Adatkezelés XML-ben

Féléves feladat

Zeneipari adatbázis XML-ben

Jegyzőkönyv

Készítette:

Hegedűs Attila László
Mérnök-informatikus
levelező
D2OVJ9

Miskolc, 2022

Tartalom

A feladat bemutatása

Ebben a feladatban az ezen félév során Adatkezelés XML-ben tantárgy óráin elsajátított módszerek kerülnek bemutatásra.

A feladat elkészítéséhez rendelkezni kell egy helyesen formált XML fájlal és a hozzá tartozó sémával. Ehhez elsőként el kell készíteni a megoldás ER-modelljét, melyet át kell alakítani XDM-re, majd ezek alapján létrehozni az XML-t. Az elkészült XML fájlhoz ezután Java nyelven programot kell készíteni, ami képes lesz az XML fájl beolvasására, szűrésére és módosítására, a W3C DOM könyvtárak segítségével.

A megoldásom bemutatása

A feladatomban egy adatbázist mutatok be, mely bejegyzéseiben különböző zeneiparral kapcsolatos egyedek szerepelnek. A gyökér neve Adatok, ezen belül a bejegyzések:

Egyedek és tulajdonságaik:

1. Zenész (ID, Név (Vezetéknév, Keresztnév), Születési dátum, Nem)
2. Zenekar (ID, Név, Műfaj, Alakult)
3. Kiadó (ID, Név, Telephely, Email)
4. Tulajdonos (ID, Név (Vezetéknév, Keresztnév), Telefonszám, Email)
5. Hangszer (ID, Név (Gyártó, Típus), Osztály)

Kapcsolatok:

- 1:1: Tulajdonos – tulajdona – Kiadó
- 1:N: Kiadó – alkalmazza – Zenekart
- 1:N: Zenész – tagja – Zenekarnak
- N:M: Zenész – játszik -Hangszeren, Tulajdonság: Mióta, Szint

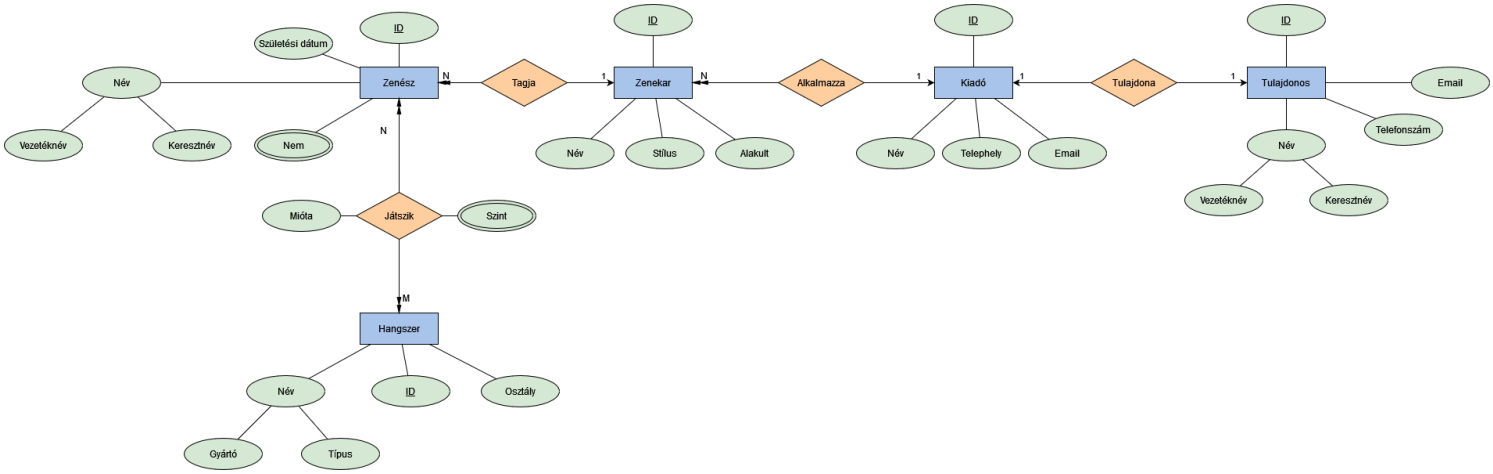
Github:

https://github.com/ati72/d2ovj9_XMLGyak/tree/main/XMLTaskD2ovj9

1. a) Feladat: ER-modell

Az ER-modellemet a draw.io internetes ábrarajzoló alkalmazás segítségével készítettem el. Az ábra elkészítése során alkalmaztam a hozzá tartozó formai megkötéseket, ezen felül színekkel is elláttam a modellt, hogy még szemléletesebb legyen.

Az egyedek világoskék négyzetekként vannak ábrázolva, a tulajdonságok világoszöld ellipszisekként. Az egyszerű tulajdonságok szimpla, az összetettek dupla körvonalakkal lettek megrajzolva. A kulcs tulajdonságok neve alá van húzva. A kapcsolatok világos narancssárga rombuszokban szerepelnek, a számosságuk nyilakkal és feliratban is fel lett tüntetve.

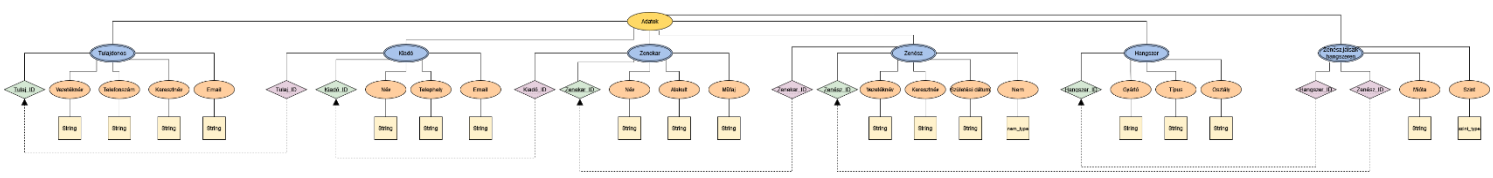


1. b) Feladat: XDM-modell

Az XDM-modellt az ER-modellből kellett átkonvertálni. Az egyedekből elemek lesznek, a tulajdonságokból gyerekelemek, a kulcsokból attribútumok stb. Az XDM-ben már feltüntetve lesznek olyan elemek is amelyek az ER-ben nem szerepelnek, ilyenek az idegen kulcsok, a gyökér elem, az elemek adattípusai.

Az ábrát az ER-hez hasonlóan a draw.io alkalmazással készítettem el. Sárga ellipszis a gyökérelem, világoskék ellipszisek az elemek, narancssárga ellipszisek a gyerekelemek, világos sárga négyzetek az adattípusok. A kulcsok világoszöld rombuszok, aláhúzott nevekkel, az idegen kulcsok lila rombuszok, szaggatott vonallal aláhúzott nevekkel. A kulcsokat és a rájuk referáló idegen kulcsokat szaggatott vonalú nyilak kapcsolják össze. Az elemek leszármazását nyilak jelölik.

Mindkét ábra megtalálható a feladathoz tartozó github repositoryban, nagyobb felbontásban.



1. c) Feladat: XML dokumentum

A modellek elkészítése után létrehoztam az általuk felvázolt XML dokumentumot.

A dokumentum létrehozásához a VSCode szövegszerkesztőt használtam. 6 többszörös előfordulást tartalmazó elemet hoztam létre a gyökérelemen belül, mindegyikből legalább 3 példány készült.

Kód

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


<adatok         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="XSDD2ovj9.xsd">
    <!-- Tulajok -->
    <tulajdonos ID="tul-01">
        <vezeteknev>Major</vezeteknev>
        <keresztnev>Anna</keresztnev>
        <telefonszam>06201234567</telefonszam>
        <email>majoranna@gmail.com</email>
    </tulajdonos>
    <tulajdonos ID="tul-02">
        <vezeteknev>Feles</vezeteknev>
        <keresztnev>Elek</keresztnev>
        <telefonszam>06309877654</telefonszam>
        <email>feles@elek.hu</email>
    </tulajdonos>
    <tulajdonos ID="tul-03">
        <vezeteknev>Ultra</vezeteknev>
        <keresztnev>Viola</keresztnev>
        <telefonszam>06709844565</telefonszam>
        <email>ultrav@gmail.com</email>
    </tulajdonos>
    <!-- Kiadók -->
    <kiado ID="kia-01" tulaj_ID="tul-01">
```

```
    <nev>Major Records</nev>
    <telephely>Eger, Széchenyi út 20</telephely>
    <email>majorrecords@gmail.com</email>
</kiado>
<kiado ID="kia-02" tulaj_ID="tul-02">
    <nev>Feles Music</nev>
    <telephely>Budapest, Rákóczi út 32</telephely>
    <email>feles_music@gmail.com</email>
</kiado>
<kiado ID="kia-03" tulaj_ID="tul-03">
    <nev>Ultra Sound</nev>
    <telephely>Székesfehérvár, Kárpát út 13</telephely>
    <email>ultrasound@hotmail.com</email>
</kiado>
<!-- Zenekarok -->
<zenekar ID="zk-01" kiado_ID='kia-01'>
    <nev>Admin</nev>
    <alakult>1999</alakult>
    <mufaj>Rock</mufaj>
</zenekar>
<zenekar ID="zk-02" kiado_ID='kia-02'>
    <nev>Quake</nev>
    <alakult>2009</alakult>
    <mufaj>Punk</mufaj>
</zenekar>
<zenekar ID="zk-03" kiado_ID='kia-03'>
    <nev>Algoritmus</nev>
    <alakult>1986</alakult>
    <mufaj>Alternatív</mufaj>
</zenekar>
```

```
<!-- Zeneszek -->
<zenesz ID="zen-01" zenekar_ID="zk-01">
    <vezeteknev>Para</vezeteknev>
    <keresztnev>Zita</keresztnev>
    <szuletett>1980</szuletett>
    <nem>N</nem>
</zenesz>
<zenesz ID="zen-02" zenekar_ID="zk-02">
    <vezeteknev>Git</vezeteknev>
    <keresztnev>Áron</keresztnev>
    <szuletett>1990</szuletett>
    <nem>F</nem>
</zenesz>
<zenesz ID="zen-03" zenekar_ID="zk-03">
    <vezeteknev>Techno</vezeteknev>
    <keresztnev>Kolos</keresztnev>
    <szuletett>1969</szuletett>
    <nem>F</nem>
</zenesz>
<!-- Hangszerek -->
<hangszer ID="hsz-01">
    <gyarto>Roland</gyarto>
    <tipus>FP-30X</tipus>
    <osztaly>Billentyűs</osztaly>
</hangszer>
<hangszer ID="hsz-02">
    <gyarto>Fender</gyarto>
    <tipus>Stratocaster</tipus>
    <osztaly>Gitár</osztaly>
</hangszer>
```

```
    <hangszer ID="hsz-03">
        <gyarto>Pearl</gyarto>
        <tipus>Roadshow</tipus>
        <osztaly>Dob</osztaly>
    </hangszer>
    <!-- Hangszerismeret -->
    <hangszerismeret hangszer_ID="hsz-01" zenesz_ID="zen-01">
        <ideje>5</ideje>
        <szint>Virtuóz</szint>
    </hangszerismeret>
    <hangszerismeret hangszer_ID="hsz-02" zenesz_ID="zen-02">
        <ideje>3</ideje>
        <szint>Alibi</szint>
    </hangszerismeret>
    <hangszerismeret hangszer_ID="hsz-03" zenesz_ID="zen-03">
        <ideje>20</ideje>
        <szint>Tapasztalt</szint>
    </hangszerismeret>
</adatok>
```

1. d) Feladat: XML Schema

Az XML elkészítését követően elkészítettem a hozzá tartozó séma fájlt. Az XML validálásához egy weboldalt alkalmaztam https://www.liquid-technologies.com/online-xsd-validator, ennek a használata során szimplán be kell másolni a két dokumentumot az adott beviteli mezőkbe és a validálás gombra kattintva megtörténik a folyamat.

Szerkezetileg a séma fájlom először leírja a gyökérelemben lévő elemeket complexType-ként, sequence-ben megadva a saját típusaimat. Ezután definiálom a kulcsokat és idegen kulcsokat. Végül a saját típusaim lettek elkészítve complexType-ként. A többértékű egyedekhez simpleType-ban készítem el a saját típusokat, ezeket a hozzájuk tartozó complexType saját típusok használják fel.

Kód

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


<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
elementFormDefault="qualified">

    <xs:element name="adatok">

        <xs:complexType>

            <xs:sequence>

                    <xs:element name="tulajdonos" type="tulaj_type"
minOccurs="1" maxOccurs="unbounded"/>

                        <xs:element name="kiado" type="kiado_type"
minOccurs="1" maxOccurs="unbounded"/>

                     <xs:element name="zenekar" type="zenekar_type"
minOccurs="1" maxOccurs="unbounded"/>

                       <xs:element name="zenesz" type="zenesz_type"
minOccurs="1" maxOccurs="unbounded"/>

                   <xs:element name="hangszer" type="hangszer_type"
minOccurs="1" maxOccurs="unbounded"/>

                                <xs:element name="hangszerismeret"
type="hangszerismeret_type" minOccurs="1" maxOccurs="unbounded"/>

            </xs:sequence>

        </xs:complexType>


        <!-- Kulcsok (Kulcs és rámutató idegen kulcs párokban)-->

        <!-- Tulaj -->
```

```
<xs:key name="tulaj_ID">
    <xs:selector xpath=".//tulajdonos" />
    <xs:field xpath="@ID" />
</xs:key>
<xs:keyref name="tulaj_ID_ref" refer="tulaj_ID">
    <xs:selector xpath=".//kiado" />
    <xs:field xpath="@tulaj_ID" />
</xs:keyref>
<!-- Kiado -->
<xs:key name="kiado_ID">
    <xs:selector xpath=".//kiado" />
    <xs:field xpath="@ID" />
</xs:key>
<xs:keyref name="kiado_ID_ref" refer="kiado_ID">
    <xs:selector xpath=".//zenekar" />
    <xs:field xpath="@kiado_ID" />
</xs:keyref>
<!-- Zenekar -->
<xs:key name="zenekar_ID">
    <xs:selector xpath=".//zenekar" />
    <xs:field xpath="@ID" />
</xs:key>
<xs:keyref name="zenekar_ID_ref" refer="zenekar_ID">
    <xs:selector xpath=".//zenesz" />
    <xs:field xpath="@zenekar_ID" />
</xs:keyref>
<!-- Zenesz -->
<xs:key name="zenesz_ID">
    <xs:selector xpath=".//zenesz" />
    <xs:field xpath="@ID" />
```

```
    </xs:key>
    <xs:keyref name="zenesz_ID_ref" refer="zenesz_ID">
        <xs:selector xpath=".//hangszerismeret" />
        <xs:field xpath="@zenesz_ID" />
    </xs:keyref>
    <!-- Hangszer -->
    <xs:key name="hangszer_ID">
        <xs:selector xpath=".//hangszer" />
        <xs:field xpath="@ID" />
    </xs:key>
    <xs:keyref name="hangszer_ID_ref" refer="hangszer_ID">
        <xs:selector xpath=".//hangszerismeret" />
        <xs:field xpath="@hangszer_ID" />
    </xs:keyref>
    <!-- 1:1 Kapcs -->
    <xs:unique name="egy_tulaj">
        <xs:selector xpath=".//kiado"></xs:selector>
        <xs:field xpath="@tulaj_ID"></xs:field>
    </xs:unique>

</xs:element>



<!-- Sajat tipusok -->
<!-- Tulaj -->
<xs:complexType name="tulaj_type">
    <xs:sequence>
        <xs:element name="vezeteknev" type="xs:string"/>
        <xs:element name="keresztnev" type="xs:string"/>
        <xs:element name="telefonszam" type="xs:string"/>
```

```
            <xs:element name="email" type="xs:string"/>
        </xs:sequence>
        <xs:attribute name="ID" type="xs:string" use="required"/>
    </xs:complexType>
    <!-- Kiado -->
    <xs:complexType name="kiado_type">
        <xs:sequence>
            <xs:element name="nev" type="xs:string"/>
            <xs:element name="telephely" type="xs:string"/>
            <xs:element name="email" type="xs:string"/>
        </xs:sequence>
        <xs:attribute name="ID" type="xs:string" use="required"/>
                <xs:attribute   name="tulaj_ID"   type="xs:string"
use="required"/>
    </xs:complexType>
    <!-- Zenekar -->
    <xs:complexType name="zenekar_type">
        <xs:sequence>
            <xs:element name="nev" type="xs:string"/>
            <xs:element name="alakult" type="xs:string"/>
            <xs:element name="mufaj" type="xs:string"/>
        </xs:sequence>
        <xs:attribute name="ID" type="xs:string" use="required"/>
                <xs:attribute   name="kiado_ID"   type="xs:string"
use="required"/>
    </xs:complexType>
    <!-- Zenesz -->
    <xs:complexType name="zenesz_type">
        <xs:sequence>
            <xs:element name="vezeteknev" type="xs:string"/>
            <xs:element name="keresztnev" type="xs:string"/>
```

```
            <xs:element name="szuletett" type="xs:string"/>
            <xs:element name="nem" type="nem_type"/>
        </xs:sequence>
        <xs:attribute name="ID" type="xs:string" use="required"/>
                <xs:attribute   name="zenekar_ID"   type="xs:string"
use="required"/>
    </xs:complexType>
    <!-- Hangszer -->
    <xs:complexType name="hangszer_type">
        <xs:sequence>
            <xs:element name="gyarto" type="xs:string"/>
            <xs:element name="tipus" type="xs:string"/>
            <xs:element name="osztaly" type="xs:string"/>
        </xs:sequence>
        <xs:attribute name="ID" type="xs:string" use="required"/>
    </xs:complexType>
    <!-- Hangszerismeret -->
    <xs:complexType name="hangszerismeret_type">
        <xs:sequence>
            <xs:element name="ideje" type="xs:string"/>
            <xs:element name="szint" type="szint_type"/>
        </xs:sequence>
                <xs:attribute   name="hangszer_ID"   type="xs:string"
use="required"/>
                 <xs:attribute   name="zenesz_ID"   type="xs:string"
use="required"/>
    </xs:complexType>
    <!-- Egyszeru tipusok a tobberteku egyedekhez -->
    <xs:simpleType name="szint_type">
        <xs:restriction base="xs:string">
            <xs:enumeration value="Virtuóz"/>
```

```
            <xs:enumeration value="Tapasztalt"/>
            <xs:enumeration value="Alibi"/>
        </xs:restriction>
    </xs:simpleType>
    <xs:simpleType name="nem_type">
        <xs:restriction base="xs:string">
            <xs:enumeration value="F"/>
            <xs:enumeration value="N"/>
        </xs:restriction>
    </xs:simpleType>
</xs:schema>
```

2. a) Feladat: Java – XML beolvasás

A fájlt beolvasó program Java nyelven íródik és felhasználja a W3C DOM könyvtárát a feladat teljesítéséhez.

A dokumentum beolvasásához szükségünk van a fájl-ra, melyet a File osztályból példányosítunk az elérési útvonalának megadásával. Példányosítanunk kell továbbá egy DocumentBuilderFactory-t a newInstance() metódusának segítségével, egy DocumentBuilder-t a DocumentBuilderFactorynk példányából a newDocumentBuilder metódussal, egy Document-et a DocumentBuilder példányunk parse metódusával, melynek paramétere az előbb megadott fájl. Ezután ezen a Document példányon végezhetjük el a beolvasást.

A beolvasáshoz egy NodeList-et hozunk létre, a getElementsByTagName() függvény segítségével. A függvény paraméterében meg kell adnunk az xml fájlban elérni kívánt elemek nevét stringként. A NodeList-be kerülnek az adott nevű node-ok, ezután a listát for ciklussal bejárva getAttribute() függvénnyel az elem adott nevű attribútumát tudjuk elérni, a getTextContent() metódussal pedig az elemek tartalmát. Ezeket változókba mentem, majd System.out.println() függvénnyel íratom ki a kimenetre. Az elemek megszámlálására egy elementCount változót implementáltam.

Kód

```
package hu.domparse.d2ovj9;

import java.io.File;
import java.io.IOException;

import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.DocumentBuilderFactory;
import javax.xml.parsers.ParserConfigurationException;

import org.w3c.dom.Document;
import org.w3c.dom.Element;
import org.w3c.dom.Node;
import org.w3c.dom.NodeList;
import org.xml.sax.SAXException;

public class DomReadD2ovj9 {

	public static void printRoot(Document doc) {
		//Gyökér kiírás
		System.out.println("Root:                " +
doc.getDocumentElement().getNodeName());

	System.out.println("=========================================
====");
	}

	public static int printOwners(Document doc, int elementCount) {
		//Tulajdonosok kiírása
		NodeList                    nListTulaj                    =
doc.getElementsByTagName("tulajdonos");
		for(int i = 0; i < nListTulaj.getLength(); i++) {
```

```
                Node nNode = nListTulaj.item(i);
                System.out.println(elementCount + ". Element: " +
nNode.getNodeName());
                if(nNode.getNodeType() == Node.ELEMENT_NODE) {
                     Element elem = (Element) nNode;
                     String uid = elem.getAttribute("ID");

                     Node                    node1                    =
elem.getElementsByTagName("vezeteknev").item(0);
                     String fname = node1.getTextContent();

                     Node                    node2                    =
elem.getElementsByTagName("keresztnev").item(0);
                     String lname = node2.getTextContent();

                     Node                    node3                    =
elem.getElementsByTagName("telefonszam").item(0);
                     String phoneNumber = node3.getTextContent();

                     Node                    node4                    =
elem.getElementsByTagName("email").item(0);
                     String email = node4.getTextContent();

                     System.out.println("ID: "+ uid);
                     System.out.println("Vezetéknév: " + fname);
                     System.out.println("Keresztnév: " + lname);
                     System.out.println("Telefonszám:      "      +
phoneNumber);
                     System.out.println("Email cím: " + email);

     System.out.println("=========================================
====");
                     elementCount++;
```

```
                }
            }
            return elementCount;
        }

        public static int printLabels(Document doc, int elementCount) {
            //Kiadók kiírása
            NodeList nListKiado = doc.getElementsByTagName("kiado");
            for(int i = 0; i < nListKiado.getLength(); i++) {
                Node nNode = nListKiado.item(i);
                System.out.println(elementCount + ". Element: " +
nNode.getNodeName());
                if(nNode.getNodeType() == Node.ELEMENT_NODE) {
                    Element elem = (Element) nNode;
                    String uid = elem.getAttribute("ID");
                    String fkeyOwner =
elem.getAttribute("tulaj_ID");

                    Node node1 =
elem.getElementsByTagName("nev").item(0);
                    String name = node1.getTextContent();

                    Node node2 =
elem.getElementsByTagName("telephely").item(0);
                    String location = node2.getTextContent();

                    Node node3 =
elem.getElementsByTagName("email").item(0);
                    String email = node3.getTextContent();

                    System.out.println("ID: "+ uid);
```

```
                    System.out.println("Tulajdonos    ID:    "    +
fkeyOwner);
                    System.out.println("Név: " + name);
                    System.out.println("Telephely: " + location);
                    System.out.println("Email cím: " + email);

     System.out.println("=========================================
====");
                    elementCount++;
               }
          }
          return elementCount;
     }

     public static int printBands(Document doc, int elementCount) {
          //Zenekarok kiírása
          NodeList                  nListZenekar                  =
doc.getElementsByTagName("zenekar");
          for(int i = 0; i < nListZenekar.getLength(); i++) {
               Node nNode = nListZenekar.item(i);
               System.out.println(elementCount  +  ".  Element:  "  +
nNode.getNodeName());
               if(nNode.getNodeType() == Node.ELEMENT_NODE) {
                    Element elem = (Element) nNode;
                    String uid = elem.getAttribute("ID");
                    String               fkeyKiado               =
elem.getAttribute("kiado_ID");

                    Node                   node1                   =
elem.getElementsByTagName("nev").item(0);
                    String name = node1.getTextContent();
```

```
                    Node                    node2                    =
elem.getElementsByTagName("alakult").item(0);
                    String est = node2.getTextContent();

                    Node                    node3                    =
elem.getElementsByTagName("mufaj").item(0);
                    String genre = node3.getTextContent();

                    System.out.println("ID: "+ uid);
                    System.out.println("Kiadó ID: " + fkeyKiado);
                    System.out.println("Név: " + name);
                    System.out.println("Alakult: " + est);
                    System.out.println("Műfaj: " + genre);

     System.out.println("=========================================
====");
                    elementCount++;
               }
          }
          return elementCount;
     }

     public static int printMusicians(Document doc, int elementCount)
{
          //Zenészek kiírása
          NodeList nListZenesz = doc.getElementsByTagName("zenesz");
          for(int i = 0; i < nListZenesz.getLength(); i++) {
               Node nNode = nListZenesz.item(i);
               System.out.println(elementCount  +  ".  Element:  "  +
nNode.getNodeName());
               if(nNode.getNodeType() == Node.ELEMENT_NODE) {
                    Element elem = (Element) nNode;
```

```
                    String uid = elem.getAttribute("ID");
                    String               fkeyZenekar               =
elem.getAttribute("zenekar_ID");

                    Node                   node1                   =
elem.getElementsByTagName("vezeteknev").item(0);
                    String fname = node1.getTextContent();

                    Node                   node2                   =
elem.getElementsByTagName("keresztnev").item(0);
                    String lname = node2.getTextContent();

                    Node                   node3                   =
elem.getElementsByTagName("szuletett").item(0);
                    String born = node3.getTextContent();

                    Node                   node4                   =
elem.getElementsByTagName("nem").item(0);
                    String gender = node4.getTextContent();

                    System.out.println("ID: "+ uid);
                    System.out.println("Zenekar     ID:     "     +
fkeyZenekar);
                    System.out.println("Vezetéknév: " + fname);
                    System.out.println("Keresztnév: " + lname);
                    System.out.println("Született: " + born);
                    System.out.println("Nem: " + gender);

     System.out.println("==========================================
====");
                    elementCount++;
                }
           }
```

```
            return elementCount;
      }

      public static int printInstruments(Document doc, int
elementCount) {
            //Hangszerek kiírása
            NodeList nListHangszer =
doc.getElementsByTagName("hangszer");
            for(int i = 0; i < nListHangszer.getLength(); i++) {
                  Node nNode = nListHangszer.item(i);
                  System.out.println(elementCount + ". Element: " +
nNode.getNodeName());
                  if(nNode.getNodeType() == Node.ELEMENT_NODE) {
                        Element elem = (Element) nNode;
                        String uid = elem.getAttribute("ID");

                        Node node1 =
elem.getElementsByTagName("gyarto").item(0);
                        String manufacturer = node1.getTextContent();

                        Node node2 =
elem.getElementsByTagName("tipus").item(0);
                        String type = node2.getTextContent();

                        Node node3 =
elem.getElementsByTagName("osztaly").item(0);
                        String instrumentClass = node3.getTextContent();

                        System.out.println("ID: "+ uid);
                        System.out.println("Gyártó: " + manufacturer);
                        System.out.println("Típus: " + type);
                        System.out.println("Osztály:        "        +
instrumentClass);
```

```
     System.out.println("========================================
====");
                        elementCount++;
                  }
            }
            return elementCount;
      }


      public static int printInstrumentKnowledge(Document doc, int
elementCount) {
            //Hangszerismeretek kiírása
            NodeList              nListHangszerismeret              =
doc.getElementsByTagName("hangszerismeret");
            for(int i = 0; i < nListHangszerismeret.getLength(); i++)
{
                  Node nNode = nListHangszerismeret.item(i);
                  System.out.println(elementCount + ". Element: " +
nNode.getNodeName());
                  if(nNode.getNodeType() == Node.ELEMENT_NODE) {
                        Element elem = (Element) nNode;
                        String              instrumentId              =
elem.getAttribute("hangszer_ID");
                        String              fkeyMusician              =
elem.getAttribute("zenesz_ID");


                        Node                    node1                 =
elem.getElementsByTagName("ideje").item(0);
                        String since = node1.getTextContent();
                        Node                    node2                 =
elem.getElementsByTagName("szint").item(0);
                        String level = node2.getTextContent();
```

```
                    System.out.println("Hangszer       ID:       "+
instrumentId);
                    System.out.println("Zenész     ID:     "     +
fkeyMusician);
                    System.out.println("Tapasztalat: " + since + "
év");
                    System.out.println("Szint: " + level);

     System.out.println("==========================================
====");
                    elementCount++;
               }
          }
          return elementCount;
     }

     public static void printXml(Document doc, int elementCount) {
          printRoot(doc);
          elementCount = printOwners(doc, elementCount);
          elementCount = printLabels(doc, elementCount);
          elementCount = printBands(doc, elementCount);
          elementCount = printMusicians(doc, elementCount);
          elementCount = printInstruments(doc, elementCount);
          elementCount        =        printInstrumentKnowledge(doc,
elementCount);
     }

     public     static     Document     createDocument()     throws
ParserConfigurationException, SAXException, IOException {
          //Fájl beolvasás
          File xmlData = new File("XMLD2ovj9.xml");
          DocumentBuilderFactory                dbf                =
DocumentBuilderFactory.newInstance();
```

```
            DocumentBuilder builder = dbf.newDocumentBuilder();
            Document doc = builder.parse(xmlData);
            doc.getDocumentElement().normalize();

            return doc;
      }

      public static void main(String[] args) {

            try {

                  Document doc = createDocument();
                  int elementCount = 1;
                  printXml(doc, elementCount);

            } catch (Exception e) {
                  System.out.println(e);
            }
      }
}
```

2. b) Feladat: Java – XML szűrés

A szűréshez az előbbi feladat módszerét használtam fel, pluszban hozzáadva egy bemeneti prompt-ot mely opcióként adja, hogy melyik elemekre szeretnénk szűrni.

Az opció megadása után a megfelelő kiíratő metódusok kerülnek meghívásra. Hozzáadásra került még két opció, melyek XPath segítségével szűrnek, tulajdonos ID-je alapján megkapjuk az email-címét, illetve zenekar nevét megadva megkapjuk a műfaját. Az opciók kiválasztásához ellenőrzött bemeneti függvényt készítettem.

Kód

```
package hu.domparse.d2ovj9;


import java.io.File;
import java.io.IOException;


import java.util.Scanner;


import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.DocumentBuilderFactory;
import javax.xml.parsers.ParserConfigurationException;
import javax.xml.xpath.XPath;
import javax.xml.xpath.XPathConstants;
import javax.xml.xpath.XPathExpression;
import javax.xml.xpath.XPathExpressionException;
import javax.xml.xpath.XPathFactory;


import org.w3c.dom.Document;
import org.w3c.dom.Element;
import org.w3c.dom.Node;
import org.w3c.dom.NodeList;
import org.xml.sax.SAXException;
```

```
public class DOMQueryD2ovj9 {


     public static void printOwners(Document doc, int elementCount) {
          NodeList                    nListTulaj                    =
doc.getElementsByTagName("tulajdonos");
          for(int i = 0; i < nListTulaj.getLength(); i++) {
               Node nNode = nListTulaj.item(i);
               System.out.println(elementCount  +  ".  Element:  "  +
nNode.getNodeName());
               if(nNode.getNodeType() == Node.ELEMENT_NODE) {
                    Element elem = (Element) nNode;
                    String uid = elem.getAttribute("ID");


                    Node                    node1                    =
elem.getElementsByTagName("vezeteknev").item(0);
                    String fname = node1.getTextContent();


                    Node                    node2                    =
elem.getElementsByTagName("keresztnev").item(0);
                    String lname = node2.getTextContent();


                    Node                    node3                    =
elem.getElementsByTagName("telefonszam").item(0);
                    String phoneNumber = node3.getTextContent();


                    Node                    node4                    =
elem.getElementsByTagName("email").item(0);
                    String email = node4.getTextContent();


                    System.out.println("ID: "+ uid);
                    System.out.println("Vezetéknév: " + fname);
                    System.out.println("Keresztnév: " + lname);
```

```
                    System.out.println("Telefonszám:       "       +
phoneNumber);
                    System.out.println("Email cím: " + email);

     System.out.println("=========================================
====");
                    elementCount++;
               }
          }
     }

     public static void printLabels(Document doc, int elementCount) {
          NodeList nListKiado = doc.getElementsByTagName("kiado");
          for(int i = 0; i < nListKiado.getLength(); i++) {
               Node nNode = nListKiado.item(i);
               System.out.println(elementCount  +  ".  Element:  "  +
nNode.getNodeName());
               if(nNode.getNodeType() == Node.ELEMENT_NODE) {
                    Element elem = (Element) nNode;
                    String uid = elem.getAttribute("ID");
                    String                fkeyOwner                =
elem.getAttribute("tulaj_ID");

                    Node                    node1                    =
elem.getElementsByTagName("nev").item(0);
                    String name = node1.getTextContent();

                    Node                    node2                    =
elem.getElementsByTagName("telephely").item(0);
                    String location = node2.getTextContent();

                    Node                    node3                    =
elem.getElementsByTagName("email").item(0);
```

```
                    String email = node3.getTextContent();


                    System.out.println("ID: "+ uid);
                    System.out.println("Tulajdonos    ID:    "    +
fkeyOwner);
                    System.out.println("Név: " + name);
                    System.out.println("Telephely: " + location);
                    System.out.println("Email cím: " + email);

     System.out.println("=========================================
====");
                    elementCount++;
               }
          }


     }


     public    static    void    printMusicians(Document    doc,    int
elementCount) {
          NodeList nListZenesz = doc.getElementsByTagName("zenesz");
          for(int i = 0; i < nListZenesz.getLength(); i++) {
               Node nNode = nListZenesz.item(i);
               System.out.println(elementCount  +  ".  Element:  "  +
nNode.getNodeName());
               if(nNode.getNodeType() == Node.ELEMENT_NODE) {
                    Element elem = (Element) nNode;
                    String uid = elem.getAttribute("ID");
                    String                fkeyBand                =
elem.getAttribute("zenekar_ID");


                    Node                  node1                   =
elem.getElementsByTagName("vezeteknev").item(0);
                    String fname = node1.getTextContent();
```

```
                    Node                    node2                   =
elem.getElementsByTagName("keresztnev").item(0);
                    String lname = node2.getTextContent();

                    Node                    node3                   =
elem.getElementsByTagName("szuletett").item(0);
                    String born = node3.getTextContent();

                    Node                    node4                   =
elem.getElementsByTagName("nem").item(0);
                    String gender = node4.getTextContent();

                    System.out.println("ID: "+ uid);
                    System.out.println("Zenekar ID: " + fkeyBand);
                    System.out.println("Vezetéknév: " + fname);
                    System.out.println("Keresztnév: " + lname);
                    System.out.println("Született: " + born);
                    System.out.println("Nem: " + gender);

     System.out.println("=========================================
====");
                    elementCount++;
                }
          }
     }

     public static void getEmail(Document doc ,String id) throws
XPathExpressionException {
          //XPath keresés
                    XPathFactory           xpathFactory           =
XPathFactory.newInstance();
                    XPath xpath = xpathFactory.newXPath();
```

```
                    XPathExpression expr = xpath.compile("//tulajdonos[@ID='"+ id +"']/email/text()");
                    Object result = expr.evaluate(doc, XPathConstants.NODESET);
                    NodeList nodes = (NodeList) result;
                    for (int i = 0; i < nodes.getLength(); i++) {

System.out.println(nodes.item(i).getNodeValue());
                    }


    }

    public static void getGenre(Document doc, String band) throws XPathExpressionException {
          XPathFactory xpathFactory = XPathFactory.newInstance();
          XPath xpath = xpathFactory.newXPath();
          XPathExpression expr = xpath.compile("//zenekar[nev='"+ band +"']/mufaj/text()");
          Object result = expr.evaluate(doc, XPathConstants.NODESET);
         NodeList nodes = (NodeList) result;
         for (int i = 0; i < nodes.getLength(); i++) {
           System.out.println(nodes.item(i).getNodeValue());
         }
    }

    public static String getQueryOption(Scanner scan) {
          System.out.println("Szűrés a következőre: \n1.Tulaj\n2.Kiadó\n3.Zenész\n4.Tulajdonos e-mail címe\n5.Zenekar műfaja\n0.Kilépés");
          String option =  Integer.toString(readInt(scan, 0, 5));
          return option;
    }
```

```
	public static String getQueryID(Scanner scan) {
		System.out.println("Adja meg a tulajdonos ID-jét");
		scan.nextLine();//A readInt nextInt-je miatt kell!
		String id = scan.nextLine();
		return id;
	}

	public static String getBandName(Scanner scan) {
		System.out.println("Adja meg a zenekar nevét");
		scan.nextLine();//A readInt nextInt-je miatt kell!
		String id = scan.nextLine();
		return id;
	}

	public static Document createDocument() throws
ParserConfigurationException, SAXException, IOException {
		//Fájl beolvasás
		File xmlData = new File("XMLD2ovj9.xml");
		DocumentBuilderFactory dbf =
DocumentBuilderFactory.newInstance();
		DocumentBuilder builder = dbf.newDocumentBuilder();
		Document doc = builder.parse(xmlData);
		doc.getDocumentElement().normalize();

		return doc;
	}

	public static int readInt(Scanner scan, int minLimit, int
maxLimit) {

		int number;
```

```
            do {
                  System.out.println("Adj meg egy opciót " + minLimit +
" és " + maxLimit + " között!");
                while (!scan.hasNextInt()) {
                    System.out.println("Ilyen opció nincs! Próbálkozz
újra!");
                    scan.next();
                }
                number = scan.nextInt();
            } while (number < minLimit || number > maxLimit);

            return number;
      }

      public static void main(String[] args) {
            try {

            Document doc = createDocument();
            int elementCount = 1;

            //Szűrési opció beolvasás
            Scanner scan = new Scanner(System.in);
            String option = getQueryOption(scan);

            if(option.equals("1")) {
                  printOwners(doc, elementCount);
            } else if(option.equals("2")) {
                  printLabels(doc, elementCount);
            } else if(option.equals("3")) {
                  printMusicians(doc, elementCount);
            } else if(option.equals("4")) {
```

```
                String id = getQueryID(scan);
                getEmail(doc, id);
            } else if(option.equals("5")) {
                String band = getBandName(scan);
                getGenre(doc, band);
            } else if (option.equals("0")) {
                System.exit(1);
                scan.close();
            }
            scan.close();
        } catch (Exception e){
            System.out.println(e);
        }
}}
```

2. c) Feladat: Java – XML módosítás

Az XML módosítására olyan függvényeket implementáltam, melyek új tulajdonost hoznak létre, letörölnek egy tulajdonos elemet, illetve módosítanak egy létező tulajdonos elemet. Készítettem egy XMLD2ovj9_edit.xml nevű fájlt, melyben könnyen észrevehető ID-vel rendelkező bejegyzéseket tettem, ezt fogom felülírni, a felülírt dokumentumot modositott_xml.xml fájlba mentem.

Új tulajdonos elem hozzáadásához a createElement() metódust használtam, ugyanígy készült két gyermek ehhez az elemhez, keresztnév és vezetéknév megadására. Az elemeket a setTextContent() függvénnyel tölthetjük meg tartalommal. Ezután a gyerekelemeket fel kell fűznünk a szülő elemekre, először a tulajdonosra azon gyermekeit, majd a tulajdonost a gyökér elemre. A tulajdonoshoz a setAttribute() függvénnyel rendeltem azonosítót.

Tulajdonos törléséhez NodeList-be gyűjtöttem a tulajdonos elemeket, majd ezen végigiterálva kerestem ki a törlendő elem ID-jét. Ha a függvény megtalálta az adott id-vel rendelkező elemet a getParentNode() függvénnyel megkeresi a szülő elemét és a removeChild() függvénnyel törli ki azt.

Tulajdonos nevének módosításához szintén bejárok egy NodeListet egy megadott ID-t keresve, majd ha ezt megtaláltam a módosítandó elemeket a getElementsByTagName() függvény segítségével eltárolom változókban. A változókra a setTextContent() metódus meghívásával állítom át az elemek tartalmát.

A módosított XML dokumentumot ezután egy új XML fájlba írom ki. Ehhez a Java TransformerFactory-jét implementálom, létrehozva egy Transformert. A DOMSource-t példányosítva megadom az XML DOM-et. A StreamResultot felhasználva megadom a fájlt, amelybe írni szeretnék, ezután a Transformer példányom hívja a transform() függvényt, melynek paraméterében megadom az XML DOM-et és az új fájlt melybe azt beleírja.

Az editOwnerName() függvény az „edit-me” ID-vel rendelkező bejegyzésben módosítja a vezeték és keresztnevet.

A deleteOwner() függvény kitörli a „delete-me” ID-jű tulajdonost.

Az addOwner() függvény új tulajdonost ír a dokumentumba.

Az addBand() függvény új zenekart ír a dokumentumba.

Az addInstrument() függvény új hangszer ír a dokumentumba.

Kód

```
package hu.domparse.d2ovj9;


import java.io.File;

import java.io.IOException;


import javax.xml.parsers.DocumentBuilder;

import javax.xml.parsers.DocumentBuilderFactory;

import javax.xml.parsers.ParserConfigurationException;
```

```
import javax.xml.transform.OutputKeys;
import javax.xml.transform.Transformer;
import javax.xml.transform.TransformerConfigurationException;
import javax.xml.transform.TransformerException;
import javax.xml.transform.TransformerFactory;
import javax.xml.transform.dom.DOMSource;
import javax.xml.transform.stream.StreamResult;

import org.w3c.dom.Document;
import org.w3c.dom.Element;
import org.w3c.dom.Node;
import org.w3c.dom.NodeList;
import org.xml.sax.SAXException;

public class DomModifyD2ovj9 {

     public static Document createDocument() throws
ParserConfigurationException, SAXException, IOException {
          //Fájl beolvasás
          File xmlData = new File("XMLD2ovj9_edit.xml");
          DocumentBuilderFactory dbf =
DocumentBuilderFactory.newInstance();
          DocumentBuilder builder = dbf.newDocumentBuilder();
          Document doc = builder.parse(xmlData);
          doc.getDocumentElement().normalize();

          return doc;
     }

     public static void executeEdit(Document doc) throws
TransformerException, TransformerConfigurationException {
```

```
          //XML átírás
        TransformerFactory            transformerFactory            =
TransformerFactory.newInstance();
        Transformer                   transformer                   =
transformerFactory.newTransformer();
        DOMSource source = new DOMSource(doc);
        StreamResult     newFile     =     new     StreamResult(new
File("modositott_xml.xml"));
          transformer.setOutputProperty(OutputKeys.INDENT, "yes");
        transformer.transform(source, newFile);
     }

     public   static   void   editOwnerName(Document   doc)   throws
TransformerConfigurationException, TransformerException {
          //Tulaj vezetéknév átírás
          NodeList                    nodeList                    =
doc.getElementsByTagName("tulajdonos");
          for(int i = 0; i < nodeList.getLength(); i++) {
               Node nNode = nodeList.item(i);
               Element element = (Element) nNode;
               if(nNode.getNodeType() == Node.ELEMENT_NODE) {
                    if(element.getAttribute("ID").equals("edit-
me")) {
                         Node                fName                =
element.getElementsByTagName("vezeteknev").item(0);
                         Node                lName                =
element.getElementsByTagName("keresztnev").item(0);
                         fName.setTextContent("Hegedus");
                         lName.setTextContent("Attila");
                    }
               }
          }
          doc.normalize();
```

```
	}

	public static void deleteOwner(Document doc) {
		//Tulaj törlése
		NodeList nodeList = doc.getElementsByTagName("tulajdonos");
		 for(int i = 0; i < nodeList.getLength(); i++) {
			Node nNode = nodeList.item(i);
			Element element = (Element) nNode;
			if(nNode.getNodeType() == Node.ELEMENT_NODE) {
				if(element.getAttribute("ID").equals("delete-me")) {
element.getParentNode().removeChild(element);
				}
			}
		 }
	}

	public static void addOwner(Document doc) {
		//Tulaj hozzáadása
		Element root = doc.getDocumentElement();
		Element newOwner = doc.createElement("tulajdonos");
		newOwner.setAttribute("ID", "UJ-TULAJ");
		Element fName = doc.createElement("vezeteknev");
		fName.setTextContent("Miklós");
		Element lName = doc.createElement("keresztnev");
		lName.setTextContent("Béla");
		newOwner.appendChild(fName);
		newOwner.appendChild(lName);
		root.appendChild(newOwner);
```

```
}

public static void addBand(Document doc) {
	//Tulaj hozzáadása
	Element root = doc.getDocumentElement();
	Element newBand = doc.createElement("zenekar");
	newBand.setAttribute("ID", "UJ-ZENEKAR");
	newBand.setAttribute("kiado_ID", "kia-01");
	Element bName = doc.createElement("nev");
	bName.setTextContent("LGT");
	Element founded = doc.createElement("alakult");
	founded.setTextContent("1971");
	Element genre = doc.createElement("mufaj");
	genre.setTextContent("Rock");
	newBand.appendChild(bName);
	newBand.appendChild(founded);
	newBand.appendChild(genre);
	root.appendChild(newBand);
}

public static void addInstrument(Document doc) {
	//Tulaj hozzáadása
	Element root = doc.getDocumentElement();
	Element newInstrument = doc.createElement("hangszer");
	newInstrument.setAttribute("ID", "UJ-HANGSZER");
	Element bName = doc.createElement("gyarto");
	bName.setTextContent("Fender");
	Element type = doc.createElement("tipus");
	type.setTextContent("Telecaster");
	Element className = doc.createElement("osztaly");
```

```
            className.setTextContent("Rock");
            newInstrument.appendChild(bName);
            newInstrument.appendChild(type);
            newInstrument.appendChild(className);
            root.appendChild(newInstrument);
      }

      public static void main(String[] args) {

            try {

                  Document doc = createDocument();
                  editOwnerName(doc);
                  deleteOwner(doc);
                  addOwner(doc);
                  addBand(doc);
                  addInstrument(doc);
                  executeEdit(doc);

            } catch (Exception e) {
                  e.printStackTrace();
            }
      }
}
```