**Tema 4 de laborator**

E2) Se dă următorul enunț: Pentru fiecare zi a lunii octombrie (se vor lua în considerare și zilele din lună în care nu au fost realizate împrumuturi) obțineți numărul de împrumuturi efectuate.

a. Încercați să rezolvați problema în SQL fără a folosi structuri ajutătoare. b. Definiți tabelul octombrie\_\*\*\* (id, data). Folosind PL/SQL populați cu date acest tabel. Rezolvați în SQL problema dată.

--E2

--a)

SELECT DT, ( SELECT COUNT(\*)

FROM rental WHERE

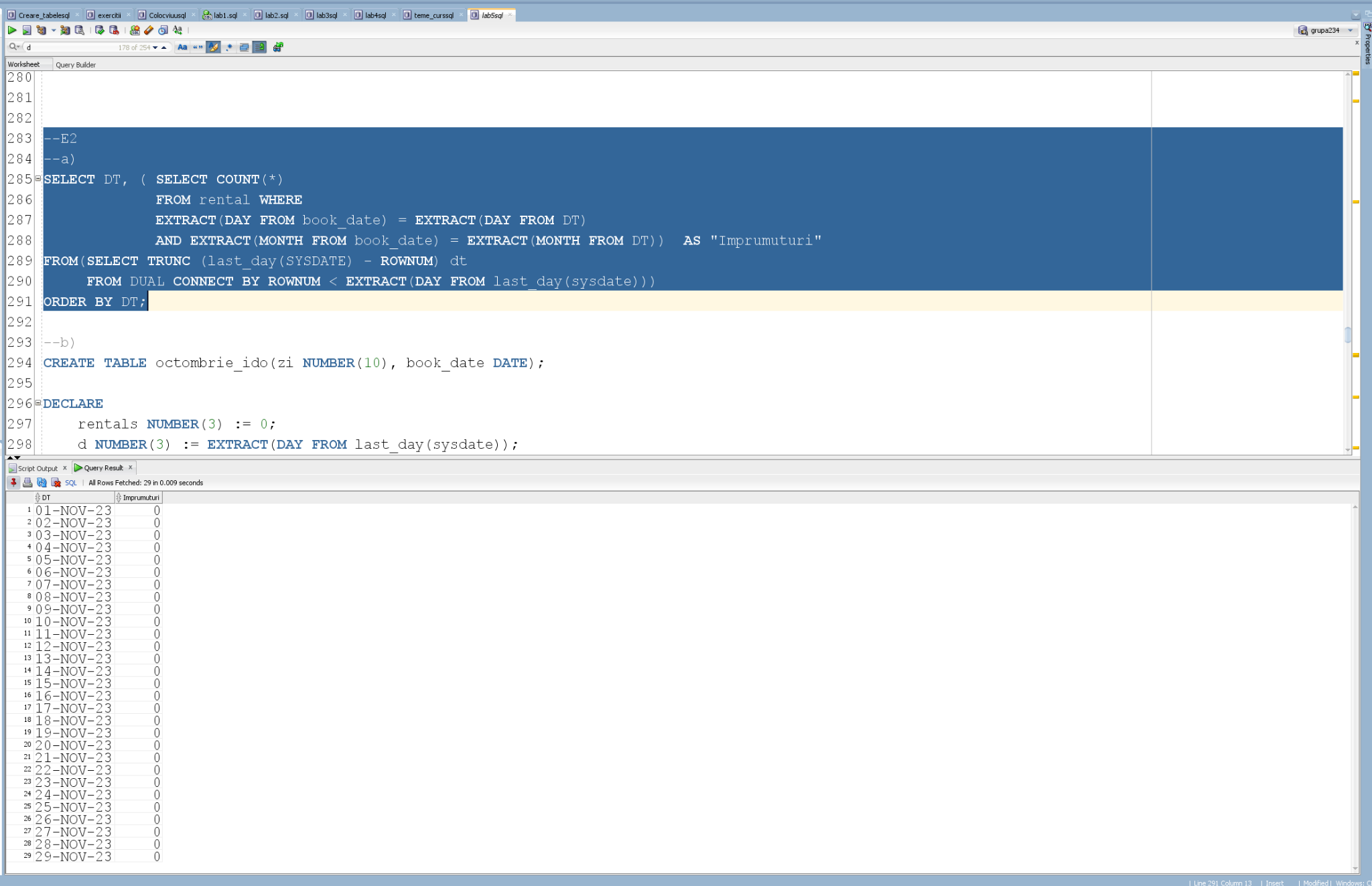
EXTRACT(DAY FROM book\_date) = EXTRACT(DAY FROM DT)

AND EXTRACT(MONTH FROM book\_date) = EXTRACT(MONTH FROM DT)) AS "Imprumuturi"

FROM(SELECT TRUNC (last\_day(SYSDATE) - ROWNUM) dt

FROM DUAL CONNECT BY ROWNUM < EXTRACT(DAY FROM last\_day(sysdate)))

ORDER BY DT;



--b)

CREATE TABLE octombrie\_ido(zi NUMBER(10), book\_date DATE);

DECLARE

rentals NUMBER(3) := 0;

d NUMBER(3) := EXTRACT(DAY FROM last\_day(sysdate));

BEGIN

FOR i IN 1..d LOOP

SELECT COUNT(\*) INTO rentals FROM rental

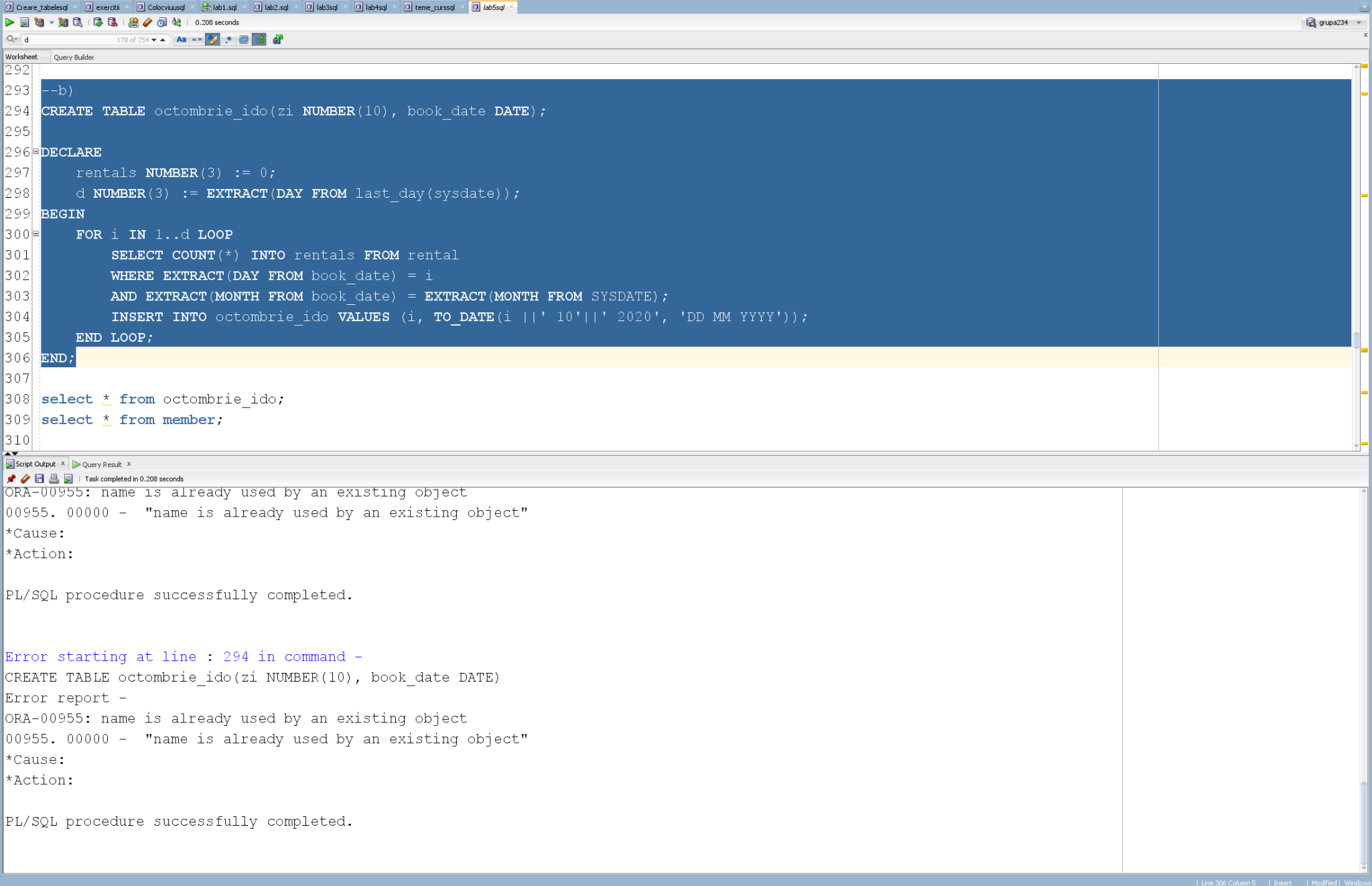
WHERE EXTRACT(DAY FROM book\_date) = i

AND EXTRACT(MONTH FROM book\_date) = EXTRACT(MONTH FROM SYSDATE);

INSERT INTO octombrie\_ido VALUES (i, TO\_DATE(i ||' 10'||' 2020', 'DD MM YYYY'));

END LOOP;

END;



E4)

Modificați problema anterioară astfel încât să afișați și următorul text:

- Categoria 1 (a împrumutat mai mult de 75% din titlurile existente)

- Categoria 2 (a împrumutat mai mult de 50% din titlurile existente)

- Categoria 3 (a împrumutat mai mult de 25% din titlurile existente)

- Categoria 4 (altfel)

DECLARE

V\_NUME member.last\_name % TYPE := '&NUME';

V\_MEMBER\_ID member.member\_id % TYPE;

V\_NR\_FILME NUMBER(5);

num NUMBER(3) := 0;

procent NUMBER(3) := 0;

BEGIN

SELECT member\_id

INTO V\_MEMBER\_ID

FROM member

WHERE last\_name = V\_NUME;

SELECT COUNT(DISTINCT title\_id)

INTO V\_NR\_FILME

FROM rental

WHERE member\_id = V\_MEMBER\_ID;

DBMS\_OUTPUT.PUT\_LINE(V\_NR\_FILME);

SELECT COUNT(title) INTO num

FROM rental JOIN title USING (title\_id);

procent := (v\_member\_id / num) \* 100;

IF procent > 75 THEN

dbms\_output.put\_line('Categoria 1');

ELSIF procent > 50 THEN

dbms\_output.put\_line('Categoria 2');

ELSIF procent > 25 THEN

dbms\_output.put\_line('Categoria 3');

ELSE

dbms\_output.put\_line('Categoria 4');

END IF;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

DBMS\_OUTPUT.PUT\_LINE('None');

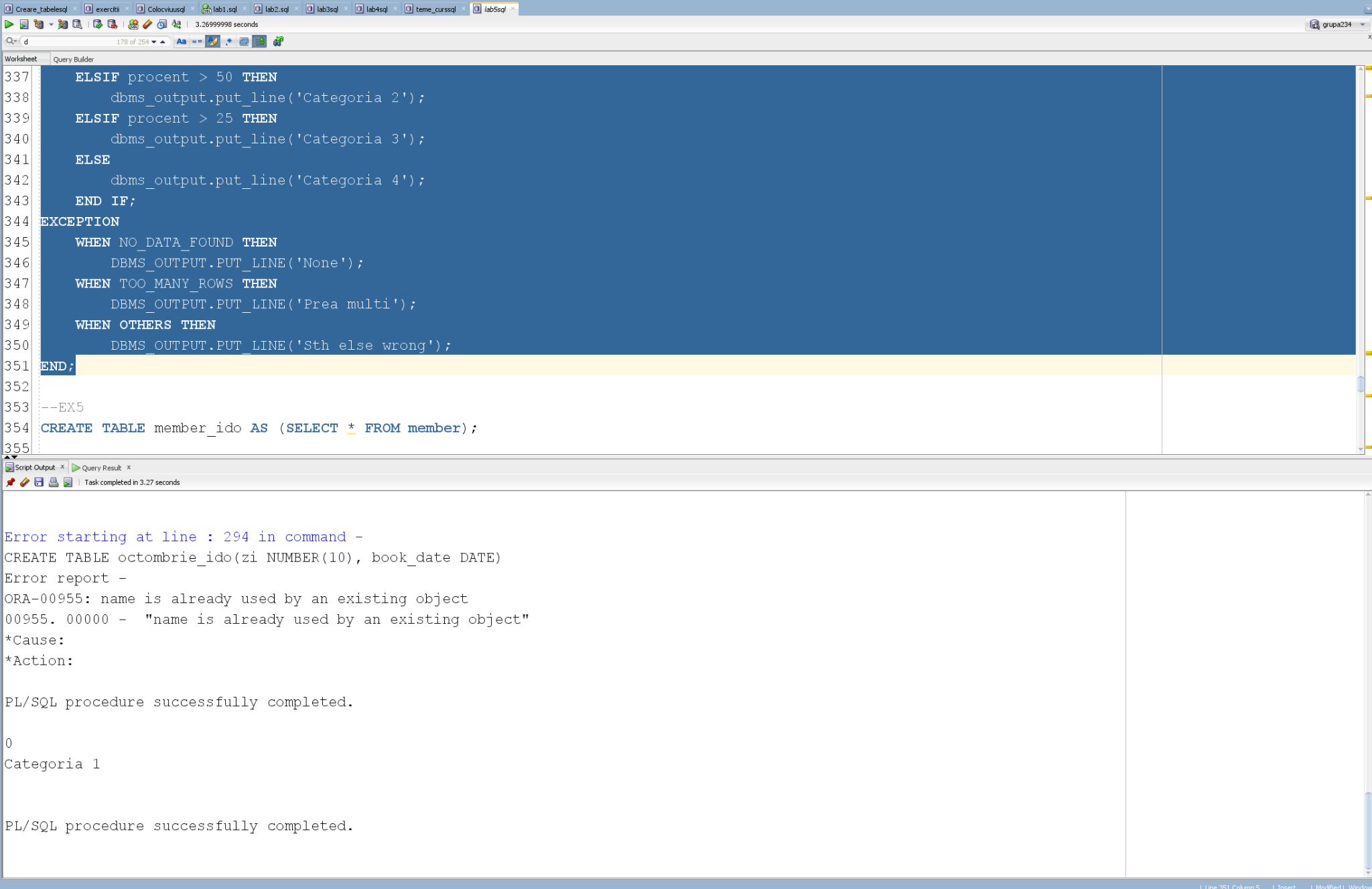
WHEN TOO\_MANY\_ROWS THEN

DBMS\_OUTPUT.PUT\_LINE('Prea multi');

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE('Sth else wrong');

END;



E5)

Creați tabelul member\_\*\*\* (o copie a tabelului member). Adăugați în acest tabel coloana discount, care va reprezenta procentul de reducere aplicat pentru membrii, în funcție de categoria din care fac parte aceștia: - 10% pentru membrii din Categoria 1 - 5% pentru membrii din Categoria 2 - 3% pentru membrii din Categoria 3 - nimic Actualizați coloana discount pentru un membru al cărui cod este dat de la tastatură. Afișați un mesaj din care să reiasă dacă actualizarea s-a produs sau nu.

CREATE TABLE member\_ido AS (SELECT \* FROM member);

ALTER TABLE member\_ido ADD CONSTRAINT PK\_member\_ido PRIMARY KEY (member\_id);

ALTER TABLE member\_ido

ADD discount NUMBER;

SET VERIFY OFF

DECLARE

id member\_ido.member\_id%TYPE := &id;

nr\_1 NUMBER;

nr\_2 NUMBER;

BEGIN

SELECT COUNT(\*)

INTO nr\_2

FROM title;

SELECT COUNT(DISTINCT title\_id)

INTO nr\_1

FROM rental r JOIN member\_ido m USING (member\_id)

GROUP BY member\_id

HAVING member\_id = id;

CASE WHEN nr\_1 \* 100 / nr\_2 >= 75 THEN

UPDATE member\_ido

SET DISCOUNT = 10

WHERE MEMBER\_ID = id;

COMMIT;

WHEN nr\_1 \* 100 / nr\_2 >= 50 THEN

UPDATE member\_ido

SET DISCOUNT = 5

WHERE MEMBER\_ID = id;

COMMIT;

WHEN nr\_1 \* 100 / nr\_2 >= 25 THEN

UPDATE member\_ido

SET DISCOUNT = 3

WHERE MEMBER\_ID = id;

COMMIT;

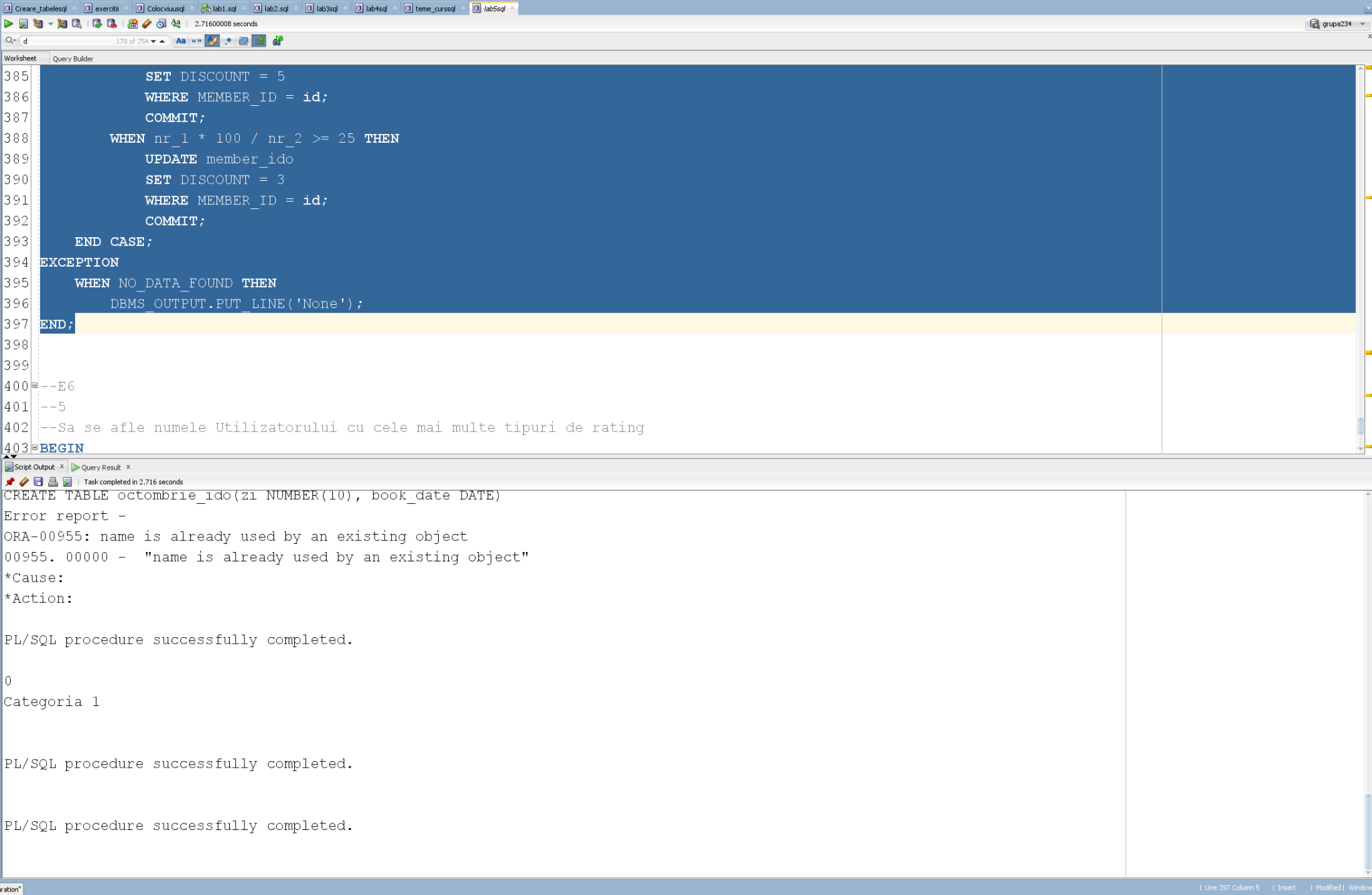
END CASE;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

DBMS\_OUTPUT.PUT\_LINE('None');

END;



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.

--5

--Sa se afle numele Utilizatorului cu cele mai multe tipuri de rating

BEGIN

SELECT username

INTO :rezultat

FROM Utilizatori u, Rating r

WHERE u.id\_utilizator = r.id\_rating

GROUP BY id\_utilizator

HAVING COUNT(\*) = (SELECT MAX(COUNT(\*))

FROM Utilizatori

GROUP BY id\_rating);

DBMS\_OUTPUT.PUT\_LINE('Utilizatorul '|| :rezultat);

END;

/

