

# Proiect baze de date

Nume: Liță Robert Cristian

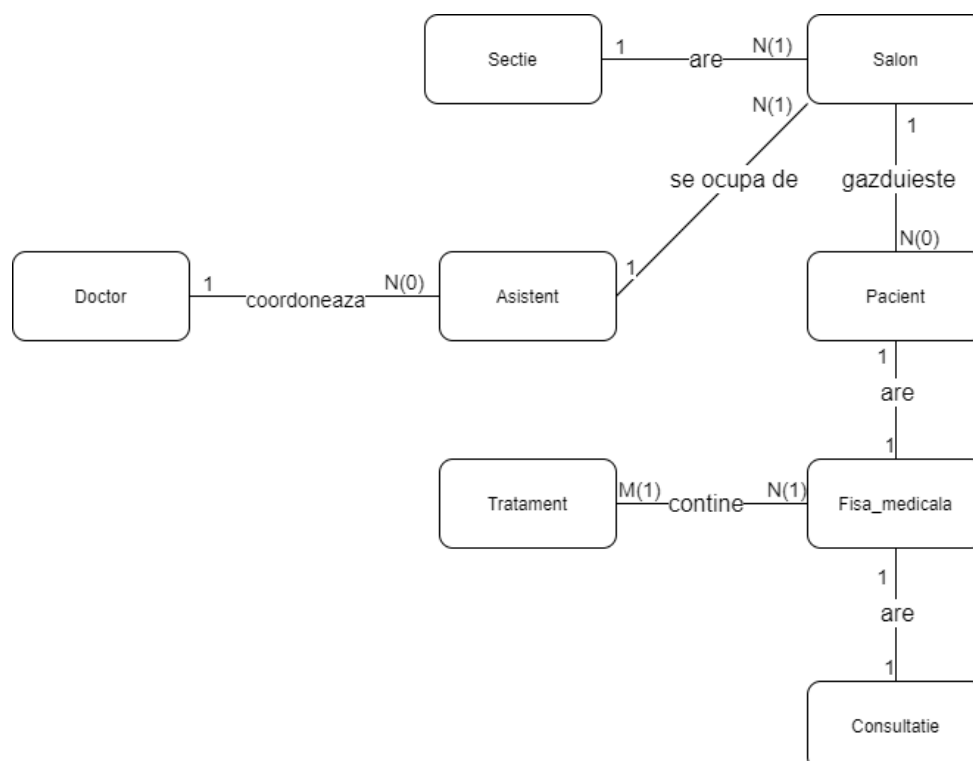
Grupă: 252

## 1) Descrierea bazei de date

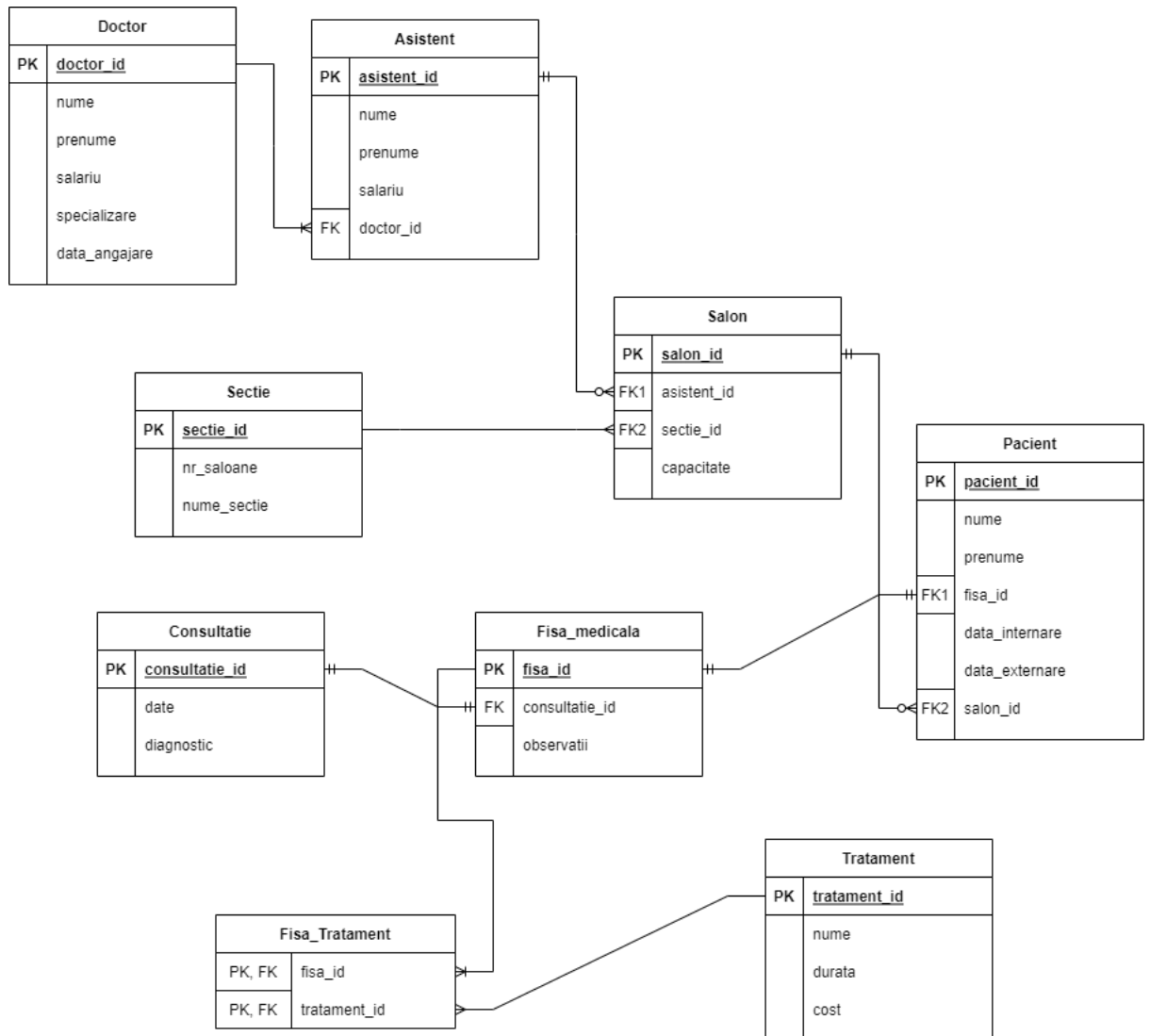
Am ales să construiesc o bază de date care să gestioneze activitatea unui spital. La spital vin pacienți, cu diferite afecțiuni, sunt supuși unei consultații, iar în funcție de diagnosticul rezultat, li se alcătuieste o fișă care conține diagnosticul, tratamentul pe care acesta trebuie să îl urmeze (intervenție chirurgicală, medicamente, regim alimentar etc.) și eventuale observații. Apoi acesta este internat la secția din spital corespunzătoare afecțiunii de care suferă, într-un salon, de care se ocupă un asistent, subordonat unui doctor.

Baza de date conține 8 entități independente (DOCTOR, ASISTENT, SALON, SECȚIE, PACIENT, CONSULTAȚIE, TRATAMENT, FIȘĂ-MEDICALĂ) dar și o tabelă asociativă între fișă medicală și tratament (FIȘĂ-TRATAMENT), rezultată din relația M-M a acestora (o fișă poate avea mai multe tratamente, atunci când pacientul suferă de mai multe afecțiuni, dar și un tratament poate apărea pe mai multe fișe).

## 2) Diagrama entitate-relație (ERD)



### 3) Diagrama conceptuală



### 4) Implementarea bazei de date în Oracle

```

create table DOCTOR(
  DOCTOR_ID number(15) constraint pk_doc PRIMARY KEY,
  NUME varchar2(30) NOT NULL,
  PRENUME varchar2(30) NOT NULL,
  SALARIU number(10,2) constraint ck_sal_valid CHECK(salariu > 0),
  SPECIALIZARE varchar2(30) NOT NULL,
  DATA_ANGAJARE date NOT NULL
);
  
```

```

create table ASISTENT(
  ASISTENT_ID number(15) constraint pk_asis PRIMARY KEY,
  
```

```

    NUME varchar2(30) NOT NULL,
    PRENUME varchar2(30) NOT NULL,
    SALARIU number(10,2) constraint ck_sal_valid_as CHECK(salariu >
0),
    DOCTOR_ID number(15) constraint fk_doc references
doctor(doctor_id) on delete cascade
);

```

```

create table CONSULTATIE(
    CONSULTATIE_ID number(15) constraint pk_cons PRIMARY KEY,
    DATA date NOT NULL,
    DIAGNOSTIC varchar2(30) NOT NULL
);

```

```

create table TRATAMENT(
    TRATAMENT_ID number(15) constraint pk_trat PRIMARY KEY,
    NUME varchar2(30) NOT NULL,
    DURATA number(5) NOT NULL,
    COST number(10,2) constraint ck_cost_valid CHECK(cost >= 0)
);

```

```

create table FISA_MEDICALA(
    FISA_ID number(15) constraint pk_fisa_med PRIMARY KEY,
    CONSULTATIE_ID number(15) constraint fk_cons references
consultatie(consultatie_id) on delete cascade,
    OBSERVATII varchar2(200)
);

```

```

create table FISA_TRATAMENT(
    TRATAMENT_ID number(30) constraint pk_ft_trat references
tratament(tratament_id) on delete cascade,
    FISA_ID number(30) constraint pk_ft_fisa references
fisa_medicala(fisa_id) on delete cascade,
    constraint pk_ft PRIMARY KEY (tratament_id, fisa_id)
);

```

```

create table SECTIE(
    SECTIE_ID number(15) constraint pk_sectie PRIMARY KEY,
    NR_SALOANE number(5) NOT NULL,
    NUME_SECTIE varchar2(30) NOT NULL
);

```

```

create table SALON(
    SALON_ID number(15) constraint pk_salon PRIMARY KEY,
    ASISTENT_ID number(15) constraint fk_salon_asis references
asistent(asistent_id) on delete cascade,
    CAPACITATE number(10) constraint ck_cap_valid CHECK(capacitate >
0),
    SECTIE_ID number(15) constraint fk_sal_sec references
sectie(sectie_id) on delete cascade
);

```

```

create table PACIENT(
    PACIENT_ID number(15) constraint pk_pac PRIMARY KEY,
    NUME varchar2(30) NOT NULL,
    PRENUME varchar2(30) NOT NULL,
    FISA_ID number(15) constraint fk_pac_fm references
fisa_medicala(fisa_id) on delete cascade,
    SALON_ID number(15) constraint fk_pac_sal references
salon(salon_id) on delete cascade,
    DATA_INTERNARE date NOT NULL,
    DATA_EXTERNARE date
);

```

## 5) Adăugarea informațiilor în tabele

```

INSERT INTO doctor (doctor_id,nume, prenume, salariu,
specializare, data_angajare)
VALUES (1, 'POPESCU', 'MIHNEA', 45000.99, 'ORTODONTIE',
'05.07.2021');
    INSERT INTO doctor (doctor_id,nume, prenume, salariu,
specializare, data_angajare)
VALUES (2, 'GEORGESCU', 'VASILE', 40000.5, 'NEUROLOGIE',
'25.10.2020');
    INSERT INTO doctor (doctor_id,nume, prenume, salariu,
specializare, data_angajare)
VALUES (3, 'OPREA', 'CORNEL', 12043.22, 'CHIRURGIE',
'24.12.2020');
    INSERT INTO doctor (doctor_id,nume, prenume, salariu,
specializare, data_angajare)
VALUES (4, 'ISTRATE', 'MIHAELA', 22222.22, 'CARDIOLOGIE',
'15.03.2021');
    INSERT INTO doctor (doctor_id,nume, prenume, salariu,

```

```

specializare, data_angajare)
VALUES (5, 'STANCU', 'LACRAMIOARA', 15500, 'RADIOLOGIE',
'30.05.2021');
    INSERT INTO doctor (doctor_id,nume, prenume, salariu,
specializare, data_angajare)
VALUES (6, 'VALERIU', 'STEFAN', 10000, 'CHIRURGIE', '07.01.2013');
    INSERT INTO doctor (doctor_id,nume, prenume, salariu,
specializare, data_angajare)
VALUES (7, 'MIHAILA', 'GETA', 9000, 'CHIRURGIE', '04.05.2020');

INSERT INTO asistent
VALUES (1, 'POPA', 'ALEXANDRU', 8000, 5);
    INSERT INTO asistent
VALUES (2, 'POPESCU', 'VICTORIA', 9000, 1);
    INSERT INTO asistent
VALUES (3, 'VASILE', '      MARIA', 4550, 4);
    INSERT INTO asistent
VALUES (4, 'TOMESCU', 'ROBERT', 5500, 2);
    INSERT INTO asistent
VALUES (5, 'IONESCU', '      TEODORA', 3245, 3);
INSERT INTO sectie VALUES (1, 10, 'DERMATOLOGIE');
INSERT INTO sectie VALUES (2, 5, 'NEUROLOGIE');
INSERT INTO sectie VALUES (3, 7, 'TERAPIE INTENSIVA');
INSERT INTO sectie VALUES (4, 6, 'NASTERI');
INSERT INTO sectie VALUES (5, 4, 'FRACTURI');
INSERT INTO salon VALUES (1, 4, 10, 2);
INSERT INTO salon VALUES (2, 1, 6, 2);
INSERT INTO salon VALUES (3, 5, 8, 2);
INSERT INTO salon VALUES (4, 5, 3, 2);
INSERT INTO salon VALUES (5, 4, 5, 3);
INSERT INTO tratament VALUES (1, 'REGIM ALIMENTAR', 14, 9000);
INSERT INTO tratament VALUES (2, 'MEDICAMENTE', 7, 200);
INSERT INTO tratament VALUES (3, 'INTERVENTIE CHIRURGICALA', 1,
1200);
INSERT INTO tratament VALUES (4, 'REPAUS FIZIC', 70, 0);
INSERT INTO tratament VALUES (5, 'INTERVENTIE CHIRURGICALA', 1,
1120.5);
INSERT INTO fisa_medicala VALUES (1, 2, null);
INSERT INTO fisa_medicala VALUES (2, 3, 'Dupa operatie, pacientul
va avea nevoie de sedinte de recuperare a mobilitatii. ');
INSERT INTO fisa_medicala VALUES (3, 4, 'Pacientul a avut parte de
o recuperare grea. ');

```

```
INSERT INTO fisa_medicala VALUES (4, 1, null);
INSERT INTO fisa_medicala VALUES (5, 5, null);
INSERT INTO fisa_tratament VALUES (1, 4);
INSERT INTO fisa_tratament VALUES (2, 3);
INSERT INTO fisa_tratament VALUES (3, 2);
INSERT INTO fisa_tratament VALUES (1, 2);
INSERT INTO fisa_tratament VALUES (5, 1);
INSERT INTO fisa_tratament VALUES (3, 4);
INSERT INTO fisa_tratament VALUES (2, 1);
INSERT INTO fisa_tratament VALUES (3, 1);
INSERT INTO fisa_tratament VALUES (5, 5);
INSERT INTO fisa_tratament VALUES (4, 2);
INSERT INTO fisa_tratament VALUES (4, 5);
INSERT INTO pacient VALUES (1, 'PASCU', 'REMUS', 3, 3,
'20.04.2010', null);
INSERT INTO pacient VALUES (2, 'IONESCU', 'GEORGE', 4, 5,
'20.04.2019', null);
INSERT INTO pacient VALUES (3, 'TOMA', 'ANDREI', 1, 1,
'01.09.2014', '11.09.2014');
INSERT INTO pacient VALUES (4, 'ENESCU', 'LIVIA', 5, 2,
'30.03.2021', null);
INSERT INTO pacient VALUES (5, 'DOVLEAC', 'FLORIN', 2, 4,
'25.11.2020', '05.12.2020');
INSERT INTO pacient VALUES (6, 'RADU', 'ANCA', 3, 3, '06.09.2021',
'12.09.2021');
INSERT INTO pacient VALUES (7, 'IONESCU', 'MIHAI', 2, 3,
'25.04.2021', '30.04.2021');
INSERT INTO pacient VALUES (8, 'IONESCU', 'GEROGICA', 1, 3,
'01.06.2021', '28.06.2021');
INSERT INTO pacient VALUES (9, 'PASCU', 'REMUS', 2, 1,
'03.01.2022', '04.01.2022');
INSERT INTO pacient VALUES (10, 'ANGHEL', 'LAURA', 3, 2,
'09.10.2021', null);
INSERT INTO pacient VALUES (11, 'PIRVULESCU', 'OTILIA', 2, 2,
'14.10.2021', null);
INSERT INTO pacient VALUES (12, 'PAVEL', 'AURICA', 3, 4,
'14.02.2019', '20.02.2019');
INSERT INTO pacient VALUES (13, 'DIACONU', 'VIOLETA', 5, 1,
'27.08.2020', null);
INSERT INTO pacient VALUES (14, 'COSMA', 'ANDREEA', 1, 4,
'19.06.2018', '24.06.2018');
```

## 6) Subprogram stocat care să utilizeze două tipuri de colecție

Pentru că este finalul anului, spitalul dorește o statistică cu privire la numărul pacienților externați din anul acesta comparativ cu anul precedent. Să se afișeze numărul de pacienți externați din anul curent și de anul trecut și diferența în procente dintre acestea.

```
CREATE OR REPLACE TYPE p_ext_2021 IS varray(20) OF VARCHAR(50);  
/  
CREATE OR REPLACE TYPE p_ext_2020 IS TABLE OF VARCHAR(50);  
/
```

```
CREATE OR REPLACE PROCEDURE bilant IS  
    pacienti_2021    p_ext_2021 := p_ext_2021();  
    pacienti_2020    p_ext_2020 := p_ext_2020();  
BEGIN  
    SELECT  
        nume  
    BULK COLLECT INTO pacienti_2021  
    FROM  
        pacient  
    WHERE  
        data_externare IS NOT NULL  
        AND to_char(data_externare, 'YYYY') = '2021';  
  
    SELECT  
        nume  
    BULK COLLECT INTO pacienti_2020  
    FROM  
        pacient  
    WHERE  
        data_externare IS NOT NULL  
        AND to_char(data_externare, 'YYYY') = '2020';  
  
    dbms_output.put_line('In anul 2021 s-au inregistrat '  
        || pacienti_2021.count  
        || ' pacienti, in comparatie cu '  
        || pacienti_2020.count  
        || ' in 2020.');
```

```
    IF pacienti_2021.count > pacienti_2020.count THEN  
        dbms_output.put_line('Astfel, s-a inregistrat o crestere  
cu '
```

```

|| pacienti_2021.count
/(pacienti_2020.count * 1.0) * 100
|| '% fata de anul trecut.');
```

**ELSE**

```

    dbms_output.put_line('Astfel, s-a inregistrat o scadere cu
,
|| pacienti_2020.count
/(pacienti_2021.count * 1.0) * 100
|| '% fata de anul trecut.');
```

**END IF;**

**END bilant;**

**/**

**EXECUTE bilant;**

```

43
44 END bilant;
45 /
46
47 EXECUTE bilant;
```

Script Output x Query Result x

Task completed in 0,091 seconds

In anul 2021 s-au inregistrat 3 pacienti, in comparatie cu 1 in 2020.  
Astfel, s-a inregistrat o crestere cu 300% fata de anul trecut.

PL/SQL procedure successfully completed.

## 7) Subprogram stocat care utilizează un tip de cursor

Pentru a ține o evidență asupra activității doctorilor și asistenților din spital, să se scrie un subprogram stocat care afișează pentru fiecare doctor, asistentul cu care colaborează și de câte saloane se ocupă aceștia în momentul respectiv.

**CREATE OR REPLACE PROCEDURE nr\_salon\_status IS**

```

v_numa_asistent    asistent.nume%TYPE;
v_numa_doctor       doctor.nume%TYPE;
v_nr_saloane        NUMBER;
CURSOR c IS
SELECT
```



```

        d.num
        || ' '
        || d.prenume,
        a.num
        || ' '
        || a.prenume,
COUNT(s.salon_id)
FROM
    doctor      d
    JOIN asistent  a ON d.doctor_id = a.doctor_id
    LEFT JOIN salon    s ON s.asistent_id = a.asistent_id
GROUP BY
    a.num,
    a.prenume,
    d.num,
    d.prenume
ORDER BY
    d.num;

BEGIN
OPEN c;
LOOP
    FETCH c INTO
        v_num_doctor,
        v_num_asistent,
        v_nr_saloane;
    EXIT WHEN c%notfound;
    IF v_nr_saloane <> 0 THEN
        dbms_output.put_line('Doctorul '
                                || v_num_doctor
                                || ' coopereaza cu asistentul '
                                || v_num_asistent
                                || ' si se ocupa de '
                                || v_nr_saloane
                                || ' saloane.');
```

ELSE

```

        dbms_output.put_line('Doctorul '
                                || v_num_doctor
                                || ' coopereaza cu asistentul '
                                || v_num_asistent
                                || ' dar momentan nu se ocupa de
niciun salon.');
```

```

        END IF;
    END LOOP;
    CLOSE c;
END nr_salon_status;
/

EXECUTE nr_salon_status;

```

```

52 END nr_salon_status;
53 /
54
55 EXECUTE nr_salon_status;

```

Script Output x Query Result x

Task completed in 0,244 seconds

Procedure NR\_SALON\_STATUS compiled

Doctorul GEORGESCU VASILE coopereaza cu asistentul TOMESCU ROBERT si se ocupa de 2 saloane.  
 Doctorul ISTRATE MIHAELA coopereaza cu asistentul VASILE MARIA dar momentan nu se ocupa de niciun salon.  
 Doctorul OPREA CORNEL coopereaza cu asistentul IONESCU TEODORA si se ocupa de 2 saloane.  
 Doctorul POPESCU MIHNEA coopereaza cu asistentul POPESCU VICTORIA dar momentan nu se ocupa de niciun salon.  
 Doctorul STANCU LACRAMIOARA coopereaza cu asistentul POPA ALEXANDRU si se ocupa de 1 saloane.

## 8) Subprogram stocat de tip funcție care să utilizeze 3 dintre tabelele definite

Pentru unii dintre pacienții spitalului nu se cunoaște data externării acestora. Știind durata tratamentelor pe care acesta le face în spital și data internării, să se estimeze pentru un pacient fără dată de externare dat ca parametru, data la care acesta ar putea fi externat.

```

CREATE OR REPLACE FUNCTION data_externare (
    v_nume_pacient pacient.nume%TYPE
) RETURN DATE IS
    data_ext pacient.data_internare%TYPE;
BEGIN
    SELECT
        nvl(p.data_externare, p.data_internare) + MAX(t.durata)
    INTO data_ext
    FROM
        pacient            p
        JOIN fisa_medicala  fm ON fm.fisa_id = p.fisa_id
        JOIN fisa_tratament ft ON fm.fisa_id = ft.fisa_id
        JOIN tratament      t ON t.tratament_id =

```

```

ft.tratament_id
  GROUP BY
    p.numa,
    p.prenume,
    p.data_externare,
    p.data_internare
  HAVING p.data_externare IS NULL
        AND lower(p.numa
                  || ' '
                  || p.prenume) = lower(v.numa_pacient);

  RETURN data_ext;
EXCEPTION
  WHEN no_data_found THEN
    raise_application_error(-20000, 'Nu exista acest pacient
sau i se cunoaste data de externare.');
```

```

  WHEN too_many_rows THEN
    raise_application_error(-20001, 'Exista mai multi pacienti
cu acest nume si fara data de externare.');
```

```

  WHEN OTHERS THEN
    raise_application_error(-20002, 'Alta eroare!');
```

```

END data_externare;
/

SELECT data_externare('PIRVULESCU OTILIA')
FROM dual; --ok

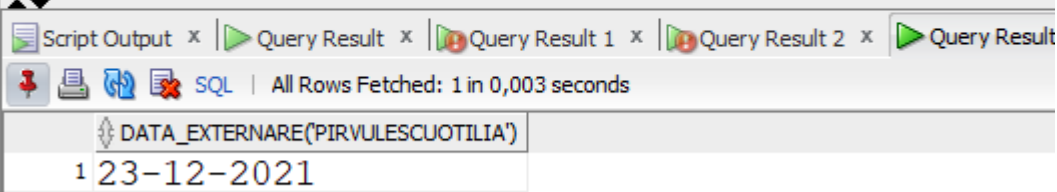
SELECT data_externare('TOMA GHEORGHE')
FROM dual; --no_data_found (nu exista acest pacient)

SELECT data_externare('PASCU REMUS')
FROM dual; --too_many_rows (exista mai multi pacienti cu acest
nume si fara data de externare)
```

```

34
35 SELECT data_externare('PIRVULESCU OTILIA')
36 FROM dual; --ok
37

```



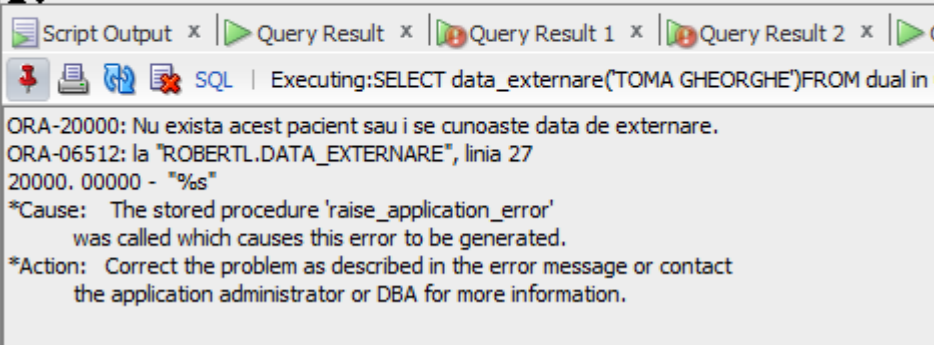
The screenshot shows the SQL Developer interface. The top pane displays the SQL query: `SELECT data_externare('PIRVULESCU OTILIA') FROM dual; --ok`. The bottom pane shows the query result in a table with one row and one column, where the value is '23-12-2021'. The status bar indicates 'All Rows Fetched: 1 in 0,003 seconds'.

DATA_EXTERNARE('PIRVULESCUOTILIA')
23-12-2021

```

38 SELECT data_externare('TOMA GHEORGHE')
39 FROM dual; --no_data_found (nu exista
40

```

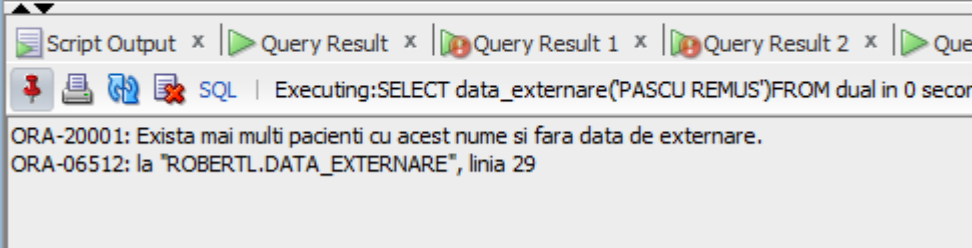


The screenshot shows the SQL Developer interface. The top pane displays the SQL query: `SELECT data_externare('TOMA GHEORGHE') FROM dual; --no_data_found (nu exista`. The bottom pane shows an error message: `ORA-20000: Nu exista acest pacient sau i se cunoaste data de externare.` The error details include the file `ORA-06512: la "ROBERTL.DATA_EXTERNARE", linia 27` and the message `20000. 00000 - "%s"`. The cause is `*Cause: The stored procedure 'raise_application_error' was called which causes this error to be generated.` The action is `*Action: Correct the problem as described in the error message or contact the application administrator or DBA for more information.`

```

41 SELECT data_externare('PASCU REMUS')
42 FROM dual; --too_many_rows (exista mai r

```



The screenshot shows the SQL Developer interface. The top pane displays the SQL query: `SELECT data_externare('PASCU REMUS') FROM dual; --too_many_rows (exista mai r`. The bottom pane shows an error message: `ORA-20001: Exista mai multi pacienti cu acest nume si fara data de externare.` The error details include the file `ORA-06512: la "ROBERTL.DATA_EXTERNARE", linia 29`.

## 9) Subprogram stocat de tip funcție care să utilizeze 3 dintre tabelele definite

Pentru un pacient dat ca parametru, să se afișeze informații despre el: în ce salon se află, în ce secție, ce doctor și ce asistent se ocupă de el.

```

CREATE OR REPLACE PROCEDURE info_pacient (
    v_num_pacient pacient.nume%TYPE
) IS

    v_num_sectie      sectie.nume_sectie%TYPE;
    v_saloon_id       salon.salon_id%TYPE;
    v_num_asistent    asistent.nume%TYPE;
    v_num_doctor       doctor.nume%TYPE;
BEGIN
    SELECT
        s.salon_id,
        se.nume_sectie,
        d.nume
        || ' '
        || d.prenume,
        a.nume
        || ' '
        || a.prenume
    INTO
        v_saloon_id,
        v_num_sectie,
        v_num_doctor,
        v_num_asistent
    FROM
        pacient      p
        JOIN salon      s ON s.salon_id = p.salon_id
        JOIN asistent   a ON s.asistent_id = a.asistent_id
        JOIN doctor      d ON d.doctor_id = a.doctor_id
        JOIN sectie      se ON se.sectie_id = s.sectie_id
    WHERE
        lower(p.nume
            || ' '
            || p.prenume) = lower(v_num_pacient);

    dbms_output.put_line('Pacientul '
        || v_num_pacient
        || ' este internat in salonul '
        || v_saloon_id
        || ' din sectia '
        || v_num_sectie);

    dbms_output.put_line('De asemenea, este ingrijit de asistentul

```

```

|| v_numa_asistent
|| ' si doctorul '
|| v_numa_doctor);

EXCEPTION
  WHEN no_data_found THEN
    raise_application_error(-20000, 'Nu exista acest
pacient.');
```

```

  WHEN too_many_rows THEN
    raise_application_error(-20001, 'Exista mai multi pacienti
cu acest nume.');
```

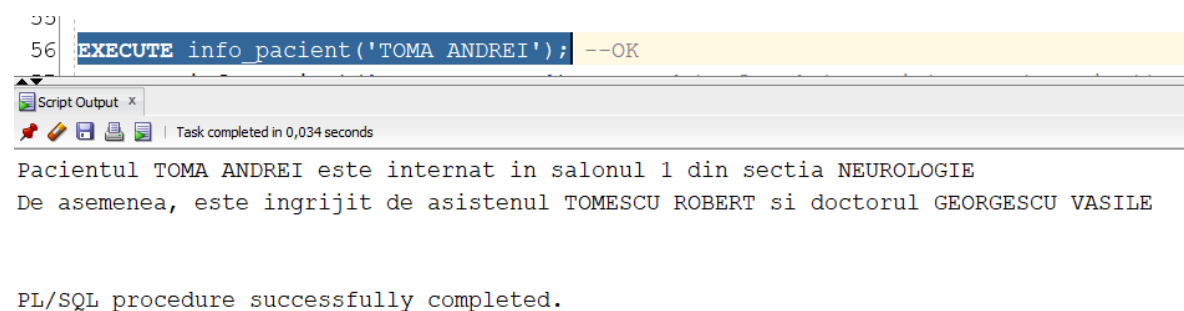
```

  WHEN OTHERS THEN
    raise_application_error(-20002, 'Alta eroare!');
```

```

END info_pacient;
/

EXECUTE info_pacient('TOMA ANDREI'); --OK
EXECUTE info_pacient('POPESCU MARA'); --no_data_found (nu exista
acest pacient)
EXECUTE info_pacient('PASCU REMUS'); --too_many_rows (exista mai
multi pacienti cu acest nume)
```



```

56 EXECUTE info_pacient('TOMA ANDREI'); --OK
```

Script Output x

Task completed in 0,034 seconds

Pacientul TOMA ANDREI este internat in salonul 1 din sectia NEUROLOGIE  
De asemenea, este ingrijit de asistenul TOMESCU ROBERT si doctorul GEORGESCU VASILE

PL/SQL procedure successfully completed.

```
57 EXECUTE info_pacient('POPESCU MARA'); --no_data_found (nu exista acest p
Script Output x
Task completed in 0,063 seconds
ORA-20000: Nu exista acest pacient.
ORA-06512: la "ROBERTL.INFO_PACIENT", linia 44
ORA-06512: la linia 1
20000. 00000 - "%s"
*Cause:      The stored procedure 'raise_application_error'
              was called which causes this error to be generated.
*Action:     Correct the problem as described in the error message or contact
              the application administrator or DBA for more information.
```

```
58 EXECUTE info_pacient('PASCU REMUS'); --too_many_row
Script Output x
Task completed in 0,071 seconds
Error starting at line : 58 in command -
BEGIN info_pacient('PASCU REMUS'); END;
Error report -
ORA-20001: Exista mai multi pacienti cu acest nume.
ORA-06512: la "ROBERTL.INFO_PACIENT", linia 46
ORA-06512: la linia 1
```

## 10) Trigger de tip LMD la nivel de comandă

Duminica și în fiecare zi după ora 19:00 se realizează verificarea aparatelor cu care se realizează consultații. Drept consecință, în aceste intervale nu se pot face modificări (inserări, ștergeri, actualizări) asupra bazei de date în tabela CONSULTAȚIE.

```
CREATE OR REPLACE TRIGGER tr_consultatie
BEFORE INSERT OR DELETE OR UPDATE ON consultatie
BEGIN
    IF to_char(sysdate, 'D') <> 7 OR to_char(sysdate, 'HH24') > 19
    THEN
        IF inserting THEN
            raise_application_error(-20001, 'Inserarea unei noi
```

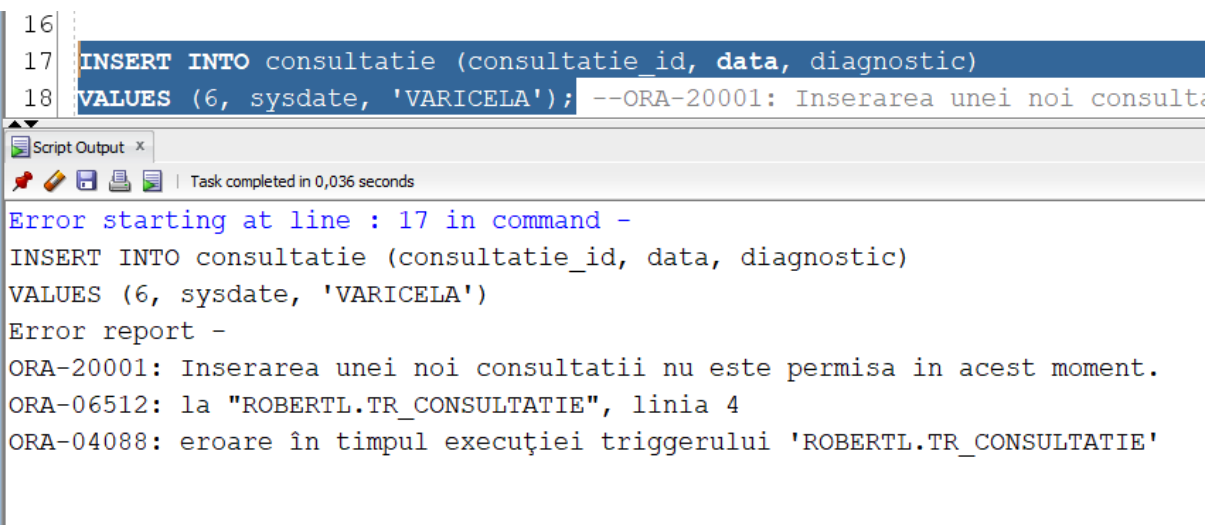
```

consultatii nu este permisa in acest moment. ');
    ELSIF deleting THEN
        raise_application_error(-20002, 'Stergerea unei
consultatii nu este permisa in acest moment. ');
    ELSE
        raise_application_error(-20003, 'Modificarea unei
consultatii nu este permisa in acest moment. ');
    END IF;

END IF;
END;
/

INSERT INTO consultatie (consultatie_id, data, diagnostic)
VALUES (6, sysdate, 'VARICELA'); --ORA-20001: Inserarea unei noi
consultatii nu este permisa in acest moment.

```



The screenshot shows a SQL script execution window with the following content:

```

16
17 INSERT INTO consultatie (consultatie_id, data, diagnostic)
18 VALUES (6, sysdate, 'VARICELA'); --ORA-20001: Inserarea unei noi consulta

```

Below the script, a message indicates the task completed in 0,036 seconds. The error report section shows the following messages:

```

Error starting at line : 17 in command -
INSERT INTO consultatie (consultatie_id, data, diagnostic)
VALUES (6, sysdate, 'VARICELA')
Error report -
ORA-20001: Inserarea unei noi consultatii nu este permisa in acest moment.
ORA-06512: la "ROBERTL.TR_CONSULTATIE", linia 4
ORA-04088: eroare în timpul execuției triggerului 'ROBERTL.TR_CONSULTATIE'

```

## 11) Trigger de tip LMD la nivel de linie.

Știind capacitatea unui salon, să nu se permită inserarea în același salon a mai multor pacienți decât capacitatea maximă a acestuia.

```

CREATE TABLE actiuni (
    utilizator    VARCHAR2(30),
    nume_bd       VARCHAR2(50),
    eveniment     VARCHAR2(20),
    nume_obiect   VARCHAR2(30),

```



```

data          DATE
);

CREATE OR REPLACE TRIGGER tr_actiuni AFTER CREATE OR DROP OR ALTER
ON SCHEMA BEGIN
    INSERT INTO actiuni VALUES (
        sys.login_user,
        sys.database_name,
        sys.sysevent,
        sys.dictionary_obj_name,
        sysdate
    );

END;
/

```

```

30
31 select * from actiuni;
32

```

Script Output x Query Result x


SQL | All Rows Fetched: 179 in 0,015 seconds

	UTILIZATOR	NUME_BD	EVENIMENT	NUME_OBIECT	DATA
1	ROBERTL	XE	DROP	DOCTOR	05-01-2022
2	ROBERTL	XE	CREATE	PK DOC	05-01-2022
3	ROBERTL	XE	CREATE	DOCTOR	05-01-2022
4	ROBERTL	XE	DROP	ASISTENT	05-01-2022
5	ROBERTL	XE	CREATE	PK ASIS	05-01-2022
6	ROBERTL	XE	CREATE	ASISTENT	05-01-2022
7	ROBERTL	XE	DROP	ASISTENT	05-01-2022
8	ROBERTL	XE	CREATE	PK ASIS	05-01-2022
9	ROBERTL	XE	CREATE	ASISTENT	05-01-2022
10	ROBERTL	XE	DROP	CONSULTATIE	05-01-2022
11	ROBERTL	XE	CREATE	PK CONS	05-01-2022

31 `select * from actiuni;`

32

Script Output x Query Result x

 | All Rows Fetched: 179 in 0,015 seconds

	UTILIZATOR	NUME_BD	EVENIMENT	NUME_OBIECT	DATA
75	ROBERTL XE		CREATE	COD	06-01-2022
76	ROBERTL XE		CREATE	COD	06-01-2022
77	ROBERTL XE		CREATE	COD	06-01-2022
78	ROBERTL XE		CREATE	COD	06-01-2022
79	ROBERTL XE		CREATE	COD	06-01-2022
80	ROBERTL XE		CREATE	COD	06-01-2022
81	ROBERTL XE		CREATE	COD	06-01-2022
82	ROBERTL XE		CREATE	COD	06-01-2022
83	ROBERTL XE		CREATE	COD	06-01-2022
84	ROBERTL XE		CREATE	COD	06-01-2022
85	ROBERTL XE		CREATE	COD	06-01-2022

### 13) Pachet care conține toate obiectele definite în cadrul proiectului

```
CREATE OR REPLACE PACKAGE pachet AS
    PROCEDURE bilant;

    PROCEDURE nr_salon_status;

    FUNCTION data_externare (
        v_num_pacient IN pacient.nume%TYPE
    ) RETURN DATE;

    PROCEDURE info_pacient (
        v_num_pacient IN pacient.nume%TYPE
    );

END pachet;
/
```

```
CREATE OR REPLACE PACKAGE BODY pachet AS

    PROCEDURE bilant IS
        pacienti_2021 p_ext_2021 := p_ext_2021();
        pacienti_2020 p_ext_2020 := p_ext_2020();
```

```

BEGIN
    SELECT
        nume
    BULK COLLECT
    INTO pacienti_2021
    FROM
        pacient
    WHERE
        data_externare IS NOT NULL
        AND to_char(data_externare, 'YYYY') = '2021';

    SELECT
        nume
    BULK COLLECT
    INTO pacienti_2020
    FROM
        pacient
    WHERE
        data_externare IS NOT NULL
        AND to_char(data_externare, 'YYYY') = '2020';

    dbms_output.put_line('In anul 2021 s-au inregistrat '
        || pacienti_2021.count
        || ' pacienti, in comparatie cu '
        || pacienti_2020.count
        || ' in 2020.');
```

IF pacienti\_2021.count > pacienti\_2020.count THEN

```

        dbms_output.put_line('Astfel, s-a inregistrat o
creștere cu '
        || pacienti_2021.count
/(pacienti_2020.count * 1.0) * 100
        || '% fata de anul trecut.');
```

ELSE

```

        dbms_output.put_line('Astfel, s-a inregistrat o
scadere cu '
        || pacienti_2020.count
/(pacienti_2021.count * 1.0) * 100
        || '% fata de anul trecut.');
```

END IF;

END bilant;

```

PROCEDURE nr_salon_status IS

    v_num_e_asistent    asistent.nume%TYPE;
    v_num_e_doctor      doctor.nume%TYPE;
    v_nr_saloane        NUMBER;
    CURSOR c IS
    SELECT
        d.nume
        || ' '
        || d.prenume,
        a.nume
        || ' '
        || a.prenume,
        COUNT(s.salon_id)
    FROM
        doctor      d
    JOIN asistent  a ON d.doctor_id = a.doctor_id
    LEFT JOIN salon      s ON s.asistent_id =
a.asistent_id
    GROUP BY
        a.nume,
        a.prenume,
        d.nume,
        d.prenume
    ORDER BY
        d.nume;

BEGIN
    OPEN c;
    LOOP
        FETCH c INTO
            v_num_e_doctor,
            v_num_e_asistent,
            v_nr_saloane;
        EXIT WHEN c%notfound;
        IF v_nr_saloane <> 0 THEN
            dbms_output.put_line('Doctorul '
                                || v_num_e_doctor
                                || ' coopereaza cu asistentul
                                || v_num_e_asistent

```

```

|| ' si se ocupa de '
|| v_nr_saloane
|| ' saloane.');
```

ELSE

```

    dbms_output.put_line('Doctorul '
|| v_num_e_doctor
|| ' coopereaza cu asistentul
,
|| v_num_e_asistent
|| ' dar momentan nu se ocupa
de niciun salon.');
```

END IF;

END LOOP;

CLOSE c;

END nr\_salon\_status;

PROCEDURE info\_pacient (

    v\_num\_e\_pacient pacient.num\_e%TYPE

) IS

    v\_num\_e\_sectie    sectie.num\_e\_sectie%TYPE;

    v\_salon\_id        salon.salon\_id%TYPE;

    v\_num\_e\_asistent  asistent.num\_e%TYPE;

    v\_num\_e\_doctor     doctor.num\_e%TYPE;

BEGIN

    SELECT

        s.salon\_id,

        se.num\_e\_sectie,

        d.num\_e

        || ' ' ,

        || d.prenume,

        a.num\_e

        || ' ' ,

        || a.prenume

    INTO

        v\_salon\_id,

        v\_num\_e\_sectie,

        v\_num\_e\_doctor,

        v\_num\_e\_asistent

    FROM

```

        pacient      p
        JOIN salon    s ON s.salon_id = p.salon_id
        JOIN asistent a ON s.asistent_id = a.asistent_id
        JOIN doctor    d ON d.doctor_id = a.doctor_id
        JOIN sectie    se ON se.sectie_id = s.sectie_id
    WHERE
        lower(p.num
            || ' '
            || p.prenume) = lower(v_num
            e_pacient);

    dbms_output.put_line('Pacientul '
        || v_num
        e_pacient
        || ' este internat in salonul '
        || v_sal
        on_id
        || ' din sectia '
        || v_num
        e_sectie);

    dbms_output.put_line('De asemenea, este ingrijit de
asistenul '
        || v_num
        e_asistent
        || ' si doctorul '
        || v_num
        e_doctor);

EXCEPTION
    WHEN no_data_found THEN
        raise_application_error(-20000, 'Nu exista acest
pacient. ');
    WHEN too_many_rows THEN
        raise_application_error(-20001, 'Exista mai multi
pacienti cu acest nume. ');
    WHEN OTHERS THEN
        raise_application_error(-20002, 'Alta eroare!');
END info_pacient;

FUNCTION data_externare (
    v_num
    e_pacient pacient.num
    e%TYPE
) RETURN DATE IS
    data_ext pacient.data_internare%TYPE;
BEGIN
    SELECT
        nvl(p.data_externare, p.data_internare) +
MAX(t.durata)
    INTO data_ext

```

```

FROM
    pacient          p
    JOIN fisa_medicala fm ON fm.fisa_id = p.fisa_id
    JOIN fisa_tratament ft ON fm.fisa_id = ft.fisa_id
    JOIN tratament    t ON t.tratament_id =
ft.tratament_id
GROUP BY
    p.nume,
    p.prenume,
    p.data_externare,
    p.data_internare
HAVING p.data_externare IS NULL
        AND lower(p.nume
                || ' '
                || p.prenume) = lower(v_nume_pacient);

RETURN data_ext;
EXCEPTION
    WHEN no_data_found THEN
        raise_application_error(-20000, 'Nu exista acest
pacient sau i se cunoaste data de externare.');
```

```

    WHEN too_many_rows THEN
        raise_application_error(-20001, 'Exista mai multi
pacienti cu acest nume si fara data de externare.');
```

```

    WHEN OTHERS THEN
        raise_application_error(-20002, 'Alta eroare!');
END data_externare;

END pachet;
/

execute pachet.bilant;
execute pachet.nr_salon_status;
execute pachet.info_pacient('PIRVULESCU OTILIA');
select data_externare('PIRVULESCU OTILIA')
from dual;
```

```

201      END data_externare;
202
203  END pachet;
204  /
205
206  execute pachet.bilant;
207  execute pachet.nr_salon_status;
208  execute pachet.info_pacient('PIRVULESCU OTILIA');
209  select data_externare('PIRVULESCU OTILIA')
210  from dual;

```

Script Output x Query Result x

Task completed in 0,067 seconds

PL/SQL procedure successfully completed.

Package PACHET compiled

Package Body PACHET compiled

```

206  execute pachet.bilant;
207  execute pachet.nr_salon_status;
208  execute pachet.info_pacient('PIRVULESCU OTILIA');
209  select data_externare('PIRVULESCU OTILIA')
210  from dual;

```

Script Output x Query Result x

Task completed in 0,051 seconds

Package Body PACHET compiled

In anul 2021 s-au inregistrat 3 pacienti, in comparatie cu 1 in 2020.  
Astfel, s-a inregistrat o crestere cu 300% fata de anul trecut.

PL/SQL procedure successfully completed.



```
207 execute pachet.nr_salon_status;
208 execute pachet.info_pacient('PIRVULESCU OTILIA');
209 select data_externare('PIRVULESCU OTILIA')
210 from dual;
```

Script Output x Query Result x  
Task completed in 0,037 seconds

Doctorul ISTRATE MIHAELA coopereaza cu asistentul VASILE MARIA dar momentan nu se ocupa de niciun salon.  
Doctorul OPREA CORNEL coopereaza cu asistentul IONESCU TEODORA si se ocupa de 2 saloane.  
Doctorul POPESCU MIHNEA coopereaza cu asistentul POPESCU VICTORIA dar momentan nu se ocupa de niciun salon.  
Doctorul STANCU LACRAMIOARA coopereaza cu asistentul POPA ALEXANDRU si se ocupa de 1 saloane.

PL/SQL procedure successfully completed.

Statements - Log

```
208 execute pachet.info_pacient('PIRVULESCU OTILIA');
209 select data_externare('PIRVULESCU OTILIA')
210 from dual;
```

Script Output x Query Result x  
Task completed in 0,034 seconds

PL/SQL procedure successfully completed.

Pacientul PIRVULESCU OTILIA este internat in salonul 2 din sectia NEUROLOGIE  
De asemenea, este ingrijit de asistenul POPA ALEXANDRU si doctorul STANCU LACRAMIOARA

PL/SQL procedure successfully completed.

Statements - Log

```
209 select data_externare('PIRVULESCU OTILIA')
210 from dual;
```

Script Output x Query Result x  
All Rows Fetched: 1 in 0,005 seconds

DATA_EXTERNARE('PIRVULESCUOTILIA')
1 23-12-2021

#### 14) Pachet care să includă tipuri de date complexe și obiecte necesare unui flux de acțiuni integrate, specifice bazei de date definite

Având în vedere situația specială prin care trecem, dorim să ne protejăm pacienții și persoanele dragi lor, astfel că spitalul a adus o nouă regulă pentru vizitatorii care vin la spital. Fiecare pacient poate invita maxim 3 vizitatori, oferindu-le drept “cheie de acces” în spital numele lor criptat folosind cifrul Cezar cu deplasare 19. Astfel, un vizitator vine la spital și prezintă numele criptat al celui pe care îl vizitează, și numele și prenumele său. Apoi numele

său, numele decriptat al celui pe care îl vizitează și data sunt inserate automat într-un tabel VIZITATOR, cu condiția ca numărul de vizitatori să nu depășească 3 și cheia să fie corectă.

Funcția `cripteaza(ume)` returnează numele criptat folosind cifrul lui Cezar cu deplasare 19. Procedura `vec_pacienți` ia toți pacienții și îi pune într-un array, îi criptează în alt array și îi afișează pe ecran.

Funcția `verif_cod(ume_criptat, nume_vizitator, prenume_vizitator)` verifică corectitudinea numelui, îl decriptează, verifică numărul de vizitatori pentru pacient, apelează procedura `<insereaza>` și returnează un mesaj corespunzător (“Codul a fost corect dar nu se mai acceptă vizitatori.”, “Codul a fost corect! Detaliile vizitatorului au fost introduse în tabel.” sau “Ne pare rău, codul este greșit.”).

Procedura `inserează(id, nume_pacient, nume_vizitator, prenume_vizitator)` inserează în tabelul VIZITATOR informațiile primite.

Pachetul folosește 2 funcții, 2 proceduri, 2 tipuri de date (array-uri și record).

```
CREATE TABLE vizitator (  
    vizitator_id    NUMBER(10)  
        CONSTRAINT pk_viz PRIMARY KEY,  
    nume            VARCHAR2(30) NOT NULL,  
    prenume         VARCHAR2(30) NOT NULL,  
    nume_pacient    VARCHAR2(30) NOT NULL,  
    data            DATE NOT NULL  
);
```

```
CREATE OR REPLACE TYPE litere IS  
    VARRAY(27) OF CHAR;
```

```
CREATE OR REPLACE TYPE pacienti IS  
    VARRAY(50) OF VARCHAR2(30);
```

```
CREATE OR REPLACE PACKAGE cod AS  
    viz vizitator%rowtype;  
    p_nume pacienti := pacienti();  
    p_nume_criptate pacienti := pacienti();
```

```
    FUNCTION cripteaza (  
        v_nume IN pacient.nume%TYPE  
    ) RETURN VARCHAR2;
```

```
    FUNCTION verific_cod (  

```

```

        v_ume_criptat    IN    pacient.ume%TYPE,
        v_ume_viz        IN    vizitator.ume%TYPE,
        v_preume_viz     IN    vizitator.preume%TYPE
    ) RETURN VARCHAR2;

PROCEDURE vec_pacienti;

PROCEDURE insereaza (
    id_viz              IN    vizitator.vizitator_id%TYPE,
    v_ume_viz           IN    vizitator.ume%TYPE,
    v_preume_viz        IN    vizitator.preume%TYPE,
    v_ume_pacient       IN    vizitator.ume_pacient%TYPE
);

END cod;
/

CREATE OR REPLACE PACKAGE BODY cod AS

    FUNCTION cripteaza (
        v_ume pacient.ume%TYPE
    ) RETURN VARCHAR2 IS

        v_ume_criptat    pacient.ume%TYPE := '';
        litere_c          litere := litere('T', 'U', 'V', 'W', 'X',
        'Y', 'Z', 'A', 'B', 'C',
        'D', 'E', 'F', 'G', 'H',
        'I', 'J', 'K', 'L', 'M',
        'N', 'O', 'P', 'Q', 'R',
        'S');
        i                  NUMBER(2) := 1;
    BEGIN
        LOOP
            EXIT WHEN i - 1 = length(v_ume);
            IF substr(v_ume, i, 1) <> ' ' THEN
                v_ume_criptat := v_ume_criptat
                    ||
                    litere_c(ascii((substr(v_ume, i, 1))) - 64);

            ELSE
                v_ume_criptat := v_ume_criptat || ' ';
            END IF;
        LOOP
    END cripteaza;

```

```

        i := i + 1;
    END LOOP;

    RETURN v_num_criptat;
END cripteaza;

PROCEDURE vec_pacienti IS
    i NUMBER(2) := 1;
BEGIN
    SELECT
        nume
        || ' '
        || prenume
    BULK COLLECT
    INTO p_num
    FROM
        pacient;

    LOOP
        EXIT WHEN i - 1 = p_num.count;
        IF p_num_criptate.count < p_num.count THEN
            p_num_criptate.extend;
        END IF;
        p_num_criptate(i) := cod.cripteaza(upper(p_num(i)));

        dbms_output.put_line(p_num(i)
            || '/'
            || p_num_criptate(i));

        i := i + 1;
    END LOOP;
END vec_pacienti;

FUNCTION verif_cod (
    v_num_criptat    pacient.nume%TYPE,
    v_num_viz        vizitator.nume%TYPE,
    v_prenume_viz    vizitator.prenume%TYPE
) RETURN VARCHAR2 IS

    raspuns    VARCHAR2(100) := 'false';

```

```

i          NUMBER(2) := 1;
id_viz     NUMBER(5);
v_ume      pacient.ume%TYPE;
nr_pac     NUMBER(5);
PRAGMA AUTONOMOUS_TRANSACTION;
BEGIN
    cod.vec_pacienti;
    LOOP
        EXIT WHEN raspuns = 'true' OR i - 1 = p_ume.count;
        IF p_ume_criptate(i) = upper(v_ume_criptat) THEN
            raspuns := 'true';
            v_ume := p_ume(i);
        END IF;

        i := i + 1;
    END LOOP;

    SELECT MAX(vizitator_id) + 1
    INTO id_viz
    FROM
        vizitator;

    SELECT COUNT(1)
    into nr_pac
    FROM vizitator
    WHERE nume_pacient = v_ume;

    IF id_viz IS NULL THEN
        id_viz := 1;
    END IF;
    IF raspuns = 'false' THEN
        raspuns := 'Ne pare rau, codul este gresit.';
        id_viz := 0;
    ELSIF nr_pac < 3 THEN
        raspuns := 'Codul a fost corect! Detaliile
vizitatorului au fost introduse in tabel.';
        cod.insereaza(id_viz, v_ume_viz, v_prenume_viz,
v_ume);
    ELSE
        raspuns := 'Codul a fost corect dar nu se mai accepta
vizitatori.';
    END IF;

```

```

        RETURN raspuns;
    END verif_cod;

    PROCEDURE insereaza (
        id_viz          vizitator.vizitator_id%TYPE,
        v_num_viz       vizitator.num%TYPE,
        v_prename_viz   vizitator.prenume%TYPE,
        v_num_pacient   vizitator.num_pacient%TYPE
    ) IS
    BEGIN
        IF id_viz <> 0 THEN
            viz.vizitator_id := id_viz;
            viz.num := upper(v_num_viz);
            viz.prenume := upper(v_prename_viz);
            viz.num_pacient := upper(v_num_pacient);
            viz.data := sysdate;
            INSERT INTO vizitator VALUES viz;
            commit;
        END IF;
    END insereaza;

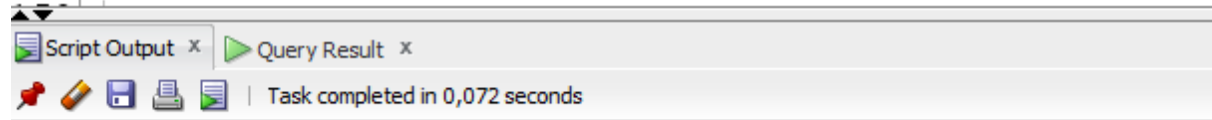
END cod;
/

```

```

154         END IF;
155     END insereaza;
156
157 END cod;
158 /

```



PL/SQL procedure successfully completed.

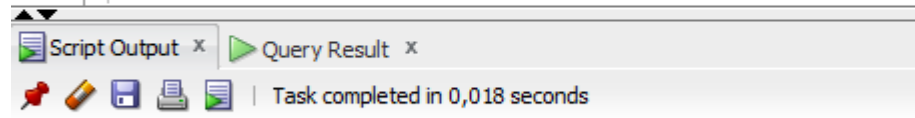
Package COD compiled

Package Body COD compiled

```

187 execute cod.vec_pacienti;
188
189

```



```

DOVLEAC FLORIN/WHOEXTV YEHKBG
RADU ANCA/KTWN TGVV
IONESCU MIHAI/BHGXLVN FBATB
PASCU REMUS/ITLVN KXFNL
ANGHEL LAURA/TGZAXE ETNKT
PIRVULESCU OTILIA/IBKONEXLVN HMBEET
PAVEL AURICA/ITOXE TNKBVT
COSMA ANDREEA/VHLFT TGWKXXT
DIACONU VIOLETA/WBTVHGN OBHEXMT

```

```
180
181 SELECT
182     cod.verif_cod('ITOXE TNKBVT', 'LITA', 'ROBERT')
183 FROM
184     dual;
```

Script Output x Query Result x	
All Rows Fetched: 1 in 0,005 seconds	
COD.VERIF_COD('ITOXETNKBT','LITA','ROBERT')	
1 Codul a fost corect! Detaliile vizitatorului au fost introduse in tabel.	

```
181 SELECT
182     cod.verif_cod('ALALA', 'LITA', 'ROBERT')
183 FROM
184     dual;
185
186
```

Script Output x Query Result x	
All Rows Fetched: 1 in 0,004 seconds	
COD.VERIF_COD('ALALA','LITA','ROBERT')	
1 Ne pare rau, codul este gresit.	

```
181 SELECT
182     cod.verif_cod('IBKONEXLVN HMBEBT', 'LITA', 'ROBERT')--PIRVULESCU OTILIA
183 FROM
184     dual;
185
186
```


Script Output x Query Result x	
All Rows Fetched: 1 in 0,004 seconds	
COD.VERIF_COD('IBKONEXLVNHMBEBT','LITA','ROBERT')--PIRVULESCUOTILIA	
1 Codul a fost corect dar nu se mai accepta vizitatori.	



187  
188  
189

```
select * from vizitator;
```

Script Output x Query Result x

 All Rows Fetched: 7 in 0,002 seconds

	VIZITATOR_ID	NUME	PRENUME	NUME_PACIENT	DATA
1	3	LITA	ROBERT	PIRVULESCU OTILIA	06-01-2022
2	1	LITA	ROBERT	PIRVULESCU OTILIA	06-01-2022
3	2	LITA	ROBERT	PIRVULESCU OTILIA	06-01-2022
4	6	LITA	ROBERT	TOMA ANDREI	06-01-2022
5	7	LITA	ROBERT	PAVEL AURICA	06-01-2022
6	4	LITA	ROBERT	TOMA ANDREI	06-01-2022
7	5	LITA	ROBERT	TOMA ANDREI	06-01-2022