M1 Informatique – Année 2022-2023
Bases de données avancées
Épreuve écrite
J. Darmont (https://eric.univ-lyon2.fr/jdarmont/), 12/12/2022

Durée : 2h – Documents autorisés – Barème fourni à titre indicatif

PL/pgSQL (12 points)

Soit la base de données dont le schéma relationnel est donné ci-dessous.

Facility (facNo, facName)
Customer (custNo, custName, custContact, custPhone, custAddress, custCity, custState, custZip)
EventRequest (eventNo, facNo#, custNo#, dateHeld, dateReq, dateAuth, status, estCost, estAudience)

1. Écrire une fonction nommée custProfile qui renvoie le nom (custName) et le numéro de téléphone (custPhone) du client (Customer) dont le numéro de client (custNo, qui est une chaîne de caractères) est passé en paramètre de la fonction.

2. Ajouter à la fonction custProfile un traitement d'exception au cas où le numéro de client n'existe pas dans la table Customer. Ne pas réécrire toute la fonction, indiquer par des numéros dans la réponse à la question 1 les endroits où il faut insérer du code et écrire le code correspondant à part en l'associant aux numéros.

3. Écrire une fonction nommée eventList permettant de retourner les informations suivantes : eventNo (chaîne), facName (chaîne), custName (chaîne), estCost (nombre) et status (chaîne) ; triées par coût estimé (estCost) décroissant.

4. Écrire un déclencheur de nom eventCheck et sa fonction associée, qui vérifie, pour chaque nouvelle demande (EventRequest) ou modification de demande, que la date d'autorisation (dateAuth) est renseignée. Si ce n'est pas le cas, interrompre l'exécution avec le message « Erreur d'autorisation ». Sinon, vérifier que la date d'autorisation (dateAuth) est supérieure ou égale à la date de demande (dateReq) et que la date tenue (dateHeld) est supérieure ou égale à la date d'autorisation (dateAuth). Si ce n'est pas le cas, interrompre l'exécution avec le message « Erreur de date ».

5. Écrire une fonction nommée facilityViews permettant de créer, pour chaque établissement (Facility), une vue de nom facilityNom_de_l'établissement et de schéma (eventNo, status, estCost, estAudience). Attention : les noms d'établissements (facName) peuvent contenir des espaces !

XQuery (8 points)

Soient les documents XML *Customers.xml* (clients) et *Orders.xml* (commandes) dont les arborescences d'éléments et d'attributs (ces derniers étant préfixés par @) sont représentées ci-dessous. Les caractères * et ? indiquent que l'élément ou l'attribut concerné peut apparaître plusieurs fois ou pas, respectivement, dans un même document.

```
Customers
.....Customer *
.....@CustomerID
.....CompanyName
.....ContactName
.....Phone
.....FullAddress
.....Address
.....City
.....Region
.....PostalCode
.....Country
```

```
Orders
.....Order *
.....CustomerID
.....EmployeeID
.....OrderDate
.....RequiredDate
.....ShipInfo
.....@ShippedDate ?
.....ShipVia
.....Freight
.....ShipName
.....ShipAddress
.....ShipCity
.....ShipRegion
.....ShipPostalCode
.....ShipCountry
```

Formuler à l'aide du langage XQuery les requêtes suivantes, en privilégiant la lisibilité du code XQuery. Tout résultat de requête doit être un fragment de code XML bien formé.

1. Nombre de clients (Customer) et de commandes (Order).

2. Noms des fournisseurs (ShipName), sans doublon.

3. Identifiant (CustomerID) et nom (CompanyName) des clients, triés par identifiants croissants, au format suivant.

```
<Customer ID="CustomerID">
     <CompanyName>CompanyName</CompanyName>
</Customer>
```

4. Commandes dont la date d'envoi (ShippedDate) est supérieure à la date prévue (RequiredDate) – cas où la date limite est dépassée.

5. Pour chaque commande, nom du contact chez le client (ContactName) et identifiant de l'employé chez le fournisseur (EmployeeID), au format suivant.

```
<Contact>
     <atCustomer>ContactName</atCustomer>
     <atSupplier>EmployeeID</atSupplier>
</Contact>
```

6. Total du fret envoyé (Freight) par client (CustomerID), au format suivant.

```
<Customer ID="CustomerID">
     <TotalFreight>TotalFreight</TotalFreight>
</Customer>
```

7. Nombre de commandes par client (CompanyName), du plus grand au plus petit nombre de commandes, au format suivant.

```
<Group>
     <CompanyName>CompanyName</CompanyName>
     <NbOrders>NbOrders</NbOrders>
</Group>
```

8. Total du fret envoyé par région (Region) et par année de la date de commande (OrderDate), trié par région et par année, au format suivant.

```
<Group>
     <Region>Region</Region>
     <Year>Year</Year>
     <TotalFreight>TotalFreight</TotalFreight>
</Group>
```

Correction PL/pgSQL

```
-- 1 + 2
create type tCust as (
      name varchar,
      phone varchar
);

create or replace function custProfile(num varchar) returns tCust as $$
      declare
            nuplet tCust;
            n integer;
      begin
            -- Test d'existence des données (question 2)
            select count(*) into n from Customer where custNo = num;
            if n = 0 then
                  raise exception 'Pas de données pour %', num;
            end if;
            -- Récupération des données (question 1)
            select custName, custPhone into nuplet from Customer
                  where custNo = num;
            return nuplet;
      end
$$ language plpgsql;

-- 3
create type tInfos as (
      eventNo varchar,
      facName varchar,
      custName varchar,
      estCost numeric,
      status varchar
);

create or replace function eventList() returns setof tInfos as $$
      declare
            nuplet tInfos;
      begin
            for nuplet in
                  select eventNo, facName, custName, estCost, status
                  from EventRequest E, Facility F, Customer C
                  where E.facNo = F.facNo and E.custNo = C.custNo
                  order by estCost desc
            loop
                  return next nuplet;
            end loop;
            return;
      end
$$ language plpgsql;

-- 4
create or replace function eventCheck() returns trigger as $$
      begin
            if new.dateAuth is null then
                  raise exception 'Erreur d''autorisation';
            elsif new.dateAuth < new.dateReq and
                  new.dateHeld < new.dateAuth then
                        raise exception 'Erreur de date';
            end if;
            return new;
      end
$$ language plpgsql;
```

```
create trigger eventCheck
before insert or update
on EventRequest
for each row
execute procedure eventCheck();

-- 5
create or replace function facilityViews() returns void as $$
      declare
            nuplet record;
      begin
            for nuplet in select * from Facility loop
                  execute     'create view facility' ||
                              replace(nuplet.facName, ' ', '') ||
                              ' as select eventNo, status, estCost, estAudience
                              from EventRequest where facNo = ' ||
                              '''' || nuplet.facNo|| '''';
            end loop;
      end
$$ language plpgsql;
```

Correction XQuery

```
(: 1 :)
let $cc := count(//Customer),
    $co := count(//Order)
return  <res>
          <NbCustomers>{$cc}</NbCustomers>
          <NbOrders>{$co}</NbOrders>
        </res>

(: 2 :)
for $o in distinct-values(//ShipName)
return <ShipName>{$o}</ShipName>

(: 3 :)
for $c in //Customer
order by $c/@CustomerID
return <Customer ID="{data($c/@CustomerID)}">
         {$c/CompanyName}
       </Customer>

(: 4 :)
for $o in //Order
where $o/ShipInfo/@ShippedDate > $o/RequiredDate
return $o

(: 5 :)
for $o in //Order,
    $c in //Customer
where $c/@CustomerID = $o/CustomerID
return  <Contact>
          <atCustomer>{data($c/ContactName)}</atCustomer>
          <atSupplier>{data($o/EmployeeID)}</atSupplier>
        </Contact>

(: 6 :)
for $o in //Order
group by $c := $o/CustomerID
let $t := sum($o/ShipInfo/Freight)
return  <Customer ID="{distinct-values(data($o/CustomerID))}">
          <TotalFreight>{$t}</TotalFreight>
        </Customer>
```

```
(: 7 :)
for $c in //Customer,
    $o in //Order
where $c/@CustomerID = $o/CustomerID
group by $g := $c/CompanyName
let $n := count($o)
order by $n descending
return    <Group>
           <CompanyName>{$g}</CompanyName>
           <NbOrders>{$n}</NbOrders>
         </Group>

(: 8 :)
for $c in //Customer,
    $o in //Order
where $c/@CustomerID = $o/CustomerID
group by $g1 := $c/FullAddress/Region,
          $g2 := year-from-date($o/OrderDate)
order by $g1, number($g2)
let $t := sum($o/ShipInfo/Freight)
return  <Group>
           <Region>{$g1}</Region>
           <Year>{$g2}</Year>
           <TotalFreight>{$t}</TotalFreight>
         </Group>
```