Exercises: XML Processing

This document defines the exercise assignments for the "Spring Data" course @ SoftUni.

Product Shop Database

In the next exercises, you will be required to **use the models** from the previous exercise for JSON processing.

1. Seed the Database

Import the data from the provided files (**`users.xml, products.xml, categories.xml`**).

Import the **users** first. When importing the products, randomly **select the buyer** and **the seller** from the existing users. Leave out some **products** that have **not been sold** (i.e. buyer is null).

Randomly **generate categories** for each product from the existing categories.

2. Query and Export Data

Write the below described queries and **export** the returned data to the specified **format**.

Query 1 – Products in Range

Get all products in a specified **price range** (e.g. 500 to 1000) which have **no buyer**. Order them by price (from lowest to highest). Select only the **product name**, **the price** and **the full name of the seller**. Export the result to XML.

products-in-range.xml

```
<?xml version="1.0" encoding="utf-8"?>
<products>
  <product name="TRAMADOL HYDROCHLORIDE" price="516.46" seller="Christine Gomez" />
  <product name="Allopurinol" price="518.50" seller="Kathy Gilbert" />
  <product name="Parsley" price="519.06" seller="Jacqueline Perez" />
  ...
</products>
```

Query 2 – Successfully Sold Products

Get all users, who have **at least 1 item sold** with a **buyer**. Order them by **last name**, then by **first name**. Select the person's **first** and **last name**. For each of the **products sold** (products with buyers), select the product's **name**, **price** and the buyer's **first** and **last name**.

users-sold-products.xml

```
<?xml version="1.0" encoding="utf-8"?>
<users>
  <user first-name="Carl" last-name="Daniels">
      <sold-products>
          <product>
              <name>Peter Island Continous sunscreen kids</name>
              <price>471.30</price>
              <buyer-first-name>Anna</buyer-first-name>
```

```
            <buyer-last-name>Parker</buyer-last-name>
        </product>
        <product>
            <name>Warfarin Sodium</name>
            <price>1379.79</price>
            <buyer-first-name>Brandon</buyer-first-name>
            <buyer-last-name>Fuller</buyer-last-name>
        </product>
        ...
    </sold-products>
</user>
...
</users>
```

Query 3 – Categories by Products Count

Get **all categories**. Order them by the **number of products** in that category (a product can be in many categories). For each category select its **name**, the **number of products**, the **average price of those products** and the **total revenue** (total price sum) of those products (regardless if they have a buyer or not).

categories-by-products.xml

```
<?xml version="1.0" encoding="utf-8"?>
<categories>
  <category name="Sports">
      <products-count>49</products-count>
      <average-price>754.327755</average-price>
      <total-revenue>36962.06</total-revenue>
  </category>
  <category name="Adult">
      <products-count>46</products-count>
      <average-price>905.283478</average-price>
      <total-revenue>41643.04</total-revenue>
  </category>
...
</categories>
```

Query 4 – Users and Products

Get all users, who have **at least 1 product sold**. Order them by the **number of products sold** (from highest to lowest), then by **last name** (ascending). Select only their **first** and **last name**, **age** and for each product - **name** and **price**.

Export the results to **XML**. Follow the format below to better understand how to structure your data.

users-and-products.xml

```
<?xml version="1.0" encoding="utf-8"?>
```

```
<users count="35">
  <user first-name="Carl" last-name="Daniels" age="59">
    <sold-products count="10">
      <product name="Finasteride" price="1374.01" />
      <product name="Peter Island Continous sunscreen kids" price="471.30" />
      <product name="Warfarin Sodium" price="1379.79" />
      <product name="Gilotrif" price="1454.77" />
      <product name="Cold and Cough" price="218.14" />
      ...
    </sold-products>
  </user>
  <user last-name="Harris">
    <sold-products count="9">
      <product name="Clarins Paris Skin Illusion - 114 cappuccino" price="811.42" />
      ...
    </sold-products>
  </user>
  ...
</users>
```

Car Dealer Database

In the next exercises, you will be required to **use the models** from the previous exercise for JSON processing.

3. Car Dealer Import Data

Import data from the provided files (**suppliers.xml, parts.xml, cars.xml, customers.xml**).

First, import the **suppliers**. When importing the **parts** set to each part a **random supplier** from the already imported suppliers. Then, when importing the cars add **between 10 and 20 random parts** to each car. Then import **all customers**. Finally, import the **sales records** by **randomly** selecting a **car,** a **customer** and the amount of **discount to be applied** (discounts can be 0%, 5%, 10%, 15%, 20%, 30%, 40% or 50%).

4. Car Dealer Query and Export Data

Write the below described queries and **export** the returned data to the specified **format**.

Query 1 – Ordered Customers

Get all **customers** ordered by their **birthdate in ascending order**. If two customers are born on the same date, **first print those, who are not young drivers** (e.g. print experienced drivers first). **Export** the list of customers **to XML** in the format provided below.

ordered-customers.xml

```
<?xml version="1.0" encoding="utf-8"?>
<customers>
  <customer>
      <id>29</id>
```

```
      <name>Louann Holzworth</name>
      <birth-date>1960-10-01T00:00:00</birth-date>
      <is-youn-driver>false</is-young-driver>
  </customer>
  <customer>
      <id>28</id>
      <name>Donnetta Soliz</name>
      <birth-date>1963-10-01T00:00:00</birth-date>
      <is-youn-driver>false</is-young-driver>
  </customer>
  ...
</customers>
```

Query 2 – Cars from Make Toyota

Get all **cars** from make **Toyota** and **order them by model alphabetically** and by **traveled distance in descending order**. **Export** the list of **cars to XML** in the format provided below.

toyota-cars.xml

```
<?xml version="1.0" encoding="utf-8"?>
<cars>
  <car id="117" make="Toyota" model="Camry Hybrid" travelled-distance="954775807" />
  <car id="112" make="Toyota" model="Camry Hybrid" travelled-distance="92275807" />
  ...
</cars>
```

Query 3 – Local Suppliers

Get all **suppliers** that **do not import parts from abroad**. Get their **id**, **name** and **the number of parts they can offer to supply**. **Export** the list of suppliers **to XML** in the format provided below.

local-suppliers.xml

```
<?xml version="1.0" encoding="utf-8"?>
<suppliers>
  <suplier id="2" name="Agway Inc." parts-count="6" />
  <suplier id="4" name="Airgas, Inc." parts-count="5" />
  ...
</suppliers>
```

Query 4 – Cars with Their List of Parts

Get all **cars along with their list of parts**. For the **car** get only **make, model** and **traveled distance** and for the **parts** get only **name** and **price**. **Export** the list of **cars and their parts to XML** in the format provided below.

cars-and-parts.xml

```
<?xml version="1.0" encoding="utf-8"?>
<cars>
```

```
  <car make="Opel" model="Omega" travelled-distance="2147483647" />
     <parts>
         <part name="Front Left Side Outer door handle" price="999.99" />
         <part name="Gudgeon pin" price="44.99" />
         <part name="Oil pump" price="100.19" />
         <part name="Transmission pan" price="106.99" />
     </parts>
  </car>
  <car make="Opel" model="Astra" travelled-distance="9223372036854775807" />
     <parts>
         <part name="Overflow tank" price="1200.99" />
         ...
     </parts>
  </car>
  ...
</cars>
```

Query 5 – Total Sales by Customer

Get all **customers** that have bought **at least 1 car** and get their **names**, **count of cars bought** and **total money spent** on cars. **Order** the result **by total money spent in descending order** and then by **total amount of cars bought** again in **descending** order. **Export** the list of customers **to XML** in the format provided below.

customers-total-sales.xml

```
<?xml version="1.0" encoding="utf-8"?>
<customers>
  <customer full-name="Hipolito Lamoreaux" bought-cars="5" spent-money="8360.48" />
  <customer full-name="Francis Mckim" bought-cars="4" spent-money="7115.50" />
  <customer full-name="Johnette Derryberry" bought-cars="4" spent-money="5337.72" />
  ...
</customer>
```

Query 6 – Sales with Applied Discount

Get all **sales** with information about the **car**, the **customer** and the **price** of the sale **with and without discount**. **Export** the list of sales **to XML** in the format provided below.

sales-discounts.xml

```
<?xml version="1.0" encoding="utf-8"?>
<sales>
  <sale>
     <car make="Peugeot" model="405" travelled-distance="92036854775807" />
     <customer-name>Donnetta Soliz</customer-name>
     <discount>0.3</discount>
     <price>1402.53</price>
```

```
    <price-with-discount>981.771</price-with-discount>
  </sale>
  <sale>
    <car make="Mercedes" model="W124" travelled-distance="2147647" />
    <customer-name>Carri Knapik</customer-name>
    <discount>0.2</discount>
    <price>254.96999999999997</price>
    <price-with-discount>203.97599999999997</price-with-discount>
  </sale>
  ...
</sales>
```