Temă

E6. Adaptați cerințele exercițiilor 5, 7 și 9 pentru diagrama proiectului prezentată la materia Baze de Date din anul I. Rezolvați aceste exerciții în PL/SQL, folosind baza de date proprie.

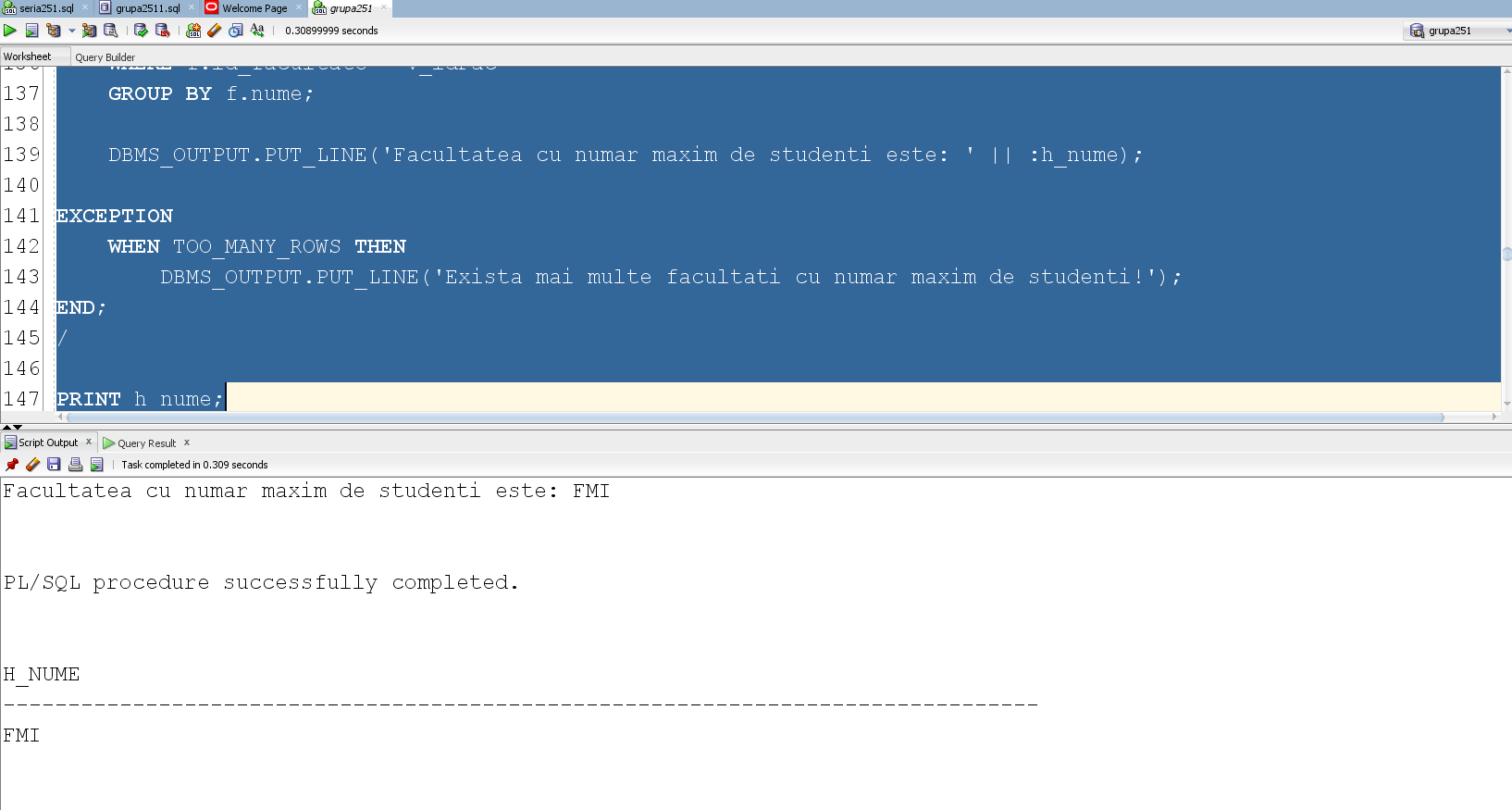
Întrucât nu am avut proiect la materia Baze de Date, am creat o structură simplificată pentru a rezolva cerințele.

| CREATE TABLE facultate\_ras  ( id\_facultate NUMBER(4) CONSTRAINT id\_facultate\_ras NOT NULL,  an\_infiintare NUMBER(4),  nume VARCHAR2(50),  profil VARCHAR2(20),  CONSTRAINT pk\_facultate\_ras PRIMARY KEY (id\_facultate)  );    CREATE TABLE student\_ras  ( id\_student NUMBER(4) CONSTRAINT id\_student\_ras NOT NULL,  nume VARCHAR2(30),  prenume VARCHAR2(30),  email VARCHAR2(60),  CONSTRAINT pk\_student\_ras PRIMARY KEY (id\_student)  );  CREATE TABLE studiaza\_ras  (  id\_facultate NUMBER(4) CONSTRAINT id\_fac\_ras NOT NULL,  id\_student NUMBER(4) CONSTRAINT id\_stud\_ras NOT NULL,  an NUMBER(1),  ciclu VARCHAR2(30),  forma\_finantare VARCHAR2(30),  CONSTRAINT pk\_studiaza\_ras PRIMARY KEY (id\_facultate, id\_student),  CONSTRAINT fk\_id\_facultate FOREIGN KEY(id\_facultate) REFERENCES facultate\_ras(id\_facultate) ON DELETE CASCADE,  CONSTRAINT fk\_id\_student FOREIGN KEY(id\_student) REFERENCES student\_ras(id\_student) ON DELETE CASCADE  );  INSERT INTO student\_ras  VALUES ('300', 'Popescu', 'Marian', 'popescu.m@gmail.com');  INSERT INTO student\_ras  VALUES ('200', 'Ailei', 'Costel', 'ailenei.c@gmail.com');  INSERT INTO student\_ras  VALUES ('400', 'Barbu', 'Miruna', 'barbu.m@gmail.com');  INSERT INTO student\_ras  VALUES ('512', 'Ali', 'Bogdan', 'ali.b@gmail.com');  INSERT INTO student\_ras  VALUES ('150', 'Marin', 'Mario', 'marin.m@gmail.com');  INSERT INTO facultate\_ras  VALUES ('1', '1990', 'FMI', 'teoretic');  INSERT INTO facultate\_ras  VALUES ('2', '1990', 'FSLS', 'teoretic');  INSERT INTO facultate\_ras  VALUES ('3', '1980', 'FSEGA', 'teoretic');  INSERT INTO facultate\_ras  VALUES ('4', '1985', 'FAA', 'teoretic');  INSERT INTO facultate\_ras  VALUES ('5', '1988', 'FASV', 'practic');  INSERT INTO studiaza\_ras  VALUES ('1', '300', '2', 'licenta', 'buget');  INSERT INTO studiaza\_ras  VALUES ('3', '300', '3', 'licenta', 'buget');  INSERT INTO studiaza\_ras  VALUES ('2', '300', '1', 'licenta', 'taxa');  INSERT INTO studiaza\_ras  VALUES ('3', '200', '1', 'master', 'taxa');  INSERT INTO studiaza\_ras  VALUES ('2', '200', '1', 'licenta', 'taxa');  INSERT INTO studiaza\_ras  VALUES ('1', '150', '2', 'licenta', 'taxa');  INSERT INTO studiaza\_ras  VALUES ('1', '400', '3', 'licenta', 'buget');  INSERT INTO studiaza\_ras  VALUES ('1', '512', '1', 'master', 'buget');  INSERT INTO studiaza\_ras  VALUES ('4', '512', '2', 'master', 'taxa');  INSERT INTO studiaza\_ras  VALUES ('5', '400', '1', 'licenta', 'taxa'); |
| --- |

(4. Definiți un bloc anonim în care să se afle numele facultății cu cei mai mulți studenți. Comentați cazul în care există cel puțin două facultăți cu număr maxim de studenți.)

5. Rezolvați problema anterioară utilizând variabile de legătură. Afișați rezultatul atât din bloc, cât și din exteriorul acestuia.

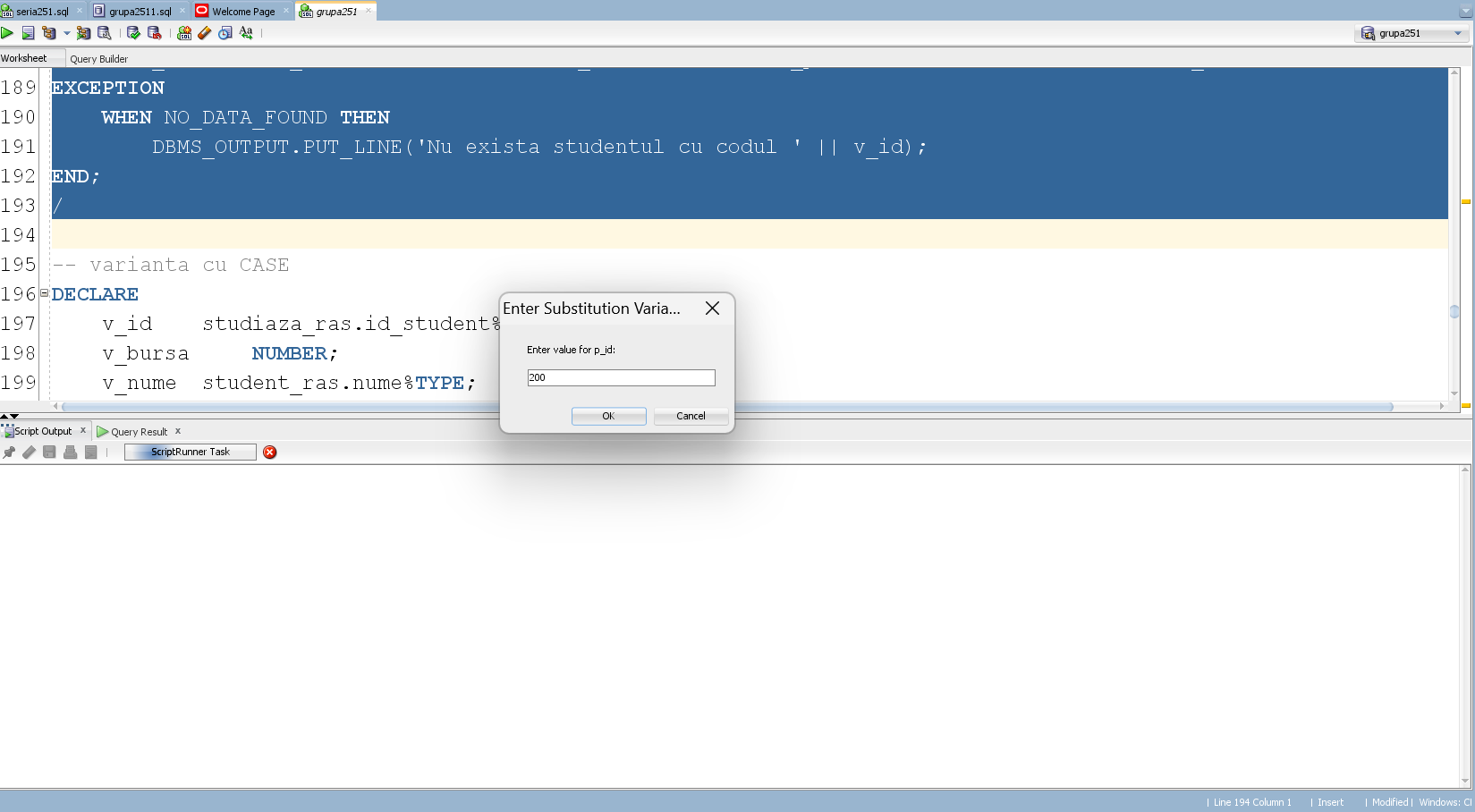
| SET SERVEROUTPUT ON;  VARIABLE h\_nume VARCHAR2(20)  DECLARE  v\_nrMaxim NUMBER;  v\_idFac studiaza\_ras.id\_facultate%TYPE;  BEGIN  -- calculam numarul maxim de studenti ai unei facultati  SELECT MAX(COUNT(id\_student))  INTO v\_nrMaxim  FROM studiaza\_ras  GROUP BY id\_facultate;    -- selectam id-ul facultatii cu numar maxim de oameni  SELECT id\_facultate  INTO v\_idFac  FROM studiaza\_ras  GROUP BY id\_facultate  HAVING COUNT(\*) = v\_nrMaxim;    -- luam numele facultatii  SELECT f.nume  INTO :h\_nume  FROM studiaza\_ras s JOIN facultate\_ras f ON (s.id\_facultate = f.id\_facultate)  WHERE f.id\_facultate = v\_idFac  GROUP BY f.nume;  DBMS\_OUTPUT.PUT\_LINE('Facultatea cu numar maxim de studenti este: ' || :h\_nume);    EXCEPTION  WHEN TOO\_MANY\_ROWS THEN  DBMS\_OUTPUT.PUT\_LINE('Exista mai multe facultati cu numar maxim de studenti!');  END;  /  PRINT h\_nume; |
| --- |

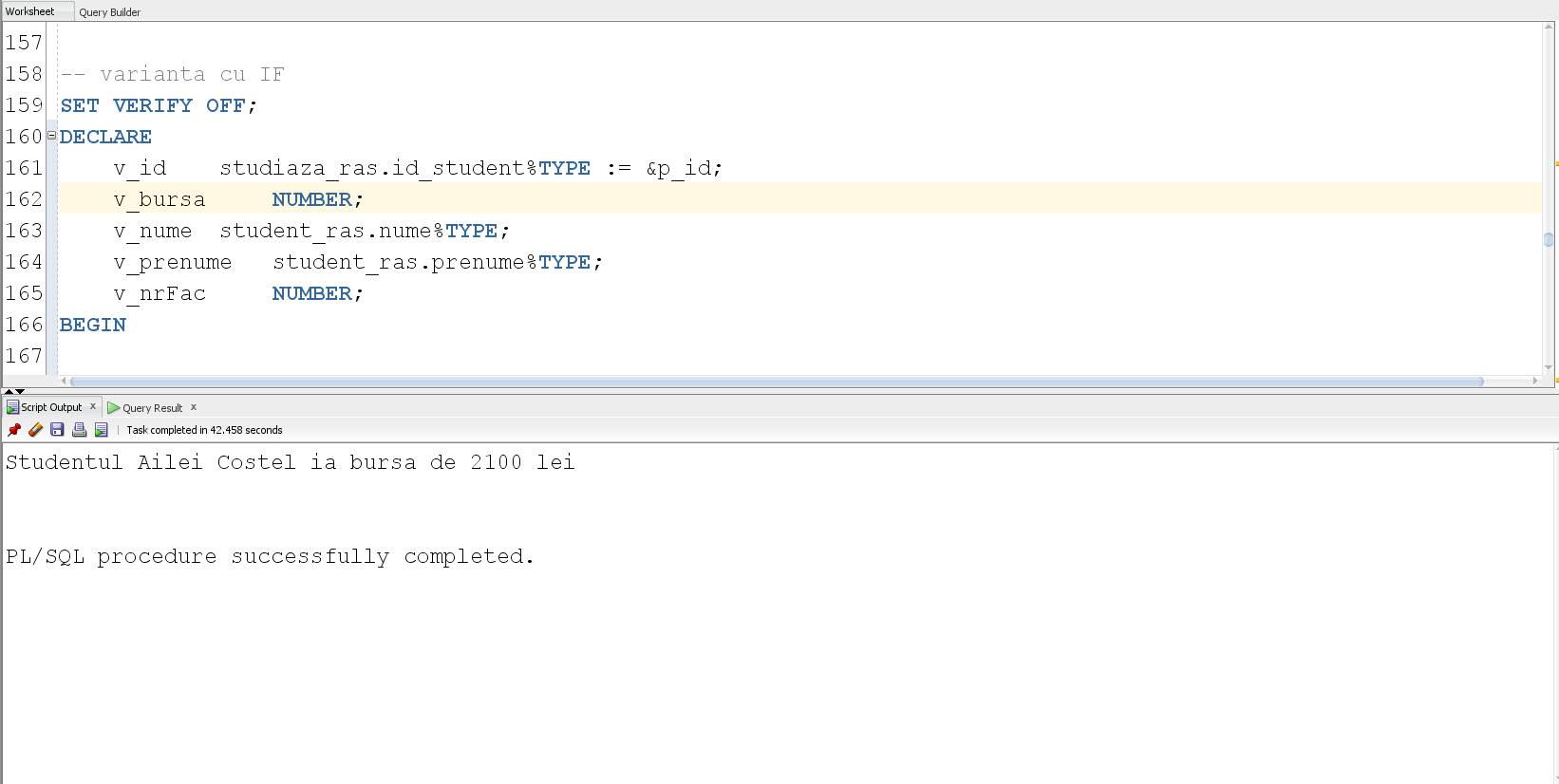


7. Determinați numele și bursa pe care o primește un student al cărui cod este dat de la tastatură. Bursa este determinată astfel: dacă e student la o singură facultate primește 1500, dacă studiază la două primește 2100, iar pentru mai mult de 2 facultăți, bursa este 3000. Afișați bursa primită. Comentați cazul în care nu există niciun student cu codul introdus.

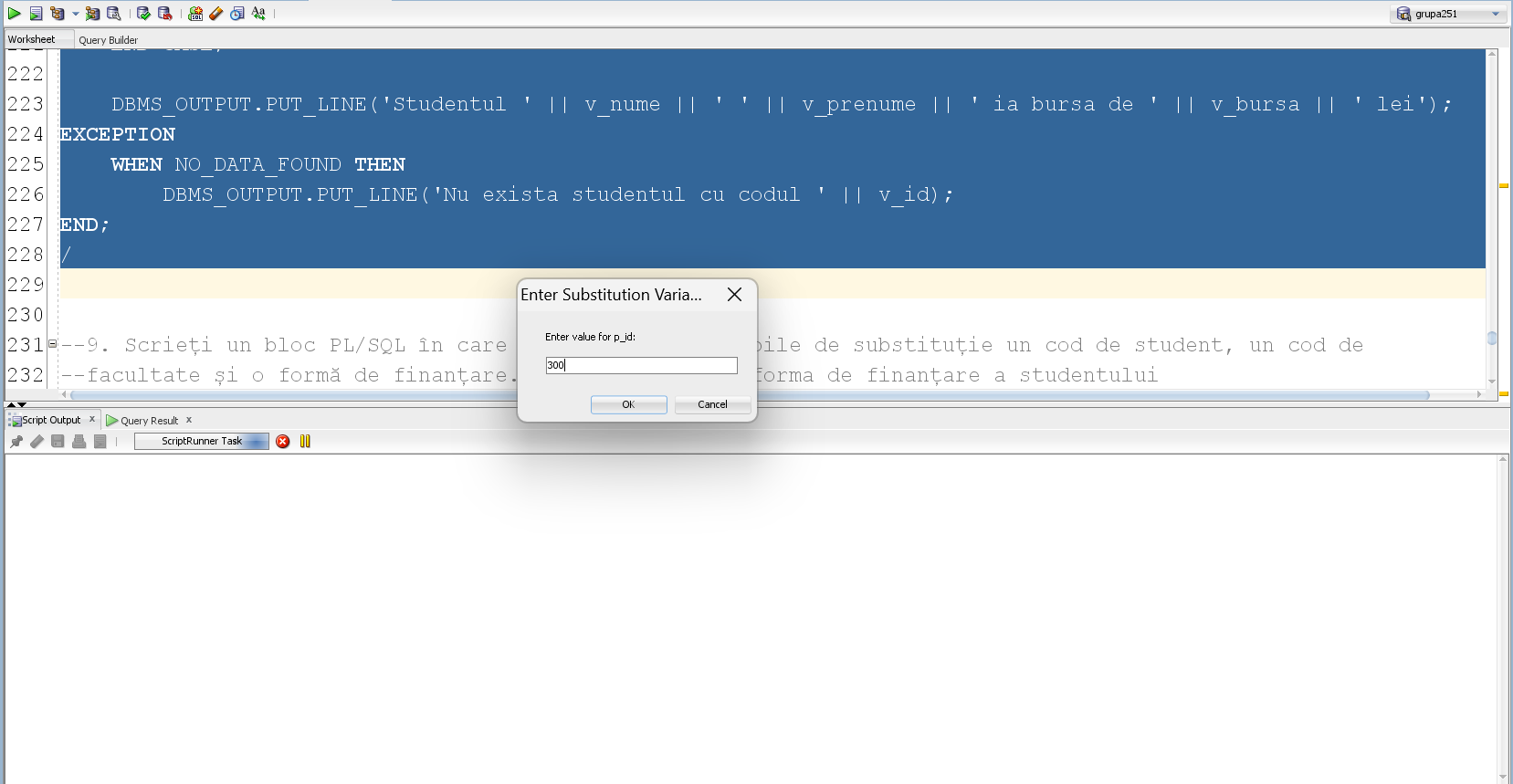
ex: 200 -> 2100, 300 -> 3000, 150 -> 1500

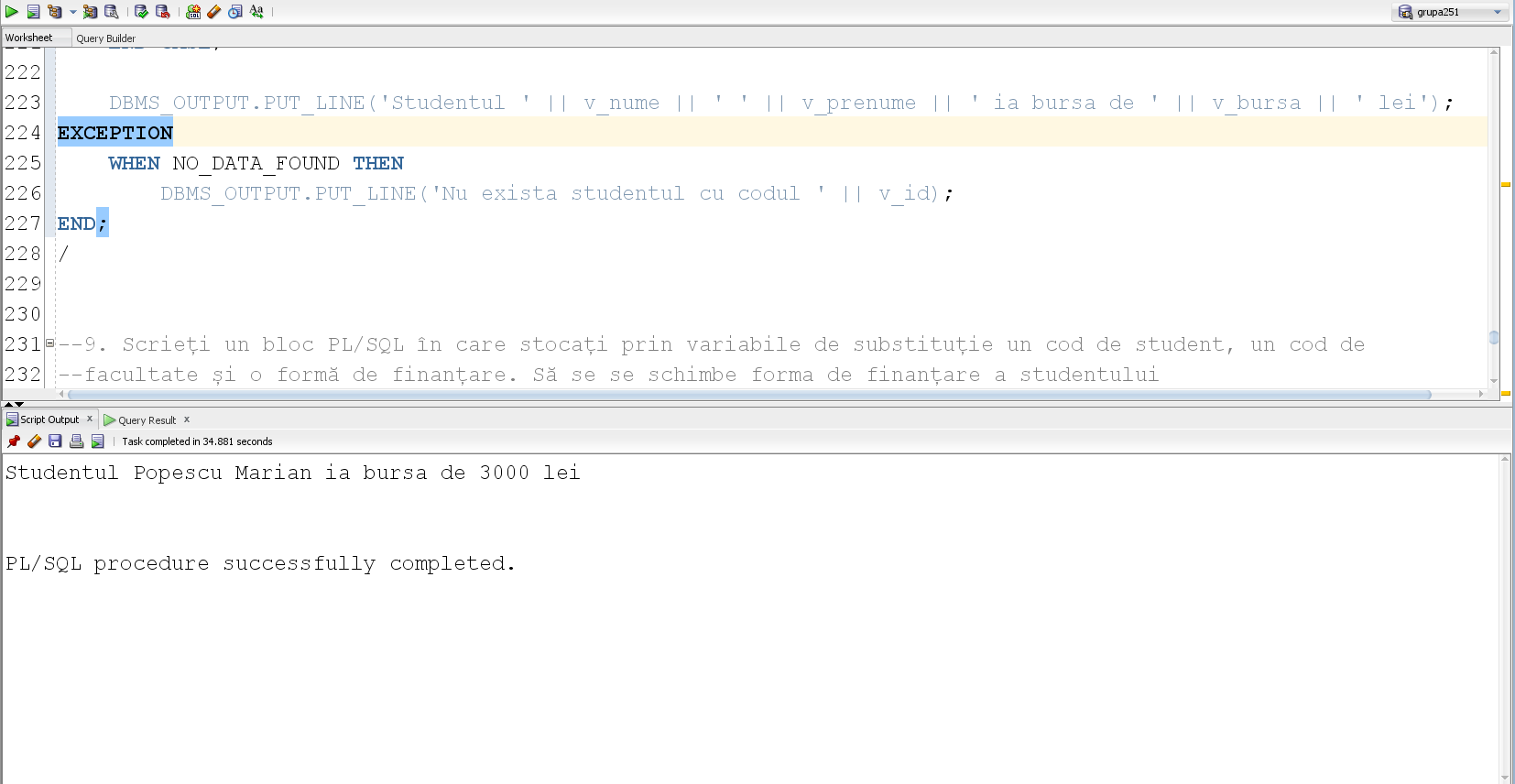
| -- varianta cu IF  SET VERIFY OFF;  DECLARE  v\_id studiaza\_ras.id\_student%TYPE := &p\_id;  v\_bursa NUMBER;  v\_nume student\_ras.nume%TYPE;  v\_prenume student\_ras.prenume%TYPE;  v\_nrFac NUMBER;  BEGIN  -- selectam numele si prenumele studentului  SELECT nume, prenume  INTO v\_nume, v\_prenume  FROM student\_ras  WHERE id\_student = v\_id;    -- vedem la cate facultati studiaza  SELECT COUNT(id\_facultate)  INTO v\_nrFac  FROM studiaza\_ras  GROUP BY id\_student  HAVING id\_student = v\_id;    IF v\_nrFac = 1 THEN  v\_bursa := 1500;  ELSIF v\_nrFac = 2 THEN  v\_bursa := 2100;  ELSE  v\_bursa := 3000;  END IF;  DBMS\_OUTPUT.PUT\_LINE('Studentul ' || v\_nume || ' ' || v\_prenume || ' ia bursa de ' || v\_bursa || ' lei');  EXCEPTION  WHEN NO\_DATA\_FOUND THEN  DBMS\_OUTPUT.PUT\_LINE('Nu exista studentul cu codul ' || v\_id);  END;  / |
| --- |





| -- varianta cu CASE  DECLARE  v\_id studiaza\_ras.id\_student%TYPE := &p\_id;  v\_bursa NUMBER;  v\_nume student\_ras.nume%TYPE;  v\_prenume student\_ras.prenume%TYPE;  v\_nrFac NUMBER;  BEGIN  -- selectam numele si prenumele studentului  SELECT nume, prenume  INTO v\_nume, v\_prenume  FROM student\_ras  WHERE id\_student = v\_id;    -- vedem la cate facultati studiaza  SELECT COUNT(id\_facultate)  INTO v\_nrFac  FROM studiaza\_ras  GROUP BY id\_student  HAVING id\_student = v\_id;    CASE v\_nrfac  WHEN 1 THEN v\_bursa := 1500;  WHEN 2 THEN v\_bursa := 2100;  ELSE v\_bursa := 3000;  END CASE;    DBMS\_OUTPUT.PUT\_LINE('Studentul ' || v\_nume || ' ' || v\_prenume || ' ia bursa de ' || v\_bursa || ' lei');  EXCEPTION  WHEN NO\_DATA\_FOUND THEN  DBMS\_OUTPUT.PUT\_LINE('Nu exista studentul cu codul ' || v\_id);  END;  / |
| --- |



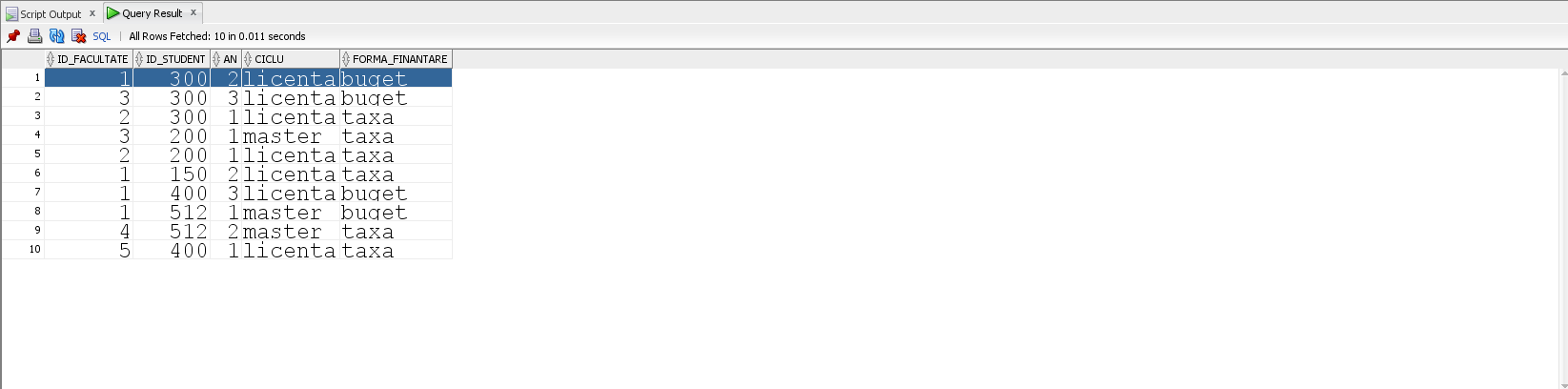


9. Scrieți un bloc PL/SQL în care stocați prin variabile de substituție un cod de student, un cod de facultate și o formă de finanțare. Să se se schimbe forma de finanțare a studentului de la facultatea respectivă conform formei de finanțare citite. Dacă modificarea s-a putut realiza (există în tabelul studiaza\_ras un student având codul respectiv care studiaza la facultatea dată) să se afișeze mesajul “Actualizare realizata”, iar în caz contrar mesajul “Nu exista studentul cu codul dat la facultatea indicata”. Anulați modificările realizate.

Se testează cu 300, 1, taxa

| DECLARE  v\_student studiaza\_ras.id\_student%TYPE := &p\_idStudent;  v\_facultate studiaza\_ras.id\_facultate%TYPE := &p\_idFacultate;  v\_finantare studiaza\_ras.forma\_finantare%TYPE := '&p\_finantare';  BEGIN  UPDATE studiaza\_ras  SET forma\_finantare = v\_finantare  WHERE id\_facultate = v\_facultate AND id\_student = v\_student;    IF SQL%ROWCOUNT = 0 THEN  DBMS\_OUTPUT.PUT\_LINE('Nu exista studentul cu codul dat la facultatea indicata');  ELSE DBMS\_OUTPUT.PUT\_LINE('Actualizare realizata');  END IF;  END;  /  SELECT \*  FROM studiaza\_ras;  ROLLBACK; |
| --- |

Înainte de rularea codului:



După rularea codului:

