CASIER VALUES('TIP-2',5,180,5);

INSERT INTO PAZNIC VALUES('BGS',6,1);

INSERT INTO PAZNIC VALUES('BGS',7,2);

INSERT INTO PAZNIC VALUES('BGS',8,3);

INSERT INTO PAZNIC VALUES('BGS',9,4);

INSERT INTO PAZNIC VALUES('BGS',10,5);

INSERT INTO TEHNICIAN VALUES(30,11,1);

INSERT INTO TEHNICIAN VALUES(5,12,2);

INSERT INTO TEHNICIAN VALUES(6,13,3);

INSERT INTO TEHNICIAN VALUES(7,14,4);

INSERT INTO TEHNICIAN VALUES(8,15,5);

INSERT INTO APROVIZIONEAZA VALUES(1,2);

INSERT INTO APROVIZIONEAZA VALUES(2,5);

INSERT INTO APROVIZIONEAZA VALUES(3,3);

INSERT INTO APROVIZIONEAZA VALUES(4,1);

INSERT INTO APROVIZIONEAZA VALUES(5,4);

INSERT INTO APROVIZIONEAZA VALUES(1,3);

INSERT INTO APROVIZIONEAZA VALUES(2,4);

INSERT INTO APROVIZIONEAZA VALUES(3,5);

INSERT INTO APROVIZIONEAZA VALUES(4,2);

INSERT INTO APROVIZIONEAZA VALUES(5,5);

INSERT INTO EXISTA VALUES(1,4,2);

INSERT INTO EXISTA VALUES(2,2,4);

INSERT INTO EXISTA VALUES(3,5,3);

INSERT INTO EXISTA VALUES(4,3,5);

INSERT INTO EXISTA VALUES(5,1,1);

INSERT INTO EXISTA VALUES(1,2,1);

INSERT INTO EXISTA VALUES(2,4,5);

INSERT INTO EXISTA VALUES(3,2,1);

INSERT INTO EXISTA VALUES(4,1,4);

INSERT INTO EXISTA VALUES(5,2,3);

COMMIT;

6. sa se afizeze locatia,programul si data istoricului pentru magazinele ale caror depozite sunt aprovizionate de un distribuitor a carui denumire se aseamana cu un cuv primit.

CREATE OR REPLACE PROCEDURE p6

(v\_den distribuitor.denumire%TYPE)

IS

TYPE locatie IS TABLE OF magazin.locatie%TYPE INDEX BY PLS\_INTEGER;

TYPE programm IS TABLE OF magazin.programm%TYPE INDEX BY PLS\_INTEGER;

TYPE dataa IS TABLE OF istoric.dataa%TYPE INDEX BY PLS\_INTEGER;

list1 locatie;

list2 programm;

list3 dataa;

BEGIN

SELECT DISTINCT m.locatie, m.programm, i.dataa

BULK COLLECT INTO list1,list2,list3

FROM MAGAZIN m, istoric i

WHERE i.cod\_magazin = m.cod\_magazin

AND

m.cod\_magazin =(SELECT cod\_magazin

FROM EXISTA

WHERE cod\_depozit=(SELECT cod\_depozit

FROM APROVIZIONEAZA

WHERE cod\_distribuitor=(SELECT cod\_distribuitor

FROM distribuitor

WHERE denumire like '%'||v\_den||'%')));

FOR j in list1.FIRST..list1.LAST loop

DBMS\_OUTPUT.PUT\_LINE(list1(j)||' '||list2(j)||' '||list3(j));

END LOOP;

exception

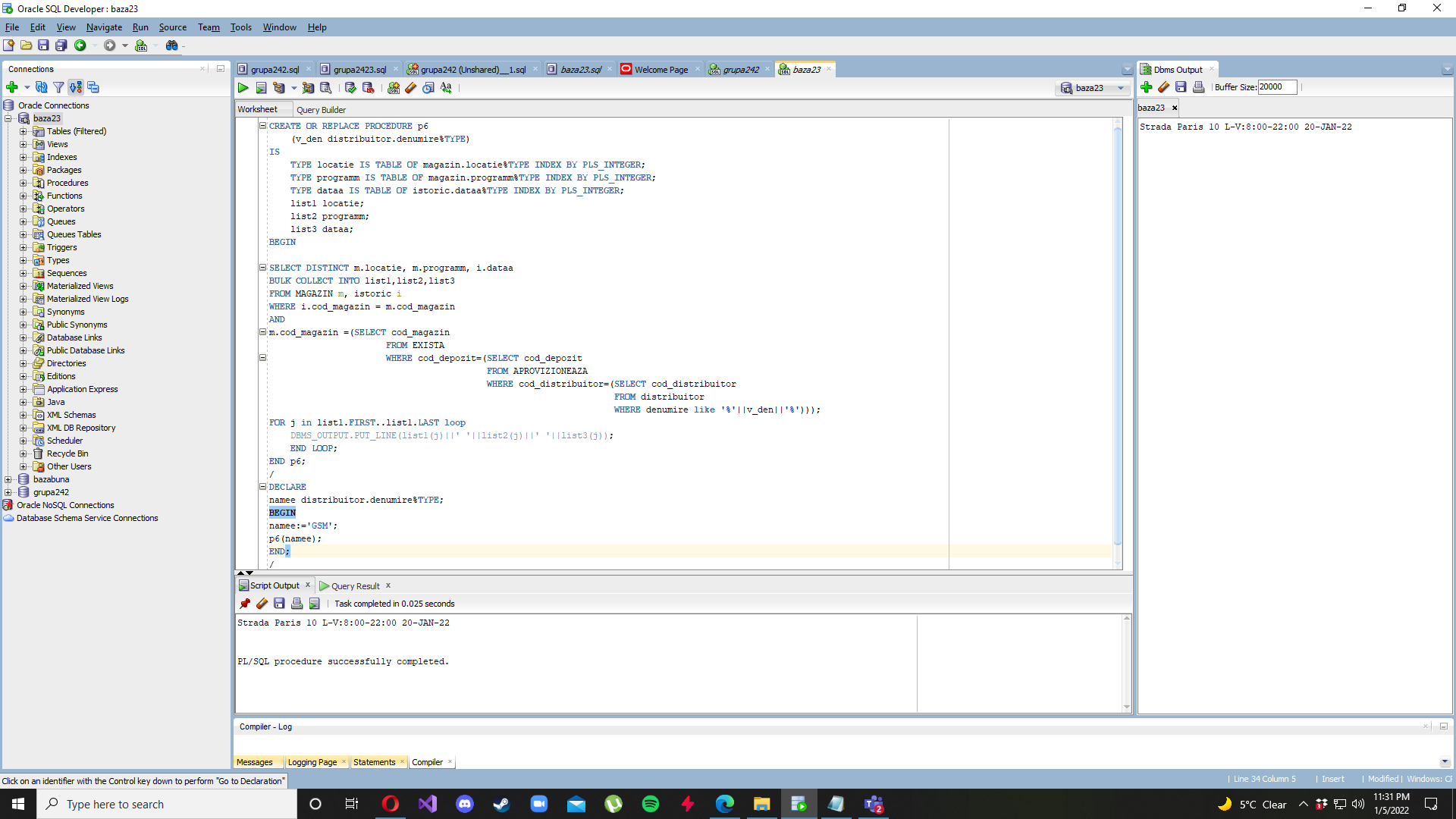
WHEN NO\_DATA\_FOUND THEN

RAISE\_APPLICATION\_ERROR(-20000,'Nu exista date pentru acest distribuitor');

WHEN OTHERS THEN

raise\_application\_error(-20002, 'Alta eroare!');

END p6;



7.Sa se afiseze numele,prenumele si salariul angajatilor ale caror nume de familie se termina in 'escu', sunt casieri si au salariu cel putn 1800. Ordonati descrescator dupa salariu

CREATE OR REPLACE PROCEDURE p7

IS

v\_nume angajat.nume%TYPE;

v\_prenume angajat.prenume%TYPE;

v\_salariu angajat.salariu%TYPE;

CURSOR c IS

SELECT nume, prenume, salariu

FROM angajat

WHERE cod\_angajat IN (SELECT cod\_angajat

FROM casier)

AND NVL(salariu,0)>1800

AND UPPER(nume) LIKE '%ESCU'

ORDER BY salariu DESC;

BEGIN

OPEN c;

LOOP

FETCH c INTO v\_nume,v\_prenume,v\_salariu;

EXIT WHEN c%NOTFOUND;

DBMS\_OUTPUT.PUT\_LINE(v\_nume||' '||v\_prenume||' este casier si are salariul '||v\_salariu);

END LOOP;

CLOSE C;

exception

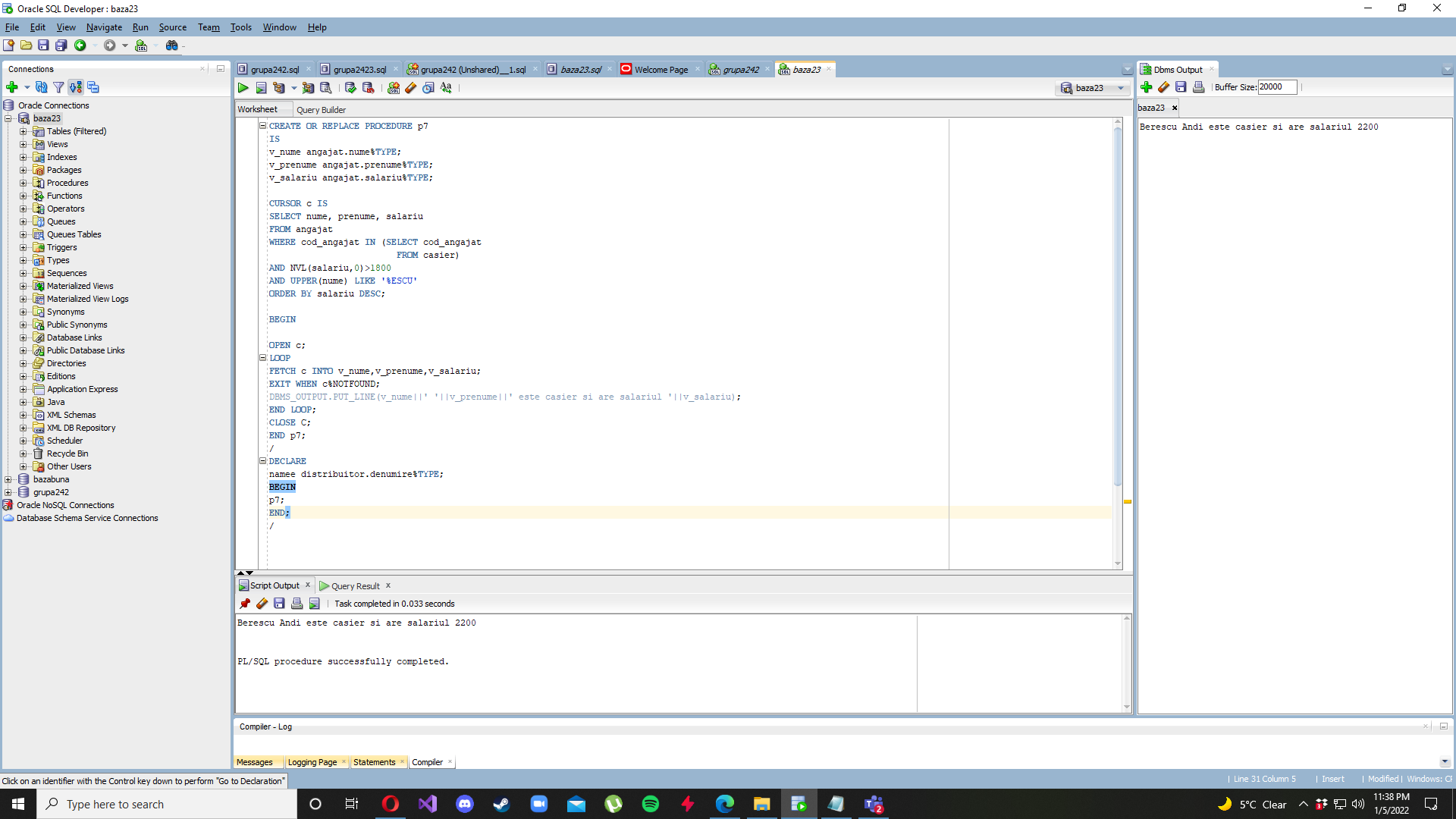
WHEN NO\_DATA\_FOUND THEN

RAISE\_APPLICATION\_ERROR(-20000,'Nu exista date');

WHEN OTHERS THEN

raise\_application\_error(-20002, 'Alta eroare!');

END p7;



8.Sa se afiseze incasarile maxime ale unui magazin situat intr-o anumita locatie in care salariul mediu al angajatilor depaseste 1000 de lei, iar casierii au uniforme de tip 2

CREATE OR REPLACE FUNCTION p8

(v\_locatie magazin.locatie%TYPE)

RETURN NUMBER IS

v\_incasari istoric.incasari%TYPE;

BEGIN

SELECT max(i.incasari)

INTO v\_incasari

FROM istoric i,magazin m,(SELECT AVG(salariu) sal, a.cod\_magazin

FROM ANGAJAT a,casier c

WHERE a.cod\_angajat = c.cod\_angajat

AND c.uniforma LIKE 'TIP-2'

GROUP BY a.cod\_magazin)ac

WHERE ac.sal>1000

AND m.cod\_magazin = i.cod\_magazin

AND m.locatie=v\_locatie

AND ac.cod\_magazin=m.cod\_magazin;

RETURN v\_incasari;

EXCEPTION

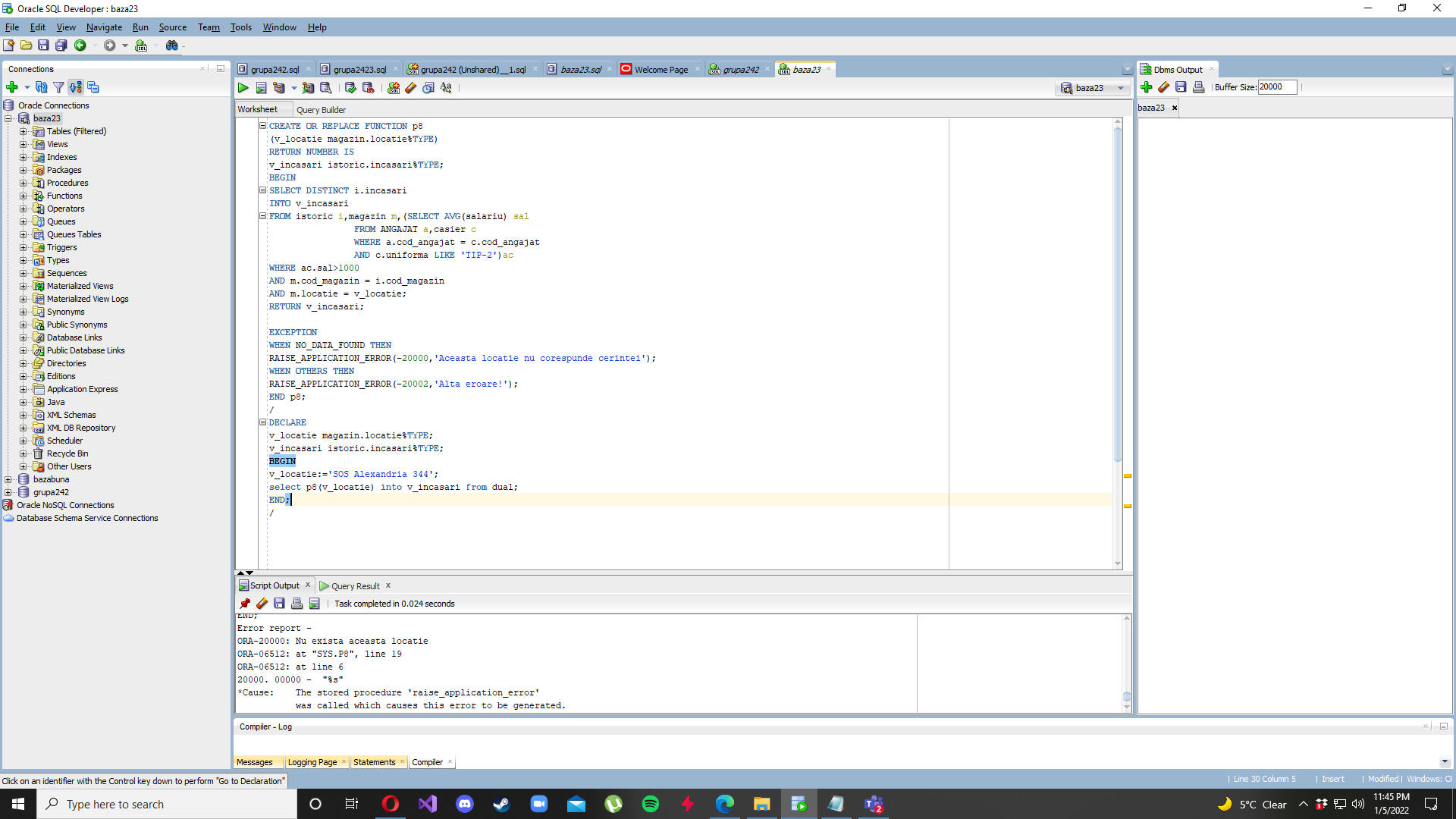
WHEN NO\_DATA\_FOUND THEN

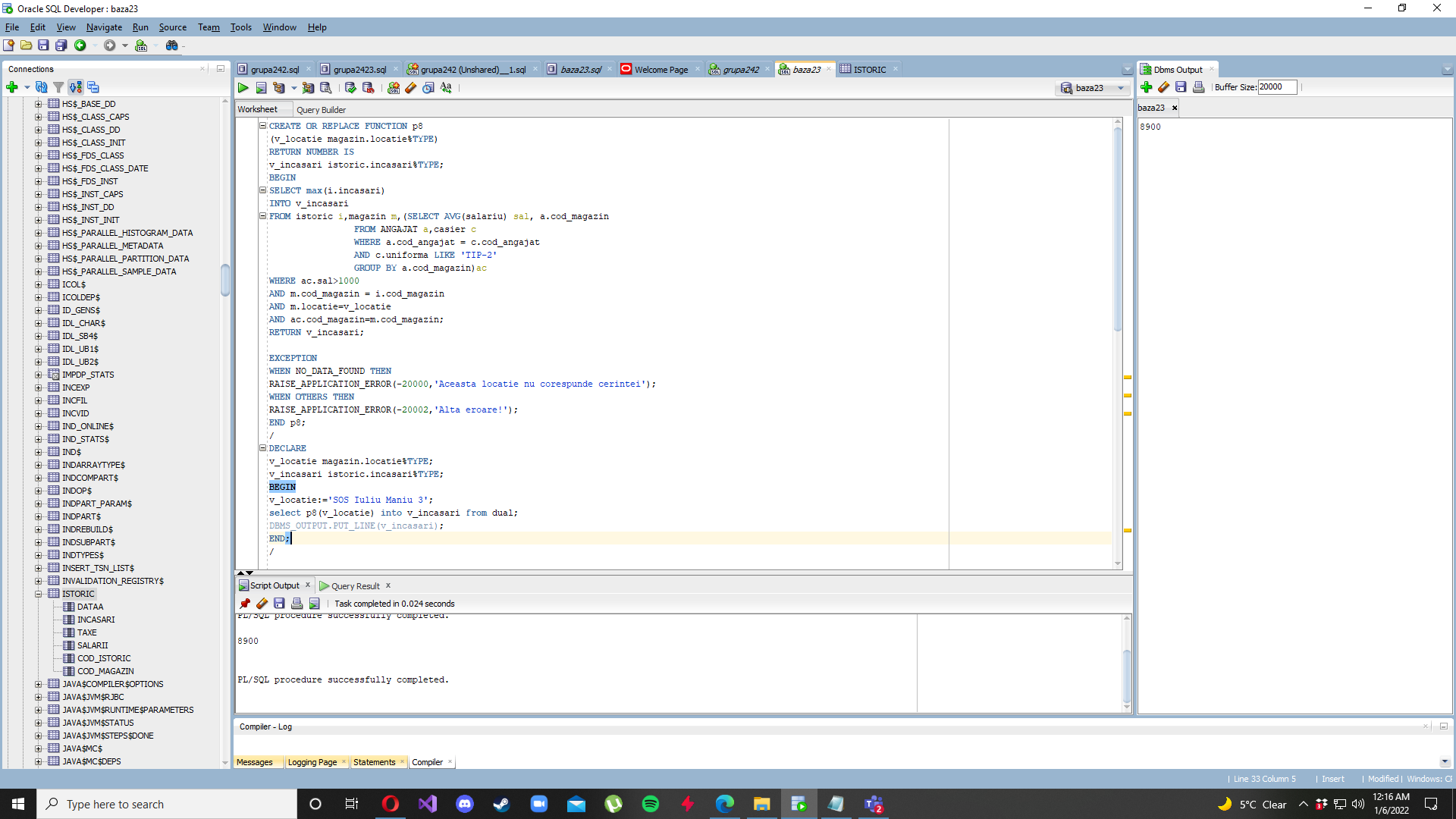
RAISE\_APPLICATION\_ERROR(-20000,'Aceasta locatie nu corespunde cerintei');

WHEN OTHERS THEN

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

END p8;





9.Sa se afiseze toate magazinele care au incasari mai mari de 10000 lei,

in care angajatii castiga un salariu mai mare de 1350 lei, dar mai mic fata de o valoare data

impreuna cu toate device-urile primite de la clienti cat si numele si prenumele acestora

CREATE OR REPLACE PROCEDURE p9

(maxi angajat.salariu%TYPE)

IS

TYPE locatie IS TABLE OF magazin.locatie%TYPE INDEX BY PLS\_INTEGER;

TYPE nume IS TABLE OF clientt.nume%TYPE;

TYPE prenume IS TABLE OF clientt.prenume%TYPE;

TYPE modell IS TABLE OF device\_defect.modell%TYPE;

v\_locatie locatie;

v\_nume nume;

v\_prenume prenume;

v\_model modell;

--11

BEGIN

SELECT m.locatie,c.prenume,c.nume,d.modell

BULK COLLECT INTO v\_locatie, v\_prenume, v\_nume, v\_model

FROM clientt c,magazin m JOIN istoric i ON(m.cod\_magazin=i.cod\_magazin)

JOIN angajat a ON(m.cod\_magazin=a.cod\_magazin)

JOIN device\_defect d ON(m.cod\_magazin=d.cod\_magazin)

WHERE c.cod\_client=d.cod\_client

AND i.incasari >1000

AND a.salariu > 1350 AND a.salariu<maxi;

FOR i in v\_locatie.FIRST..v\_locatie.LAST loop

DBMS\_OUTPUT.PUT\_LINE(v\_locatie(i)||' '||v\_prenume(i)||' '||v\_nume(i)||' '||v\_model(i));

END LOOP;

EXCEPTION

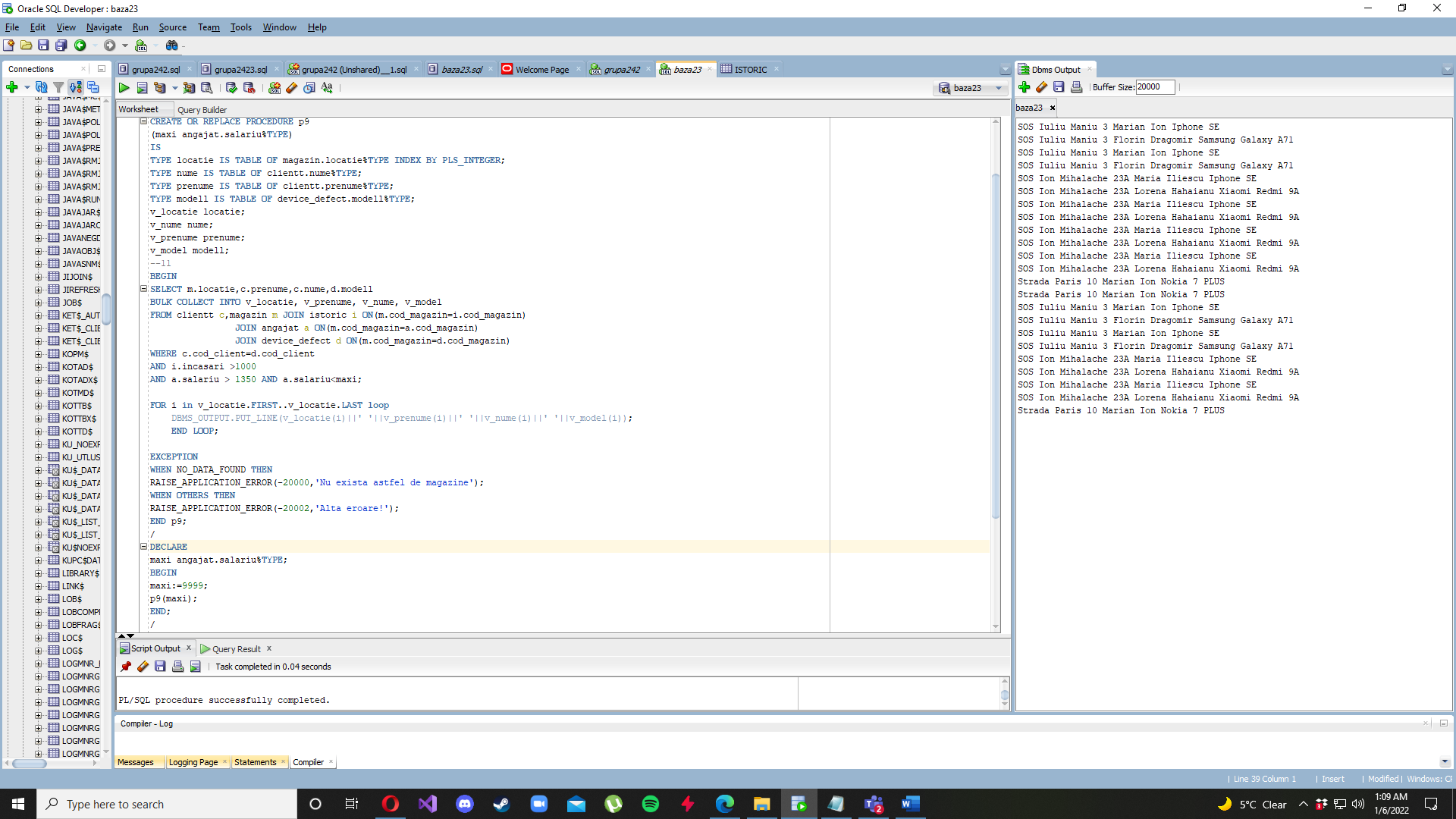
WHEN NO\_DATA\_FOUND THEN

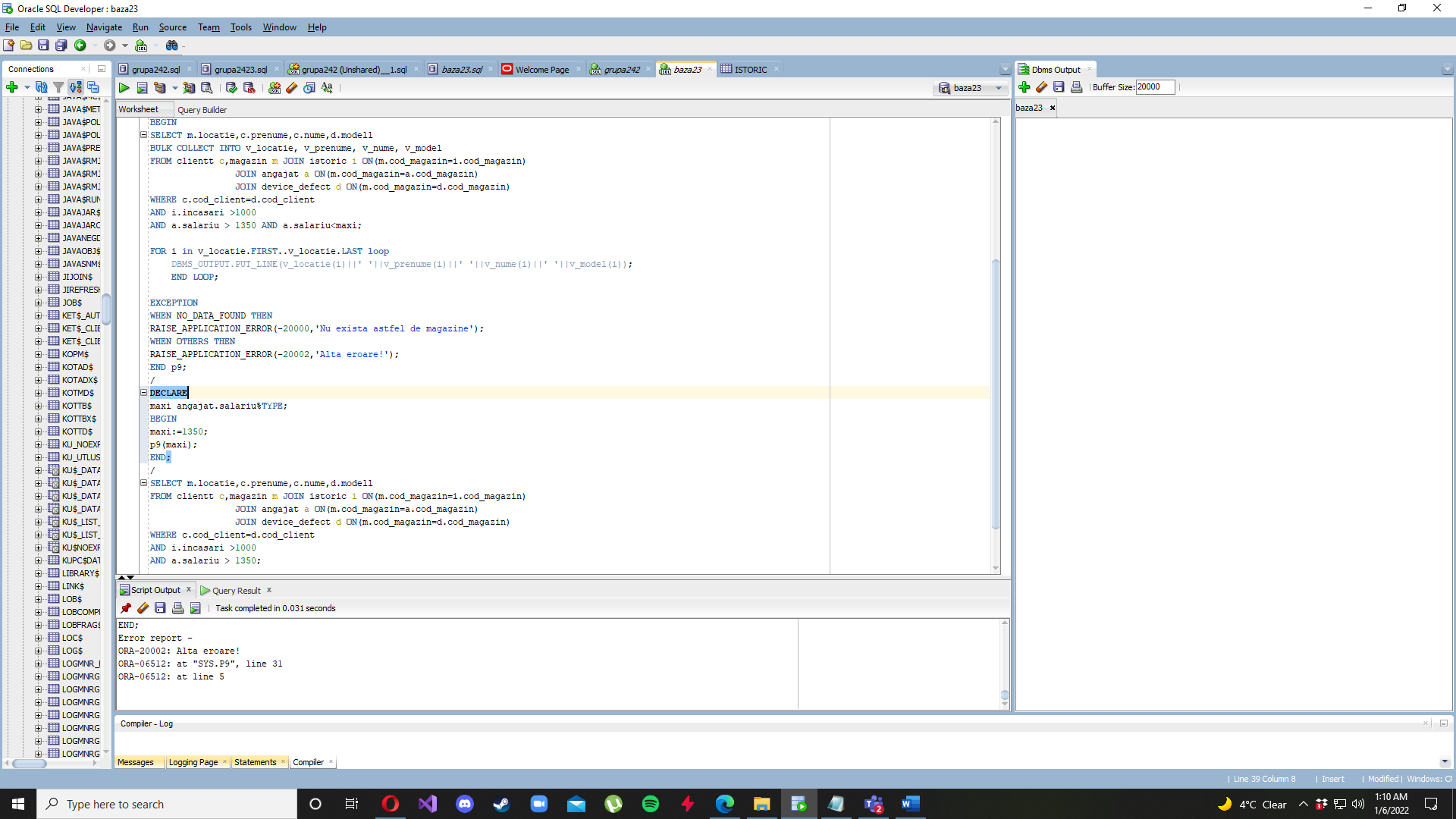
RAISE\_APPLICATION\_ERROR(-20000,'Nu exista astfel de magazine');

WHEN OTHERS THEN

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

END p9;





/\*Sa se creeze un trigger care sa opreasca comenzile de insert,update sau delete care se executa in tabelul magazin in afara programului\*/

CREATE OR REPLACE TRIGGER trig1 BEFORE

INSERT OR UPDATE OR DELETE ON magazin

BEGIN

IF ( to\_char(sysdate, 'D') = 1 ) OR (to\_char(sysdate, 'D') = 7) OR( to\_char(sysdate, 'HH24') NOT BETWEEN 7 AND 22 ) THEN

raise\_application\_error(-20001, 'Modificarile pot fi facute doar in timpul programului de functionare');

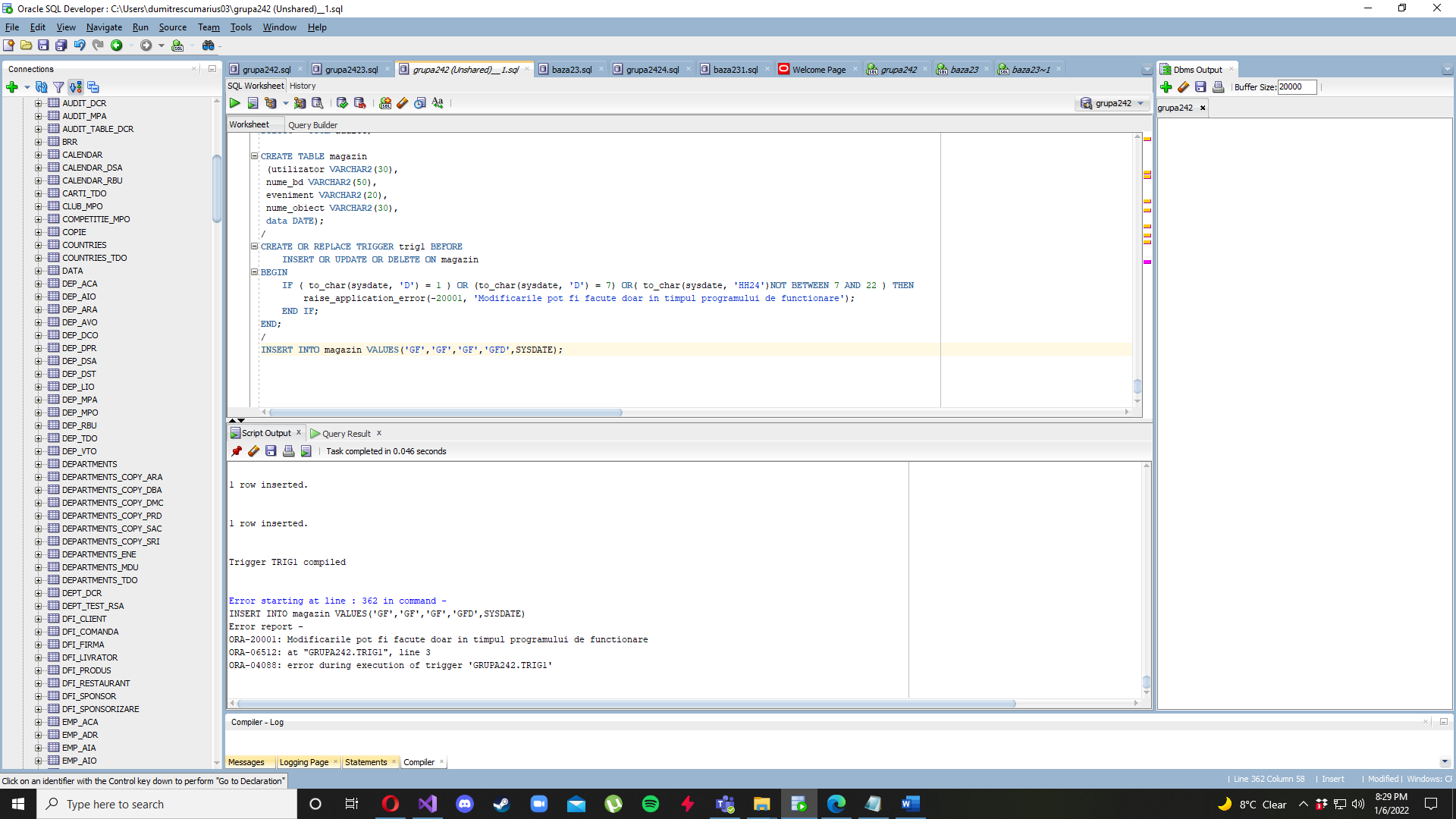
END IF;

END;

/

DROP TRIGGER trig1\_1;

/



/\*Sa se creeze un trigger care sa opreasca comenzile de insert,update sau delete care se executa in tabelul magazin in afara programului\*/

CREATE OR REPLACE TRIGGER trig2 BEFORE

INSERT OR UPDATE OF programm ON magazin

FOR EACH ROW

declare

ora\_i NUMBER;

ora\_f NUMBER;

v magazin.programm%TYPE;

punct1 NUMBER;

BEGIN

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

IF UPDATING OR DELETING THEN

v:=:OLD.programm;

ora\_i:=TO\_NUMBER(SUBSTR(v,5,1));

punct1:=INSTR(v,'-',3);

ora\_f:=TO\_NUMBER(SUBSTR(v,punct1+1,2));

IF ( to\_char(sysdate, 'D') = '1' ) OR (to\_char(sysdate, 'D') = '7') OR( TO\_NUMBER(to\_char(sysdate, 'HH24')) < ora\_i) OR( TO\_NUMBER(to\_char(sysdate, 'HH24'))>ora\_f ) THEN

raise\_application\_error(-20001, 'Modificarile pot fi facute doar in timpul programului de functionare');

END IF;

end if;

if inserting then

DBMS\_OUTPUT.PUT\_LINE('Magazinul a fost introdus cu succes. Acesta poate fi modificat exclusiv in timpul programului de functionare');

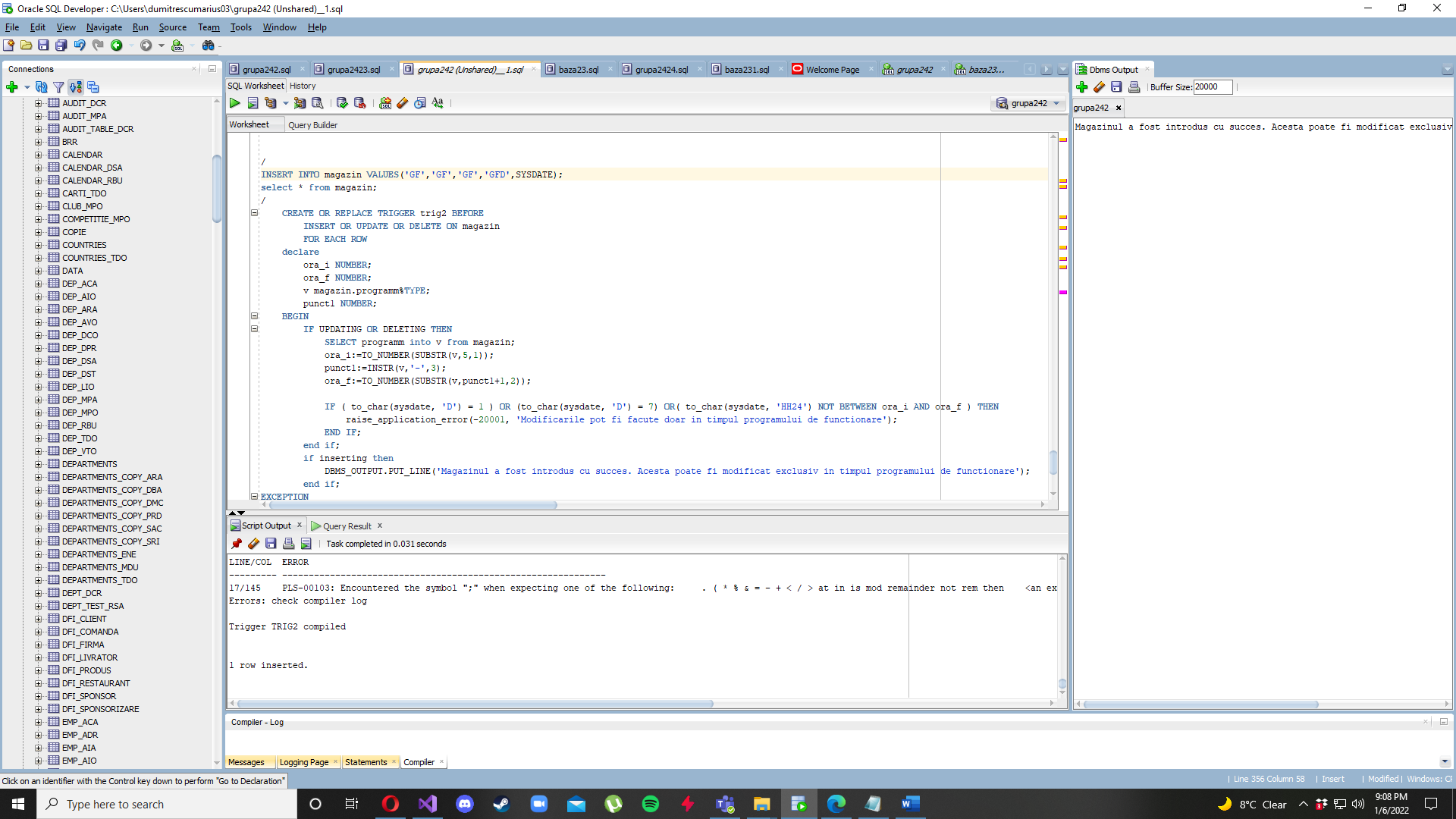
end if;

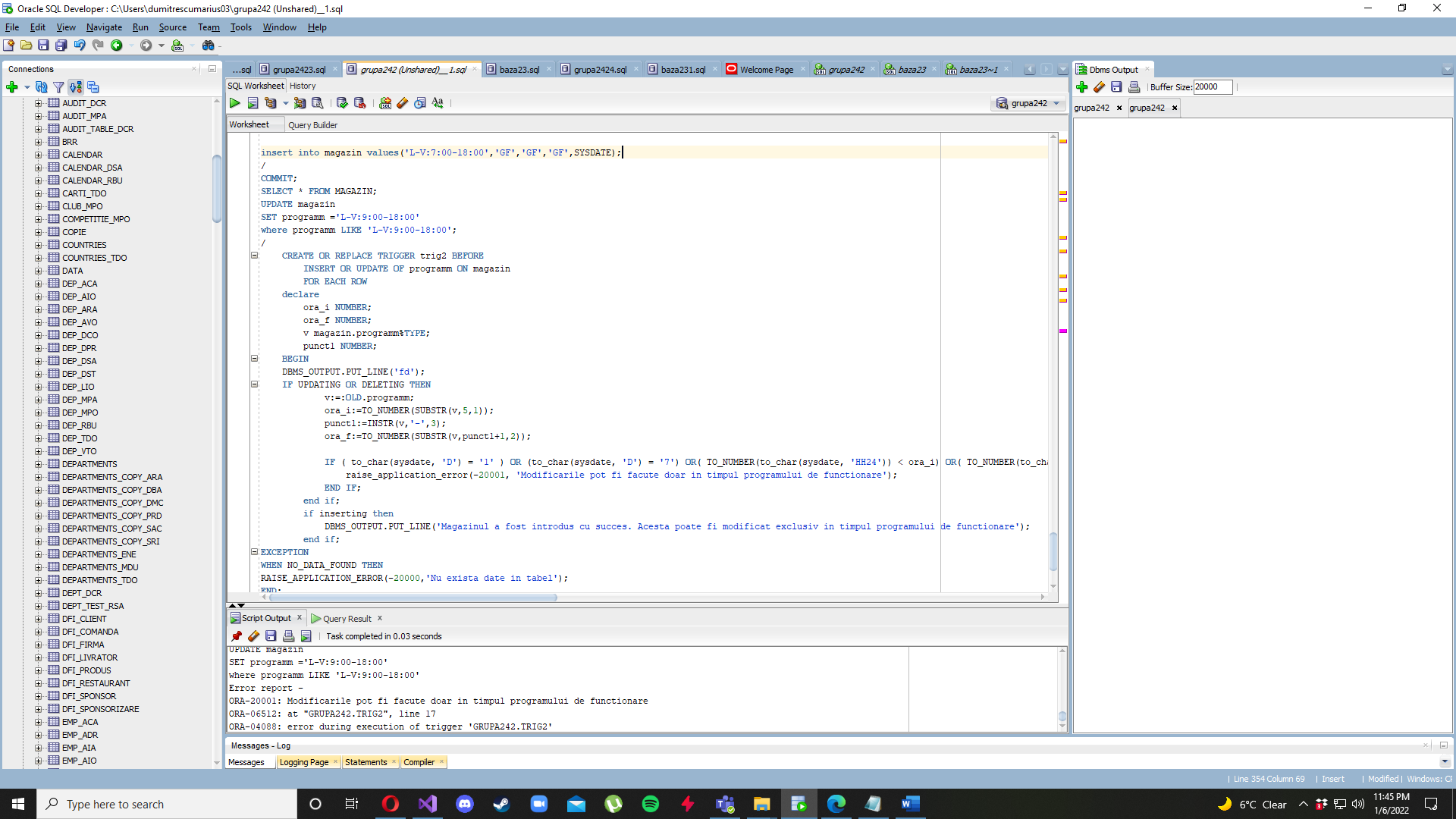
EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RAISE\_APPLICATION\_ERROR(-20000,'Nu exista date in tabel');

END;





/\*Sa se creeze un trigger care sa opreasca comenzile de create,drop sau alter care se executa in afara programului. In cazul in care se respecta programul 5-22, se vor insera in tabelul audit date referitoare la user,tabel,data etc;\*/

CREATE TABLE auditt

(utilizator VARCHAR2(30),

nume\_bd VARCHAR2(50),

eveniment VARCHAR2(20),

nume\_obiect VARCHAR2(30),

data DATE);

/

CREATE OR REPLACE TRIGGER trig3

BEFORE CREATE OR DROP OR ALTER ON SCHEMA

BEGIN

IF to\_number(to\_char(sysdate,'HH24'))>22 OR to\_number(to\_char(sysdate,'HH24'))<5 then

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

RAISE\_APPLICATION\_ERROR(-20002,'Nu sunt permise astfel de modificari in acest interval orar. Reveniti dupa 5 AM!');

ELSE

INSERT INTO auditt

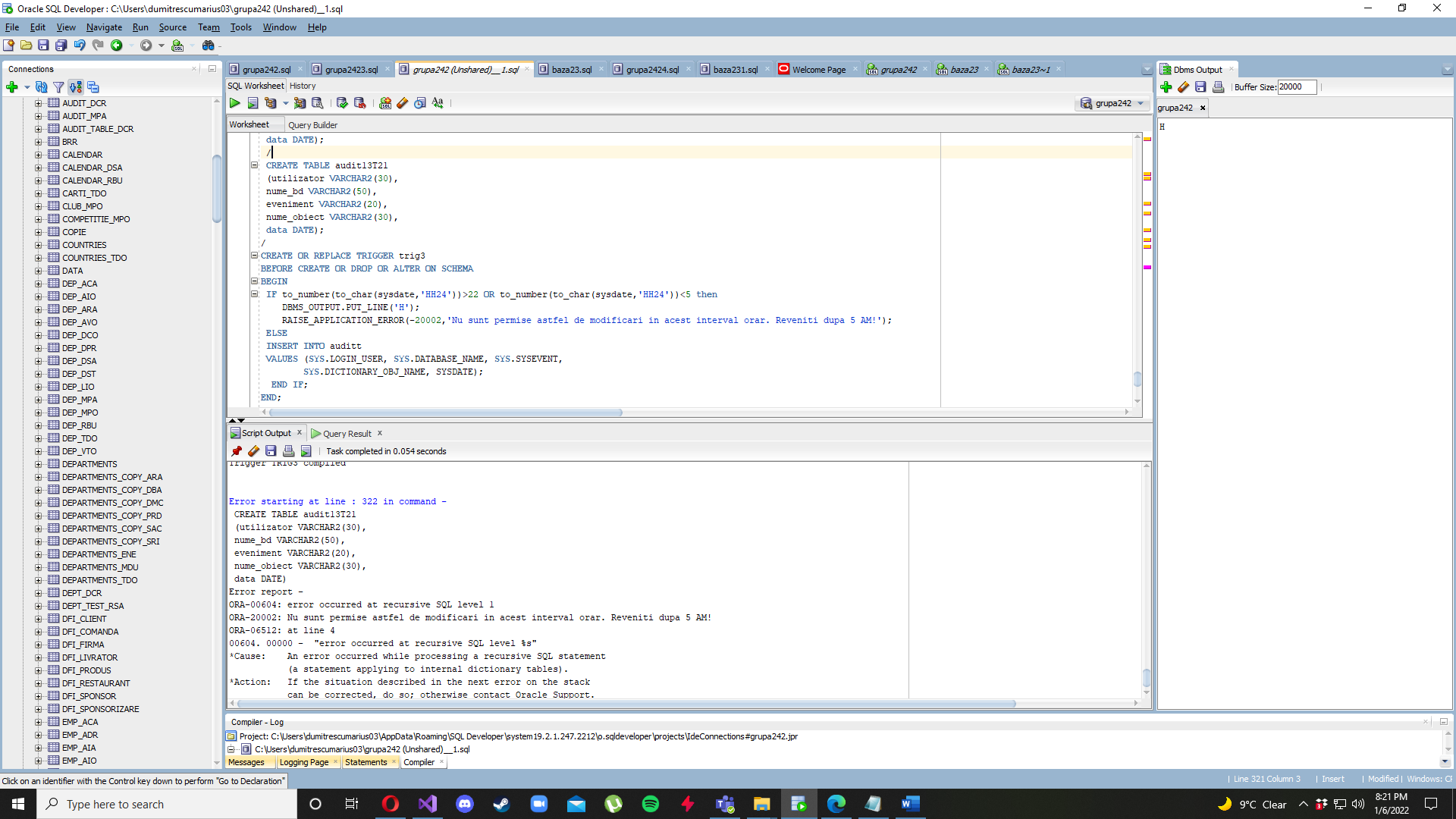
VALUES (SYS.LOGIN\_USER, SYS.DATABASE\_NAME, SYS.SYSEVENT,

SYS.DICTIONARY\_OBJ\_NAME, SYSDATE);

END IF;

END;

/



13.PACHET

CREATE OR REPLACE PACKAGE pachet AS

PROCEDURE p6

(v\_den distribuitor.denumire%TYPE);

PROCEDURE p7;

FUNCTION p8

(v\_locatie magazin.locatie%TYPE) RETURN NUMBER;

PROCEDURE p9

(maxi angajat.salariu%TYPE);

END pachet;

/

CREATE OR REPLACE PACKAGE BODY pachet AS

PROCEDURE p6

(v\_den distribuitor.denumire%TYPE)

IS

TYPE locatie IS TABLE OF magazin.locatie%TYPE INDEX BY PLS\_INTEGER;

TYPE programm IS TABLE OF magazin.programm%TYPE INDEX BY PLS\_INTEGER;

TYPE dataa IS TABLE OF istoric.dataa%TYPE INDEX BY PLS\_INTEGER;

list1 locatie;

list2 programm;

list3 dataa;

BEGIN

SELECT DISTINCT m.locatie, m.programm, i.dataa

BULK COLLECT INTO list1,list2,list3

FROM MAGAZIN m, istoric i

WHERE i.cod\_magazin = m.cod\_magazin

AND

m.cod\_magazin =(SELECT cod\_magazin

FROM EXISTA

WHERE cod\_depozit=(SELECT cod\_depozit

FROM APROVIZIONEAZA

WHERE cod\_distribuitor=(SELECT cod\_distribuitor

FROM distribuitor

WHERE denumire like '%'||v\_den||'%')));

FOR j in list1.FIRST..list1.LAST loop

DBMS\_OUTPUT.PUT\_LINE(list1(j)||' '||list2(j)||' '||list3(j));

END LOOP;

exception

WHEN NO\_DATA\_FOUND THEN

RAISE\_APPLICATION\_ERROR(-20000,'Nu exista date pentru acest distribuitor');

WHEN OTHERS THEN

raise\_application\_error(-20002, 'Alta eroare!');

END p6;

------------

PROCEDURE p7

IS

v\_nume angajat.nume%TYPE;

v\_prenume angajat.prenume%TYPE;

v\_salariu angajat.salariu%TYPE;

CURSOR c IS

SELECT nume, prenume, salariu

FROM angajat

WHERE cod\_angajat IN (SELECT cod\_angajat

FROM casier)

AND NVL(salariu,0)>1800

AND UPPER(nume) LIKE '%ESCU'

ORDER BY salariu DESC;

BEGIN

OPEN c;

LOOP

FETCH c INTO v\_nume,v\_prenume,v\_salariu;

EXIT WHEN c%NOTFOUND;

DBMS\_OUTPUT.PUT\_LINE(v\_nume||' '||v\_prenume||' este casier si are salariul '||v\_salariu);

END LOOP;

CLOSE C;

exception

WHEN NO\_DATA\_FOUND THEN

RAISE\_APPLICATION\_ERROR(-20000,'Nu exista date');

WHEN OTHERS THEN

raise\_application\_error(-20002, 'Alta eroare!');

END p7;

---------

FUNCTION p8

(v\_locatie magazin.locatie%TYPE)

RETURN NUMBER IS

v\_incasari istoric.incasari%TYPE;

BEGIN

SELECT max(i.incasari)

INTO v\_incasari

FROM istoric i,magazin m,(SELECT AVG(salariu) sal, a.cod\_magazin

FROM ANGAJAT a,casier c

WHERE a.cod\_angajat = c.cod\_angajat

AND c.uniforma LIKE 'TIP-2'

GROUP BY a.cod\_magazin)ac

WHERE ac.sal>1000

AND m.cod\_magazin = i.cod\_magazin

AND m.locatie=v\_locatie

AND ac.cod\_magazin=m.cod\_magazin;

RETURN v\_incasari;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RAISE\_APPLICATION\_ERROR(-20000,'Aceasta locatie nu corespunde cerintei');

WHEN OTHERS THEN

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

END p8;

------------

PROCEDURE p9

(maxi angajat.salariu%TYPE)

IS

TYPE locatie IS TABLE OF magazin.locatie%TYPE INDEX BY PLS\_INTEGER;

TYPE nume IS TABLE OF clientt.nume%TYPE;

TYPE prenume IS TABLE OF clientt.prenume%TYPE;

TYPE modell IS TABLE OF device\_defect.modell%TYPE;

v\_locatie locatie;

v\_nume nume;

v\_prenume prenume;

v\_model modell;

--11

BEGIN

SELECT m.locatie,c.prenume,c.nume,d.modell

BULK COLLECT INTO v\_locatie, v\_prenume, v\_nume, v\_model

FROM clientt c,magazin m JOIN istoric i ON(m.cod\_magazin=i.cod\_magazin)

JOIN angajat a ON(m.cod\_magazin=a.cod\_magazin)

JOIN device\_defect d ON(m.cod\_magazin=d.cod\_magazin)

WHERE c.cod\_client=d.cod\_client

AND i.incasari >1000

AND a.salariu > 1350 AND a.salariu<maxi;

FOR i in v\_locatie.FIRST..v\_locatie.LAST loop

DBMS\_OUTPUT.PUT\_LINE(v\_locatie(i)||' '||v\_prenume(i)||' '||v\_nume(i)||' '||v\_model(i));

END LOOP;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RAISE\_APPLICATION\_ERROR(-20000,'Nu exista astfel de magazine');

WHEN OTHERS THEN

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

END p9;

end pachet;

/

declare

v\_locatie magazin.locatie%TYPE;

v\_incasari istoric.incasari%TYPE;

BEGIN

v\_locatie:='SOS Iuliu Maniu 3';

pachet.p6('GSM');

pachet.p7;

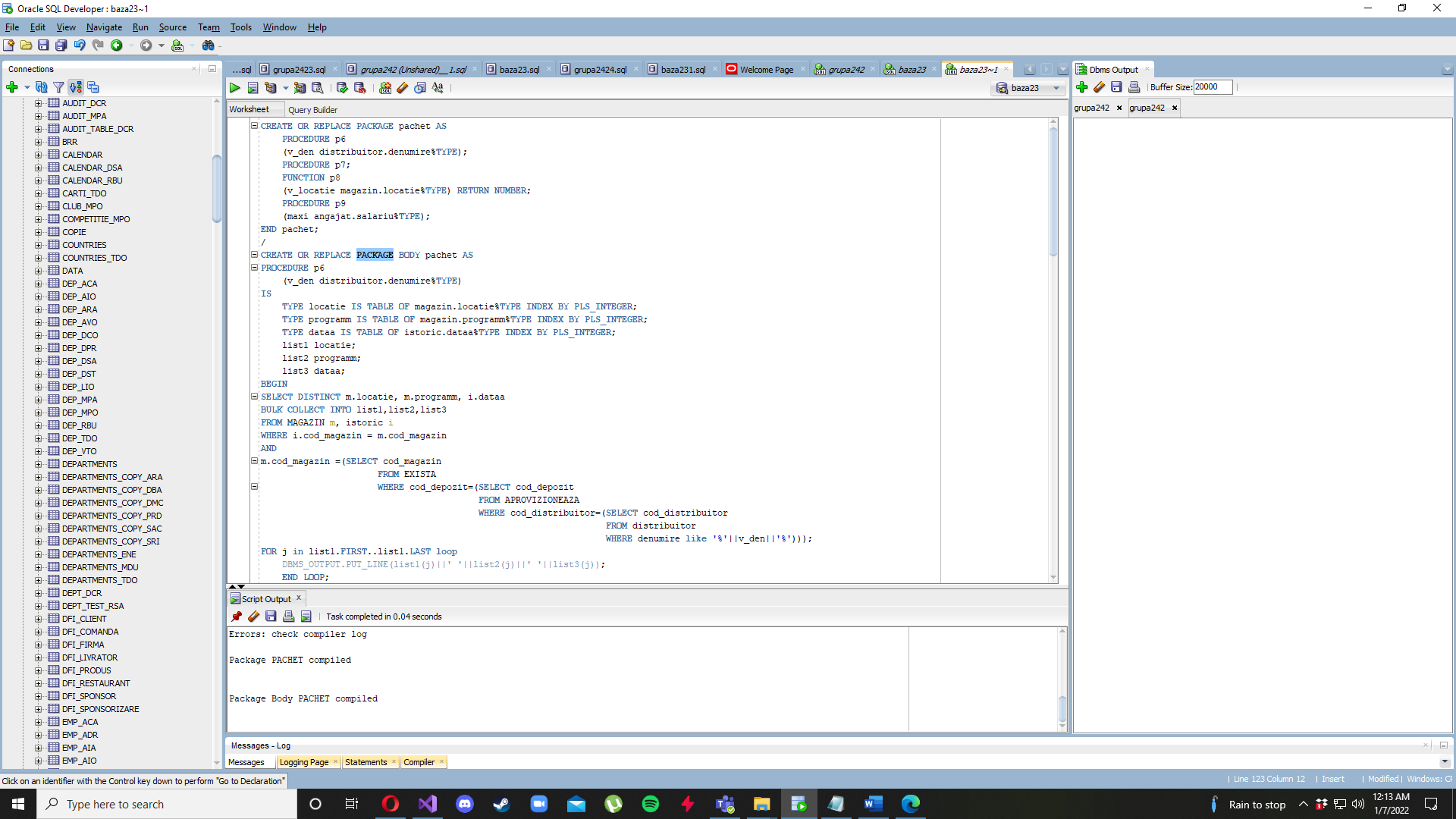
v\_incasari:=pachet.p8(v\_locatie);

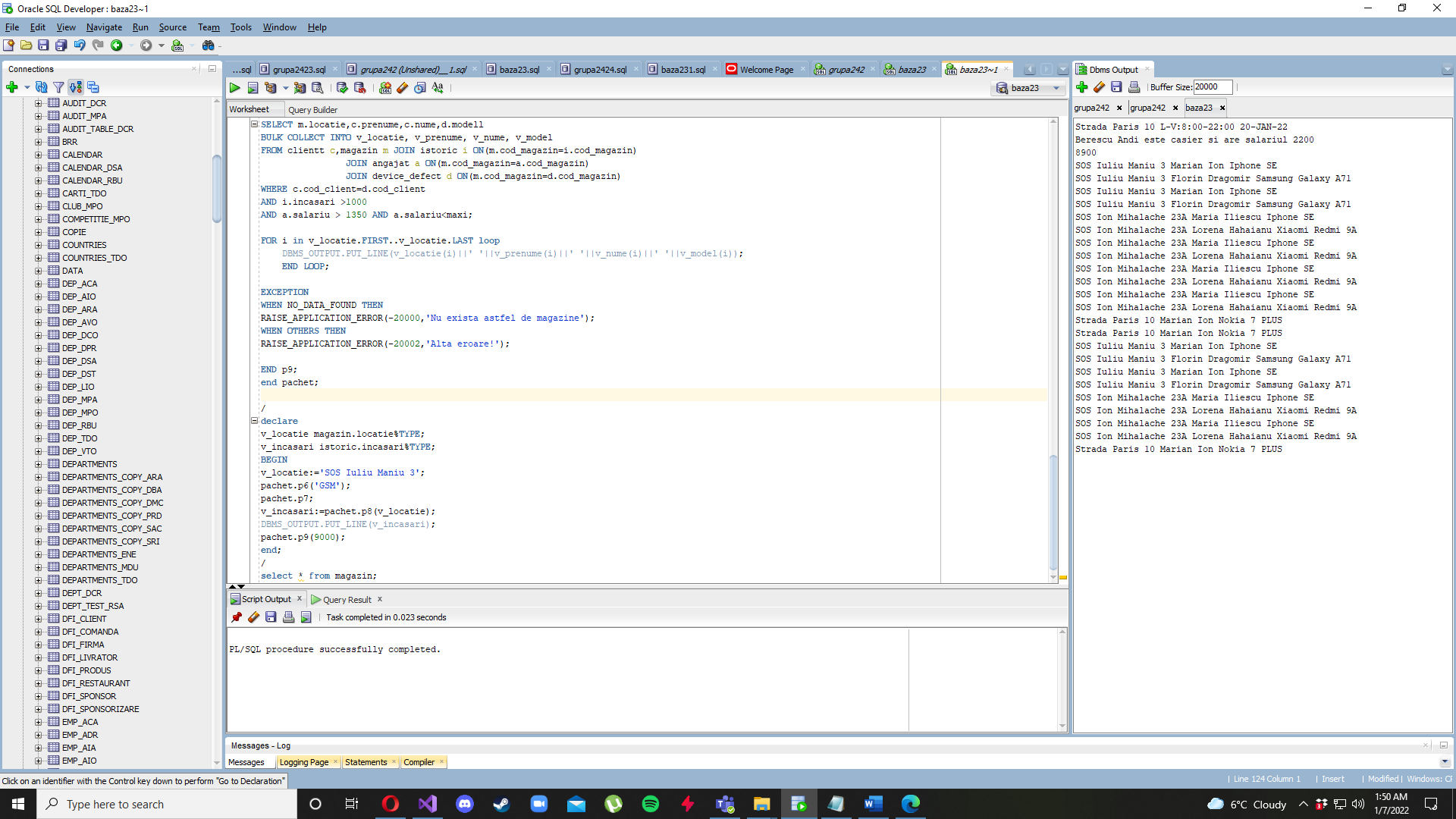
DBMS\_OUTPUT.PUT\_LINE(v\_incasari);

pachet.p9(9000);

end;

/





14.

--functie care primeste un nume si imi spune daca este client sau nu

--functie care primeste nume angajat si returneaza locatia magazinului unde lucreaza

--procedura care afiseaza denumirea distribuitorului,depozitului si programul magazinului pentru un anumit distribuitor

--procedura care afiseaza numele si prenumele clientilor care au cumparat un device, dar au si depus unul

CREATE OR REPLACE PACKAGE pachet2 AS

PROCEDURE numePre;

FUNCTION findClient

(v\_nume clientt.nume%TYPE) RETURN NUMBER;

FUNCTION findAngajat

(v\_nume angajat.nume%TYPE) RETURN VARCHAR2;

PROCEDURE distrDepMag

(v\_distr distribuitor.denumire%TYPE);

END pachet2;

/

CREATE OR REPLACE PACKAGE BODY pachet2 AS

FUNCTION findClient (v\_nume clientt.nume%TYPE)

RETURN NUMBER IS

v\_bool NUMBER;

v\_nume1 clientt.nume%TYPE;

BEGIN

v\_bool := 1;

SELECT

nume

INTO v\_nume1

FROM

clientt

WHERE

nume LIKE v\_nume;

RETURN v\_bool;

EXCEPTION

WHEN too\_many\_rows THEN

return null;

WHEN OTHERS THEN

return null;

END findClient;

PROCEDURE numepre IS

TYPE numepre\_record IS RECORD (

v\_nume clientt.nume%TYPE,

v\_prenume clientt.prenume%TYPE,

modell produs.modell%TYPE,

modell2 device\_defect.modell%TYPE

);

TYPE arr IS TABLE OF numepre\_record INDEX BY PLS\_INTEGER;

v arr;

BEGIN

SELECT c.nume,prenume,p.modell,d.modell "modell2"

BULK COLLECT INTO v

FROM clientt c, produs p, device\_defect d

WHERE c.cod\_client = p.cod\_client and c.cod\_client=d.cod\_client;

FOR i IN v.FIRST..v.LAST loop

DBMS\_OUTPUT.PUT\_LINE(v(i).v\_nume||' '||v(i).v\_prenume||' '||v(i).modell||' '||v(i).modell2);

end loop;

exception

WHEN NO\_DATA\_FOUND THEN

RAISE\_APPLICATION\_ERROR(-20000,'Nu exista astfel de clienti');

WHEN OTHERS THEN

raise\_application\_error(-20002, 'Alta eroare!');

END numepre;

---------------------------

PROCEDURE distrDepMag (v\_distr distribuitor.denumire%TYPE)

IS

TYPE distrDepMag\_record IS RECORD (

v\_den distribuitor.denumire%TYPE,

v\_den2 depozit.denumire%TYPE,

v\_loc2 magazin.locatie%TYPE

);

TYPE arr IS TABLE OF distrDepMag\_record INDEX BY PLS\_INTEGER;

v arr;

--54

BEGIN

SELECT d.denumire,de.denumire "denumire2", m.locatie

BULK COLLECT INTO v

FROM distribuitor d,depozit de,magazin m,exista e, aprovizioneaza a

WHERE d.cod\_distribuitor=a.cod\_distribuitor and a.cod\_depozit=de.cod\_depozit and de.cod\_depozit=e.cod\_depozit and e.cod\_magazin=m.cod\_magazin;

--60

FOR i IN v.FIRST..v.LAST loop

DBMS\_OUTPUT.PUT\_LINE(v(i).v\_den||' '||v(i).v\_den2||' '||v(i).v\_loc2);

end loop;

exception

WHEN NO\_DATA\_FOUND THEN

RAISE\_APPLICATION\_ERROR(-20000,'Nu exista date pentru acest distribuitor');

WHEN OTHERS THEN

raise\_application\_error(-20002, 'Alta eroare!');

END distrDepMag;

--------------

FUNCTION findAngajat (v\_nume angajat.nume%TYPE)

RETURN VARCHAR2 IS

v\_loc2 magazin.locatie%TYPE;

BEGIN

v\_loc2:='';

SELECT

m.locatie

INTO v\_loc2

FROM

magazin m,angajat a

WHERE

m.cod\_magazin = a.cod\_magazin

AND a.nume LIKE v\_nume;

RETURN v\_loc2;

exception

WHEN NO\_DATA\_FOUND THEN

v\_loc2:='';

WHEN OTHERS THEN

v\_loc2:='';

END findAngajat;

END pachet2;

/

declare

vr NUMBER;

nume angajat.nume%TYPE;

BEGIN

pachet2.numePre;

vr:=pachet2.findclient('Ion');

nume:=pachet2.findangajat('Ion');

pachet2.distrdepmag('hgh');

--DBMS\_OUTPUT.PUT\_LINE(v\_incasari);

end;

