NOUAR MOHAMMED ELHAFNOUI GROUPE 2

LOW HOMEWORK

QUESTION 1:

<xsl:template> : Définit une règle **de** transformation pour un ou plusieurs éléments XML

<xsl:apply-templates> : Permet une transformation récursive des éléments XML.

<xsl:call-template> : Utilisé pour la **réutilisation** de blocs de transformation

Fonctions en XSLT : Permettent de **manipuler du texte, des nombres, des dates**, ou de faire des tests logiques.

QUESTION 2 :

exo 1 :

1. Lister tous les albums en ordre alphabétique ascendant :

```
for $a in doc("albums.xml")//album

order by $a/titre

return $a/titre
```

2. Les albums publiés après 1970:

```
for $a in doc("albums.xml")//album

where xs:integer($a/date/annee) > 1970

return $a/titre
```

3. Les auteurs ayant participé à plus d'un album:

```
for $auteur in distinct-values(doc("albums.xml")//auteur)

let $albums := doc("albums.xml")//album[auteur = $auteur]

where count($albums) > 1

return <auteur nom="{$auteur}" nbAlbums="{count($albums)}"/>
```

4. Trouver l'album le plus récent de chaque série:

```
for $s in distinct-values(doc("albums.xml")//album/@serie)
```

```
let $albums := doc("albums.xml")//album[@serie = $s]
let $recent := max(for $a in $albums return xs:integer($a/date/annee))
let $dernier := $albums[date/annee = $recent]
return
  <serie nom="{$s}">
    {$dernier/titre}
  </serie>
```

5. Regrouper les albums par série et compter le nombre d'albums par série:

```
for $s in distinct-values(doc("albums.xml")//album/@serie)
let $albums := doc("albums.xml")//album[@serie = $s]
return <serie nom="{$s}" nb-albums="{count($albums)}"/>
```

6. Trouver la série ayant le plus d'albums

```
let $groupes :=
  for $serie in distinct-values(doc("albums.xml")//album/@serie)
  let $count := count(doc("albums.xml")//album[@serie = $serie])
  return <serie nom="{$serie}" nb="{$count}"/>

return max($groupes, (), fn:data(@nb))
```

Pour afficher la série elle-même (pas juste le nombre) :

```
let $groupes :=
  for $serie in distinct-values(doc("albums.xml")//album/@serie)
  let $count := count(doc("albums.xml")//album[@serie = $serie])
  return <serie nom="{$serie}" nb="{$count}"/>

let $max := max($groupes/@nb)
```

```
return $groupes[@nb = $max]
```

7. Les années où le plus d’albums ont été publiés

```
let $groupes :=
  for $annee in distinct-values(doc("albums.xml")//annee)
  let $count := count(doc("albums.xml")//annee[. = $annee])
  return <annee valeur="{$annee}" nb="{$count}"/>

let $max := max($groupes/@nb)
return $groupes[@nb = $max]
```

8. Albums avec plus de 10 ans d’écart du précédent de la même série

```
for $serie in distinct-values(doc("albums.xml")//album/@serie)
let $albums :=
  for $a in doc("albums.xml")//album[@serie = $serie]
  order by xs:integer($a/date/annee)
  return $a
for $i in 2 to count($albums)
let $current := $albums[$i]
let $prev := $albums[$i - 1]
where abs(xs:integer($current/date/annee) - xs:integer($prev/date/annee)) > 10
return $current/titre
```

9. Auteurs ayant participé à plusieurs séries différentes

```
for $auteur in distinct-values(doc("albums.xml")//auteur)
let $series := distinct-values(doc("albums.xml")//album[auteur = $auteur]/@serie)
where count($series) > 1
return <auteur nom="{$auteur}" nbSeries="{count($series)}"/>
```

10. Auteur ayant écrit le plus d’albums

```
let $auteurs :=
  for $a in distinct-values(doc("albums.xml")//auteur)
  let $nb := count(doc("albums.xml")//album[auteur = $a])
  return <auteur nom="{$a}" nb="{$nb}"/>

let $max := max($auteurs/@nb)
return $auteurs[@nb = $max]
```

11. Albums avec le même titre mais série différente

```
for $titre in distinct-values(doc("albums.xml")//titre)
let $albums := doc("albums.xml")//album[titre = $titre]
let $series := distinct-values($albums/@serie)
where count($series) > 1
return
  <titre nom="{$titre}">
    { for $a in $albums return <album serie="{$a/@serie}"
numero="{$a/@numero}"/> }
  </titre>
```

12. Fonction pour les albums les plus anciens d’un auteur (ex: Hergé)

```
declare function local:albums-plus-anciens($auteur as xs:string) as element()* {
  let $albums := doc("albums.xml")//album[auteur = $auteur]
  let $min := min(for $a in $albums return xs:integer($a/date/annee))
  return $albums[date/annee = $min]
};

local:albums-plus-anciens("Hergé")
```

13. Ajouter l’auteur "Uderzo" à l’album numéro 1 de la série "Tintin" (XQUF)

```
copy $mod := doc("albums.xml")
modify (
  for $a in $mod//album[@numero="1" and @serie="Tintin"]
  return insert node <auteur>Uderzo</auteur> after $a/titre
)
return $mod
```

14. Ajouter un attribut editeur="La plume" à l’album 3 de "Astérix"

```
copy $mod := doc("albums.xml")
modify (
  for $a in $mod//album[@numero="3" and @serie="Astérix"]
  return insert node attribute editeur {"La plume"} into $a
)
return $mod
```

15. Ajouter "Hergé" à tous les albums Tintin sans auteur

```
copy $mod := doc("albums.xml")
modify (
  for $a in $mod//album[@serie="Tintin"]
  where empty($a/auteur)
  return insert node <auteur>Hergé</auteur> after $a/titre
)
return $mod
```

16. Changer serie="Astérix" → serie="Astérix et Obélix"

```
copy $mod := doc("albums.xml")
modify (
```

```
  for $a in $mod//album[@serie="Astérix"]
  return replace value of node $a/@serie with "Astérix et Obélix"
)
return $mod
```

17. Supprimer les albums de Tintin publiés avant 1950

```
copy $mod := doc("albums.xml")
modify (
  for $a in $mod//album[@serie="Tintin"]
  where xs:integer($a/date/annee) < 1950
  return delete node $a
)
return $mod
```

18. Ajouter 1 an à la date des albums Astérix après 1980

```
copy $mod := doc("albums.xml")
modify (
  for $a in $mod//album[@serie="Astérix"]
  where xs:integer($a/date/annee) > 1980
  return replace value of node $a/date/annee with xs:string(xs:integer($a/date/annee) +
1)
)
return $mod
```

19. Changer <album> en <Premier_album> si numero=1

```
copy $mod := doc("albums.xml")
modify (
  for $a in $mod//album[@numero="1"]
```

```
  return rename node $a as "Premier_album"
)
return $mod
```

20. Ajouter un nouvel album à la fin de la série Tintin

```
copy $mod := doc("albums.xml")
modify (
  let $last := $mod//album[last()]
  return insert node
    <album numero="25" serie="Tintin">
      <titre>Le Nouveau Mystère</titre>
      <auteur>Hergé</auteur>
      <date>
        <mois>mars</mois>
        <annee>2025</annee>
      </date>
    </album>
  after $last
)
return $mod
```

exo2:

```
(: EXERCICE 2 : Films et Artistes - TD XQuery :)

(: 1. Titre, genre, pays pour tous les films avant 1970 :)
for $f in doc("Films.xml")//FILM
where xs:integer($f/@annee) < 1970
```

```
return <film> {$f/TITRE, $f/GENRE, $f/PAYS} </film>

(: 2. Rôles joués par Bruce Willis :)
for $film in doc("Films.xml")//FILM
for $role in $film/ROLES/ROLE
where $role/PRENOM = "Bruce" and $role/NOM = "Willis"
return $role

(: 3. Rôles joués par Bruce Willis avec titre et personnage :)
for $film in doc("Films.xml")//FILM
for $role in $film/ROLES/ROLE
where $role/PRENOM = "Bruce" and $role/NOM = "Willis"
return <role>
  <titre>{$film/TITRE/text()}</titre>
  <personnage>{$role/INTITULE/text()}</personnage>
</role>

(: 4. Nom du metteur en scène du film "Vertigo" :)
let $film := doc("Films.xml")//FILM[TITRE = "Vertigo"]
let $id := $film/MES/@idref
let $artiste := doc("Artistes.xml")//ARTISTE[@id = $id]
return concat($artiste/ARTPNOM, " ", $artiste/ARTNOM)

(: 5. Chaque artiste avec les films qu'il a réalisés :)
for $a in doc("Artistes.xml")//ARTISTE
```

```
let $films := doc("Films.xml")//FILM[MES/@idref = $a/@id]
where count($films) > 0
return <artiste nom="{concat($a/ARTPNOM, ' ', $a/ARTNOM)}">
  { for $f in $films return <film>{$f/TITRE/text()}</film> }
</artiste>

(: 6. Âge du metteur en scène à la sortie de chaque film :)
for $film in doc("Films.xml")//FILM
let $id := $film/MES/@idref
let $artiste := doc("Artistes.xml")//ARTISTE[@id = $id]
let $naiss := xs:integer($artiste/ANNEENAISS)
let $annee := xs:integer($film/@annee)
where exists($naiss)
return <film titre="{$film/TITRE}" age="{$annee - $naiss}"/>

(: 7. Pour chaque genre, les titres des films :)
for $g in distinct-values(doc("Films.xml")//FILM/GENRE)
let $films := doc("Films.xml")//FILM[GENRE = $g]
return <genre nom="{$g}">
  { for $f in $films return <titre>{$f/TITRE/text()}</titre> }
</genre>

(: 8. Artistes ayant joué et réalisé un film :)
for $a in doc("Artistes.xml")//ARTISTE
let $id := $a/@id
```

```
let $films := doc("Films.xml")//FILM[
  MES/@idref = $id and
  ROLES/ROLE[NOM = $a/ARTNOM and PRENOM = $a/ARTPNOM]
]
where count($films) > 0
return <artiste nom="{concat($a/ARTPNOM, ' ', $a/ARTNOM)}">
  { for $f in $films return <film annee="{$f/@annee}">{$f/TITRE/text()}</film> }
</artiste>
```