JEGYZŐKÖNYV

Adatkezelés XML környezetben

Féléves feladat:

EMAG felépítése

Készítette: **Sándor Máté**
Neptunkód: **AQUSP7**
Dátum: **2022. 11. 30.**

Tartalomjegyzék

1. Feladat 3
1.a Az adatbázis ER modell 3
1.b Az adatbázis konvertálása XDM modellre 4
1.c Az XDM modell alapján XML dokumentum készítése 4
1.d Az XML dokumentum alapján XMLSchema készítése 9
2. Feladat 12
2.a Adatolvasás, kiírás konzolra – fájlba 12
2.b Adatmódosítás 15
2.c Adatlekérdezés 19

1. Feladat

Egy webáruház, ebben az esetben az EMAG felépítésének XML környezetben való megvalósítása. A szerkezet főbb komponensei az egyedek, gyerekelemei az ezekhez kapcsolódó tulajdonságok, a kapcsolatok pedig az alkotórészek közötti összefüggések. Gyökér elemként az „Emag” elem szolgál. Ennek gyerekelemei a többi elemek (cim, gyarto, termek, tartalmazza, vevo, nev). A „tartalmazza” elem a N:M kapcsolat tulajdonsága alapján lett létrehozva, egy „darabszam” gyerekelemmel. A többértékű tulajdonságok legalább 3 elemmel rendelkeznek a dokumentumban.

1.a Az adatbázis ER modell

Az **összérték attribútum** származtatott, mivel a darabszámból és az egységárból számolható.

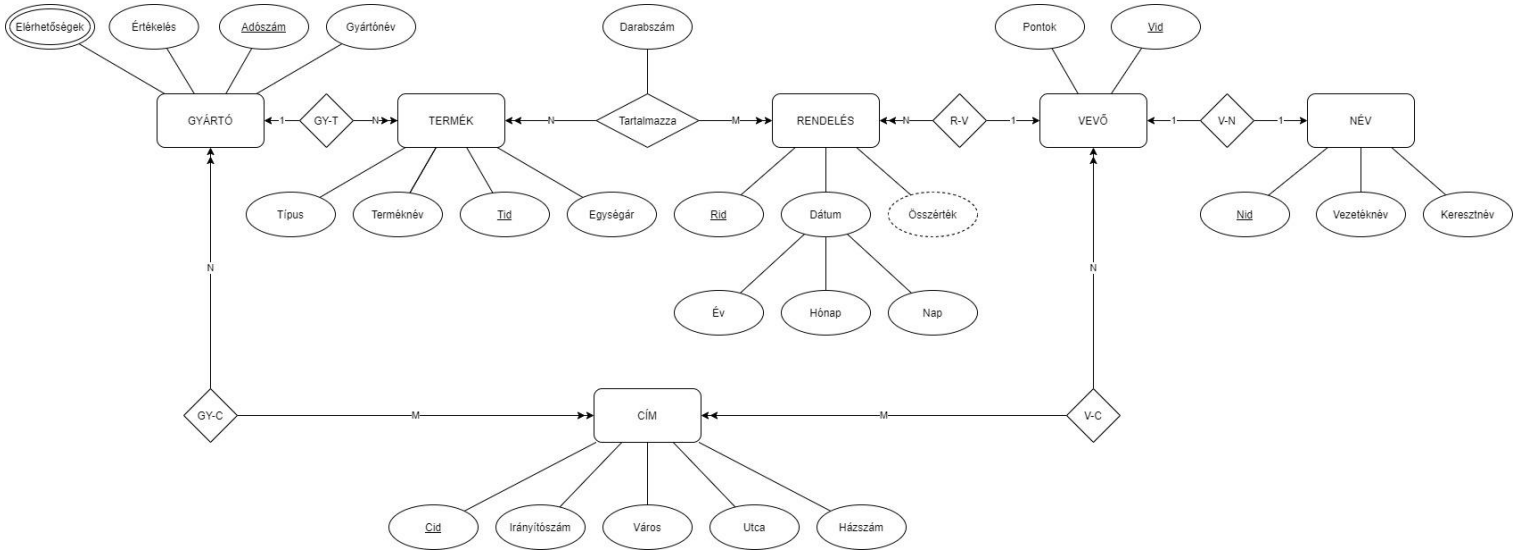
Cím kapcsolatok: Egy címhez tartozhat több vevő és több gyártó is, valamint egy gyártónak vagy vevőnek több címe is lehet.

Tartalmazza kapcsolat: Egy rendelés több terméket is tartalmazhat, és egy termék több rendelésben is lehet. ezért N-M kapcsolat jön létre amihez létrehoztam a Tartalmazza elemet.

GY-T kapcsolat: Egy gyártóhoz több termék is tartozhat, de egy terméket csak egy gyártó szabadalmaztat.

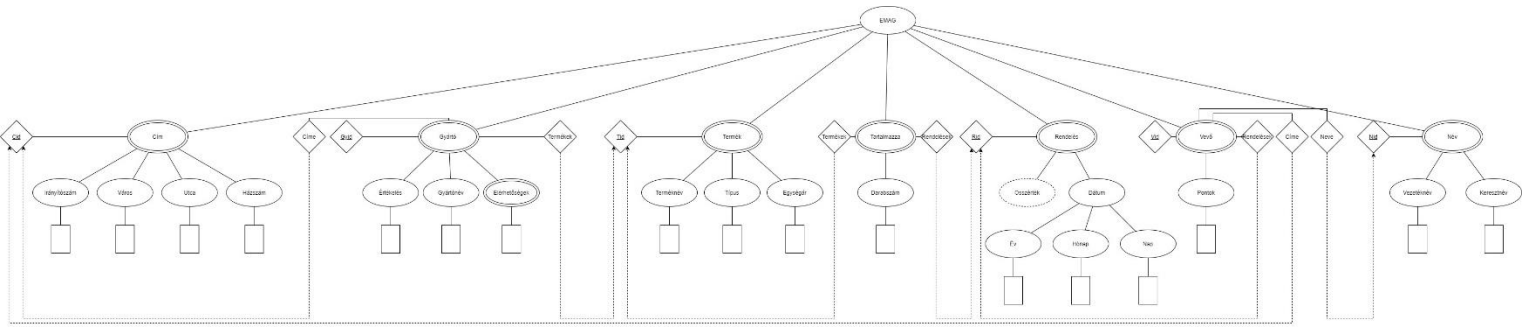
R-V kapcsolat: Egy vevőnek több rendelése lehet, de egy rendeléshez csakis egy vevő tartozhat.

V-N kapcsolat: Egy vevőnek csak egy neve lehet (vezetéknév, keresztnév)



1.b Az adatbázis konvertálása XDM modellre

XML-ről XDM-re konvertáláskor figyelembe vettem hogy a kulcsokat rombuszokba kell átírni. Létrehoztam a csatoláshoz a megfelelő idegen kulcsokat és azt a kulcsokhoz hasonlóan mellé illesztettem rombuszokba. Ebbe a fajta ábrába a gyökér elemet is meg kell jelenteni, ezt az ábra tetejére tettem. Az egyedetket dupla hurok jelöli a szintaktikának megfelelően. Az összes attribútum alá egy téglalap kerül. A kulcsokat és az idegen kulcsokat szaggatott vonallal a egfelelő irányba összekötöttem.



1.c Az XDM modell alapján XML dokumentum készítése

A fölötti ábra alapján létrehoztam egy XMLAQUSP7.xml fájlt. Mindegyik egyedhez csináltam legalább 3 példát. Az egyértelműség kedvéért kommentáltam minden egyed „tömb” előtt. Például a „Cím”-ek elé hogy <-- Címek -->. Az ábra alapján kódoltam bele az idegen kulcsokat is. Az idegen kulcsok és a kulcsok értelmezése a következő:

Cid → Cim identification → Cim kulcs

FGyCid → Foreign Gyarto Cim identification → Idegen Gyarto Cím kulcs

A kulcsoknál a bennük tárolt adat is tükrözi a származását. Például az első Cím Cid-je „c1” a másodiké „c2”.

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

<Emag xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="XMLSchemaAQUSP7.xsd">

    <!-- Cimek-->
    <Cim Cid="c1">
        <iranyitoszam>3720</iranyitoszam>
        <varos>Sajokaza</varos>
        <utca>Majus 1 ut</utca>
        <hazszam>11</hazszam>
    </Cim>

    <Cim Cid="c2">
        <iranyitoszam>3720</iranyitoszam>
```

```
    <varos>Sajoivanka</varos>
    <utca>Petofi Sandor</utca>
    <hazszam>2</hazszam>
</Cim>

<Cim Cid="c3">
    <iranyitoszam>3700</iranyitoszam>
    <varos>Kazincbarcika</varos>
    <utca>Majus 1 ut</utca>
    <hazszam>3</hazszam>
</Cim>

<Cim Cid="c4">
    <iranyitoszam>3530</iranyitoszam>
    <varos>Miskolc</varos>
    <utca>Szent istvan</utca>
    <hazszam>47</hazszam>
</Cim>

<Cim Cid="c5">
    <iranyitoszam>3530</iranyitoszam>
    <varos>Miskolc</varos>
    <utca>Joyson utca</utca>
    <hazszam>1</hazszam>
</Cim>

<!-- Gyartok-->
<Gyarto Gyid="gy1" FGyTid="t1" FGyCid="c4">
    <ertekeles>5</ertekeles>
    <gyartonev>Makita</gyartonev>
    <elerhetosegek>Makita.eu</elerhetosegek>
</Gyarto>

<Gyarto Gyid="gy2" FGyTid="t2" FGyCid="c5">
    <ertekeles>2</ertekeles>
    <gyartonev>LaptopShop</gyartonev>
    <elerhetosegek></elerhetosegek>
</Gyarto>
```

```
<Gyarto Gyid="gy3" FGyTid="t3" FGyCid="c2">
    <ertekeles>4</ertekeles>
    <gyartonev>Ikea</gyartonev>
    <elerhetosegek>Ikea.com</elerhetosegek>
</Gyarto>

<!-- Termekek-->
<Termek Tid="t1">
    <termeknev>Elektromos funyiro</termeknev>
    <tipus>Elektronika</tipus>
    <egysegar>70000</egysegar>
</Termek>

<Termek Tid="t2">
    <termeknev>Thinkpad T450</termeknev>
    <tipus>Elektronika</tipus>
    <egysegar>200000</egysegar>
</Termek>

<Termek Tid="t3">
    <termeknev>Skorgjorn parna</termeknev>
    <tipus>Butor</tipus>
    <egysegar>8000</egysegar>
</Termek>

<!-- Tartalmazza-->
<Tartalmazza Taid="t1" FTaTid="t1" FTaRid="r1">
    <darab>1</darab>
</Tartalmazza>

<Tartalmazza Taid="t2" FTaTid="t2" FTaRid="r2">
    <darab>2</darab>
</Tartalmazza>

<Tartalmazza Taid="t3" FTaTid="t3" FTaRid="r3">
    <darab>1</darab>
</Tartalmazza>
```

```
<!-- Rendelesek-->
<Rendeles Rid="r1">
    <osszertek>270000</osszertek>
    <datum>
        <ev>2022</ev>
        <honap>11</honap>
        <nap>22</nap>
    </datum>
</Rendeles>

<Rendeles Rid="r2">
    <osszertek></osszertek>
    <datum>
        <ev>2022</ev>
        <honap>10</honap>
        <nap>5</nap>
    </datum>
</Rendeles>

<Rendeles Rid="r3">
    <osszertek></osszertek>
    <datum>
        <ev>2021</ev>
        <honap>12</honap>
        <nap>21</nap>
    </datum>
</Rendeles>

<!-- Vevok-->
<Vevo Vid="v1" FVeNid="n1" FVeCid="c1" FVeRid="r1">
                <felnott>Igen</felnott>
    <pontok>110</pontok>
</Vevo>

<Vevo Vid="v2" FVeNid="n2" FVeCid="c2" FVeRid="r2">
                <felnott>Nem</felnott>
    <pontok>0</pontok>
```

```
    </Vevo>

    <Vevo Vid="v3" FVeNid ="n3" FVeCid="c3" FVeRid="r3">
                    <felnott>Igen</felnott>
        <pontok>10</pontok>
    </Vevo>

    <!-- Nevek-->
    <Nev Nid="n1">
        <vezeteknev>Sandor</vezeteknev>
        <keresztnev>Mate</keresztnev>
    </Nev>

    <Nev Nid="n2">
        <vezeteknev>Joska</vezeteknev>
        <keresztnev>Gyurka</keresztnev>
    </Nev>

    <Nev Nid="n3">
        <vezeteknev>Gabor</vezeteknev>
        <keresztnev>Aron</keresztnev>
    </Nev>


</Emag>
```

1.d Az XML dokumentum alapján XMLSchema készítése

Létrehoztam az XSDAQUSP7.xsd fájlt, majd összecsatoltam a .xml fájlommal. Ez az elöző kódsor elején látható. Felépítésnek a következőt választottam:

- Saját típus
- Gyökérelem és annak fő elemei
- Kulcsok
- Idegen kulcsok
- Unique
- Végül a fő elemek gyermekeinek definiálása

A Saját típusnak a egy szimpla felnőtt e típust hoztam létre amelynek a két lehetséges válasza az „Igen” és a „Nem”. A kulcsokat és az idegen kulcsokat a W3 nak megfelelő szemantikával oldottam meg.

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

        <xs:simpleType name="felnottE">
      <xs:restriction base="xs:token">
         <xs:enumeration value="Igen" />
         <xs:enumeration value="Nem" />
      </xs:restriction>
   </xs:simpleType>

        <!-- Felépítés-->

   <xs:element name="Emag">

                <!-- Fő elemek-->

      <xs:complexType>
         <xs:sequence>
            <xs:element maxOccurs="unbounded" name="Cim"/>
            <xs:element maxOccurs="unbounded" name="Gyarto"/>
            <xs:element maxOccurs="unbounded" name="Termek"/>
            <xs:element maxOccurs="unbounded" name="Tartalmazza"/>
            <xs:element maxOccurs="unbounded" name="Rendeles"/>
            <xs:element maxOccurs="unbounded" name="Vevo"/>
            <xs:element maxOccurs="unbounded" name="Nev"/>
         </xs:sequence>
      </xs:complexType>

                <!-- Kulcsok-->

                <xs:key name="cimKulcs">
         <xs:selector xpath="Cim" />
         <xs:field xpath="@Cid" />
      </xs:key>

      <xs:key name="gyartoKulcs">
         <xs:selector xpath="Gyarto" />
         <xs:field xpath="@Gyid" />
      </xs:key>
```

```
<xs:key name="termekKulcs">
   <xs:selector xpath="Termek" />
   <xs:field xpath="@nev" />
</xs:key>

<xs:key name="tartalmazzaKulcs">
   <xs:selector xpath="Tartalmazza" />
   <xs:field xpath="@Taid" />
</xs:key>

<xs:key name="vevoKulcs">
   <xs:selector xpath="Vevo" />
   <xs:field xpath="@Vid" />
</xs:key>

<xs:key name="nevKulcs">
   <xs:selector xpath="Nev" />
   <xs:field xpath="@Nid" />
</xs:key>

<!-- Idegen kulcsok -->

<xs:keyref refer="cimKulcs" name="cimVevoIdegenKulcs">
   <xs:selector xpath="Vevo" />
   <xs:field xpath="@FVeCid" />
</xs:keyref>

            <xs:keyref refer="cimKulcs" name="cimTartalmazzaIdegenKulcs">
   <xs:selector xpath="Tartalmazza" />
   <xs:field xpath="@FTaCid" />
</xs:keyref>

<xs:keyref refer="termekKulcs" name="termekGyartoIdegenKulcs">
   <xs:selector xpath="Gyarto" />
   <xs:field xpath="@FGyTid" />
</xs:keyref>

            <xs:keyref refer="termekKulcs" name="termekTartalmazzaIdegenKulcs">
   <xs:selector xpath="Tartalmazza" />
   <xs:field xpath="@FTaTid" />
</xs:keyref>

<xs:keyref refer="rendelesKulcs" name="rendelesTartalmazzaIdegenKulcs">
   <xs:selector xpath="Tartalmazza" />
   <xs:field xpath="@FTaRid" />
</xs:keyref>

            <xs:keyref refer="rendelesKulcs" name="rendelesVevoIdegenKulcs">
   <xs:selector xpath="Vevo" />
   <xs:field xpath="@FVeRid" />
</xs:keyref>

            <!-- Unique -->
            <xs:unique name="nevVevoIdegenKulcs">
                  <xs:selector xpath="Nev"/>
                  <xs:field xpath="@FVeNid"/>
```

```
            </xs:unique>

        </xs:element>

        <!-- További elemek-->
        <xs:element name="Cim">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="iranyitoszam" type="xs:integer" />
            <xs:element name="varos" type="xs:string" />
            <xs:element name="utca" type="xs:string" />
                                <xs:element name="hazszam" type="xs:integer" />
        </xs:sequence>
    </xs:complexType>
</xs:element>

        <xs:element name="Gyarto">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="ertekeles" type="xs:integer" />
            <xs:element name="gyartonev" type="xs:string" />
            <xs:element name="elerhetosegek" type="xs:string" maxOccurs="unbounded"/>

        </xs:sequence>
    </xs:complexType>
</xs:element>

        <xs:element name="Termek">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="termeknev" type="xs:string" />
            <xs:element name="tipus" type="xs:string" />
            <xs:element name="egysegar" type="xs:integer" />
        </xs:sequence>
    </xs:complexType>
</xs:element>

        <xs:element name="Tartalmazza">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="darab" type="xs:integer" />
        </xs:sequence>
    </xs:complexType>
</xs:element>

        <xs:element name="Rendeles">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="osszertek" type="xs:integer" />
            <xs:element name="datum">
                                    <xs:complexType>
                                            <xs:element name="ev" type="xs:string" />
                                            <xs:element name="honap" type="xs:string" />
                                            <xs:element name="nap" type="xs:string" />
                                    </xs:complexType>
                                </xs:element>
        </xs:sequence>
```

```
      </xs:complexType>
   </xs:element>

      <xs:element name="Vevo">
      <xs:complexType>
         <xs:sequence>
            <xs:element name="pontok" type="xs:integer" />
                              <xs:element name="felnott" type="felnottE"/>
         </xs:sequence>
      </xs:complexType>
   </xs:element>

      <xs:element name="Nev">
      <xs:complexType>
         <xs:sequence>
            <xs:element name="vezeteknev" type="xs:string" />
                              <xs:element name="keresztnev" type="xs:string" />
         </xs:sequence>
      </xs:complexType>
   </xs:element>


</xs:schema>
```

2. Feladat

DOM program készítés

A DOM programok Eclipse programozói környezetben készültek Java nyelven. A feladat leírásnak megfelelően található a szerkezete a GitHubomon (MateSandor).

2.a Adatolvasás, kiírás konzolra – fájlba

A DomReadAQUSP7.java fájl létrehozása után importáltam bele a Dom feladatokhoz szükséges csomagokat. Ezek a parserek: DocumentBuilder, DocumentBuilder factory, ParserConfigurationException. A DOM-ok: Document, Element, Node, NodeList. Valamint importáltam még a SAXexceptiont is.

A file beolvasáshoz a java.io.File-t használva adtam át a programnak az XMLAQUSP7.xml fájlomat, amit a fentebb említett csomagok segítségével könnyen kezeltem. A lekérdezésekhez többször át kell alakítani a fájlt, de végül Document formába kerül.

A kód logikája röviden az, hogy ciklikusan ellenőrizzük a sorokat amíg Egyedet nem találunk. Ha egyedet talál akkor kiírja a nevét, majd a gyermek elemei nevét és értékeit.

```
package hu.domparse.aqusp7;


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 DomReadAQUSP7 {

	public static void main(String[] args) throws ParserConfigurationException, SAXException, IOException {
		//Forrás file
		File file = new File("XMLAQUSP7.xml");

		DocumentBuilderFactory factory = DocumentBuilderFactory.newInstance();
		DocumentBuilder dBuilder = factory.newDocumentBuilder();

		Document doc = dBuilder.parse(file);

		doc.getDocumentElement().normalize();

		//Gyökérelem
		System.out.println("Root element: " + doc.getDocumentElement().getNodeName());
		//Gyerekelemek lementése
		NodeList nList = (NodeList) doc.getDocumentElement();

		for(int i = 0; i < nList.getLength(); i++)
	{
	   Node node = nList.item(i);
	   if(!(node.getNodeName().equals("#comment")||node.getNodeName().equals("#text")))
	   {
		System.out.println("\nJelenlegi elem: " + node.getNodeName());

		if(node.getNodeType() == Node.ELEMENT_NODE)
		{
			Element elem = (Element) node;

			NodeList nList2 = elem.getChildNodes();
```

```
                    for(int j = 0; j < nList2.getLength(); j++)
                {
                        Node node2 = nList2.item(j);

                                        if(node2.getNodeType() == Node.ELEMENT_NODE)
                  {
                                                System.out.println(node2.getNodeName()
+ " : " + node2.getTextContent());

                                        }
                                    }
              }
        }
      }



              }
        }
```

2.b Adatmódosítás

Az egyszerűség kedvéért az importokat nem írom le újra hiszen szinte ugyanazok. A filebeolvasás és átalakítás az előzővel megegyező módon történt.

A módosítások amiket csináltam a következők:

- Attribútum értéket változtat → Ha a város „Miskolc” változtassa a várost „Szirma”-ra
- Gyerekelem törlés → Ha az egyed neve „Gyarto” törölje az értékelés gyerekelemét.
- Gyerekelem adat módosítás → Ha a dátum „2022” helyette írja azt, hogy now
- Új gyerekelem felvétele → Ha az egyet neve „Tartalmazza”, hozzon létre hozzá egy „kupon” nevű gyermek elemet (aminek az értéke legyen a kódban előre beállítva id függőre).
- Majd kiírja a változtatott xml fájlt konzolra.

A kódban lépésről lépésre kommentáltam, hogy érthető legyen éppen mit csinál a program, de azért megjegyzés képpen a legtöbbet a Nodeokról Nodeokra váltás volt szempontban, valamint a feltételek. Ezek metódusai a ChildNodes(), getChildNodes(), getElementsByTagName(), getNodeName() és a getNodeText(). Ahol számot kellett kinyerni, ott még parse-olni is kellett Integerre.

```
package hu.domparse.aqusp7;

import java.io.File;

import javax.xml.parsers.DocumentBuilderFactory;
import javax.xml.parsers.DocumentBuilder;
import javax.xml.transform.Transformer;
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.NodeList;
import org.w3c.dom.Node;
import org.w3c.dom.Element;
import org.w3c.dom.NamedNodeMap;

public class DomModifyAQUSP7
{
   public static void main(String argv[])
   {
      try
      {
         File inputFile = new File("XMLAQUSP7.xml");
```

```
        DocumentBuilderFactory docFactory = DocumentBuilderFactory.newInstance();
        DocumentBuilder docBuilder = docFactory.newDocumentBuilder();

        Document doc = docBuilder.parse(inputFile);

        // ------------------------------------------------------------------------------------------------------------
-----------------

        //Attribútum értékeket változtat, feltétellel for loopban az összes címre.

        NodeList cimList = doc.getElementsByTagName("Cim");
        for(int i = 0; i < cimList.getLength(); i++)
        {
            Node cim = doc.getElementsByTagName("Cim").item(i);
            NamedNodeMap attr = cim.getAttributes();
            Node nodeAttr = attr.getNamedItem("Cid");
            nodeAttr.setTextContent("lakcim" + (i+1));

            NodeList list = cim.getChildNodes();

            // for loop ami a gyerekelemek számáig megy
            for(int temp = 0; temp < list.getLength(); temp++)
            {
                Node node = list.item(temp);

                if (node.getNodeType() == Node.ELEMENT_NODE)
                {
                    Element eElement = (Element) node;

                    // A varos nevű gyerekelemnél teljesül
                    if ("varos".equals(eElement.getNodeName()))
                    {
                        // Ha a varos egyenlő ezzel
                        if ("Miskolc".equals(eElement.getTextContent()))
                        {
                            // Változtassa meg erre
                            eElement.setTextContent("Szirma");
                        }
                    }
                }
            }
        }

        // ------------------------------------------------------------------------------------------------------------
-----------------

        //Gyerekelem törlés

        NodeList gyartoList = doc.getElementsByTagName("Gyarto");
        for(int i = 0; i < gyartoList.getLength(); i++)
        {
            // Kilistázza a jelenlegi gyarto egyedet
            Node gyarto = doc.getElementsByTagName("Gyarto").item(i);

            // Lekéri annak gyerekelemeit
            NodeList childNodes = gyarto.getChildNodes();
```

```
            // Végigmegy a gyerekelemeken
            for (int count = 0; count < childNodes.getLength(); count++)
            {
                Node node = childNodes.item(count);
                // Ha a gyerekelem neve "ertekeles"
                if("ertekeles".equals(node.getNodeName()))
                {
                    // Akkor törölje
                    gyarto.removeChild(node);
                }
            }
        }

        // ----------------------------------------------------------------------------------------------------
-----------------

        // A dátumon belül módosítja az évbe írtakat

        NodeList datumList = doc.getElementsByTagName("datum");
        for(int i = 0; i < datumList.getLength(); i++)
        {
            // Kilistázza a datum egyedeket
            Node datum = doc.getElementsByTagName("datum").item(i);

            // Lekéri annak gyerekelemeit
            NodeList childNodes = datum.getChildNodes();

            // for loop ami a gyerekelemek számáig megy
            for(int temp = 0; temp < childNodes.getLength(); temp++)
            {
                Node node = childNodes.item(temp);

                // Ellenőrzés hogy a kapott egyed elem-e
                if(node.getNodeType() == Node.ELEMENT_NODE)
                {
                    Element eElement = (Element) node;

                    // Az ev nevű gyerekelemnél teljesül
                    if("ev".equals(eElement.getNodeName()))
                    {
                        // Ha az év egyenlő ezzel
                        if ("2022".equals(eElement.getTextContent()))
                        {
                            // Változtassa meg "now"-ra
                            eElement.setTextContent("now");
                        }
                    }
                }
            }
        }
        // ----------------------------------------------------------------------------------------------------
-----------------

        // Új gyerekelemet vesz fel a Tartalmazza egyedbe - "kupon", majd az FRid attribútum értéke
alapján állít neki értéket

        NodeList tartalmazzaList = doc.getElementsByTagName("Tartalmazza");
```

```
            for (int i = 0; i < tartalmazzaList.getLength(); i++)
            {
               Node tartalmazza = tartalmazzaList.item(i);

               // Lekéri az "FRid" attribútum értékét és eltárolja az "id"-ben
               String id = tartalmazza.getAttributes().getNamedItem("FTaRid").getTextContent();

               // Létrehozza az új "kupon" elemet
               Element kupon = doc.createElement("kupon");
               tartalmazza.appendChild(kupon);

               // Az "id" értéke alapján ad értéket az új "kupon" elemnek
               if ("r1".equals(id))
               {
                  kupon.appendChild(doc.createTextNode("0"));
               }
               if ("r2".equals(id))
               {
                  kupon.appendChild(doc.createTextNode("30"));
               }
               if ("r3".equals(id))
               {
                  kupon.appendChild(doc.createTextNode("0"));
               }
            }
            // ---------------------------------------------------------------------------------------------------------
-----------------

            // Tratalom konzolra írása:

            TransformerFactory transformerFactory = TransformerFactory.newInstance();
            Transformer transformer = transformerFactory.newTransformer();

            DOMSource source = new DOMSource(doc);

            System.out.println("-----Módosított fájl-----");
            StreamResult consoleResult = new StreamResult(System.out);
            transformer.transform(source, consoleResult);

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

2.c Adatlekérdezés

A csomag importálások megegyeznek az előbbiekkel, a file(xml) beolvasással együtt. Mivel ebben a programban lekérdezéseket kellett létrehozni ki is találtam párat:

- Írja ki a címeket az irányítószám kihagyásával.
- Számolja ki, hogy összesen hány féle termék van az adatbázisban.
- Számolja ki, hogy mennyi átlagosan egy termék ára.
- Listázza ki azon gyártók nevét, amelyek értékelése 3 nál jobb
- Számolja ki, hogy a Vevők hány százaléka felnőtt, majd írja ki legfeljebb 2 tizedes törtig ezt a százalékot.

A kódot végig lehet követni a kommenteim alapján, A DomModify után ebben a kódban nem jelenik meg új, ismeretlen metódus. A kódot //**…**// részekkel láttam el, ez nem szolgál semmilyen jelentést, csupán az a célja hogy könnyebben átlátható legyen a kód.

```
package hu.domparse.aqusp7;

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 DomQueryAQUSP7
{
        public static void main(String[] args) throws ParserConfigurationException, SAXException,
IOException
   {
                //Forrás file
                File file = new File("XMLAQUSP7.xml");

                DocumentBuilderFactory dbFactory = DocumentBuilderFactory.newInstance();
                DocumentBuilder dBuilder = dbFactory.newDocumentBuilder();

                Document doc = dBuilder.parse(file);
                doc.getDocumentElement().normalize();



      System.out.println("----------------\n");
      System.out.println("Elso lekerdezes:");

      // Címek kilistázása
      NodeList nList1 = doc.getElementsByTagName("Cim");
```

```
      // Végigfut az "Cim"-nek a gyerekelemein, kihagyva az "iranyitoszamot"-t
      for(int i = 0; i < nList1.getLength(); i++)
      {
         Node node1 = nList1.item(i);
          System.out.println("\nJelenlegi elem: " + node1.getNodeName());

          if(node1.getNodeType() == Node.ELEMENT_NODE)
          {
                  Element elem = (Element) node1;

                  System.out.println("Cid: " + elem.getAttribute("Cid"));

                  NodeList nList11 = elem.getChildNodes();

                  for(int j = 0; j < nList11.getLength(); j++)
            {
                          Node node11 = nList11.item(j);

                                        if(node11.getNodeType() == Node.ELEMENT_NODE)
                {
                                                if(!node11.getNodeName().equals("iranyitoszam"))
                 {
                                                        System.out.println(node11.getNodeName()
+ " : " + node11.getTextContent());
                                                }
                                        }
                                }
          }
                  }



//*****************************************************************************
*********************************************
   System.out.println("\n----------------\n");
   System.out.println("Masodik lekerdezes:\n");
   //Számolja meg a termékek számát

   // Termekek kilistázása
   NodeList nList2 = doc.getElementsByTagName("Termek");
   int termekDarabSzam = 0;

   // Végigfut az "Cim"-nek a gyerekelemein, kihagyva az "iranyitoszamot"-t
   for(int i = 0; i < nList2.getLength(); i++)
   {
          Node node2 = nList2.item(i);
      if(node2.getNodeType() == Node.ELEMENT_NODE)
      {
         if(((Element) node2).getAttribute("Tid")!=null){
                  termekDarabSzam++;
         }
      }
         }
   System.out.println("Termekek darabszama: " + termekDarabSzam);
```

```
//*****************************************************************************
*******************************************
   System.out.println("\n----------------\n");
   System.out.println("Harmadik lekerdezes:\n");
   //Számolja ki a termékek árainak átlagát

   // Termekek kilistázása
   NodeList nList3 = doc.getElementsByTagName("Termek");
   int termekSum = 0;

   // Végigfut az "Termékek"-en és azon belül az "egysegar" akat szummázza egy változóba
   for(int i = 0; i < nList3.getLength(); i++)
   {
         Node node3= nList3.item(i);
      if(node3.getNodeType() == Node.ELEMENT_NODE)
      {
         NodeList nList33 = ((Element) node3).getChildNodes();
         for (int j = 0; j < nList33.getLength(); j++) {
                  Node node33 = nList33.item(j);
                  if(node33.getNodeName().equals("egysegar"))
            {
                                    termekSum += Integer.parseInt(node33.getTextContent());
                              }
                        }

      }
         }
System.out.println("Termekek összege: " + termekSum);

//*****************************************************************************
*******************************************
System.out.println("\n----------------\n");
System.out.println("Negyedik lekerdezes:\n");
//Listázza ki azokat azoknak a gyártóknak a nevét, akiknek az értékelése jobb mint 3

// Gyartok kilistázása
NodeList nList4 = doc.getElementsByTagName("Gyarto");
// Végigfut az "Gyarto"-kon és kilistázza azoknak a nevét, akiknek az "elerhetosegek" gyermekelemén
nincs ertek
System.out.println("Gyartok 3 nal jobb ertekelessel: ");
for(int i = 0; i < nList4.getLength(); i++)
{
         Node node4= nList4.item(i);
   if(node4.getNodeType() == Node.ELEMENT_NODE)
   {
         NodeList nList4Child = ((Element) node4).getChildNodes();
         for (int j = 0; j < nList4Child.getLength(); j++) {
                  Node node4Child = nList4Child.item(j);
                  if(node4Child.getNodeName().equals("ertekeles"))
                  {
                                    if(Integer.parseInt(node4Child.getTextContent())>3) {
                                             for (int k = 0; k < nList4Child.getLength(); k++) {
                                                      Node node4Chosen = nList4Child.item(k);
                                                      if
(node4Chosen.getNodeName().equals("gyartonev")) {
```

```
        System.out.println(node4Chosen.getTextContent());
                                                }
                                        }
                                }
                        }
                        }
                }

   }
//*****************************************************************************
*****************************************
System.out.println("\n----------------\n");
System.out.println("Otodik lekerdezes:\n");
//Számolja ki hogy a vevők hány százaléka felnőtt

//Vevok kilistázása
NodeList nList5 = doc.getElementsByTagName("Vevo");
//Végigfut az "Vevo"-kön mekeresi a felnőtteket és elmenti a darabszámukat, majd százalékot számít
int felnottDarab = 0;
for(int i = 0; i < nList5.getLength(); i++)
{
        Node node5= nList5.item(i);
  if(node5.getNodeType() == Node.ELEMENT_NODE)
  {
        NodeList nList5Child = ((Element) node5).getChildNodes();
        for (int j = 0; j < nList5Child.getLength(); j++) {
                Node node5Child = nList5Child.item(j);
                if(node5Child.getNodeName().equals("felnott"))
                {
                        if (node5Child.getTextContent().equals("Igen")) {
                                felnottDarab++;
                        }
                }
        }
        }

  }
//(double) parse és formázás szükséges
System.out.println("A vevok "+String.format("%.2f",  ((double)felnottDarab / nList5.getLength() *
100))+"%-a felnott ");
   }

}
```