Laborator 9:
SUBPROGRAME in PL/SQL (partea II - functii)

Un subprogram este un bloc PL/SQL cu nume (spre deosebire de blocurile anonime) care poate primi parametri şi poate fi invocat dintr-un anumit mediu (de exemplu, SQL*Plus, Oracle Forms, Oracle Reports etc.)

Subprogramele sunt bazate pe structura de bloc PL/SQL. Similar, ele conţin o parte declarativă opţională, o parte executabilă obligatorie şi o parte de tratare de excepţii opţională. Exista 2 tipuri de subprograme:

1. **proceduri**;
2. **funcţii** (trebuie să conţină cel puţin o comandă RETURN);

Subprogramele pot fi:

- **locale** (în cadrul unui alt bloc PL/SQL sau subprogram)
- **stocate** (create cu comanda CREATE) - odată create, procedurile şi funcţiile sunt stocate în baza de date, motiv pentru care se numesc subprograme stocate.

Sintaxa simplificată pentru crearea unei funcţii este următoarea:

```
[CREATE [OR REPLACE] ]  FUNCTION  nume_funcţie [ (lista_parametri) ]
   RETURN tip_de_date
   {IS | AS}
   [declaraţii locale]
BEGIN
   partea  executabilă
[EXCEPTION
   partea de tratare a excepţiilor]
END [nume_funcţie];
```

Lista de parametri conţine specificaţii de parametri separate prin virgulă de forma:

nume_parametru mod_parametru tip_parametru;

O functie îndeplineste urmatoarele condiţii:

- ✓ Acceptă numai parametri de tip IN
- ✓ Acceptă numai tipuri de date SQL, nu şi tipuri specifice PL/SQL
- ✓ Returnează valori de tipuri de date SQL.
- ✓ Nu modifică tabelul care este blocat pentru comanda respectiva (mutating tables)
- ✓ Poate fi folosită în lista de expresii a comenzii SELECT, în clauzele WHERE si HAVING, CONNECT BY, START WITH, ORDER BY, GROUP BY, clauza VALUES a comenzii INSERT, clauza SET a comenzii UPDATE.

În cazul în care se modifică un obiect (vizualizare, tabel etc) de care depinde un subprogram, acesta este invalidat. Revalidarea se face fie prin recrearea subprogramului fie prin comanda:

ALTER FUNCTION nume_functie COMPILE;

Ştergerea unei funcţii se realizează prin comenzile:

DROP PROCEDURE nume_proc;
DROP FUNCTION nume_functie;

Informaţii despre funcţiile deţinute de utilizatorul curent se pot obţine interogând vizualizarea USER_OBJECTS din dicţionarul datelor.

```
SELECT   OBJECT_NAME, OBJECT_TYPE, STATUS
FROM     USER_OBJECTS
WHERE    OBJECT_TYPE IN ('FUNCTION');
```

Obs: STATUS – starea subprogramului (validă sau invalidă).

Codul complet al unui subprogram poate fi vizualizat folosind următoarea sintaxă:

```
SELECT   TEXT
FROM     USER_SOURCE
WHERE    NAME = 'nume_subprogram'
ORDER BY LINE;
```

Eroarea apărută la compilarea unui subprogram poate fi vizualizată folosind următoarea sintaxă:

```
SELECT   LINE, POSITION, TEXT
FROM     USER_ERRORS
WHERE    NAME = 'nume';
```

Erorile pot fi vizualizate şi prin intermediul comenzii SHOW ERRORS.
Descrierea specificaţiei unui subprogram se face prin comanda DESCRIBE.

Când este apelată o procedură PL/SQL, sistemul **Oracle** furnizează două metode pentru definirea parametrilor actuali:

- specificarea explicită prin nume
- specificarea prin poziţie

Exemplu: **subprog(a tip_a, b_tip_b, c tip_c, d tip_d)**

- specificare prin poziţie: **subprog(var_a, var_b, var_c, var_d);**

- specificare prin nume: **subprog(b=>var_b, c=>var_c, d=>var_d, a=>var_a);**

- specificare prin nume şi poziţie: **subprog(var_a, var_b, d=>var_d, c=>var_c);**

Probleme rezolvate

a). Funcţii locale

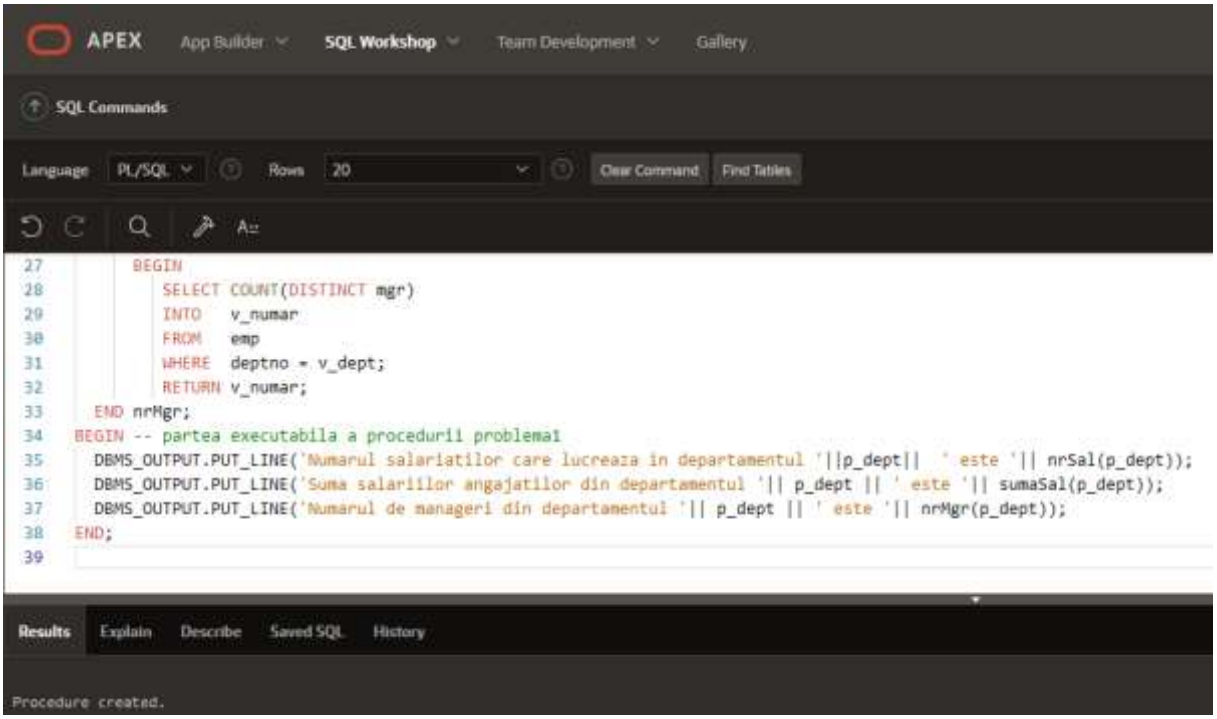
1. Să se creeze o procedură stocată care pentru un anumit cod de departament (dat ca parametru) calculează prin intermediul unor funcţii locale numărul de salariaţi care lucrează în el, suma salariilor şi numărul managerilor salariaţilor care lucrează în departamentul respectiv.

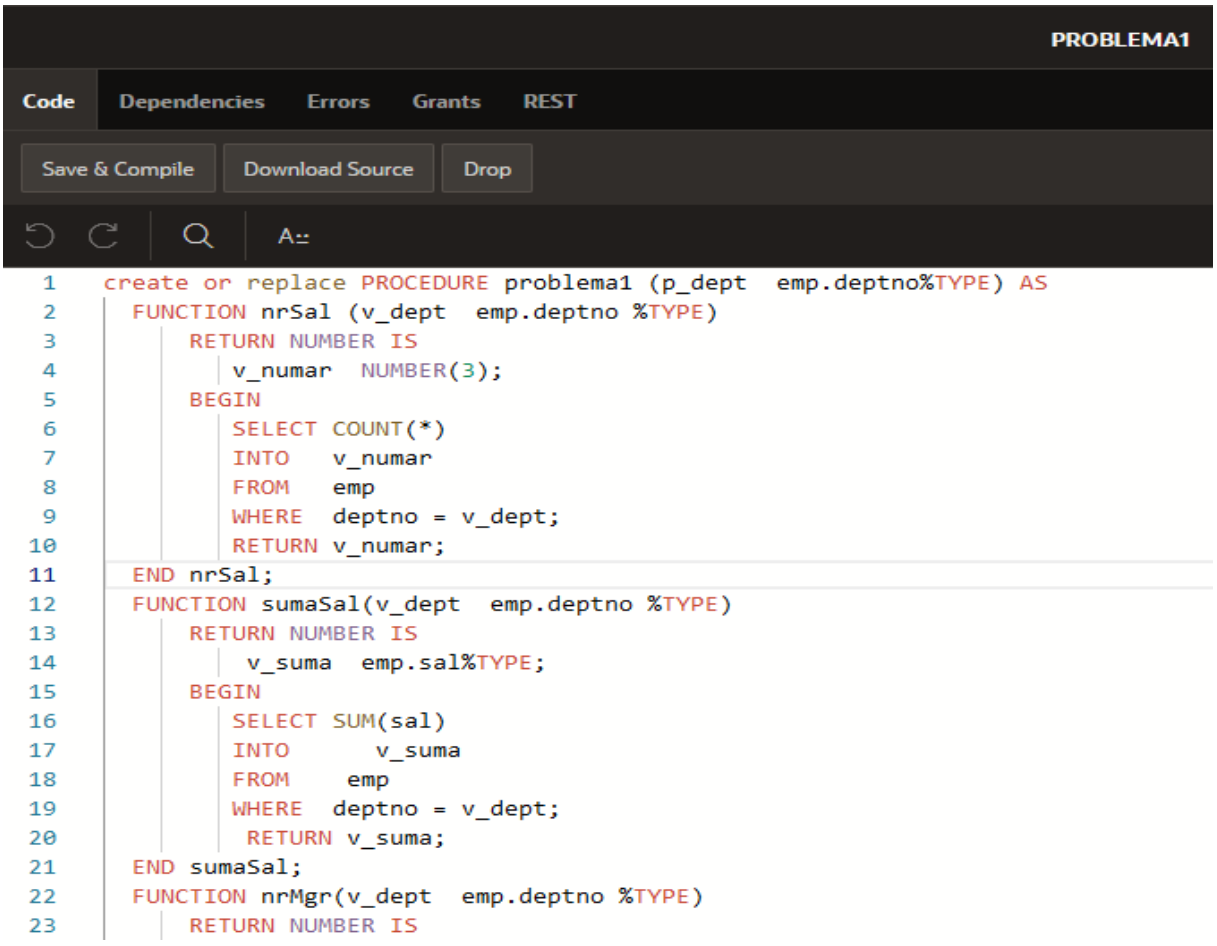
Solutie:

```
CREATE OR REPLACE PROCEDURE problema1 (p_dept  emp.deptno%TYPE) AS
 FUNCTION nrSal (v_dept  emp.deptno %TYPE)
    RETURN NUMBER IS
      v_numar  NUMBER(3);
    BEGIN
      SELECT COUNT(*)
      INTO   v_numar
      FROM   emp
      WHERE  deptno = v_dept;
      RETURN v_numar;
 END nrSal;

 FUNCTION sumaSal(v_dept  emp.deptno %TYPE)
    RETURN NUMBER IS
       v_suma  emp.sal%TYPE;
    BEGIN
      SELECT SUM(sal)
      INTO      v_suma
      FROM    emp
      WHERE  deptno = v_dept;
       RETURN v_suma;
 END sumaSal;

 FUNCTION nrMgr(v_dept  emp.deptno %TYPE)
    RETURN NUMBER IS
      v_numar  NUMBER(3);
    BEGIN
      SELECT COUNT(DISTINCT mgr)
      INTO   v_numar
      FROM   emp
      WHERE  deptno = v_dept;
      RETURN v_numar;
 END nrMgr;
```

```
BEGIN -- partea executabila a procedurii problema1_pnu
 DBMS_OUTPUT.PUT_LINE('Numarul salariatilor care lucreaza in departamentul '||p_dept||  ' este '|| nrSal(p_dept));
 DBMS_OUTPUT.PUT_LINE('Suma salariilor angajatilor din departamentul '|| p_dept || ' este '|| sumaSal(p_dept));
```

```
 DBMS_OUTPUT.PUT_LINE('Numarul de manageri din departamentul '|| p_dept || ' este '|| nrMgr(p_dept));
END;
```

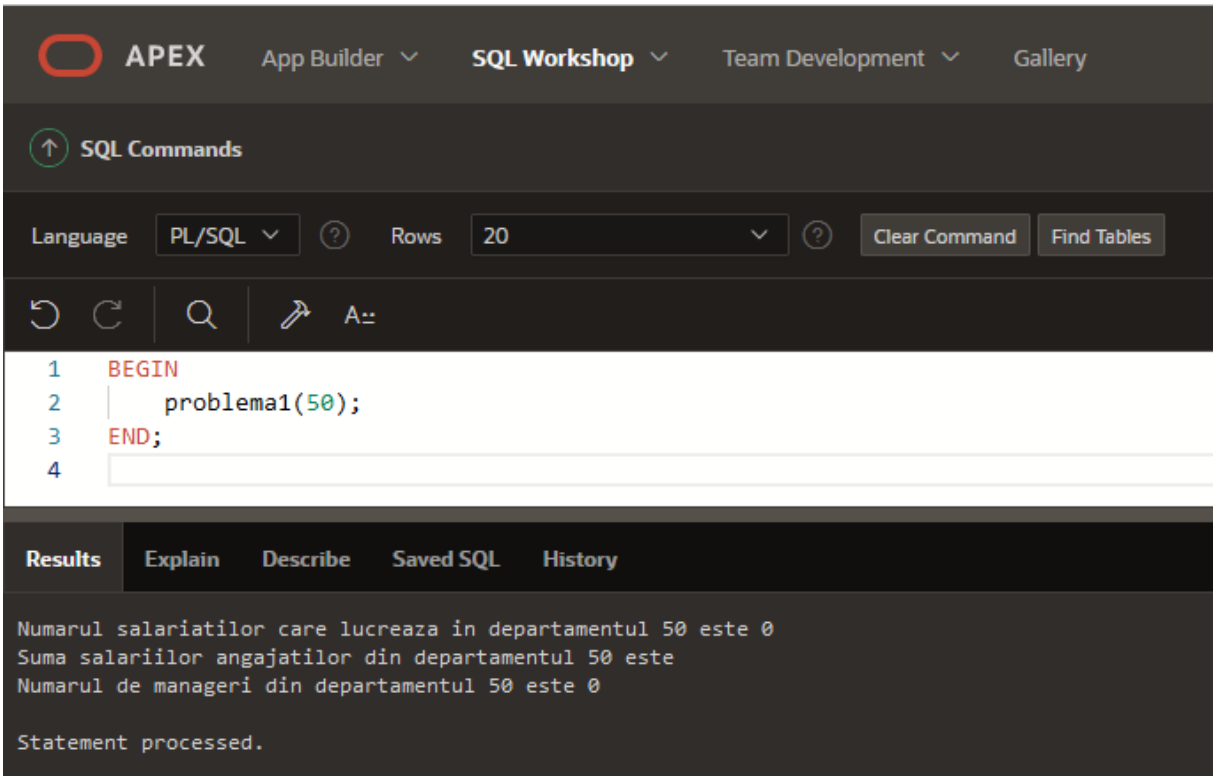




```
22   FUNCTION nrMgr(v_dept  emp.deptno %TYPE)
23      RETURN NUMBER IS
24         v_numar  NUMBER(3);
25      BEGIN
26         SELECT COUNT(DISTINCT mgr)
27         INTO   v_numar
28         FROM   emp
29         WHERE  deptno = v_dept;
30         RETURN v_numar;
31   END nrMgr;
32 BEGIN -- partea executabila a procedurii problema1_pnu
33   DBMS_OUTPUT.PUT_LINE('Numarul salariatilor care lucreaza in departamentul '||p_dept||  ' este '|| nrSal(p_dept));
34   DBMS_OUTPUT.PUT_LINE('Suma salariilor angajatilor din departamentul '|| p_dept || ' este '|| sumaSal(p_dept));
35   DBMS_OUTPUT.PUT_LINE('Numarul de manageri din departamentul '|| p_dept || ' este '|| nrMgr(p_dept));
36 END;
```

Pasul 2 – Executia procedurii construite prin apelare intr-un bloc anonim:

```
BEGIN
      problema1(50);
END;
```

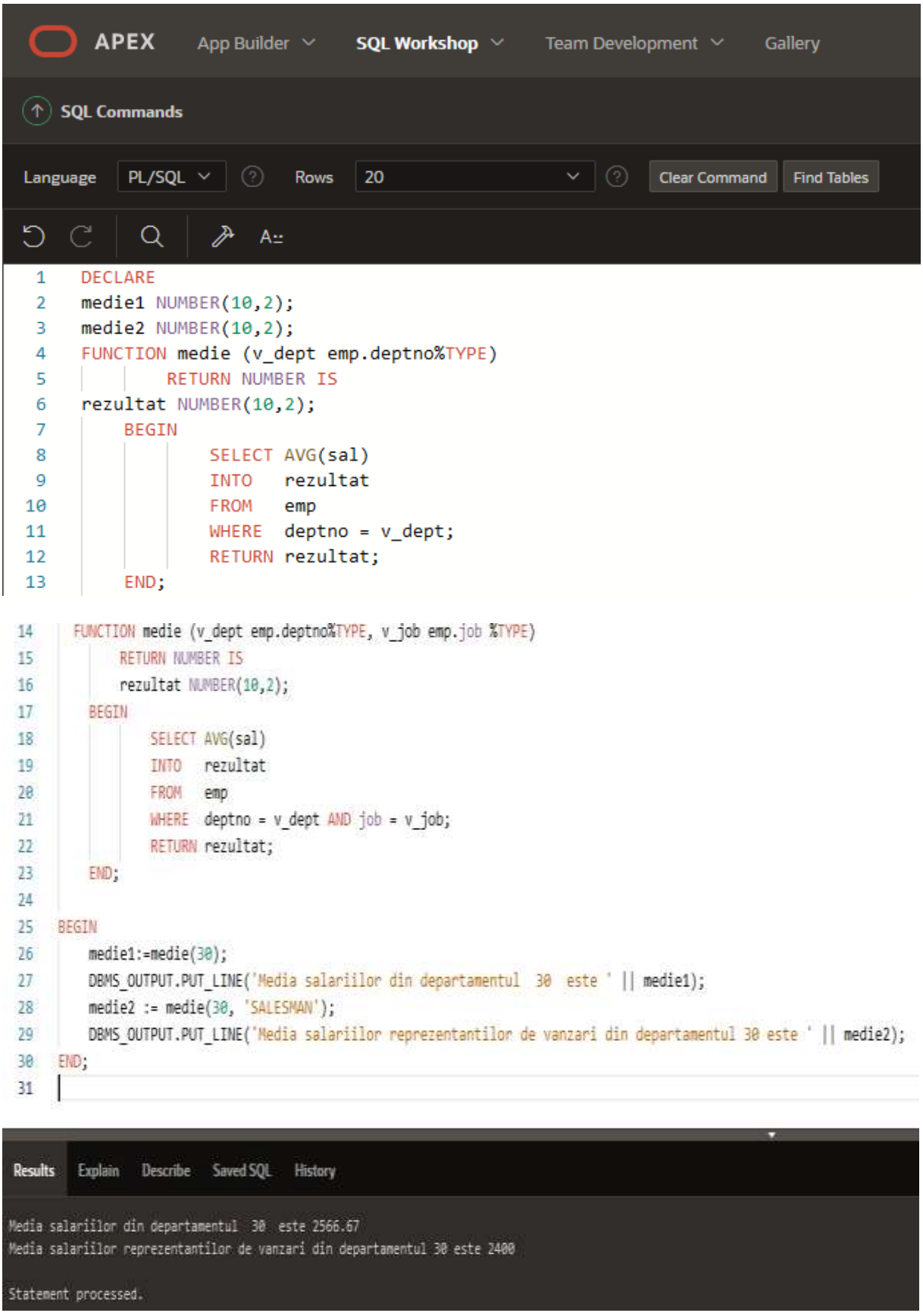


2. Să se creeze două funcţii (locale) supraîncărcate (overloaded) care să calculeze media salariilor astfel:
- prima funcţie va avea ca argument codul departamentului, adică funcţia calculează media salariilor din departamentul specificat
- a doua funcţie va avea două argumente, unul reprezentând codul departamentului, iar celălalt reprezentând job-ul, adică funcţia va calcula media salariilor dintr-un anumit departament şi care aparţin unui job specificat

Solutie:

```
DECLARE
        medie1 NUMBER(10,2);
        medie2 NUMBER(10,2);
FUNCTION medie (v_dept emp.deptno%TYPE)
        RETURN NUMBER IS
        rezultat NUMBER(10,2);
        BEGIN
                SELECT AVG(sal)
                INTO   rezultat
                FROM   emp
                WHERE  deptno = v_dept;
                RETURN rezultat;
        END;
 FUNCTION medie (v_dept emp.deptno%TYPE, v_job emp.job %TYPE)
        RETURN NUMBER IS
        rezultat NUMBER(10,2);
        BEGIN
                SELECT AVG(sal)
                INTO   rezultat
                FROM   emp
                WHERE  deptno = v_dept AND job = v_job;
                RETURN rezultat;
        END;
```

```
BEGIN
        medie1:=medie(30);
        DBMS_OUTPUT.PUT_LINE('Media salariilor din departamentul  30  este ' ||
medie1);
        medie2 := medie(30, 'SALESMAN');
        DBMS_OUTPUT.PUT_LINE('Media salariilor reprezentantilor de vanzari din
departamentul 30 este ' || medie2);
END;
```

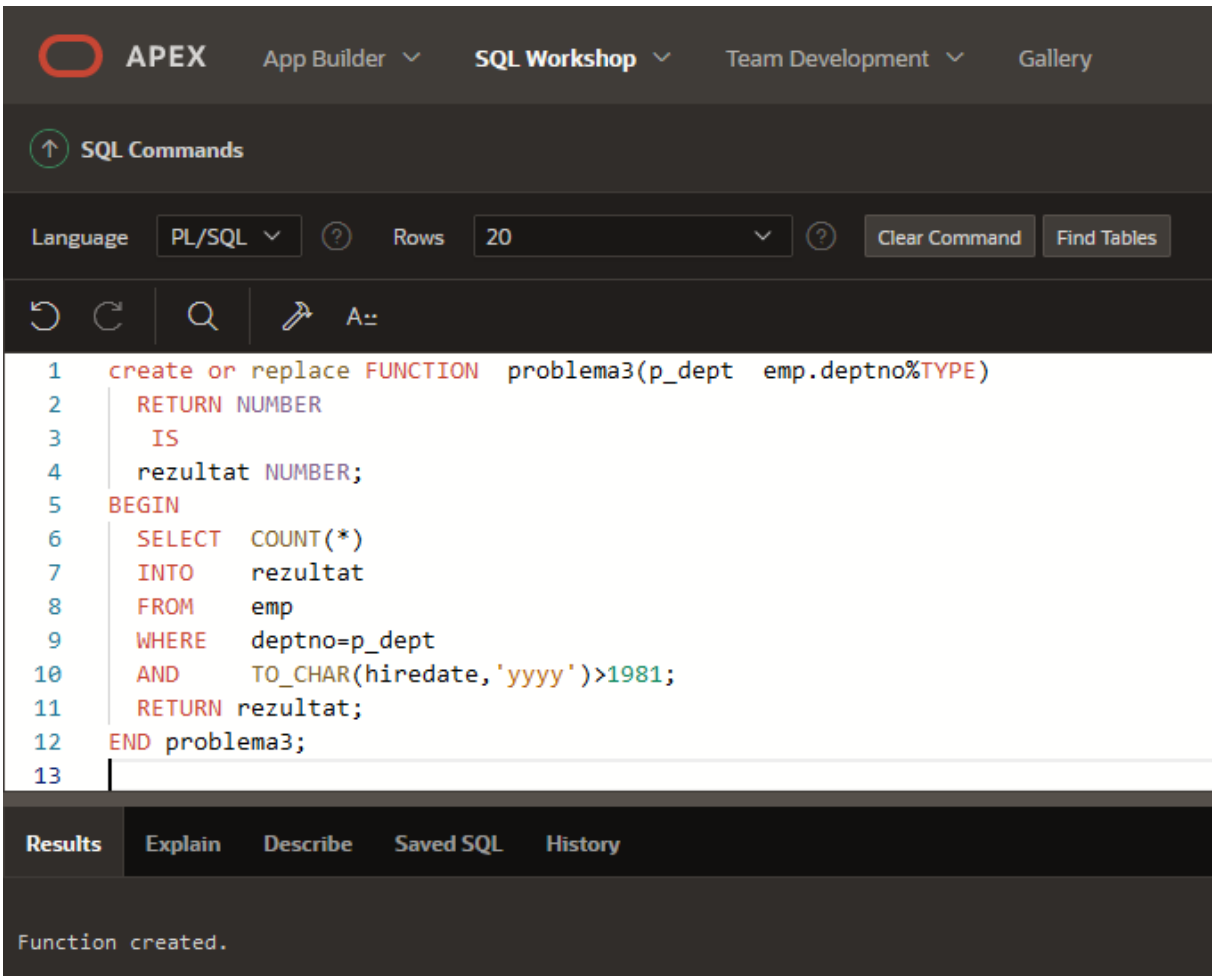

APEX
App Builder
SQL Workshop
Team Development
Gallery
SQL Commands
Language
PL/SQL
Rows
20
Clear Command
Find Tables
DECLARE
medie1 NUMBER(10,2);
medie2 NUMBER(10,2);
FUNCTION medie (v_dept emp.deptno%TYPE)
RETURN NUMBER IS
rezultat NUMBER(10,2);
BEGIN
SELECT AVG(sal)
INTO rezultat
FROM emp
WHERE deptno = v_dept;
RETURN rezultat;
END;
FUNCTION medie (v_dept emp.deptno%TYPE, v_job emp.job %TYPE)
RETURN NUMBER IS
rezultat NUMBER(10,2);
BEGIN
SELECT AVG(sal)
INTO rezultat
FROM emp
WHERE deptno = v_dept AND job = v_job;
RETURN rezultat;
END;
BEGIN
medie1:=medie(30);
DBMS_OUTPUT.PUT_LINE('Media salariilor din departamentul 30 este ' || medie1);
medie2 := medie(30, 'SALESMAN');
DBMS_OUTPUT.PUT_LINE('Media salariilor reprezentantilor de vanzari din departamentul 30 este ' || medie2);
END;
Results
Explain
Describe
Saved SQL
History
Media salariilor din departamentul 30 este 2566.67
Media salariilor reprezentantilor de vanzari din departamentul 30 este 2400
Statement processed.

b) Funcţii stocate

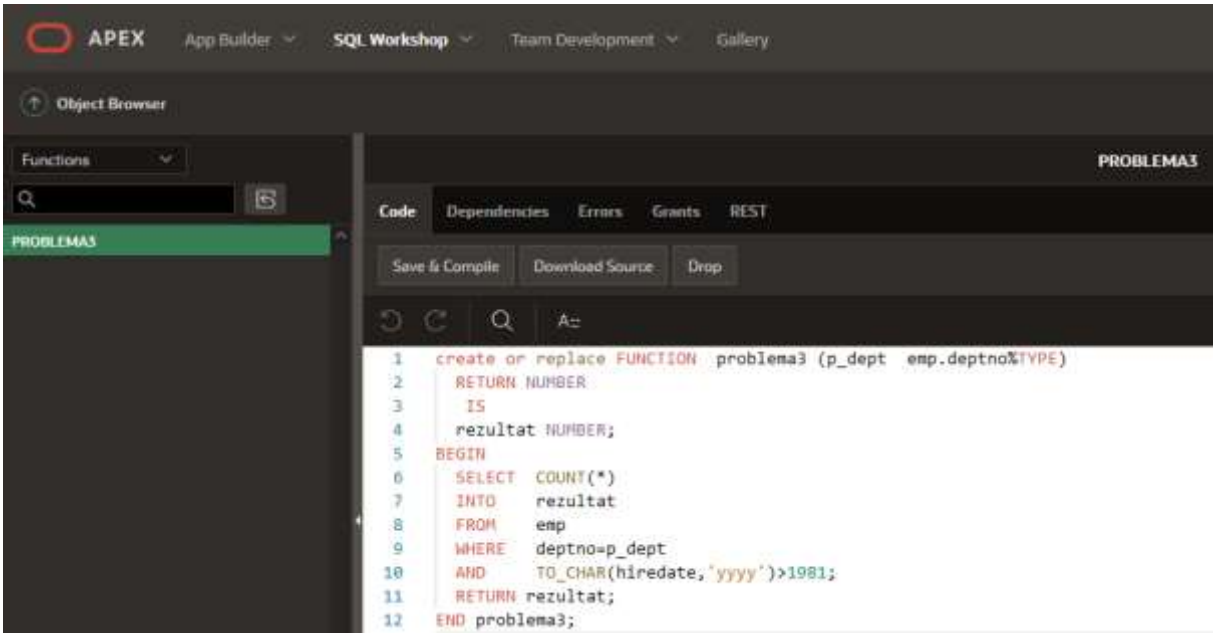
3. Să se creeze o funcţie stocată care determină numărul de salariaţi din emp angajaţi după 1981, într-un departament dat ca parametru. Să se apeleze această funcţie prin diferite modalităţi:
- printr-o comandă SELECT;
- într-un bloc PL/SQL.

Solutie:
Pasul 1 – Crearea functiei problema3:

```
CREATE OR REPLACE FUNCTION  problema3 (p_dept  emp.deptno%TYPE)
 RETURN NUMBER
  IS
 rezultat NUMBER;
BEGIN
 SELECT  COUNT(*)
 INTO    rezultat
 FROM    emp
 WHERE   deptno=p_dept
 AND     TO_CHAR(hire_date,'yyyy')>1981;
 RETURN rezultat;
END problema3;
```

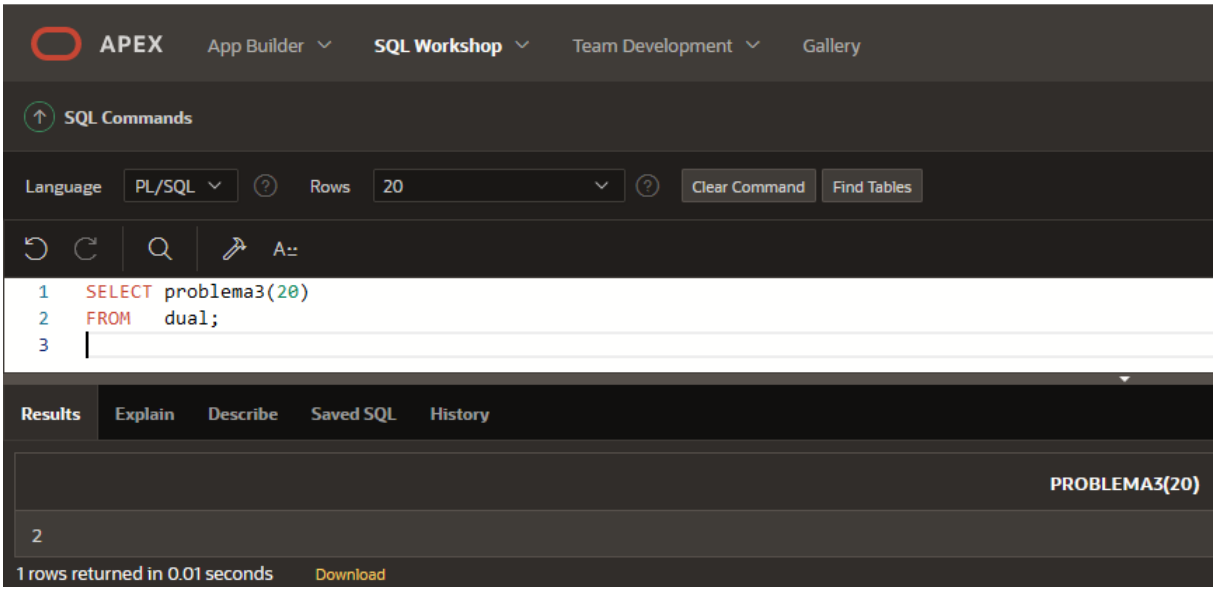


Pasul 2 – Verificarea stocarii functiei. Se acceseaza **OBJECT BROWSER**; se selecteaza **FUNCTIONS**, iar apoi *se selecteaza numele functiei dorite din lista de functii*. Continutul codului functiei va aparea in zona principala a ecrestrei respective:



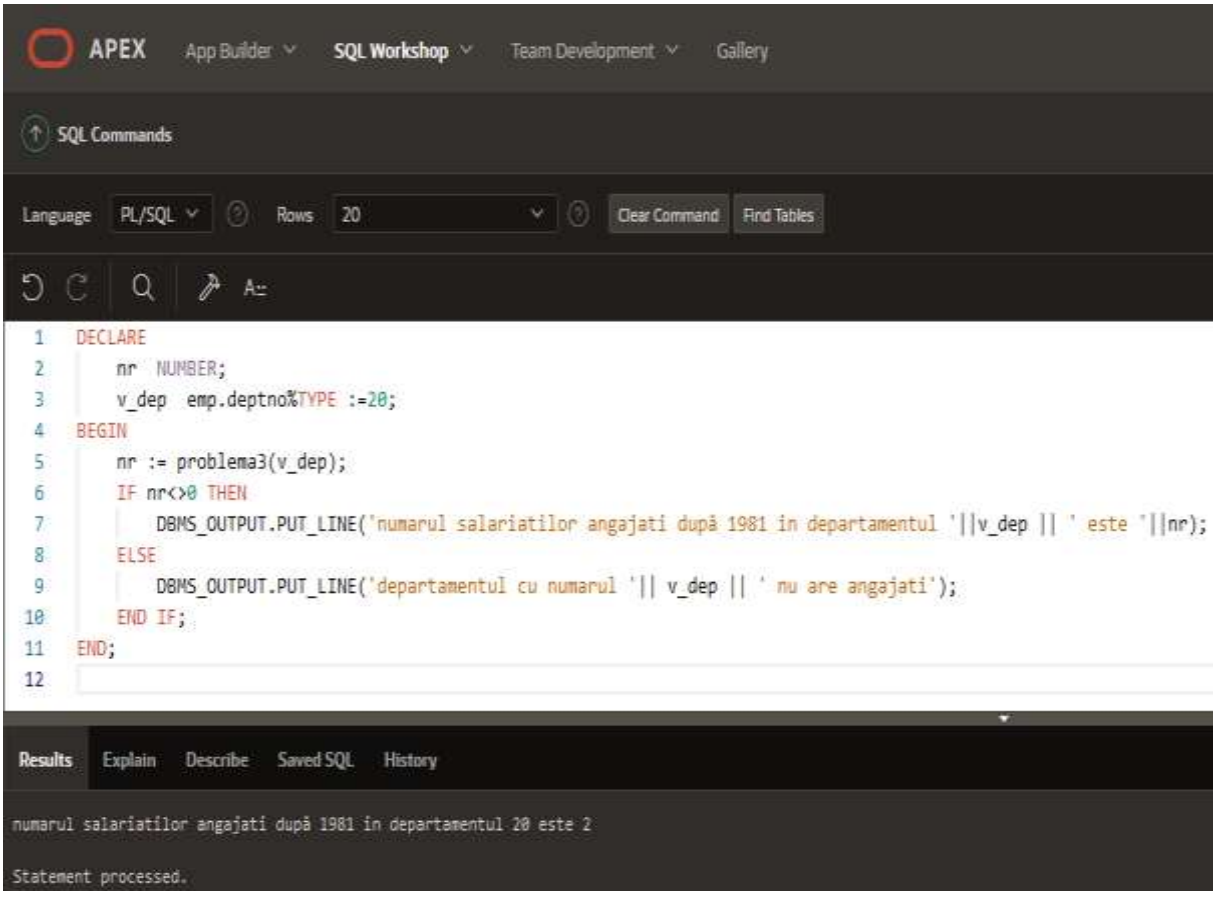
Pasul 3 – Se executa functia construita la pasul 1, printr-un apel in care parametrul este 20 cu semnificatia ca se prelucreaza datele pentru departamentul cu numarul 20.

1)
SELECT problema3(20)
FROM dual;



Pasul 4 – Se construieste un bloc anonim in care se apeleaza functia construita la pasul 1 cu un parametru dat ca variabila.

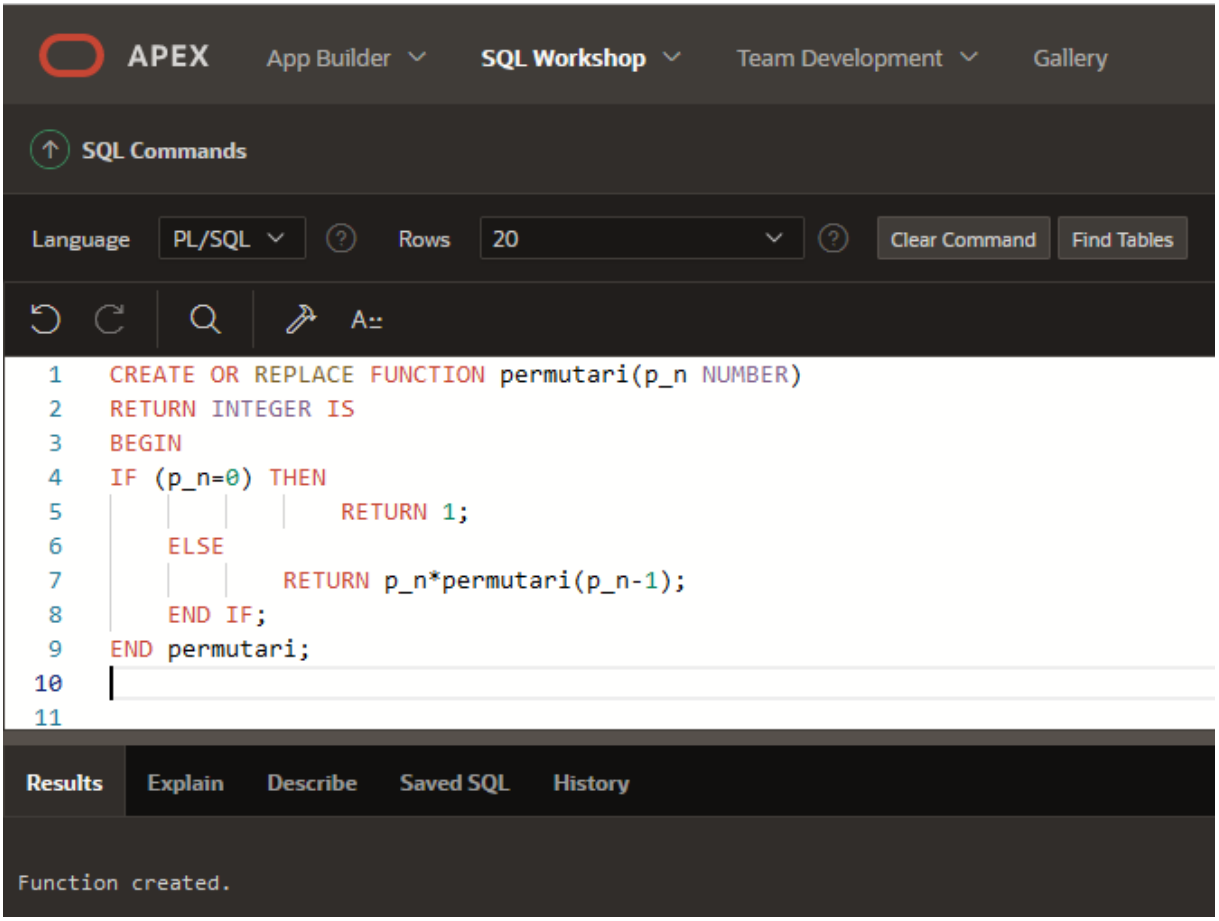
```
2)
DECLARE
   nr  NUMBER;
   v_dep  emp.deptno%TYPE :=20;
BEGIN
        nr := problema3(v_dep);
        IF nr<>0 THEN
        DBMS_OUTPUT.PUT_LINE('numarul salariatilor angajati după 1981 in departamentul '||v_dep || ' este '||nr);
        ELSE
        DBMS_OUTPUT.PUT_LINE('departamentul cu numarul '|| v_dep || ' nu are angajati');
        END IF;
END;
```



4. Să se calculeze recursiv numărul de permutări ale unei mulţimi cu n elemente, unde n va fi transmis ca parametru.

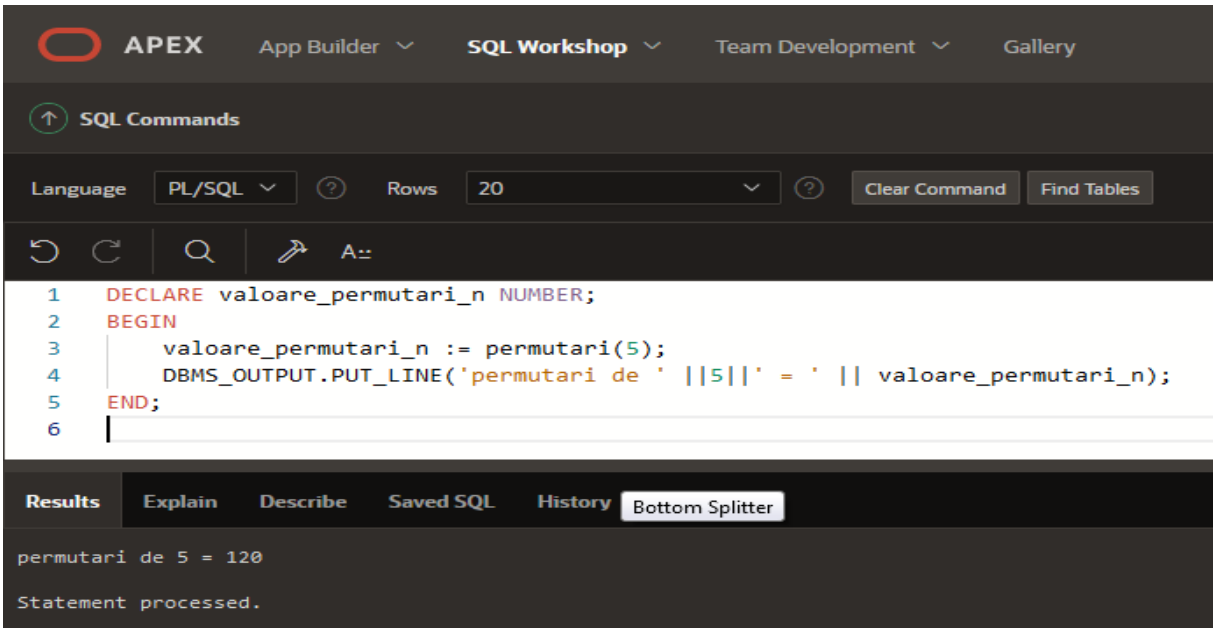
Solutie:
Pasul 1 – Se construieste functia permutari.

```
CREATE OR REPLACE FUNCTION permutari(p_n NUMBER)
RETURN INTEGER IS
BEGIN
IF (p_n=0) THEN
                RETURN 1;
        ELSE
                RETURN p_n*permutari(p_n-1);
        END IF;
END permutari;
```



Pasul 2 – Intr-un bloc anonim se apeleaza functia construita la pasul 1 cu parametrul 5.

```
DECLARE valoare_permutari_n NUMBER;
BEGIN
   valoare_permutari_n := permutari(5);
   DBMS_OUTPUT.PUT_LINE('permutari de ' ||5||' = ' || valoare_permutari_n);
END;
```

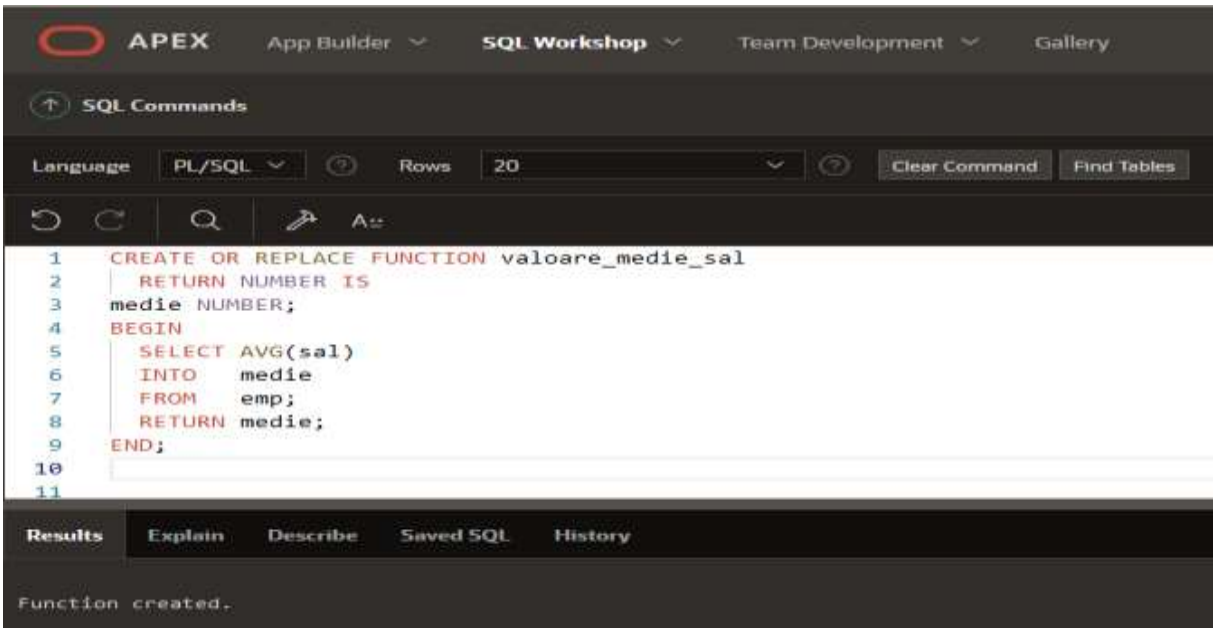


5. Să se afişeze numele, job-ul şi salariul angajaţilor al căror salariu este mai mare decât media salariilor din tabelul emp.

Solutie:

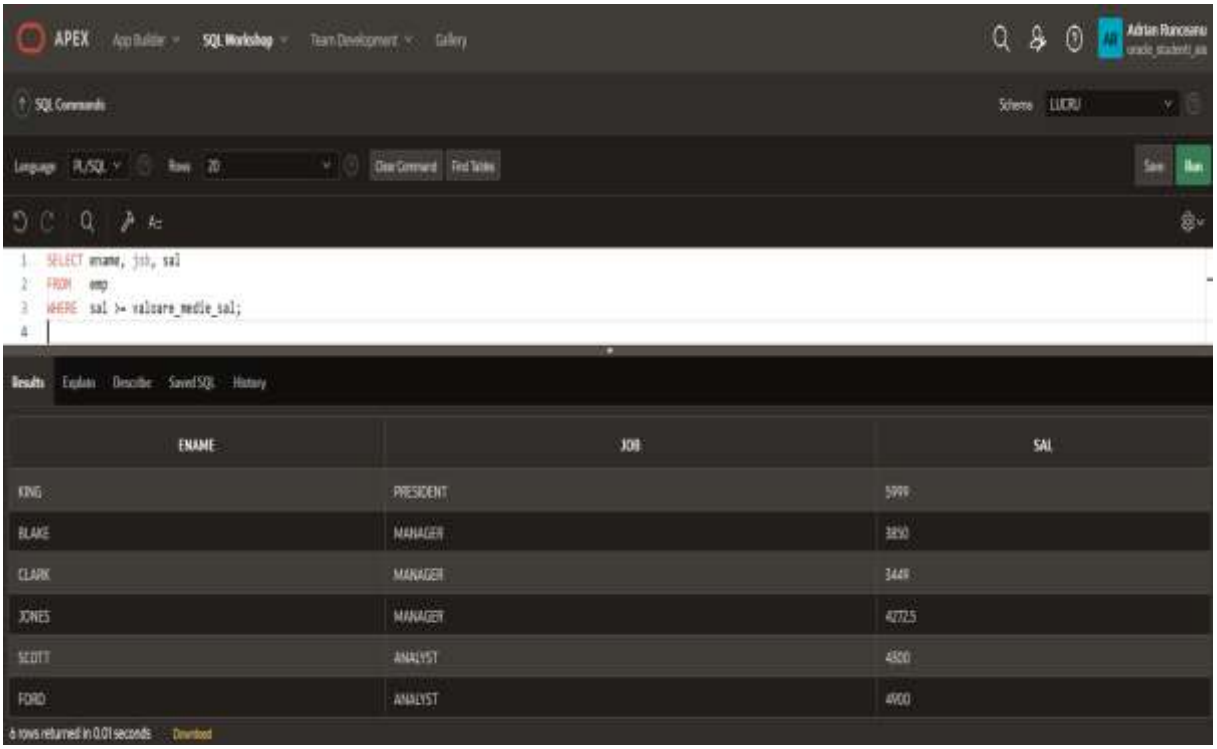
Pasul 1 – Se creeaza functia valoare_medie_sal pentru calculul mediei salariilor angajatilor din tabela EMP.

```
CREATE OR REPLACE FUNCTION valoare_medie_sal
 RETURN NUMBER IS
medie NUMBER;
BEGIN
 SELECT AVG(sal)
 INTO     medie
 FROM   emp;
 RETURN medie;
END;
```



Pasul 2 – Se scrie o interogare care rezolva cerinta problemei.

```
SELECT ename, job, sal
FROM   emp
WHERE  sal >= valoare_medie_sal;
```



6. a) Analizaţi subprogramele create anterior în vizualizarea din dicţionarul datelor USER_OBJECTS.

b) Regăsiţi codul unuia dintre subprogramele create anterior în vizualizarea USER_SOURCE.

c) Aflaţi tipul parametrilor uneia dintre procedurile create anterior utilizând comanda DESCRIBE.

Probleme propuse spre rezolvare

1. Să se creeze o funcţie stocată care determină numărul de salariaţi care au fost angajaţi ulterior tuturor subalternilor unui manager al cărui cod este dat ca parametru. Să se apeleze această funcţie într-un bloc PL/SQL.

2. Să se creeze trei funcţii locale cu acelaşi nume care să calculeze numărul de salariaţi astfel:
- prima funcţie va avea ca argument codul departamentului, adică funcţia calculează numărul de salariaţi din departamentul specificat;
- a doua funcţie va avea două argumente, unul reprezentând codul departamentului, iar celălalt reprezentând anul angajării, adică funcţia va calcula numărul de salariaţi din departament şi care au fost angajaţi într-un anumit an;
- a treia funcţie va avea trei argumente, unul reprezentând codul departamentului, unul reprezentând anul angajării, iar celălalt grila de salarizare, adică funcţia va calcula numărul de salariaţi din departament, care au fost angajaţi într-un anumit an şi au salariul într-o anumită grilă de salarizare.

3. Să se modifice salariul unui angajat al cărui cod este introdus ca parametru astfel încât să devină media salariilor angajaţilor care câştigă comision dintr-un departament dat ca parametru.

4. Să se scrie o funcţie care să întoarcă, pentru un angajat al cărui cod este specificat ca parametru, vechimea exprimată în luni. Să se utilizeze funcţia într-o instrucţiune SELECT care să întoarcă numele angajaţilor, salariul şi numărul de luni lucrate.

Bibliografie web:

https://www.w3resource.com/
https://www.bullraider.com/database/pl-sql/pl-sql-examples
https://www.oracletutorial.com/plsql-tutorial/plsql-function/
https://docs.oracle.com/database/121/LNPLS/subprograms.htm#LNPLS00825