Database and Information Systems, Test Variant: PL/SQL
Duration of test: 75min
Maximal number of points/minimal number of points: **15/8**

Only an result satisfying the whole task can be evaluated by non-zero points.

Tasks

1. Create procedure AddComplaint with parameters (`p_nID, p_complaintorder, p_duration, p_price`) that inserts the complaint into the table `Complaint`. The insertion is possible only in two conditions:
   (a) The purchase with the specified `p_nID` exists in the table `Complaint`. If does not exist, the procedure prints on the screen "The purchase with the nID = `p_nID` does not exist."
   (b) Total price of the complaints of the specified purchase can not be higher than the price of the specified purchase. If the total price of the complaints should be higher, the insert of the complaint is avoided and the product is taken out of production (i.e. attribute `lastProductionYear` is set to current year) and the procedure prints on the screen "Product with the serial number `serialNr` has been taken out of production."

```
create or replace procedure AddComplaint
(p_nID Complaint.nid%type,  p_complaintorder Complaint.complaintorder%type,
p_duration Complaint.duration%type,  p_price Complaint.price%type)
as
  v_count int;
  v_price_r int;
  v_price_n int;
  v_pID Purchase.pID%type;
begin
  select count(*) into v_count from Purchase where nID=p_nID;
  if (v_count = 1) then
     select sum(price) into v_price_r from Complaint where nID=p_nID;
     select price into v_price_n from Purchase where nID=p_nID;
     if (v_price_r + p_price > v_price_n) then
       select pID into v_pID from Purchase where nID = p_nID;
       update Product set lastProductionYear = EXTRACT(YEAR from SYSDATE)
       where pID=v_pID;
       dbms_output.put_line('Product with the serial number '|| v_pID ||'
       has been taken out of production.');
     else
        insert into Complaint
        values (p_nID, p_complaintorder, p_duration, p_price);
        dbms_output.put_line('Complaint has been saved.');
     end if;
  else
    dbms_output.put_line('The purchase with the nID = ' || p_nID || '
     does not exist.');
  end if;
  commit;
exception
  when others then
    rollback;
end;
```

2. Create procedure CustomerReport with one parameter `p_order` of the datatype varchar that prints the customers in the order according to the parameter `p_order`.

   Possible values of the parameter `p_order`:

   (a) **'name'** - the list of the customer is ordered lexicographically by their names.

   (b) **'purchase'** - the list of the customer is ordered by their numbers of purchases from the most frequent to the least frequent ones.

   (c) **'complaint'** - the list of the customer is ordered by their numbers of complaints from the most frequent to the least frequent ones.

```
create or replace procedure CustomerReport (p_order varchar2)
as
  CURSOR a is select name from Customer order by name;
  CURSOR b is select Customer.cID, name, count(nID)
  from Customer, Purchase
  where Customer.cID=Purchase.cID
  group by Customer.cID, name
  order by count(nID) desc;
  CURSOR c is select Customer.cID, name, count(Complaint.nID)
  from Customer, Purchase, Complaint
  where Customer.cID=Purchase.cID
  and Purchase.nID=Complaint.nID
  group by Customer.cID, name
  order by count(Complaint.nID) desc;
begin
  if (p_order = 'name') then
    dbms_output.put_line('List of the customers ordered by their name:');
    for row in a loop
      dbms_output.put_line(row.name);
    end loop;
  elsif (p_usporadani = 'purchase') then
    dbms_output.put_line('List of the customers ordered by their purchases:');
    for row in b loop
      dbms_output.put_line(row.name);
    end loop;
  else
    dbms_output.put_line('List of the customers ordered by their complaints:');
    for row in c loop
      dbms_output.put_line(row.name);
    end loop;
  end if;
end;
```