CERINTE

1. Definiti blocuri PL/SQL si tratati 5 exceptii predefinite (in afara de NO_DATA_FOUND si TOO_MANY_ROWS).
2. Care bloc e mai eficient dintre exemplul 3.15 (ambele variante) si 3.16? Folositi un pachet pentru timp.
3. De ce nu a fost necesar sa se trateze exceptiile NO_DATA_FOUND si TOO_MANY_ROWS in exemplul 3.8?
4. Creati o tabela in care sa inserati niste date (cu diacritice). Utilizand un bloc PL/SQL, inserati date in tabela si afisati-le.

REZOLVARI

1. Pentru a rezolva aceasta cerinta, am folosit 3 blocuri PL/SQL. In primul bloc am tratat exceptiile INVALID_NUMBER, ZERO_DIVIDE si CASE_NOT_FOUND, in al doilea bloc am tratat exceptia DUP_VAL_ON_INDEX, iar in ultimul bloc am tratat exceptia COLLECTION_IS_NULL.
   Am inclus tot codul, inclusiv crearea tabelelor si inserarea datelor, impreuna cu niste poze pentru a arata rezultatele. Tabelele si datele au fost necesare pentru primele 2 blocuri. Pentru primul bloc, am modificat in “select into” conditia “where lower(titlu) like …” sa fie “like ‘%stad%’ “, “like ‘%meri%’ “, “like ‘%of%’ “, “like ‘%dook%’ “.

```
CREATE TABLE ALBUM_AUX (
 id_album NUMBER(4) PRIMARY KEY,
 titlu VARCHAR(30) NOT NULL
);

CREATE TABLE PIESA_AUX (
 id_piesa NUMBER(4) PRIMARY KEY,
 id_album NUMBER(4) NOT NULL,
 titlu VARCHAR(30) NOT NULL,
 durata VARCHAR(5) NOT NULL,
 FOREIGN KEY (id_album) REFERENCES ALBUM_AUX(id_album)
);

INSERT INTO ALBUM_AUX
VALUES(100, 'Stadium Arcadium');
INSERT INTO ALBUM_AUX
VALUES(110, 'Master of Puppets');
INSERT INTO ALBUM_AUX
VALUES(120, 'American Idiot');
INSERT INTO ALBUM_AUX
VALUES(130, 'Dookie');
```

```
INSERT INTO PIESA_AUX
VALUES(1, 100, 'Snow (Hey Oh)', 335);
INSERT INTO PIESA_AUX
VALUES(2, 100, 'Dani California', 283);
INSERT INTO PIESA_AUX
VALUES(3, 110, 'Master Of Puppets', 515);
INSERT INTO PIESA_AUX
VALUES(4, 110, 'Orion', 502);
INSERT INTO PIESA_AUX
VALUES(5, 120, 'Boulevard of Broken Dreams', 494);
INSERT INTO PIESA_AUX
VALUES(6, 120, 'Wake Me Up When September Ends', 286);
INSERT INTO PIESA_AUX
VALUES (7, 120, 'Whatsername', '253s');
INSERT INTO PIESA_AUX
VALUES(8, 130, 'Basket Case', 183);
INSERT INTO PIESA_AUX
VALUES(9, 130, 'When I Come Around', 178);
COMMIT;

declare
   t_id_album       album_aux.id_album%type;
   t_nr_total_piese number;
   t_nr_piese       number;
   t_raport         number;
begin
   select id_album
   into t_id_album
   from album_aux
   where lower(titlu) like '%dook%';
   -- '%stad'  - execution with no exceptions
   -- '%meri%' - INVALID_NUMBER
   -- '%of%'   - ZERO_DIVIDE
   -- '%dook'% - CASE_NOT_FOUND

   select count(1)
   into t_nr_total_piese
   from piesa_aux
   where id_album = t_id_album;

   select count(1)
   into t_nr_piese
   from piesa_aux
```

```
    where to_number(durata) < 300
    and id_album = t_id_album;

    t_raport := t_nr_total_piese / t_nr_piese;

    dbms_output.put_line('ID Album: ' || t_id_album);
    dbms_output.put_line('Nr. piese cu durata < 300: ' || t_nr_piese);
    dbms_output.put_line('Raport: ' || t_nr_total_piese || '/' || t_nr_piese || ' = ' ||
t_nr_total_piese / t_nr_piese);

    case
        when t_raport >= 2 then dbms_output.put_line('Categorie: B (raport >= 2)');
        when t_raport > 1 then dbms_output.put_line('Categorie: A (raport > 1)');
    end case;

    exception
        when invalid_number then
            dbms_output.put_line('INVALID_NUMBER: Invalid conversion from string to
number.');
        when zero_divide then
            dbms_output.put_line('ZERO_DIVIDE: No songs found with a duration of under
300 seconds.');
        when case_not_found then
            dbms_output.put_line('CASE_NOT_FOUND: Nu exista o categorie.');
end;
```

```
PL/SQL procedure successfully completed.
```

Dbms Output

Buffer Size: 20000

SGBD-231

```
ID Album: 100
Nr. piese cu durata < 300: 1
Raport: 2/1 = 2
Categorie: B (raport >= 2)
```

Script Output x | Query Result x

Task completed in 0.059 seconds

```
PL/SQL procedure successfully completed.
```

Dbms Output

Buffer Size: 20000

SGBD-231 ×

```
ZERO_DIVIDE: No songs found with a duration of under 300 seconds.
```

Script Output x | Query Result x

Task completed in 0.043 seconds

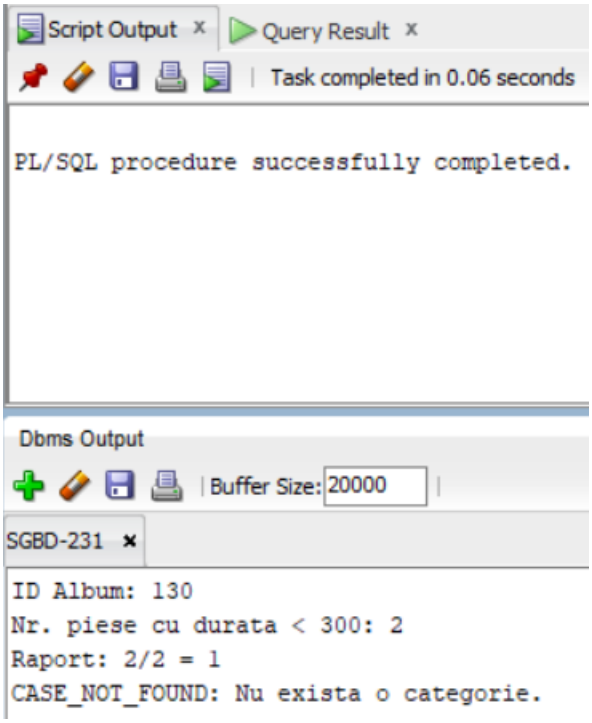
```
PL/SQL procedure successfully completed.
```

Dbms Output

Buffer Size: 20000

SGBD-231 ×

```
INVALID_NUMBER: Invalid conversion from string to number
```




```
declare
   t_id_album album_aux.id_album%type := &id;
   t_titlu    album_aux.titlu%type    := '&titlu';
begin
   insert into album_aux
   values (t_id_album, t_titlu);

   exception
      when dup_val_on_index then
         dbms_output.put_line('Deja exista un album cu acel ID!');
end;
/
rollback;
```

Daca se introduce un ID care inca nu exista, blocul va rula. Altfel, va intra pe exceptie (identificatorul deja exista).

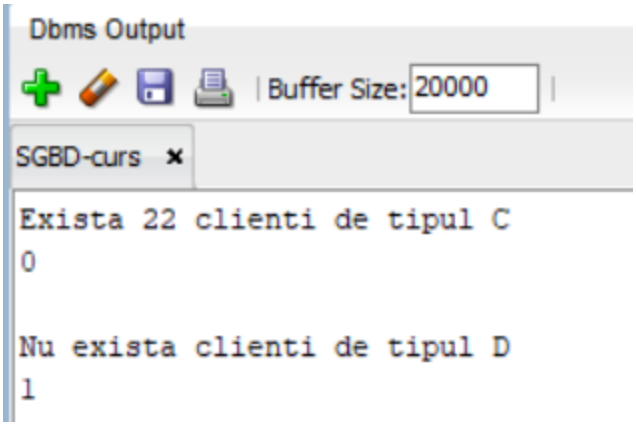
```
Script Output x   Query Result x
Task completed in 2.182 seconds
            dbms_output.put_line('Deja exista un album cu acel ID!');
end;

PL/SQL procedure successfully completed.

Dbms Output
Buffer Size: 20000
SGBD-231 x
Deja exista un album cu acel ID!
```

```
declare
   type t_num_table is table of number;
   t_nums t_num_table;
   t_aux  number;
begin
   t_aux := t_nums(1);
   dbms_output.put_line('First element: t_aux');

   exception
      when collection_is_null then
         dbms_output.put_line('Error - null collection');
end;
```

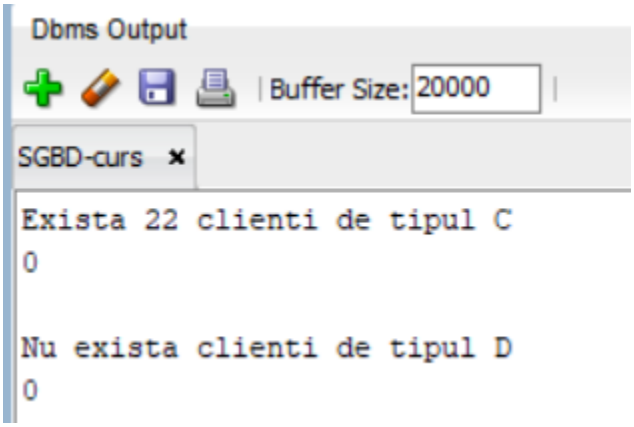
```
Script Output x   Query Result x
Task completed in 0.081 seconds

PL/SQL procedure successfully completed.

Dbms Output
Buffer Size: 20000
SGBD-231 x
Error - null collection
```

2. Pe un rand am afisat rezultatul cererii, si pe urmatorul rand timpul de executie. In concluzie, a doua varianta pentru 3.15 si varianta 3.16 sunt cele mai rapide.

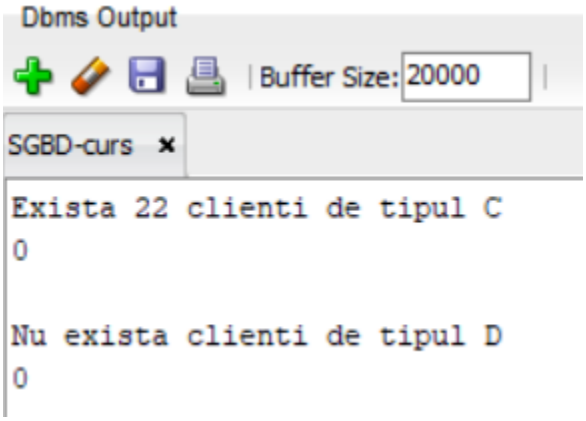
```
UNDEFINE p_clasificare
declare
   t_result varchar2(30);
   start_time number;
begin
   start_time := dbms_utility.get_time();

 SELECT
 CASE WHEN COUNT(*) = 0
 THEN 'Nu exista clienti de tipul ' ||
 UPPER('&&p_clasificare')
 WHEN COUNT(*) = 1
 THEN 'Exista 1 client de tipul ' ||
 UPPER('&&p_clasificare')
 ELSE 'Exista '|| COUNT(*) ||
 ' clienti de tipul ' ||
 UPPER('&&p_clasificare')
 END "INFO CLIENTI"
 into t_result
 FROM clasific_clienti
 WHERE clasificare = UPPER('&p_clasificare')
 AND id_categorie = 1;

 dbms_output.put_line(t_result);
 dbms_output.put_line(dbms_utility.get_time() - start_time);
end;
```



```
UNDEFINE p_clasificare
declare
   t_result varchar2(30);
   start_time number;
begin
   start_time := dbms_utility.get_time();

 SELECT
 CASE COUNT(*)
 WHEN 0
 THEN 'Nu exista clienti de tipul ' ||
 UPPER('&&p_clasificare')
 WHEN 1
 THEN 'Exista 1 client de tipul ' ||
 UPPER('&&p_clasificare')
 ELSE 'Exista '|| COUNT(*) ||
 ' clienti de tipul ' ||
 UPPER('&&p_clasificare')
 END "INFO CLIENTI"
 into t_result
 FROM clasific_clienti
 WHERE clasificare = UPPER('&p_clasificare')
 AND id_categorie = 1;

 dbms_output.put_line(t_result);
 dbms_output.put_line(dbms_utility.get_time() - start_time);
end;
```



```
UNDEFINE p_clasificare
DECLARE
 v_nr NATURAL;
 v_clasificare CHAR(1) := UPPER('&p_clasificare');
 mesaj VARCHAR2(100);
 start_time number;
BEGIN
   start_time := dbms_utility.get_time();

 SELECT COUNT(*) INTO v_nr
 FROM clasific_clienti
 WHERE clasificare = v_clasificare
 AND id_categorie = 1;
 mesaj := CASE
 WHEN v_nr = 0 THEN
 'Nu exista clienti de tipul '||
 v_clasificare
 WHEN v_nr =1 THEN
 'Exista 1 client de tipul '||
 v_clasificare
 ELSE
 'Exista ' ||v_nr || ' clienti de tipul '||
 v_clasificare
 END;
 DBMS_OUTPUT.PUT_LINE(mesaj);
 dbms_output.put_line(dbms_utility.get_time() - start_time);
END;
```



3. NO_DATA_FOUND: aceasta exceptie se aplica pentru clauzele de tip "select ... into" (cand nu se gasesc randuri); nu ar avea logica sa se aplice pentru restul (delete/update/insert). Chiar daca nu s-au gasit randuri de sters sau modificat, sau nu s-a inserat nimic, acest fapt nu va genera o eroare.
   TOO_MANY_ROWS: aceasta exceptie se aplica pentru clauzele de tip "select into" (cand sunt returnate mai multe randuri); deleteul si updateul pot sterge mai multe randuri, iar insertul, in acest bloc, insereaza un singur rand.

4. Am creat o tabela in care am introdus niste date cu diacritice; mai intai am incercat sa declar tipul de date cu “VARCHAR2(9)” (adica pe bytes) ceea ce nu a mers, iar ulterior am rezolvat problema folosind “VARCHAR2(9 CHAR)” (adica pe numar de caractere).

```
create table capitala (
   id_capitala number(2) primary key,
   nume_tara        varchar2(9 char),
   nume_capitala    varchar2(9 char)
);

begin
   insert into capitala
   values (1, 'Franța', 'Paris');
   insert into capitala
   values (2, 'Moldova', 'Chișinău');
   insert into capitala
   values (3, 'România', 'București');

   for record in (select id_capitala, nume_tara, nume_capitala from capitala) loop
      dbms_output.put_line('ID: ' || record.id_capitala || ', Tara: ' ||
      record.nume_tara || ', Capitala: ' || record.nume_capitala);
   end loop;
end;
/
rollback;
```

```
PL/SQL procedure successfully completed.
```

Dbms Output

Buffer Size: 20000

SGBD-231

```
ID: 1, Tara: Franța, Capitala: Paris
ID: 2, Tara: Moldova, Capitala: Chișinău
ID: 3, Tara: România, Capitala: București
```