

JEGYZŐKÖNYV

Adatkezelés XML környezetben

Féléves feladat

Készítette: **Tisza Marcell**

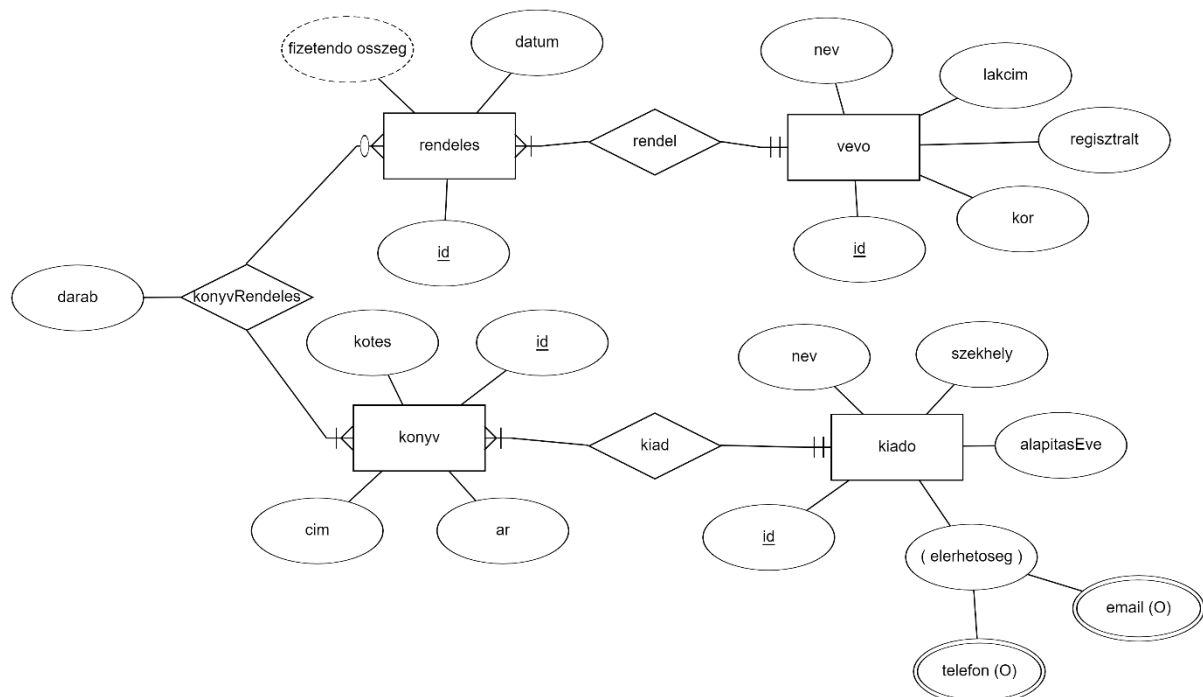
Neptunkód: **NLVHSR**

A feladat leírása:

Egy modern könyvesbolt adatbázisának a mintáját készítettem el, amely a bolt weboldalának a tartalmát képezi. Az adatbázis nyilvántartja a könyvesbolt által forgalmazott könyveket (információkat szolgáltat az árakról, valamint a könyvek kötésének típusáról stb.), a rendeléseket és a hozzájuk kapcsolódó vásárlói adatokat. Emellett a könyvek kiadóiról is tartalmaz néhány információt (mint pl. székhelyük vagy az alapításuk éve), amit majd az oldalon meg lehet jeleníteni. Tartalmazza a vásárlók adatait is, mint nevüket, korukat, lakcímüket, ahova majd a rendelést leszállíthatják, illetve tartalmazza még a regisztráció dátumát. Egy rendelés az adatbázisban mindig egy vevőhöz köthető, viszont egy vevő nevén lehet több rendelés is. Továbbá egy rendelés könyv(ek)et tartalmaz (egyet vagy többet). Ahol természetesen egy könyv több rendelésnek is része lehet, feltéve, hogy van elég belőle készleten. Mindemellett számon van még tartva, hogy egy könyv hányszor szerepel egy adott rendelésben. Ezen paramétereivel az adatbázis lehetőséget biztosít a vállalatnak, hogy nagyobb rálátása legyen az áruház készleteire, illetve a tranzakciók számontartására.

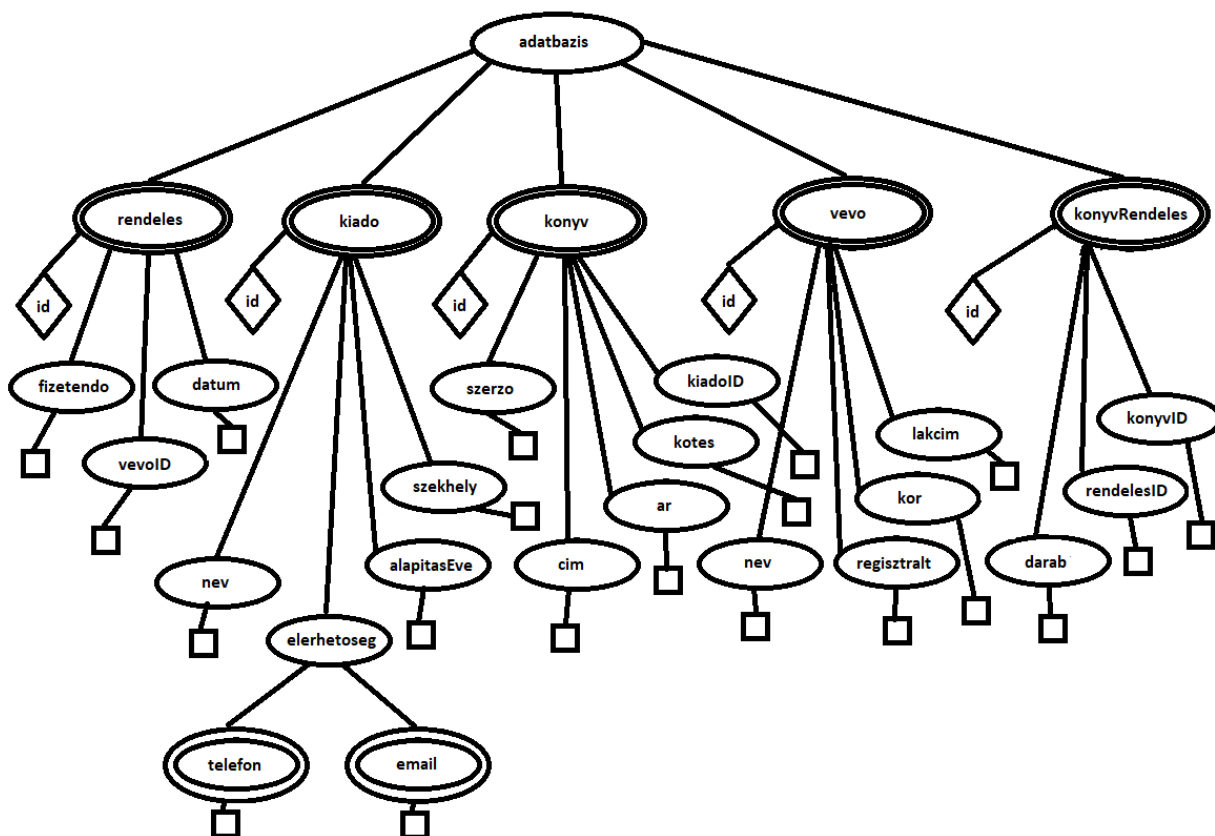
1. feladat

1a) Az adatbázis ER modell:



1b) Az adatbázis konvertálása XDM modellre

Az XDM modell az ER modelltől eltérően egy extra elemet is tartalmaz, névlegesen a *konyvRendeles*-t, amire azért van szükség, hogy meg tudjuk jeleníteni az ER modellben lévő N-M kapcsolatot. Továbbá ennek az elemnek lesz a tulajdonsága az a mennyiség, amennyit egy adott könyvből a kosárba rakott a vásárló. Az 1-N kapcsolatok a modellben kulcs-idegen kulcs párokként vannak megvalósítva. (ezt a kapcsolatot az *xmlschema* ellenőrzi). Ezeken a módosításokon kívül a modell többi tagja változatlan az ER modellhez képest



1c) Az XDM modell alapján XML dokumentum készítése

```
XMLNLVHSR.xml x
1 <?xml version="1.0" encoding="UTF-8"?>
2
3 <adatbazis xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4   xsi:noNamespaceSchemaLocation="XMLNLVHSR.xsd">
5
6   <!-- RENDELÉSEK -->
7   <rendeles id="r1">
8     <datum>2021-10-26</datum>
9     <fizetendo>2500</fizetendo>
10    <vevoID>v1</vevoID>
11  </rendeles>
12
13   <rendeles id="r2">
14     <datum>2021-10-30</datum>
15     <fizetendo>3200</fizetendo>
16     <vevoID>v2</vevoID>
17   </rendeles>
18
19   <rendeles id="r3">
20     <datum>2021-11-02</datum>
21     <fizetendo>5000</fizetendo>
22     <vevoID>v1</vevoID>
23   </rendeles>
24
25
26   <!-- KÖNYV RENDELÉSEK -->
27   <konyvRendeles id="kr1">
28     <rendelesID>r1</rendelesID>
29     <konyvID>k1</konyvID>
30     <darab>1</darab>
31   </konyvRendeles>
32   <konyvRendeles id="kr2">
33     <rendelesID>r1</rendelesID>
34     <konyvID>k2</konyvID>
35     <darab>2</darab>
36   </konyvRendeles>
37
38   <konyvRendeles id="kr3">
39     <rendelesID>r2</rendelesID>
40     <konyvID>k1</konyvID>
41     <darab>1</darab>
42   </konyvRendeles>
43   <konyvRendeles id="kr4">
44     <rendelesID>r2</rendelesID>
45     <konyvID>k2</konyvID>
46     <darab>1</darab>
47   </konyvRendeles>
48   <konyvRendeles id="kr5">
49     <rendelesID>r2</rendelesID>
50     <konyvID>k3</konyvID>
51     <darab>1</darab>
52   </konyvRendeles>
53
54   <konyvRendeles id="kr6">
55     <rendelesID>r3</rendelesID>
56   </konyvRendeles>
57
58   <!-- VEVŐK -->
59   <vevo id="v1">
60     <nev>Olajos Lajos</nev>
61     <lakcim>1077 Budapest, Wesselényi u. 55</lakcim>
62     <kor>23</kor>
63     <regisztralt>2000-01-01</regisztralt>
64   </vevo>
65
66   <vevo id="v2">
67     <nev>Varga Katalin</nev>
68     <lakcim>3530 Miskolc, Rácz György u. 8</lakcim>
69     <kor>49</kor>
70     <regisztralt>2012-10-12</regisztralt>
71   </vevo>
72
73
74
75
76
```

```

77 <!-- KÖNYVEK -->
78< konyv id="k1">
79   <szerso>Szilasi László</szerso>
80   <cim>Tavaszi hadjárat</cim>
81   <ar>1500</ar>
82   <kotes>kemény</kotes>
83   <kiadoID>kial1</kiadoID>
84 </konyv>
85
86< konyv id="k2">
87   <szerso>Györfy Ákos</szerso>
88   <cim>A távollodásban</cim>
89   <ar>500</ar>
90   <kotes>puha</kotes>
91   <kiadoID>kial1</kiadoID>
92 </konyv>
93
94< konyv id="k3">
95   <szerso>Andrew S. Tanenbaum</szerso>
96   <szerso>David J. Wetherall</szerso>
97   <cim>Számítógép-hálózatok</cim>
98   <ar>5000</ar>
99   <kotes>kemény</kotes>
100   <kiadoID>kial1</kiadoID>
101 </konyv>
102
103
104 <!-- KIADÓK -->
105< kiado id="kia1">
106   <nev>Magvető Könyvkiadó</nev>
107   <szekhely>Budapest</szekhely>
108   <alapitasEve>1955</alapitasEve>
109< elerhetoseg>
110   <telefon>+36-1/323-3981</telefon>
111   <telefon>+36-1/235-5032</telefon>
112   <email>magveto.kiado@lira.hu</email>
113 </elerhetoseg>
114 </kiado>
115
116< kiado id="kia2">
117   <nev>Panem</nev>
118   <szekhely>Budapest</szekhely>
119   <alapitasEve>1972</alapitasEve>
120< elerhetoseg>
121   <telefon>+36-1/460-0272</telefon>
122   <email>webbolt@panem.hu</email>
123 </elerhetoseg>
124 </kiado>
125
126 </adatbazis>

```

1d) Az XML dokumentum alapján XMLSchema készítése

```

1 <?xml version="1.0" encoding="UTF-8"?>
2< xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">
3<   <xs:element name="adatbazis">
4<     <xs:complexType>
5<       <xs:sequence>
6<         <xs:element name="rendeles" type="rendelesTipus" minOccurs="0" maxOccurs="unbounded"/>
7<         <xs:element name="konyvRendeles" type="konyvRendelesTipus" maxOccurs="unbounded"/>
8<         <xs:element name="vevo" type="vevoTipus" maxOccurs="unbounded"/>
9<         <xs:element name="konyv" type="konyvTipus" maxOccurs="unbounded"/>
10<         <xs:element name="kiado" type="kiadoTipus" maxOccurs="unbounded"/>
11<       </xs:sequence>
12<     </xs:complexType>
13<     <xs:key name="vevoKulcs">
14<       <xs:selector xpath="vevo"/>
15<       <xs:field xpath="@id"/>
16<     </xs:key>
17<     <xs:key name="konyvKulcs">
18<       <xs:selector xpath="konyv"/>
19<       <xs:field xpath="@id"/>
20<     </xs:key>
21<     <xs:key name="kiadoKulcs">
22<       <xs:selector xpath="kiado"/>
23<       <xs:field xpath="@id"/>
24<     </xs:key>
25<     <xs:key name="rendelesKulcs">
26<       <xs:selector xpath="rendeles"/>
27<       <xs:field xpath="@id"/>
28<     </xs:key>
29<     <xs:keyref name="rendeles-vevo" refer="vevoKulcs">
30<       <xs:selector xpath="rendeles/vevoID"/>
31<       <xs:field xpath="."/>

```

```

32      </xs:keyref>
33      <xs:keyref name="konyvRendeles-konyv" refer="konyvKulcs">
34        <xs:selector xpath="konyvRendeles/konyvID"/>
35        <xs:field xpath="."/>
36      </xs:keyref>
37      <xs:keyref name="konyv-kiado" refer="kiadoKulcs">
38        <xs:selector xpath="konyv/kiadoID"/>
39        <xs:field xpath="."/>
40      </xs:keyref>
41      <xs:keyref name="konyvRendeles-rendeles" refer="rendelesKulcs">
42        <xs:selector xpath="konyvRendeles/rendelesID"/>
43        <xs:field xpath="."/>
44      </xs:keyref>
45    </xs:element>
46
47    <xs:complexType name="vevoTipus">
48      <xs:sequence>
49        <xs:element name="nev" type="xs:string"/>
50        <xs:element name="lakcim" type="xs:string"/>
51        <xs:element name="kor" type="xs:unsignedInt"/>
52        <xs:element name="regisztralt" type="xs:date"/>
53      </xs:sequence>
54      <xs:attribute name="id" type="xs:ID" use="required"/>
55    </xs:complexType>
56
57    <xs:complexType name="konyvTipus">
58      <xs:sequence>
59        <xs:element name="szerzo" type="xs:string" maxOccurs="unbounded"/>
60        <xs:element name="cim" type="xs:string"/>
61        <xs:element name="ar" type="xs:int"/>
62        <xs:element name="kotes" type="kotesTipus"/>
63        <xs:element name="kiadoID" type="xs:string"/>
64      </xs:sequence>
65      <xs:attribute name="id" type="xs:ID" use="required"/>
66    </xs:complexType>
67
68    <xs:simpleType name="kotesTipus">
69      <xs:restriction base="xs:string">
70        <xs:enumeration value="kemeny"/>
71        <xs:enumeration value="puha"/>
72      </xs:restriction>
73    </xs:simpleType>
74
75    <xs:complexType name="kiadoTipus">
76      <xs:sequence>
77        <xs:element name="nev" type="xs:string"/>
78        <xs:element name="szekhely" type="xs:string"/>
79        <xs:element name="alapitasEve" type="xs:unsignedInt"/>
80        <xs:element name="elerhetoseg" type="elerhetosegTipus"/>
81      </xs:sequence>
82      <xs:attribute name="id" type="xs:ID" use="required"/>
83    </xs:complexType>
84
85    <xs:complexType name="elerhetosegTipus">
86      <xs:sequence>
87        <xs:element name="telefon" type="telefonTipus" minOccurs="1" maxOccurs="unbounded"/>
88        <xs:element name="email" type="emailTipus" minOccurs="0" maxOccurs="unbounded"/>
89      </xs:sequence>
90    </xs:complexType>
91
92    <xs:simpleType name="emailTipus">
93      <xs:restriction base="xs:string">
94        <xs:pattern value="[w.]+@[w.]+\.[w+]"/>
95      </xs:restriction>
96    </xs:simpleType>
97
98    <xs:simpleType name="telefonTipus">
99      <xs:restriction base="xs:string">
100        <xs:pattern value="+36-\d{1,2}/\d{3}-\d{4}"/>
101      </xs:restriction>
102    </xs:simpleType>
103
104    <xs:complexType name="rendelesTipus">
105      <xs:sequence>
106        <xs:element name="datum" type="xs:date"/>
107        <xs:element name="fizetendo" type="xs:unsignedInt"/>
108        <xs:element name="vevoID" type="xs:string"/>
109      </xs:sequence>
110      <xs:attribute name="id" type="xs:ID" use="required"/>
111    </xs:complexType>
112
113    <xs:complexType name="konyvRendelesTipus">
114      <xs:sequence>
115        <xs:element name="rendelesID" type="xs:string"/>
116        <xs:element name="konyvID" type="xs:string"/>
117        <xs:element name="darab" type="xs:unsignedInt"/>
118      </xs:sequence>
119      <xs:attribute name="id" type="xs:ID" use="required"/>
120    </xs:complexType>
121  </xs:schema>

```

2. feladat

2a) adatolvasás – DOMReadNLVHSR.java

```
DOMReadNLVHSR.java ×
1 package hu.dompars.nlvhsr;
2
3 import java.io.File;
4 import java.io.IOException;
5
6 import javax.xml.parsers.DocumentBuilder;
7 import javax.xml.parsers.DocumentBuilderFactory;
8 import javax.xml.parsers.ParserConfigurationException;
9
10 import org.w3c.dom.Document;
11 import org.w3c.dom.Element;
12 import org.w3c.dom.Node;
13 import org.w3c.dom.NodeList;
14 import org.xml.sax.SAXException;
15
16 public class DOMReadNLVHSR {
17
18     public static void main(String[] args) {
19         try {
20
21             // A DOM objektum létrehozása az XML dokumentumból
22             File xmlFile = new File("XMLNLVHSR.xml");
23             DocumentBuilderFactory factory = DocumentBuilderFactory.newInstance();
24             DocumentBuilder dBuilder = factory.newDocumentBuilder();
25             Document doc = dBuilder.parse(xmlFile);
26             doc.getDocumentElement().normalize();
27
28             // Gyökér elem kiírása (adatbázis)
29             System.out.println("Root element: " + doc.getDocumentElement().getNodeName() + "\n");
30
31             // Az adatbázis elemeinek kiírása tulajdonságaikkal együtt formázva
32             String[] tagNames = { "vevo", "konyv", "kiado", "rendeles", "konyvRendeles" };
33             for (String tagName : tagNames) {
34                 NodeList nodeList = doc.getElementsByTagName(tagName);
35                 String message = tagName + " elemek:";
36                 System.out.println("\n" + message);
37                 System.out.println("-".repeat(message.length()) + "\n");
38
39                 // Elemek tulajdonságainak és azonosítójának kiírása
40                 for (int i = 0; i < nodeList.getLength(); i++) {
41                     Node nNode = nodeList.item(i);
42                     if (nNode.getNodeType() == Node.ELEMENT_NODE) {
43                         Element elem = (Element)nNode;
44
45                         // Azonosító kiírása
46                         String id = elem.getAttribute("id");
47                         System.out.println(">>> ID: " + id);
48
49                         // Tulajdonságok (gyermek elemek) kiírása
50                         String nodeContent = "";
51                         NodeList childNodes = elem.getChildNodes();
52                         for (int j = 0; j < childNodes.getLength(); j++) {
53                             if (childNodes.item(j).getTextContent().trim() != "") {
54                                 nodeContent = normalizeText(childNodes.item(j).getTextContent().trim());
55                                 System.out.println(childNodes.item(j).getNodeName() + ": " + nodeContent);
56                             }
57                         }
58                     }
59                 }
60                 System.out.println();
61             }
62
63             // Esetleges hibák kezelése
64         } catch (SAXException | IOException | ParserConfigurationException ex) {
65             System.out.println("Some error occurred\nDescription:\n" + ex.getMessage());
66             ex.printStackTrace();
67         }
68     }
69
70     // Szöveg formázása a szép megjelenés érdekében
71     private static String normalizeText(String text) {
72         text = text.replaceAll("\n", " ");
73         text = text.replaceAll("\s+", " ");
74         return text;
75     }
76
77 }
```

2b) adatmódosítás – DOMModifyNLVHSR.java

```
DOMModifyNLVHSR.java x
1 package hu.domparsen.lvhhsr;
2
3 import java.io.File;
4 import java.io.IOException;
5 import java.io.UnsupportedEncodingException;
6
7 import javax.xml.parsers.DocumentBuilder;
8 import javax.xml.parsers.DocumentBuilderFactory;
9 import javax.xml.parsers.ParserConfigurationException;
10 import javax.xml.transform.OutputKeys;
11 import javax.xml.transform.Transformer;
12 import javax.xml.transform.TransformerException;
13 import javax.xml.transform.TransformerFactory;
14 import javax.xml.transform.dom.DOMSource;
15 import javax.xml.transform.stream.StreamResult;
16
17 import org.w3c.dom.Document;
18 import org.w3c.dom.Element;
19 import org.w3c.dom.Node;
20 import org.w3c.dom.NodeList;
21 import org.xml.sax.SAXException;
22
23 public class DOMModifyNLVHSR {
24
25     public static void main(String[] args) {
26         try {
27             // A DOM objektum létrehozása az XML dokumentumból
28             File xmlFile = new File("XMLNLVHSR.xml");
29             DocumentBuilderFactory factory = DocumentBuilderFactory.newInstance();
30             DocumentBuilder dBuilder = factory.newDocumentBuilder();
31             Document document = dBuilder.parse(xmlFile);
32             document.getDocumentElement().normalize();
33
34             // A v1-es azonosítójú vásárló korának módosítása 19-re
35             NodeList nodes = document.getElementsByTagName("vevo");
36             for (int i = 0; i < nodes.getLength(); i++) {
37                 Node node = nodes.item(i);
38                 if (node.getNodeType() == Node.ELEMENT_NODE) {
39                     if (node.getAttribute("id").getTextContent().equals("v1")) {
40                         NodeList childNodes = node.getChildNodes();
41                         for (int j = 0; j < childNodes.getLength(); j++) {
42                             Node childNode = childNodes.item(j);
43                             if (childNode.getNodeName().equals("kor")) {
44                                 childNode.setTextContent("19");
45                             }
46                         }
47                     }
48                 }
49             }
50
51             // Az k1-es azonosítójú könyv árának módosítása 5000-re
52             nodes = document.getElementsByTagName("konyv");
53             for (int i = 0; i < nodes.getLength(); i++) {
54                 Node node = nodes.item(i);
55                 if (node.getNodeType() == Node.ELEMENT_NODE) {
56                     if (node.getAttribute("id").getTextContent().equals("k1")) {
57                         NodeList childNodes = node.getChildNodes();
58                         for (int j = 0; j < childNodes.getLength(); j++) {
59                             Node childNode = childNodes.item(j);
60                             if (childNode.getNodeName().equals("ar")) {
61                                 childNode.setTextContent("5000");
62                             }
63                         }
64                     }
65                 }
66             }
67
68             //A kial-es kiado elérhetőségeinek a kibővítése a valami@gmail.com email címmel
69             nodes = document.getElementsByTagName("kiado");
70             for (int i = 0; i < nodes.getLength(); i++) {
71                 Node node = nodes.item(i);
72                 if (node.getNodeType() == Node.ELEMENT_NODE) {
73                     if (node.getAttribute("id").getTextContent().equals("kial1")) {
74                         NodeList childNodes = node.getChildNodes();
75                         for (int j = 0; j < childNodes.getLength(); j++) {
76                             Node childNode = childNodes.item(j);
77                             if (childNode.getNodeName().equals("elerhetoseg")) {
78                                 Element newElement = document.createElement("email");
79                                 newElement.setTextContent("valami@gmail.com");
80                                 childNode.appendChild(newElement);
81                             }
82                         }
83                     }
84                 }
85             }
86
87             // Kiírás
88             File myFile = new File("XMLNLVHSR.xml");
89             writeXml(document, myFile);
90         }
91         // Esetleges hibák kezelése
92         catch (ParserConfigurationException | SAXException | IOException | TransformerException ex) {
```



```

93     System.out.println("Some error occured\nDescription:\n" + ex.getMessage());
94     ex.printStackTrace();
95 }
96 }
97
98 // A módosított XML dokumentum kiíratása fájlba és a konzolra
99 private static void writeXml(Document doc, File output) throws TransformerException, UnsupportedEncodingException {
100     TransformerFactory transformerFactory = TransformerFactory.newInstance();
101     Transformer transf = transformerFactory.newTransformer();
102     transf.setOutputProperty(OutputKeys.ENCODING, "UTF-8");
103     transf.setOutputProperty(OutputKeys.INDENT, "yes");
104     transf.setOutputProperty("{http://xml.apache.org/xslt}indent-amunt", "2");
105
106     DOMSource source = new DOMSource(doc);
107
108     StreamResult console = new StreamResult(System.out);
109     StreamResult file = new StreamResult(output);
110
111     transf.transform(source, console);
112     transf.transform(source, file);
113 }
114 }

```

2c) adatlekérdezés – DOMQueryNLVHSR.java

```

DOMQueryNLVHSR.java x
1 package hu.domparsel.nlvhsr;
2
3 import java.io.File;
4 import java.io.IOException;
5
6 import javax.xml.parsers.DocumentBuilder;
7 import javax.xml.parsers.DocumentBuilderFactory;
8 import javax.xml.parsers.ParserConfigurationException;
9 import javax.xml.xpath.XPath;
10 import javax.xml.xpath.XPathConstants;
11 import javax.xml.xpath.XPathExpressionException;
12 import javax.xml.xpath.XPathFactory;
13
14 import org.w3c.dom.Document;
15 import org.w3c.dom.Element;
16 import org.w3c.dom.Node;
17 import org.w3c.dom.NodeList;
18 import org.xml.sax.SAXException;
19
20 public class DOMQueryNLVHSR {
21
22     public static void main(String[] args) {
23         try {
24             // A DOM objektum létrehozása az XML dokumentumból
25             File xmlFile = new File("XMLNLVHSR.xml");
26             DocumentBuilderFactory factory = DocumentBuilderFactory.newInstance();
27             DocumentBuilder dBuilder = factory.newDocumentBuilder();
28             Document document = dBuilder.parse(xmlFile);
29             document.getDocumentElement().normalize();
30
31             // XPath objektum létrehozása
32             XPath xPath = XPathFactory.newInstance().newXPath();
33
34             //-----LEKÉRDEZÉSEK-----
35
36             // 3000 Ft-nál nagyobb értékű rendelések
37             String expression = "adatbazis/rendeles[fizetendo>3000]";
38
39             // v2 azonosítójú vásárló
40             //String expression = "adatbazis/vevo[@id='v2']";
41
42             // 20 és 30 év közötti vásárlók
43             //String expression = "adatbazis/vevo[kor>20 and kor<30]";
44
45             // Az összes adatbázis bejegyzése
46             //String expression = "adatbazis/*";
47
48             //-----
49
50             // A query expression kiértékelése
51             NodeList nodeList = (NodeList) xPath.compile(expression).evaluate(document, XPathConstants.NODESET);
52
53             // Az eredményül kapott elemek kiírása
54             for (int j = 0; j < nodeList.getLength(); j++) {
55                 Node node = nodeList.item(j);
56                 System.out.println("\n>>> Elem típusa: " + node.getNodeName());
57                 Node nNode = nodeList.item(j);
58                 if (nNode.getNodeType() == Node.ELEMENT_NODE) {
59                     Element elem = (Element)nNode;
60
61                     // Azonosító kiírása
62                     String id = elem.getAttribute("id");

```

```

63         System.out.println("ID: " + id);
64
65         // Tulajdonságok (gyermek elemek) kiírása
66         String nodeContent = "";
67         NodeList childNodes = elem.getChildNodes();
68         for (int k = 0; k < childNodes.getLength(); k++) {
69             if (childNodes.item(k).getTextContent().trim() != "") {
70                 nodeContent = normalizeText(childNodes.item(k).getTextContent().trim());
71                 System.out.println(childNodes.item(k).getNodeName() + ": " + nodeContent);
72             }
73         }
74     }
75     System.out.println();
76 }
77
78 // Esetleges hibák kezelése
79 catch(ParserConfigurationException | IOException | SAXException | XPathExpressionException ex) {
80     System.out.println("Some error occurred\nDescription:\n" + ex.getMessage());
81     ex.printStackTrace();
82 }
83 }
84
85 // Szöveg formázása a szép megjelenés érdekében
86 private static String normalizeText(String text) {
87     text = text.replaceAll("\\n", " ");
88     text = text.replaceAll("\\s+", " ");
89     return text;
90 }
91 }

```