**Tema PLSQL 2**

**E2. Definiți un tip colecție denumit tip\_orase\_\*\*\*. Creați tabelul excursie\_\*\*\* cu următoarea structură:**

**cod\_excursie NUMBER(4), denumire VARCHAR2(20), orase tip\_orase\_\*\*\* (ce va conține lista**

**orașelor care se vizitează într-o excursie, într-o ordine stabilită; de exemplu, primul oraș din listă va fi**

**primul oraș vizitat), status (disponibilă sau anulată).**

**a. Inserați 5 înregistrări în tabel.**

**b. Actualizați coloana orase pentru o excursie specificată:**

**- adăugați un oraș nou în listă, ce va fi ultimul vizitat în excursia respectivă;**

**- adăugați un oraș nou în listă, ce va fi al doilea oraș vizitat în excursia respectivă;**

**- inversați ordinea de vizitare a două dintre orașe al căror nume este specificat;**

**- eliminați din listă un oraș al cărui nume este specificat.**

**c. Pentru o excursie al cărui cod este dat, afișați numărul de orașe vizitate, respectiv numele orașelor.**

**d. Pentru fiecare excursie afișați lista orașelor vizitate.**

**e. Anulați excursiile cu cele mai puține orașe vizitate.**

**a. Cod SQL:**

create or replace type tip\_orase\_sma is table of varchar2(20);

create table excursie\_sma(

cod\_excursie number(4),

denumire varchar2(20),

orase tip\_orase\_sma,

status varchar2(11)

)nested table orase store as orase\_tab;

-- adaugare 5 inregistrari

insert into excursie\_sma values (1, 'Excursie 1', tip\_orase\_sma('Bacau', 'Bucuresti', 'Brasov'), 'disponibila');

insert into excursie\_sma values (2, 'Excursie 2', tip\_orase\_sma('Amsterdam', 'Haga', 'Rotterdam', 'Eindhoven'), 'disponibila');

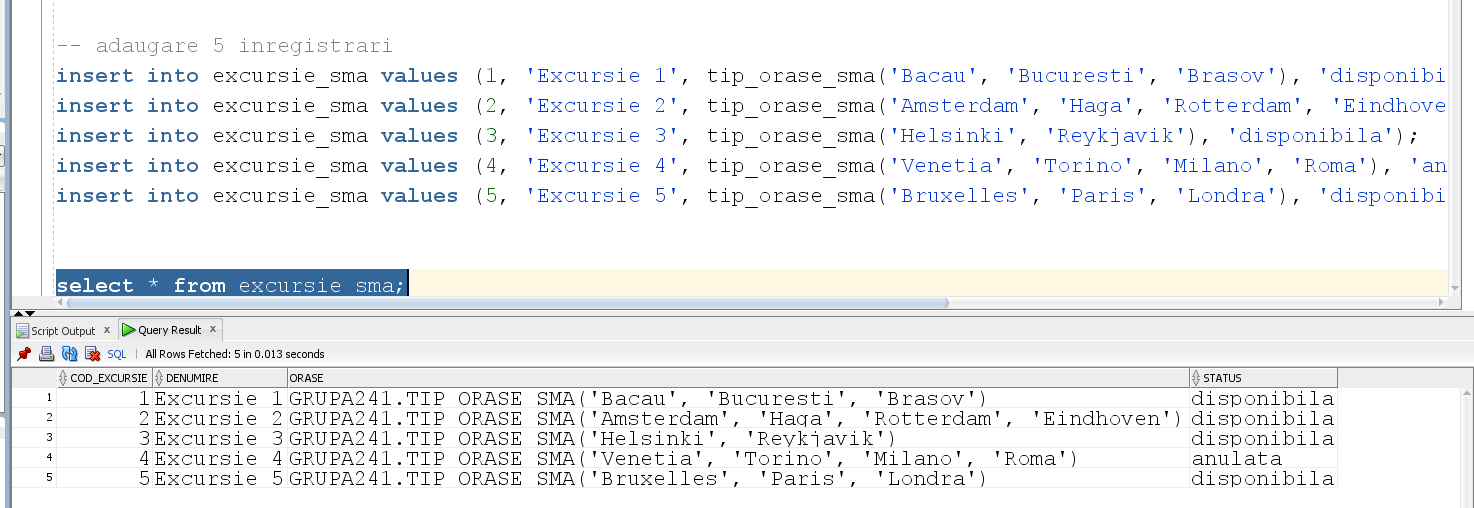
insert into excursie\_sma values (3, 'Excursie 3', tip\_orase\_sma('Helsinki', 'Reykjavik'), 'disponibila');

insert into excursie\_sma values (4, 'Excursie 4', tip\_orase\_sma('Venetia', 'Torino', 'Milano', 'Roma'), 'anulata');

insert into excursie\_sma values (5, 'Excursie 5', tip\_orase\_sma('Bruxelles', 'Paris', 'Londra'), 'disponibila');

commit;

**Print screen-uri:**



**b) Cod sql:**

declare --adaugare la final

v\_cod\_exc number := 1;

v\_noul\_oras varchar2(20) := 'OrasNouF';

v\_orase\_tip tip\_orase\_sma;

begin

select orase

into v\_orase\_tip

from excursie\_sma

where cod\_excursie = v\_cod\_exc for update;

v\_orase\_tip := v\_orase\_tip multiset union tip\_orase\_sma(v\_noul\_oras);

update excursie\_sma

set orase = v\_orase\_tip

where cod\_excursie = v\_cod\_exc;

end;

/

declare --adaugare al doilea

v\_cod\_exc number := 2;

v\_noul\_oras varchar2(20) := 'orasnou2!';

v\_orase\_tip tip\_orase\_sma;

v\_new\_orase tip\_orase\_sma := tip\_orase\_sma(v\_noul\_oras);

v\_index number;

begin

select orase

into v\_orase\_tip

from excursie\_sma

where cod\_excursie = v\_cod\_exc

for update;

v\_index := 2; --indexul la care inseram

-- shiftam orasele deja existente

for i in reverse v\_index..v\_orase\_tip.last-1 loop

v\_orase\_tip(i + 1) := v\_orase\_tip(i);

end loop;

--inseram orasul pe poz 2

v\_orase\_tip(v\_index) := v\_noul\_oras;

update excursie\_sma

set orase = v\_orase\_tip

where cod\_excursie = v\_cod\_exc;

end;

/

declare

v\_cod\_exc number := 1;

v\_oras1 varchar2(20) := 'Bacau';

v\_oras2 varchar2(20) := 'Bucuresti';

v\_orase\_tip tip\_orase\_sma;

begin

select orase

into v\_orase\_tip

from excursie\_sma

where cod\_excursie = v\_cod\_exc for update;

--inlocuire orase

for i in v\_orase\_tip.first .. v\_orase\_tip.last loop

if v\_orase\_tip(i) = v\_oras1 then

v\_orase\_tip(i) := v\_oras2;

elsif v\_orase\_tip(i) = v\_oras2 then

v\_orase\_tip(i) := v\_oras1;

end if;

end loop;

update excursie\_sma

set orase = v\_orase\_tip

where cod\_excursie = v\_cod\_exc;

end;

/

--stergere din excursia 4 torino

declare

v\_cod\_exc number := 4;

v\_oras\_to\_remove varchar2(20) := 'Torino';

v\_orase\_tip tip\_orase\_sma;

begin

--blocare rand pt update

select orase

into v\_orase\_tip

from excursie\_sma

where cod\_excursie = v\_cod\_exc for update;

--gasire si stergere oras

for i in v\_orase\_tip.first .. v\_orase\_tip.last loop

if v\_orase\_tip(i) = v\_oras\_to\_remove then

v\_orase\_tip.delete(i);

exit;

end if;

end loop;

--actualizare tabel cu lista de orase modificata

update excursie\_sma

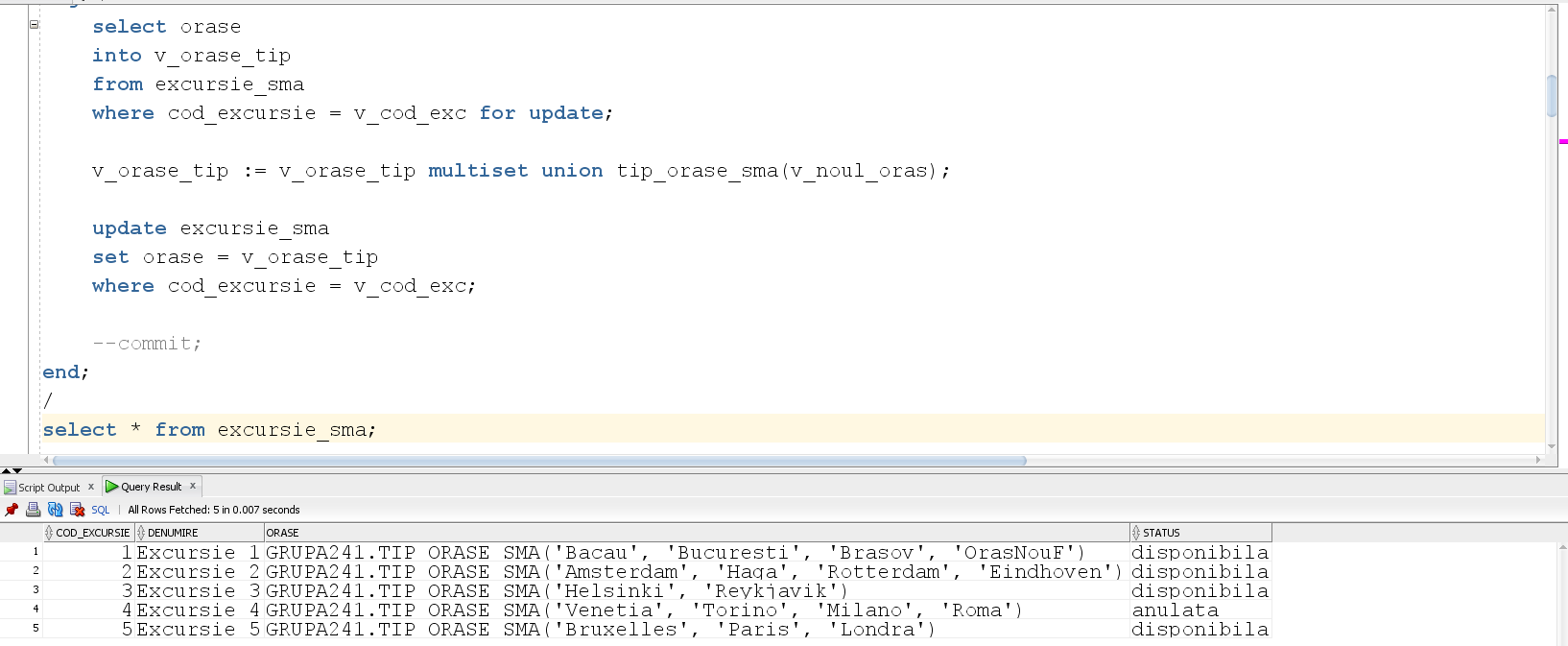
set orase = v\_orase\_tip

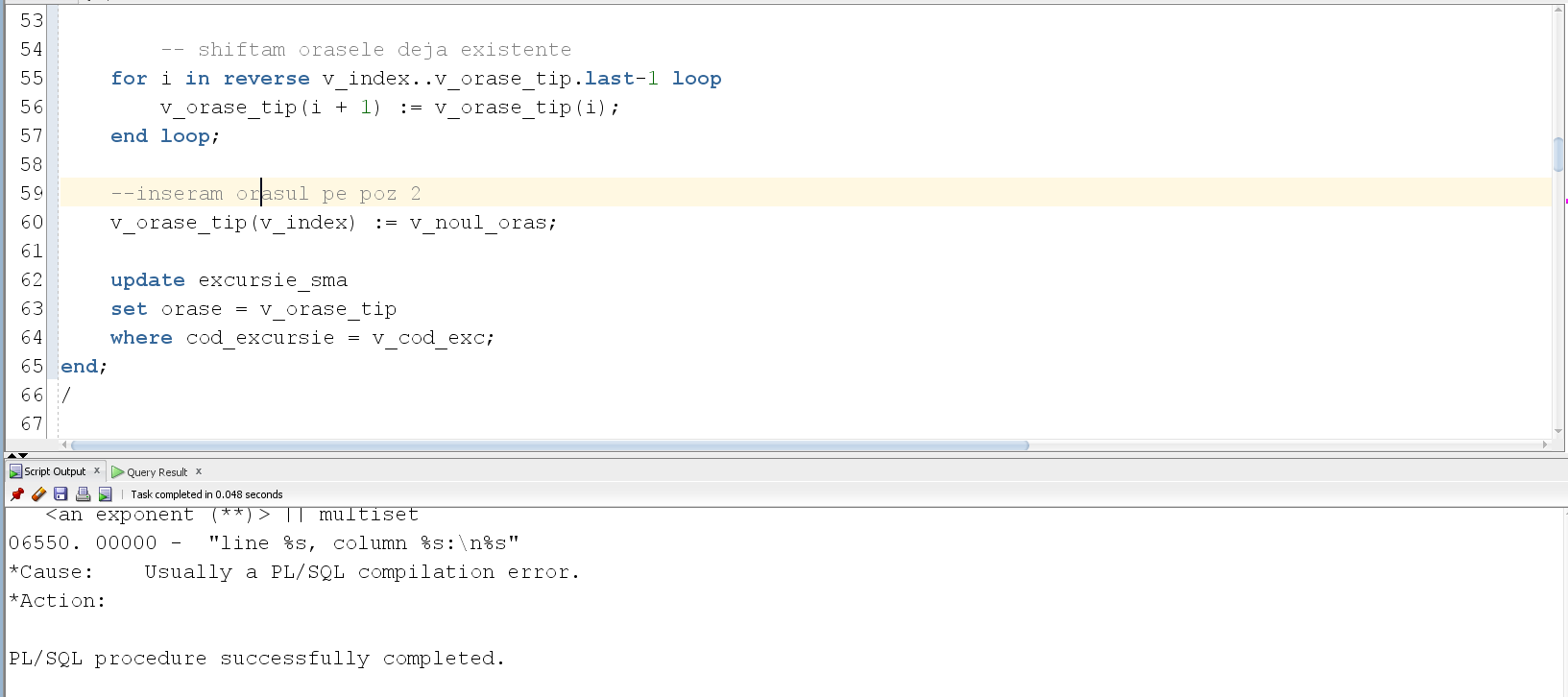
where cod\_excursie = v\_cod\_exc;

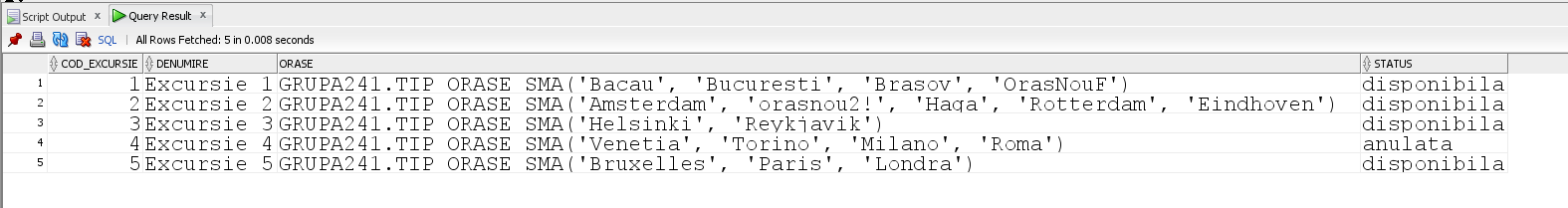
end;

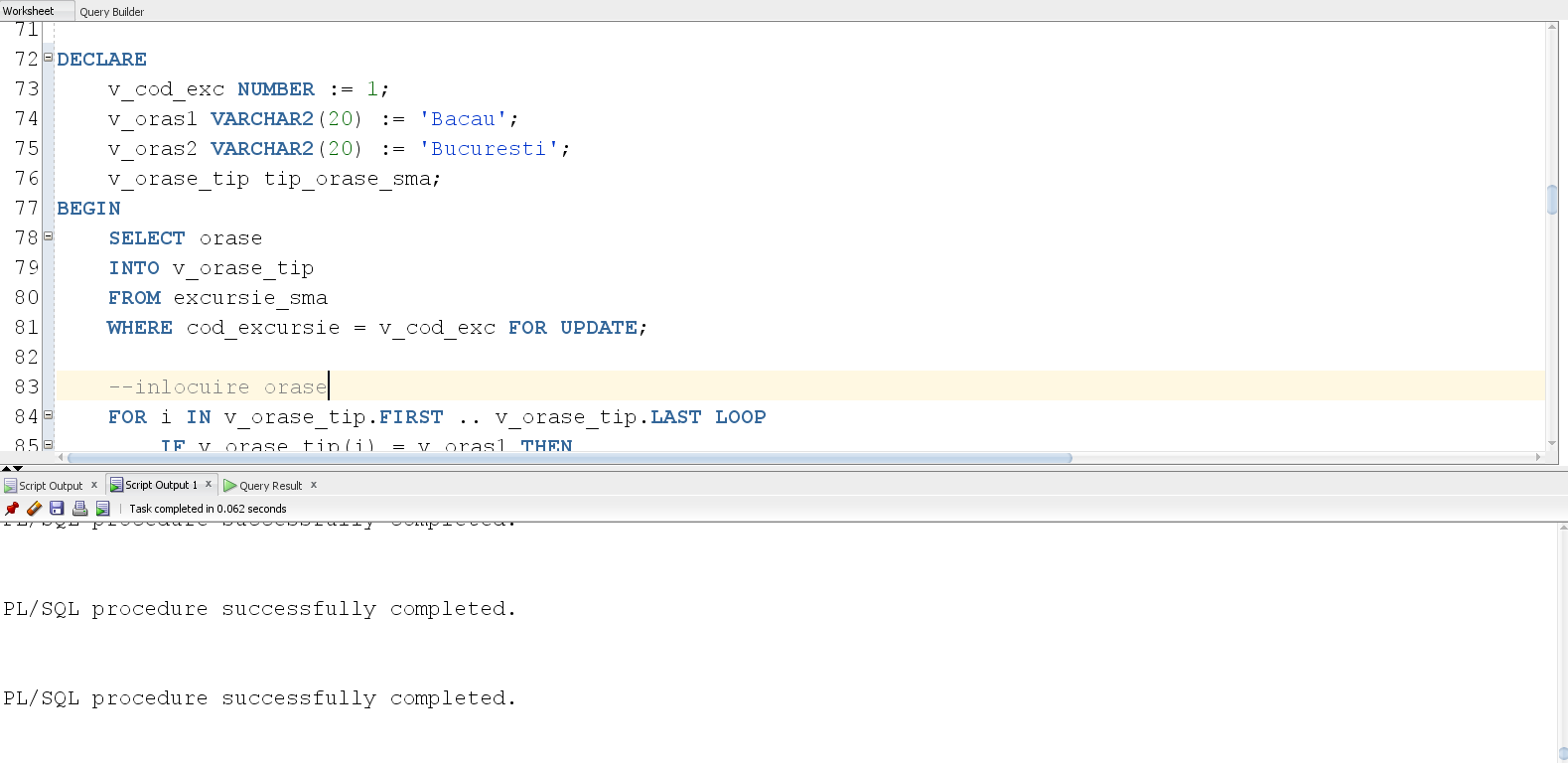
/

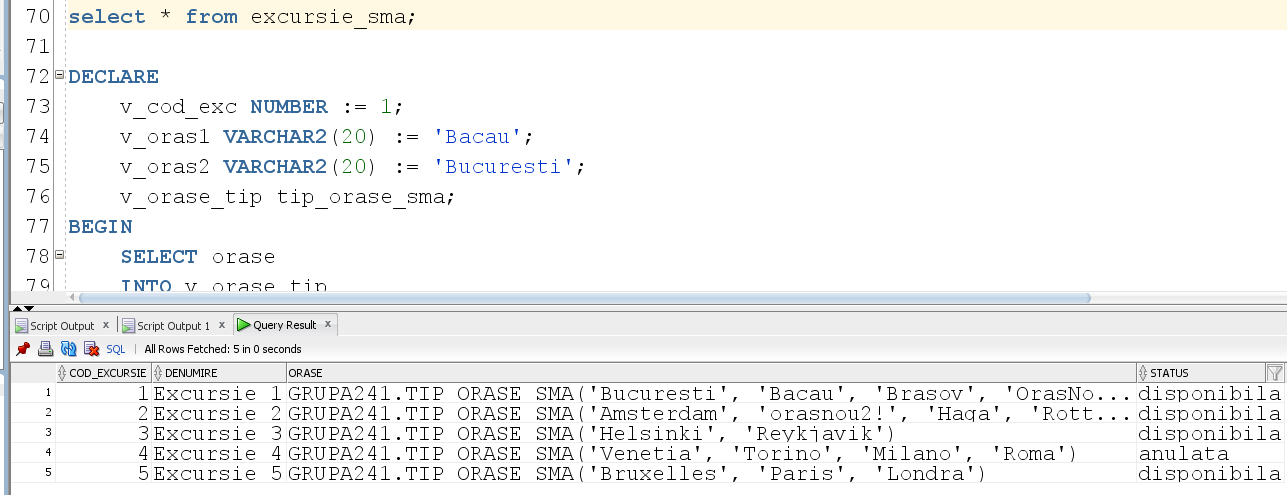
**Print screen-uri:**

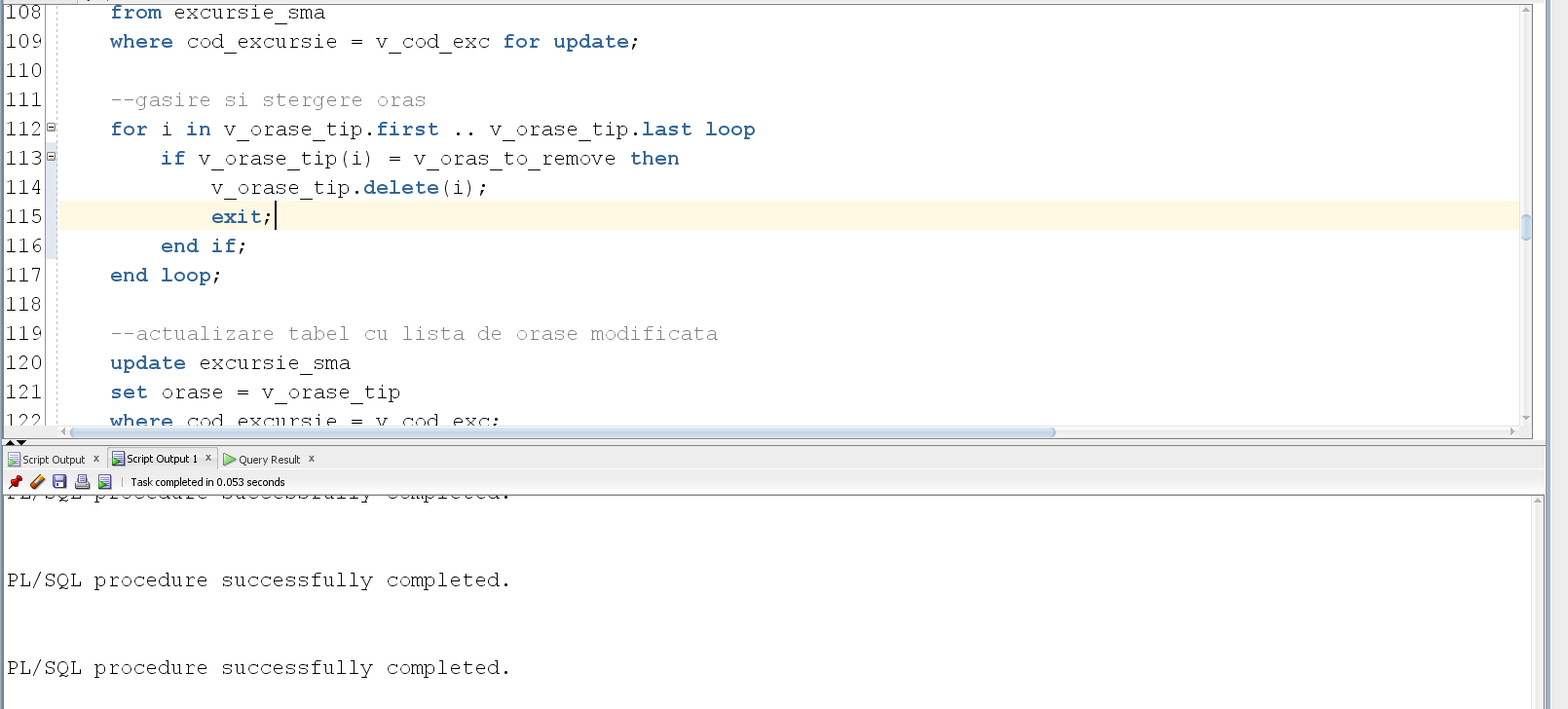


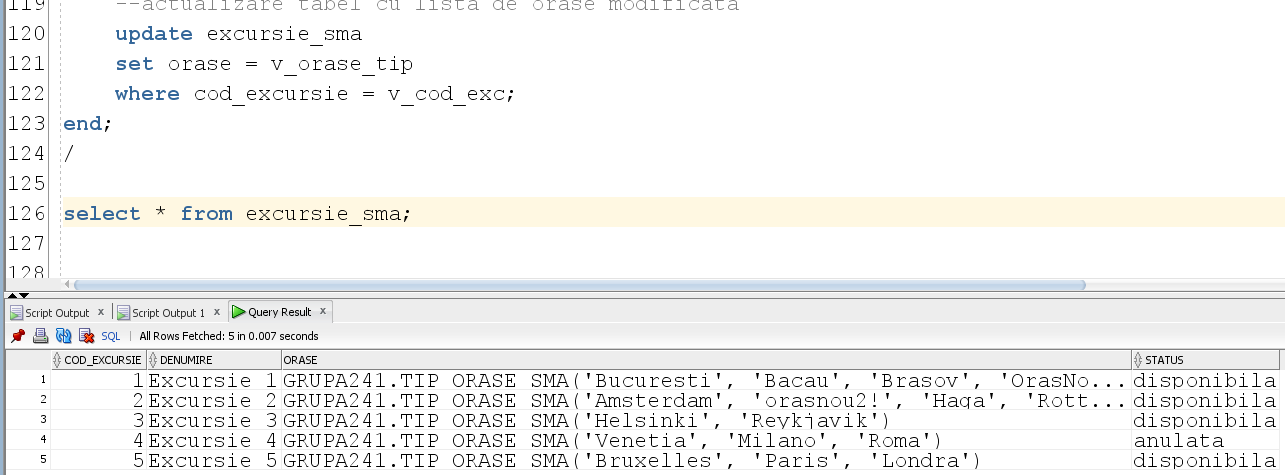












C) **Cod sql:**

--c

declare

v\_cod\_exc number := &p\_cod;

v\_orase\_tip tip\_orase\_sma;

begin

select orase

into v\_orase\_tip

from excursie\_sma

where cod\_excursie = v\_cod\_exc;

--afisare nr orase vizitate

dbms\_output.put\_line('Numar de orase vizitate: ' || v\_orase\_tip.count);

-- afisare nume oras

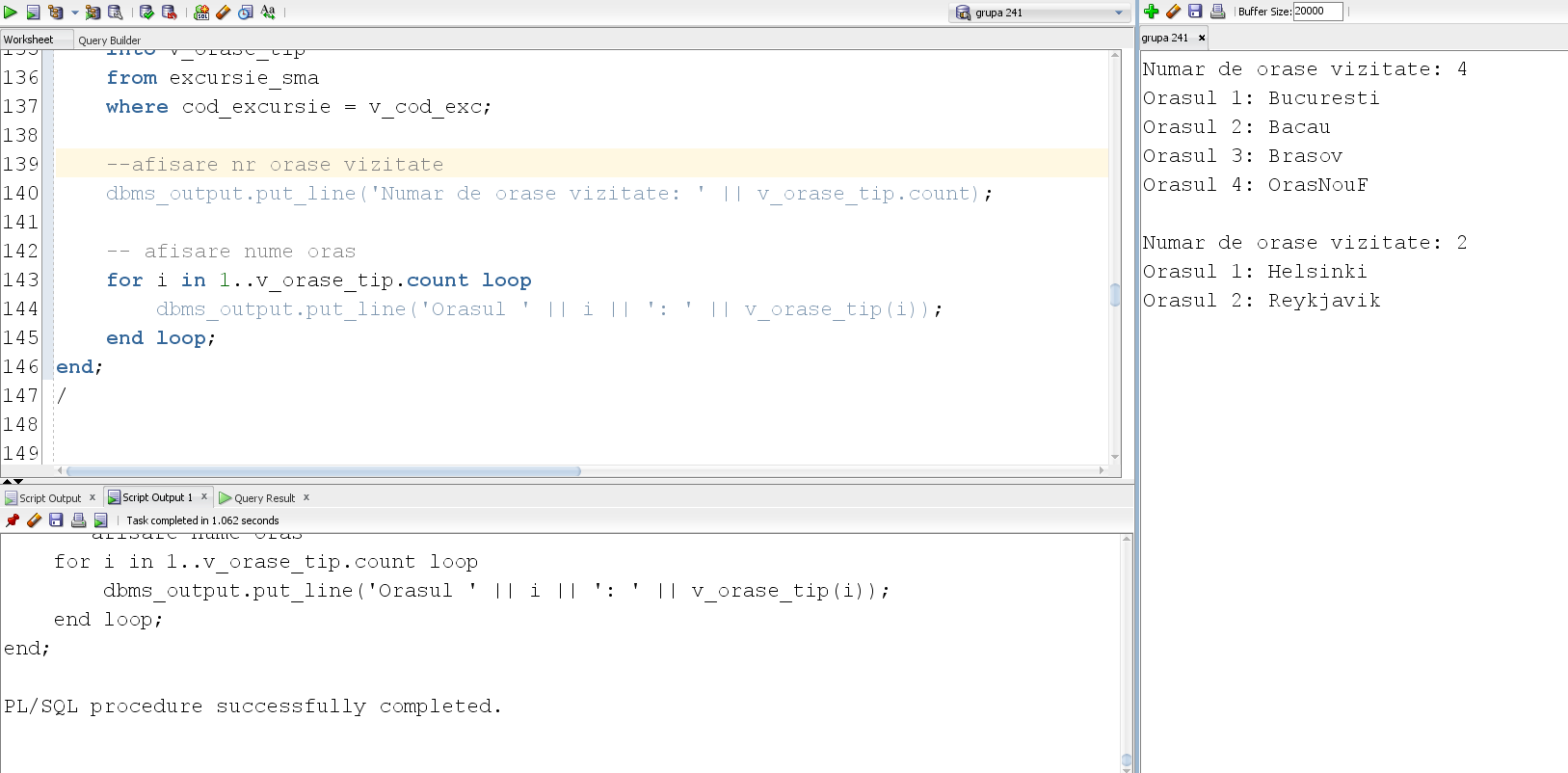
for i in 1..v\_orase\_tip.count loop

dbms\_output.put\_line('Orasul ' || i || ': ' || v\_orase\_tip(i));

end loop;

end;

/



d) Cod Sql:

--d

declare

cursor excursii\_cursor is

select cod\_excursie, orase

from excursie\_sma;

v\_cod\_exc number;

v\_orase\_tip tip\_orase\_sma;

begin

for excursie\_rec in excursii\_cursor loop

v\_cod\_exc := excursie\_rec.cod\_excursie;

v\_orase\_tip := excursie\_rec.orase;

--afisare lista cu orasele vizitate pt fiecare exc

dbms\_output.put\_line('Cod Excursie: ' || v\_cod\_exc);

dbms\_output.put\_line('Orase Vizitate:');

for i in 1..v\_orase\_tip.count loop

dbms\_output.put\_line(' ' || v\_orase\_tip(i));

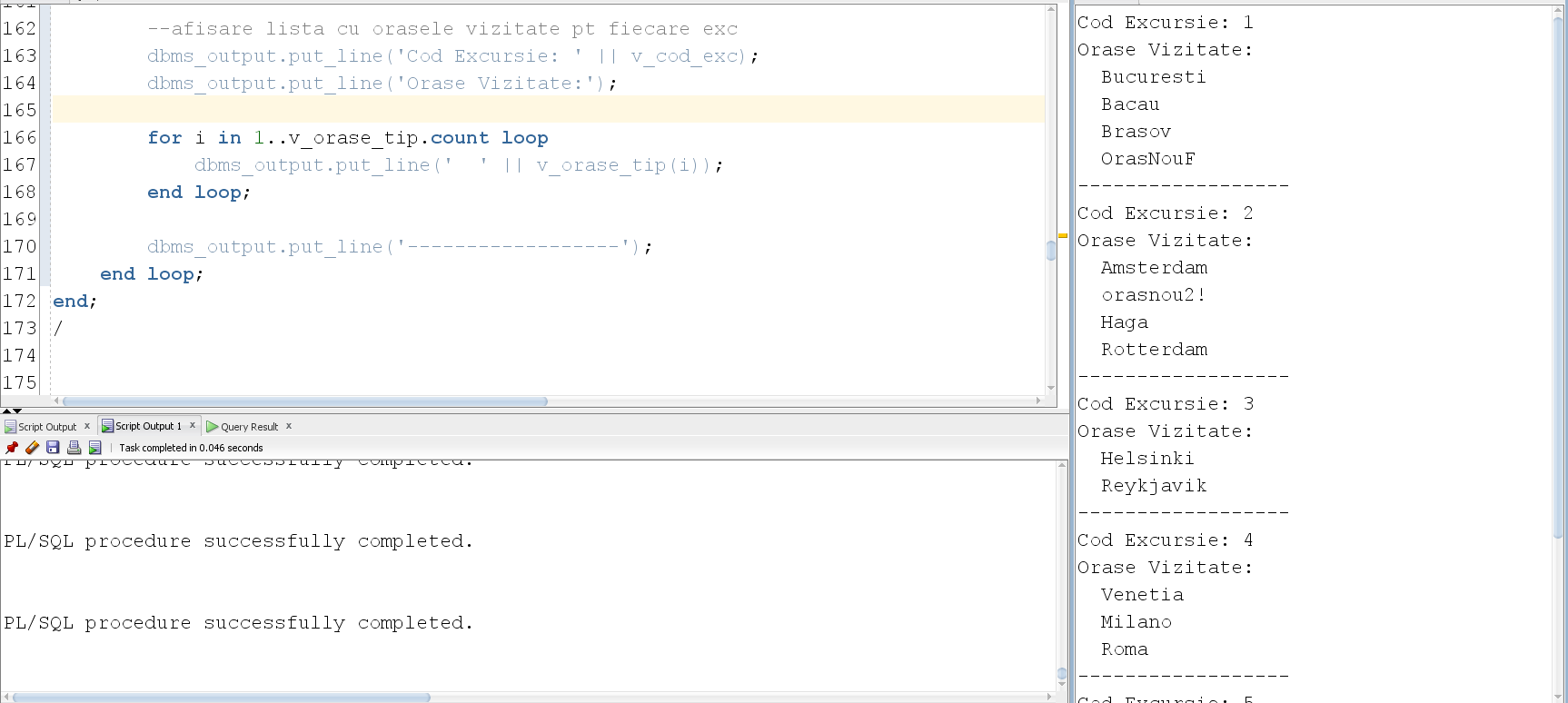
end loop;

dbms\_output.put\_line('------------------');

end loop;

end;

/



e) Cod Sql:

--e anulare excursie cu cele mai putine orase vizitate

declare

cursor excursii\_cursor is

select cod\_excursie, orase

from excursie\_sma;

v\_cod\_exc number;

v\_orase\_tip tip\_orase\_sma;

v\_min\_orase number := null;

procedure update\_excursie\_status(p\_cod\_exc number, p\_status varchar2) is

begin

update excursie\_sma

set status = p\_status

where cod\_excursie = p\_cod\_exc;

end;

begin

-- nr minim de orase vizitate

for excursie\_rec in excursii\_cursor loop

v\_orase\_tip := excursie\_rec.orase;

if v\_min\_orase is null or v\_orase\_tip.count < v\_min\_orase then

v\_min\_orase := v\_orase\_tip.count;

end if;

end loop;

--anulare exc cu min de orase

for excursie\_rec in excursii\_cursor loop

v\_cod\_exc := excursie\_rec.cod\_excursie;

v\_orase\_tip := excursie\_rec.orase;

if v\_orase\_tip.count = v\_min\_orase then

update\_excursie\_status(v\_cod\_exc, 'anulata');

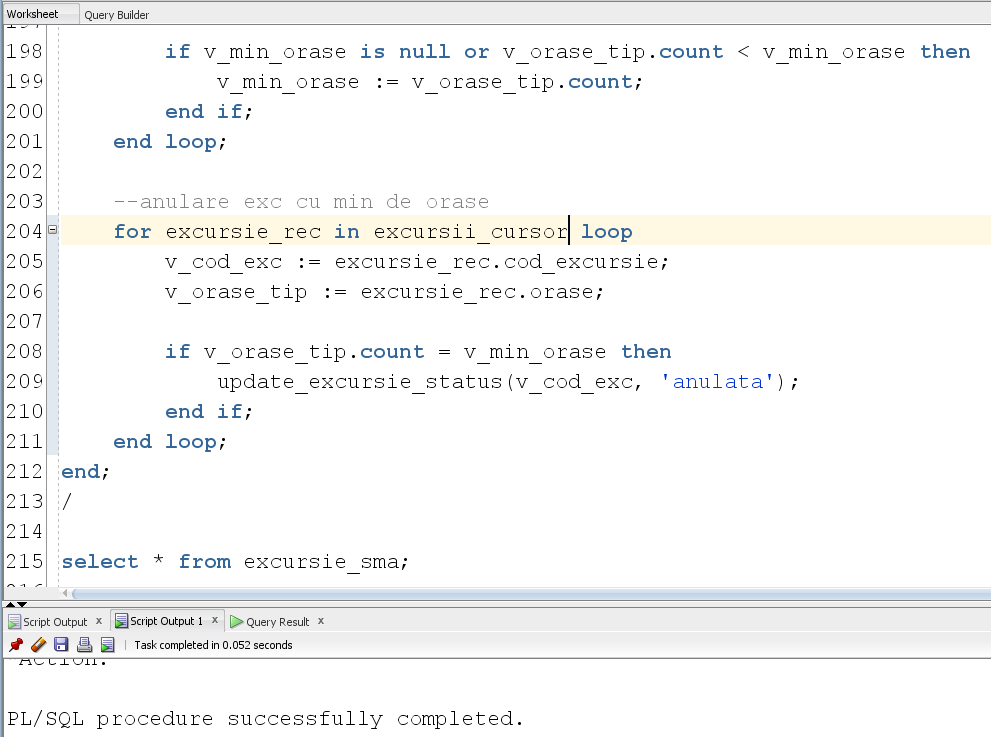
end if;

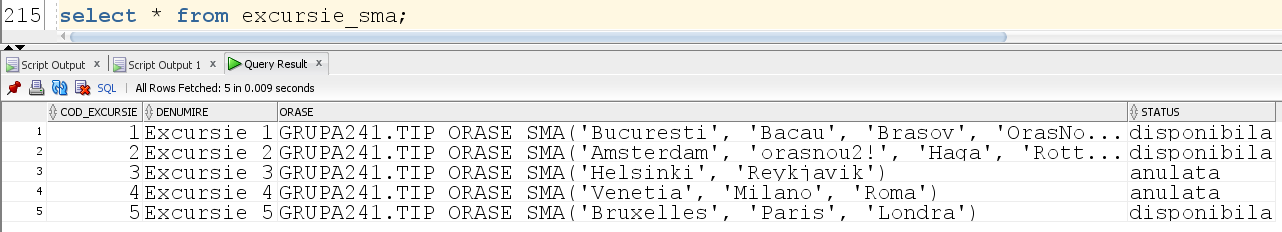
end loop;

end;

/

**Print screen-uri:**





E4: Adaptați cerințele exercițiilor 9 și 10 pentru diagrama proiectului prezentată la materia Baze de Date din anul I. Rezolvați aceste exerciții în PL/SQL, folosind baza de date proprie.

9) Definiți tipul subordonati\_\*\*\* (vector, dimensiune maximă 10, menține numere). Creați tabelul manageri\_\*\*\* cu următoarele câmpuri: cod\_mgr NUMBER(10), nume VARCHAR2(20), lista subordonati\_\*\*\*. Introduceți 3 linii în tabel. Afișați informațiile din tabel. Ștergeți tabelul creat, apoi tipul.

\*Am adaptat pentru artisti (manageri) si subordonati(asistenti)\*

**Cod:**

--definirea tipului pentru subordonati (artisti asistati)

CREATE TYPE asistenti\_artisti AS VARRAY(10) OF VARCHAR2(20);

--crearea tabelului pentru manageri(artisti)

CREATE TABLE artisti\_manageri (

cod\_mgr NUMBER(10),

nume VARCHAR2(20),

lista\_asistenti asistenti\_artisti

);

INSERT INTO artisti\_manageri VALUES (1, 'Ne-Yo', asistenti\_artisti('Asistent1', 'Asistent2', 'Asistent3'));

INSERT INTO artisti\_manageri VALUES (2, 'Rihanna', asistenti\_artisti('Asistent4', 'Asistent5'));

INSERT INTO artisti\_manageri VALUES (3, 'USHER', asistenti\_artisti('Asistent6', 'Asistent7', 'Asistent8', 'Asistent9'));

--afisare

SELECT \* FROM artisti\_manageri;

-- stergere

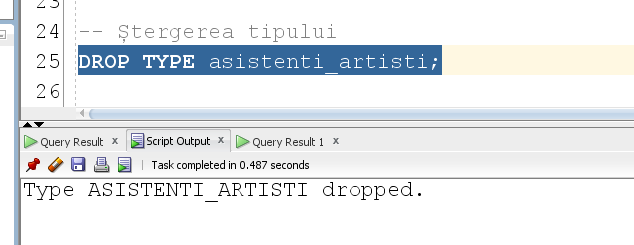
DROP TABLE artisti\_manageri;

DROP TYPE asistenti\_artisti;

**Print screen-uri:**







10) Creați tabelul emp\_test\_\*\*\* cu coloanele employee\_id și last\_name din tabelul employees. Adăugați în acest tabel un nou câmp numit telefon de tip tablou imbricat. Acest tablou va menține pentru fiecare salariat toate numerele de telefon la care poate fi contactat. Inserați o linie nouă în tabel. Actualizați o linie din tabel. Afișați informațiile din tabel. Ștergeți tabelul și tipul.

\*Am adaptat pentru artisti si numerele acestora de telefon\*

**Cod:**

--crearea tipului pentru numerele de telefon

CREATE TYPE telefoane\_artisti AS VARRAY(10) OF VARCHAR2(15);

--crearea tabelului pentru artisti

CREATE TABLE artisti (

artist\_id NUMBER(10),

nume\_artist VARCHAR2(50),

telefon telefoane\_artisti

);

--inserarea artist oin tabel

INSERT INTO artisti VALUES (1, 'NumeArtist1', telefoane\_artisti('123456789', '987654321'));

SELECT \* FROM artisti;

--actualizarea unui nr de tlf pt un artist

UPDATE artisti

SET telefon = telefoane\_artisti('987654321', '987654322')

WHERE artist\_id = 1;

--afis dupa actualizare

SELECT \* FROM artisti;

--stergerea tabelului

DROP TABLE artisti;

--stergerea tipului

DROP TYPE telefoane\_artisti;

