M1 Informatique – Année 2024-2025
**Bases de données avancées**
**TD n° 7 (XQuery) : Requêtes complexes**
J. Darmont (https://eric.univ-lyon2.fr/jdarmont/), 21/08/24

**Préliminaires**

1. Télécharger les documents XML suivants, qui concernent des commandes (`orders.xml`) de livres (`books.xml`) par des clients (`customers.xml`), et les placer dans un nouveau dossier.
   - https://eric.univ-lyon2.fr/jdarmont/docs/customers.xml
   - https://eric.univ-lyon2.fr/jdarmont/docs/orders.xml
   - https://eric.univ-lyon2.fr/jdarmont/docs/books.xml

2. Dans BaseX, créer une nouvelle base de données et indiquer le **dossier** contenant les trois documents XML comme source de données.

**Exercice 1 : Jointures**

Formuler les requêtes suivantes en XQuery.

1 Liste des clients.

2 Noms des clients qui ont commandé des livres (formuler la condition de jointure à l'aide d'une clause `where`).

3 Noms distincts des clients qui ont commandé des livres (utiliser la fonction `distinct-values` et formuler la condition de jointure à l'aide d'un prédicat de chemin).

4 Noms des clients distincts qui ont commandé des livres (utiliser la fonction `exists` et, à partir de maintenant, formuler toute condition de jointure à l'aide d'une clause `where` sauf si c'est explicitement demandé). Conclusion ?

5 Noms des clients qui n'ont commandé aucun livre.

6 Noms des clients distincts qui ont commandé au moins un livre en quantité supérieure à 1.

7 Liste des commandes indiquant le nom du client, la date de la commande, le titre du livre et la quantité commandée. Formater le résultat comme suit.

```
<order>
    <customer>CUSTOMER_NAME</customer>
    <date>ORDER_DATE</date>
    <book>BOOK_TITLE
        <qty>QUANTITY_ORDERED</qty>
    </book>
</order>
```

8 Même requête que la question 7 en formulant les conditions de jointure à l'aide de prédicats de chemin.

9 Noms des clients qui ont commandé des livres de science-fiction.

**Exercice 2 : Agrégation et regroupement**

Formuler les requêtes suivantes en XQuery.

1 Nombre total de livres dans le catalogue.

2 Prix moyen des livres de genre « Computer ».

3 Prix minimum, moyen et maximum des livres.

4 Nombre de livres par auteur.

5 Nombre et prix moyens des livres par genre, triés par genre. À partir de cette question, toujours trier le résultat en fonction du ou des éléments de regroupement.

6 Longueur moyenne des descriptions par genre.

7 Plus grande longueur moyenne des descriptions par genre.

8 Nombre de livres par auteur et par genre.

9 Quantité totale commandée par livre (indiquer les titres des livres).

10 Quantité totale commandée par genre.

**Correction**

```
(: Exercice 1 :)

(: 1 :)
/directory/customer

(: 2 :)
for  $c in /directory/customer,
     $o in /list/order
where $c/@id = $o/customer/@id
return $c/name

(: 3 :)
for  $c in /directory/customer,
     $o in distinct-values(/list/order/customer[@id = $c/@id])
return $c/name

(: 4 :)
for $c in /directory/customer
where exists(
     for $o in /list/order
     where $o/customer/@id = $c/@id (: pour une meilleure lisibilité :)
     return $o)
return $c/name
(: avec exists, plus besoin de distinct-values() :)

(: 5 :)
for $c in /directory/customer
where not(exists(
     for $o in /list/order
     where $o/customer/@id = $c/@id
     return $o))
return $c/name

(: 6 :)
for  $c in /directory/customer,
     $o in /list/order
where $o/customer/@id = $c/@id
and $o//qty > 1
return $c/name


(: 7 :)
for  $c in /directory/customer,
     $b in /catalog/book,
     $o in /list/order
where $o/customer/@id = $c/@id
and $o/book/@id = $b/@id
let $q := $o/book[@id = $b/@id]/qty
return   <order>
              <customer>{data($c/name)}</customer>
              <date>{data($o/date)}</date>
              <book>{data($b/title)}
                  <qty>{data($q)}</qty>
              </book>
         </order>
```

```
(: 8 :)
for  $c in /directory/customer,
     $b in /catalog/book,
     $o in /list/order[customer/@id = $c/@id and book/@id = $b/@id]
let $q := $o/book[@id = $b/@id]/qty
return    <order>
               <customer>{data($c/name)}</customer>
               <date>{data($o/date)}</date>
               <book>{data($b/title)}
                   <qty>{data($q)}</qty>
               </book>
          </order>

(: 9 :)
for  $c in /directory/customer,
     $b in /catalog/book,
     $o in /list/order
where $o/customer/@id = $c/@id
and $o/book/@id = $b/@id
and $b/genre = "Science Fiction"
return $c/name
```

**(: Exercice 2 :)**

```
(: 1 :)
count(/catalog/book)

(: 2 :)
avg(//book[genre = "Computer"]/price)

(: 3 :)
let  $priceList := //book/price,
     $m := min($priceList),
     $a := avg($priceList),
     $M := max($priceList)
return    <price>
               <min>{$m}</min>
               <avg>{$a}</avg>
               <max>{$M}</max>
          </price>

(: 4 :)
for $b in /catalog/book
group by $a := $b/author
return <nb_books author="{data($a)}">{count($b)}</nb_books>

(: 5 :)
for $b in /catalog/book
group by $g := $b/genre
let  $c := count($b),
     $a := avg($b/price)
order by $g
return    <genre name="{data($g)}">
               <number_books>{$c}</number_books>
               <avg_price>{$a}</avg_price>
          </genre>

(: 6 :)
for $b in /catalog/book
group by $g := $b/genre
let $a := avg($b/string-length(description))
order by $g
return    <avg_desc_length genre="{data($g)}">{$a}</avg_desc_length>
```

```
(: 7 :)
let $maxal := max(      for $b in /catalog/book
                        group by $g := $b/genre
                        let $a := avg($b/string-length(description))
                        order by $g
                        return      <avg_desc_length genre="{data($g)}">
                                          {$a}
                                    </avg_desc_length>      )
return <max_avg_desc_length>{$maxal}</max_avg_desc_length>

(: 8 :)
for $b in /catalog/book
group by $a := $b/author, $g := $b/genre
let $c := count($b)
order by $a, $g
return <nb_books author="{data($a)}" genre="{data($g)}">{$c}</nb_books>

(: 9 :)
for  $b in /catalog/book,
     $o in //order/book
where $o/@id = $b/@id
group by $t := $b/title
let $q := sum($o/qty)
order by $t
return    <book>{data($t)}
               <total_qty>{$q}</total_qty>
          </book>

(: 10 :)
for  $b in /catalog/book,
     $o in //order/book
where $o/@id = $b/@id
group by $g := $b/genre
let $q := sum($o/qty)
order by $g
return    <genre>{data($g)}
               <total_qty>{$q}</total_qty>
          </genre>
```