ЗАДАЧА 1

Со цел да се спојат дадените датотеки во еден фајл, напишав квери кое ја постигнува таа цел, тоа се наоѓа во датотека во рамки на овој фолдер именувана како systemQuery.xquery.

а) За ова барање, напишав XML Schema за System.xml во xml-schema.xsd. Сите побарани ограничувања се напишани тука во рамки на овој фајл.

б) За ова барање го извршив гореспоменатото квери кое даде резултат соодветен на тоа што беше предложено како шема и изглед на xml фајл според побарувањата и дадената структура на засебните фајлови.

в) Не се соочив со никакви проблеми до сега. Со цел попрактична работа, направив во рамки на ентитетот Артист да се наоѓаат информации за ДЈ,Група или Пејач. На овој начин е овозможен брз пристап до таквите информации. Албумите се дел од артистите, поврзувајќи ги соодветните албуми со соодветни артисти. Исто така, ентитетот ЦД е дел од албум, а Рентата е дел од соодветен Клиент. Временски гледано, побавно ќе биде доколку сакаме да ги пребараме сите ЦД-ињa, бидејќи истите се наоѓаат во рамки на Албум, а албумите пак се вгнездени во Артист. Исто така од практичен аспект, иако сето ова работи, е попроблематично, поради тоа што треба да се прават joins.

ЗАДАЧА 2

XQuery изрази

а) Прикажи ги сите артисти (сите информации) заедно со насловите и датумите на издавање на сите албуми кои ги имаат снимено.

```
xquery version "3.1";

for $artist in doc("Artists.xml")//ARTISTS/ARTIST
return
  <ARTIST>
    <NAME>{ $artist/NAME/text() }</NAME>
    <COUNTRY>{ $artist/COUNTRY/text() }</COUNTRY>
    <GENRE>{ $artist/GENRE/text() }</GENRE>
    <ALBUMS>{
      for $album in doc("Albums.xml")//ALBUMS/ALBUM[@ARTIST_ID = $artist/@ID]
      return
        <ALBUM>
          <NAME>{ $album/NAME/text() }</NAME>
          <RELEASE_YEAR>{ $album/RELEASE_YEAR/text()}</RELEASE_YEAR>
        </ALBUM>
    }</ALBUMS>
  </ARTIST>
```

---

b) Најди ги првите три члена со најголем број изнајмувања.

```
xquery version "3.1";

let $rentals := doc("Rent.xml")//RENTS
let $rentCounts := for $clientID in distinct-values($rentals/RENT/@CLIENT_ID)
                   let $count := count($rentals/RENT[@CLIENT_ID eq $clientID])
                   order by $count descending
                   return <ClientRentCount>
                          <ClientID>{$clientID}</ClientID>
                          <RentCount>{$count}</RentCount>
```

```
                    </ClientRentCount>
return subsequence($rentCounts, 1, 3)
```
——

c) Врати ги сите албуми кои биле изнајмени најмалку три пати во периодот од јануари до март 2020.

```
xquery version "3.1";

for $album in doc("Albums.xml")//ALBUM
where count(
   for $rent in doc("Rent.xml")//RENT[matches(FROM_DATE, '^[0-9]{2}/0(1|2|3)/2020$')],
      $cd in doc("CatalogCD.xml")//CD
   where $rent/@CD_ID = $cd/@ID and $cd/@ALBUM_ID = $album/@ID
   return $rent
) ge 3
return $album
```
——

d) Прикажи го клиентот кој најмногу изнајмувал од најизнајмениот албум заедно со информациите за албумот

```
xquery version "3.1";

let $allAlbums:=doc("Albums.xml")//ALBUM
let $allRentals:=doc("Rent.xml")//RENT
let $allCDs:=doc("CatalogCD.xml")//CD
let $allClients:=doc("Clients.xml")//CLIENT
let $rentAlbum := (
   for $album in $allAlbums
   let $album_counter := count(
      for $rent in $allRentals,
      $cd in $allCDs
      where $rent/@CD_ID = $cd/@ID and $cd/@ALBUM_ID = $album/@ID
      return $rent
   )
   order by $album_counter descending
   return $album
)[1]

return (
   for $client in $allClients
   let $cnt := count(
      for $rent in $allRentals,
         $cd in $allCDs
      where $rent/@CD_ID = $cd/@ID and $cd/@ALBUM_ID = $rentAlbum/@ID and $rent/
@CLIENT_ID = $client/@ID
      return $rent
   )
   order by $cnt descending
   return <MOST_RENTED>
            {$rentAlbum}
            {$client}
      </MOST_RENTED>
)[1]
```
——-

e) Прикажи го просечниот број на изнајмувања на секој албум посебно.

```
xquery version "3.1";
```

```
let $allCDs:=doc("CatalogCD.xml")//CD
let $allRentals:=doc("Rent.xml")//RENT
let $counter:=count($allRentals)
for $album in doc("Albums.xml")//ALBUM
let $rentals := count(
   for $cd in $allCDs,
      $rent in $allRentals
   where $cd/@ALBUM_ID = $album/@ID and $rent/@CD_ID = $cd/@ID
   return $rent
)
return <RESULT><ALBUM_ID>{$album/@ID}</ALBUM_ID><COUNT>{$rentals}</COUNT>
<AVERAGE>{$rentals div $counter}</AVERAGE></RESULT>
```

---

f) Врати го вкупниот профит кој е направен за секој албум посебно.

```
xquery version "3.1";

let $albums := doc("Albums.xml")//ALBUMS/ALBUM
let $rentals := doc("Rent.xml")//RENTS/RENT
   for $album in $albums
   let $albumID := $album/@ID
   let $albumName := $album/NAME
   let $albumPrice := xs:decimal(substring($album/PRICE, 2))
let $totalRentalCount := count(
   for $rent in $rentals,
      $cd in doc("CatalogCD.xml")//CD
      where $rent/@CD_ID = $cd/@ID and $cd/@ALBUM_ID = $album/@ID
      return $rent
)
let $totalProfit := $albumPrice * $totalRentalCount
   return
      <Album>
         <ID>{$albumID}</ID>
         <Name>{$albumName}</Name>
         <TotalRentalCount>{$totalRentalCount}</TotalRentalCount>
         <TotalProfit>{$totalProfit}</TotalProfit>
      </Album>
```

---

g) Најди ја групата чиј албум е најмногу пати изнајмен, но постои барем едно CD кое во моментот не е изнајмено.

```
xquery version "3.1";

let $allGroups := doc("Groups.xml")//GROUP
let $allArtists := doc("Artists.xml")//ARTIST
let $allAlbums := doc("Albums.xml")//ALBUM
let $allRents := doc("Rent.xml")//RENT
let $allCDs := doc("CatalogCD.xml")//CD

let $rentedAlbumGroup :=
   for $group in $allGroups
   let $artist := $allArtists[@ID = $group/@ID]
   let $mostRented :=
      let $albumCounts :=
         for $album in $allAlbums[@ARTIST_ID = $group/@ID]
         let $rents :=
            for $rent in $allRents
            let $cd := $allCDs[@ID = $rent/@CD_ID and @ALBUM_ID = $album/@ID]
            where $cd
```

```
            return $rent
         return count($rents)
      let $maxCount := max($albumCounts)
      let $maxAlbum :=
         for $album in $allAlbums[@ARTIST_ID = $group/@ID]
         let $rents :=
            for $rent in $allRents
            let $cd := $allCDs[@ID = $rent/@CD_ID and @ALBUM_ID = $album/@ID]
            where $cd
            return $rent
         where count($rents) = $maxCount
         return $album
      return $maxAlbum

   let $availableCDs :=
      for $cd in $allCDs
      where $cd/@ALBUM_ID = $mostRented/@ID and not(exists(
         for $rent in $allRents
         where $rent/@CD_ID = $cd/@ID and not(exists($rent/RETURN_DATE))
         return $rent
      ))
      return $cd

   where $mostRented
   order by count($availableCDs) descending
   return <RESULT>{$group}{$mostRented}</RESULT>

return $rentedAlbumGroup[1]
```

— —

h) Прикажи ги сите клиенти кои барем еднаш изнајмиле CD во времетраење пократко од 10 дена.

```
let $allClients := doc("Clients.xml")//CLIENT
for $client in $allClients
where exists(
   let $rents := doc("Rent.xml")//RENT[@CLIENT_ID = $client/@ID and exists(RETURN_DATE)]
   for $rent in $rents
   let $fromDate := xs:date(concat(substring($rent/FROM_DATE, 7, 4), '-', substring($rent/
FROM_DATE, 1, 2), '-', substring($rent/FROM_DATE, 4, 2)))
   let $returnDate := xs:date(concat(substring($rent/RETURN_DATE, 7, 4), '-', substring($rent/
RETURN_DATE, 1, 2), '-', substring($rent/RETURN_DATE, 4, 2)))
   let $differenceDays := xs:integer(($returnDate - $fromDate) div xs:dayTimeDuration("P1D"))
   where $differenceDays lt 10
   return $rent
)
return <CLIENT id="{$client/@ID}"> {$client/*}</CLIENT>
```

i) Најди го омилениот артист на секој клиент посебно (сите информации за артистот од кој клиентот има изнајмено најмногу албуми, ако има повеќе вратете го првиот).

```
xquery version "3.1";
let $allArtists := doc("Artists.xml")//ARTIST
let $allRents := doc("Rent.xml")//RENT
let $allCDs := doc("CatalogCD.xml")//CD
let $allAlbums := doc("Albums.xml")//ALBUM
for $client in doc("Clients.xml")//CLIENT
let $art := (
```

```
    for $artist in $allArtists
    let $counter := count(
        for $rent in $allRents,
            $cd in $allCDs,
            $album in $allAlbums
        where $client/@ID = $rent/@CLIENT_ID and $rent/@CD_ID = $cd/@ID and $cd/@ALBUM_ID
= $album/@ID and $album/@ARTIST_ID = $artist/@ID
        return $rent
    )
    order by $counter descending
    return <FAVOURITE_ARTIST>{$artist} <COUNT>{$counter}</COUNT></FAVOURITE_ARTIST>
)[1]
return <CLIENT_FAVOURITE>{$client} {$art}</CLIENT_FAVOURITE>
```

---

j) Најди го најомилениот артист (артистот кој е омилен на најголемиот број клиенти).

```
xquery version "3.1";
let $allArtists:=doc("Artists.xml")//ARTIST
let $allRentals:=doc("Rent.xml")//RENT
let $allCDs:=doc("CatalogCD.xml")//CD
let $allAlbums:=doc("Albums.xml")//ALBUM
let $favoriteArtist :=
    for $artist in $allArtists
    let $counter := count(
        for $rent in $allRentals,
            $cd in $allCDs,
            $album in $allAlbums
        where $rent/@CD_ID = $cd/@ID and $cd/@ALBUM_ID = $album/@ID and $album/
@ARTIST_ID = $artist/@ID
        return $rent
    )
    order by $counter descending
    return <FAVOURITE_ARTIST>{$artist} <COUNT>{$counter}</COUNT></FAVOURITE_ARTIST>
return subsequence($favoriteArtist, 1, 1)
```

---

k) Напиши кориснички дефинирана функција за генерирање месечен извештај. Месецот и годината се задаваат како влезни параметри, а во извештајот за тековниот месец ќе се вратат профитот направен од сите албуми и бројот на продажби по албум за секој артист посебно.

```
declare function local:salesReport($month as xs:integer, $year as xs:integer) as element() {
    let $dateToMatch := doc("Rent.xml")//RENT[matches(FROM_DATE, fn:concat('^0?',$month,'/
[0-9]{2}/',$year, '$'))]
    let $report :=
        for $artist in doc("Artists.xml")//ARTIST
        let $albums :=
            for $album in doc("Albums.xml")//ALBUM[@ARTIST_ID = $artist/@ID]
            let $price := xs:decimal(substring($album/PRICE, 2))
            let $counter := count(
                for $rent in $dateToMatch
                let $cd := doc("CatalogCD.xml")//CD[@ID = $rent/@CD_ID and @ALBUM_ID = $album/
@ID]
                where $cd
                return $rent
```

```
            )
            let $profit := $price * $counter
            return
                <ALBUM id="{$album/@ID}">
                    {$album/*}
                    <PROFIT>{$profit}</PROFIT>
                    <SALES>{$counter}</SALES>
                </ALBUM>
        let $totalProfit := sum($albums/PROFIT)
        return
            <ARTIST id="{$artist/@ID}">
                {$artist/*}
                <TOTAL_PROFIT>{$totalProfit}</TOTAL_PROFIT>
                <ALBUMS>{$albums}</ALBUMS>
            </ARTIST>
    return <SalesReport>{$report}</SalesReport>
};

local:salesReport(4, 2020)
```