JEGYZŐKÖNYV

Adatkezelés XML környezeteben

Féléves feladat

Készítette: **Tisza Marcell**

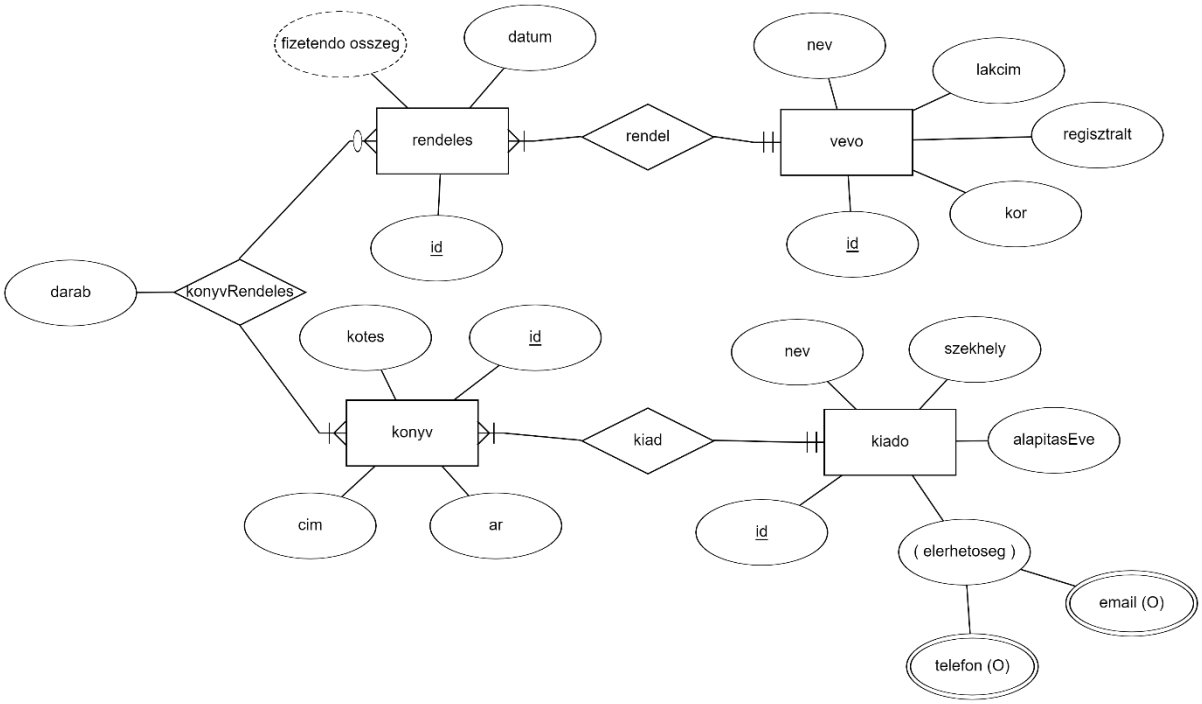
Neptunkód: **NLVHSR**

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.

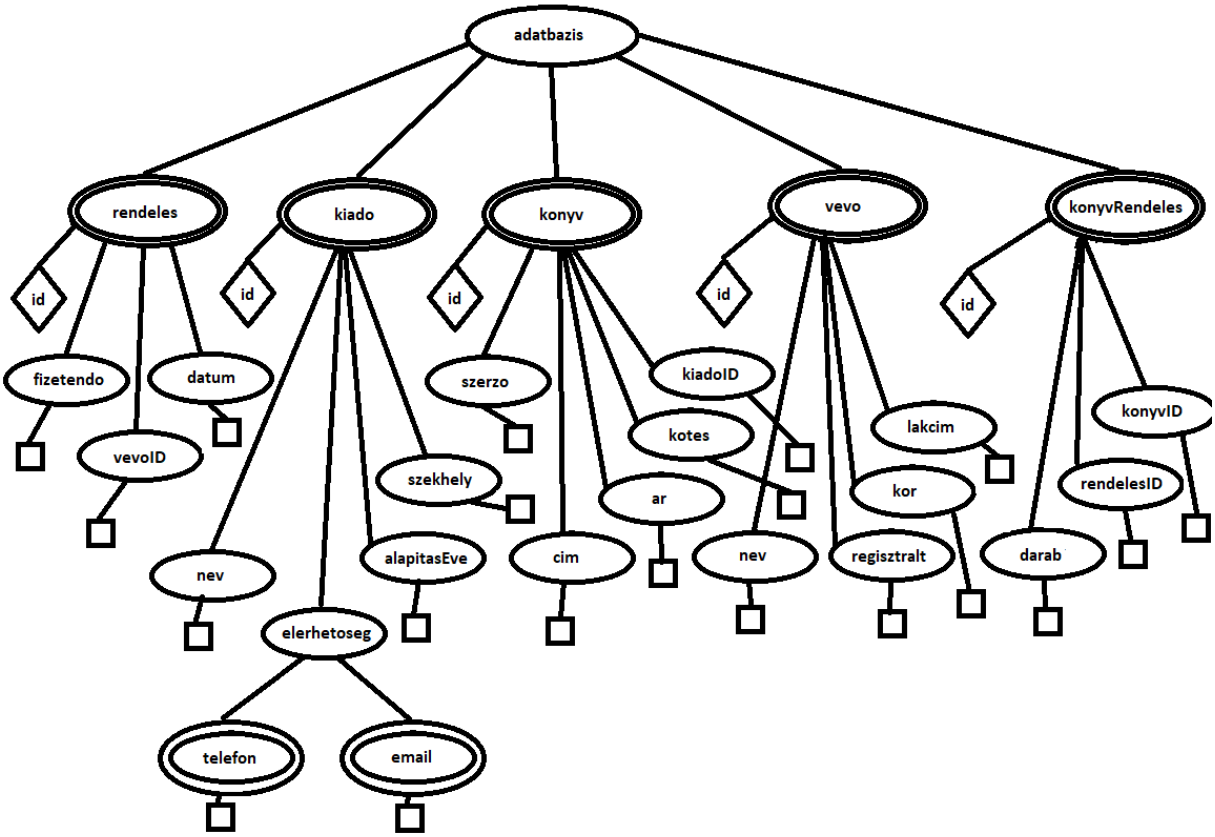
1. feladat

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 tuljdonsága 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

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

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

<!-- RENDELÉSEK -->
    <rendeles id="r1">
        <datum>2021-10-26</datum>
        <fizetendo>2500</fizetendo>
        <vevoID>v1</vevoID>
    </rendeles>

    <rendeles id="r2">
        <datum>2021-10-30</datum>
        <fizetendo>3200</fizetendo>
        <vevoID>v2</vevoID>
    </rendeles>

    <rendeles id="r3">
        <datum>2021-11-02</datum>
        <fizetendo>5000</fizetendo>
        <vevoID>v1</vevoID>
    </rendeles>


<!-- KÖNYV RENDELÉSEK -->
    <konyvRendeles id="kr1">
        <rendelesID>r1</rendelesID>
        <konyvID>k1</konyvID>
        <darab>1</darab>
    </konyvRendeles>
    <konyvRendeles id="kr2">
        <rendelesID>r1</rendelesID>
        <konyvID>k2</konyvID>
        <darab>2</darab>
    </konyvRendeles>

    <konyvRendeles id="kr3">
        <rendelesID>r2</rendelesID>
        <konyvID>k1</konyvID>
        <darab>1</darab>
    </konyvRendeles>
    <konyvRendeles id="kr4">
        <rendelesID>r2</rendelesID>
        <konyvID>k2</konyvID>
        <darab>1</darab>
    </konyvRendeles>
    <konyvRendeles id="kr5">
        <rendelesID>r2</rendelesID>
        <konyvID>k3</konyvID>
        <darab>1</darab>
    </konyvRendeles>

    <konyvRendeles id="kr6">
        <rendelesID>r3</rendelesID>


<!-- VEVŐK -->
    <vevo id="v1">
        <nev>Olajos Lajos</nev>
        <lakcim>1077 Budapest, Wesselényi u. 55</lakcim>
        <kor>23</kor>
        <regisztralt>2000-01-01</regisztralt>
    </vevo>

    <vevo id="v2">
        <nev>Varga Katalin</nev>
        <lakcim>3530 Miskolc, Rácz György u. 8</lakcim>
        <kor>49</kor>
        <regisztralt>2012-10-12</regisztralt>
    </vevo>
```

```xml
<!-- KÖNYVEK -->
    <konyv id="k1">
        <szerzo>Szilasi László</szerzo>
        <cim>Tavaszi hadjárat</cim>
        <ar>1500</ar>
        <kotes>kemeny</kotes>
        <kiadoID>kia1</kiadoID>
    </konyv>

    <konyv id="k2">
        <szerzo>Győrffy Ákos</szerzo>
        <cim>A távolodásban</cim>
        <ar>500</ar>
        <kotes>puha</kotes>
        <kiadoID>kia1</kiadoID>
    </konyv>

    <konyv id="k3">
        <szerzo>Andrew S. Tanenbaum</szerzo>
        <szerzo>David J. Wetherall</szerzo>
        <cim>Számítógép-hálózatok</cim>
        <ar>5000</ar>
        <kotes>kemeny</kotes>
        <kiadoID>kia1</kiadoID>
    </konyv>


<!-- KIADÓK -->
    <kiado id="kia1">
       <nev>Magvető Könyvkiadó</nev>
       <szekhely>Budapest</szekhely>
       <alapitasEve>1955</alapitasEve>
       <elerhetoseg>
           <telefon>+36-1/323-3981</telefon>
           <telefon>+36-1/235-5032</telefon>
           <email>magveto.kiado@lira.hu</email>
       </elerhetoseg>
    </kiado>

    <kiado id="kia2">
        <nev>Panem</nev>
        <szekhely>Budapest</szekhely>
        <alapitasEve>1972</alapitasEve>
        <elerhetoseg>
            <telefon>+36-1/460-0272</telefon>
            <email>webbolt@panem.hu</email>
        </elerhetoseg>
    </kiado>

</adatbazis>
```

1d) Az XML dokumentum alapján XMLSchema készítése

XMLNLVHSR.xsd

```xml
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">
    <xs:element name="adatbazis">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="rendeles" type="rendelesTipus" minOccurs="0" maxOccurs="unbounded"/>
                <xs:element name="konyvRendeles" type="konyvRendelesTipus" maxOccurs="unbounded"/>
                <xs:element name="vevo" type="vevoTipus" maxOccurs="unbounded"/>
                <xs:element name="konyv" type="konyvTipus" maxOccurs="unbounded"/>
                <xs:element name="kiado" type="kiadoTipus" maxOccurs="unbounded"/>
            </xs:sequence>
        </xs:complexType>
        <xs:key name="vevoKulcs">
            <xs:selector xpath="vevo"/>
            <xs:field xpath="@id"/>
        </xs:key>
        <xs:key name="konyvKulcs">
            <xs:selector xpath="konyv"/>
            <xs:field xpath="@id"/>
        </xs:key>
        <xs:key name="kiadoKulcs">
            <xs:selector xpath="kiado"/>
            <xs:field xpath="@id"/>
        </xs:key>
        <xs:key name="rendelesKulcs">
            <xs:selector xpath="rendeles"/>
            <xs:field xpath="@id"/>
        </xs:key>
        <xs:keyref name="rendeles-vevo" refer="vevoKulcs">
            <xs:selector xpath="rendeles/vevoID"/>
            <xs:field xpath="."/>
```

```
        </xs:keyref>
        <xs:keyref name="konyvRendeles-konyv" refer="konyvKulcs">
            <xs:selector xpath="konyvRendeles/konyvID"/>
            <xs:field xpath="."/>
        </xs:keyref>
        <xs:keyref name="konyv-kiado" refer="kiadoKulcs">
            <xs:selector xpath="konyv/kiadoID"/>
            <xs:field xpath="."/>
        </xs:keyref>
        <xs:keyref name="konyvRendeles-rendeles" refer="rendelesKulcs">
            <xs:selector xpath="konyvRendeles/rendelesID"/>
            <xs:field xpath="."/>
        </xs:keyref>
    </xs:element>

    <xs:complexType name="vevoTipus">
        <xs:sequence>
            <xs:element name="nev" type="xs:string"/>
            <xs:element name="lakcim" type="xs:string"/>
            <xs:element name="kor" type="xs:unsignedInt"/>
            <xs:element name="regisztralt" type="xs:date"/>
        </xs:sequence>
        <xs:attribute name="id" type="xs:ID" use="required"/>
    </xs:complexType>

    <xs:complexType name="konyvTipus">
        <xs:sequence>
            <xs:element name="szerzo" type="xs:string" maxOccurs="unbounded"/>
            <xs:element name="cim" type="xs:string"/>
            <xs:element name="ar" type="xs:int"/>
            <xs:element name="kotes" type="kotesTipus"/>
            <xs:element name="kiadoID" type="xs:string"/>
        </xs:sequence>
        <xs:attribute name="id" type="xs:ID" use="required"/>
    </xs:complexType>

    <xs:simpleType name="kotesTipus">
        <xs:restriction base="xs:string">
            <xs:enumeration value="kemeny"/>
            <xs:enumeration value="puha"/>
        </xs:restriction>
    </xs:simpleType>

    <xs:complexType name="kiadoTipus">
        <xs:sequence>
            <xs:element name="nev" type="xs:string"/>
            <xs:element name="szekhely" type="xs:string"/>
            <xs:element name="alapitasEve" type="xs:unsignedInt"/>
            <xs:element name="elerhetoseg" type="elerhetosegTipus"/>
        </xs:sequence>
        <xs:attribute name="id" type="xs:ID" use="required"/>
    </xs:complexType>

    <xs:complexType name="elerhetosegTipus">
        <xs:sequence>
            <xs:element name="telefon" type="telefonTipus" minOccurs="1" maxOccurs="unbounded"/>
            <xs:element name="email" type="emailTipus" minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>

    <xs:simpleType name="emailTipus">
        <xs:restriction base="xs:string">
            <xs:pattern value="[\w.]+@\w+\.\w+"/>
        </xs:restriction>
    </xs:simpleType>

    <xs:simpleType name="telefonTipus">
        <xs:restriction base="xs:string">
            <xs:pattern value="\+36-\d{1,2}/\d{3}-\d{4}"/>
        </xs:restriction>
    </xs:simpleType>

    <xs:complexType name="rendelesTipus">
        <xs:sequence>
            <xs:element name="datum" type="xs:date"/>
            <xs:element name="fizetendo" type="xs:unsignedInt"/>
            <xs:element name="vevoID" type="xs:string"/>
        </xs:sequence>
        <xs:attribute name="id" type="xs:ID" use="required"/>
    </xs:complexType>

    <xs:complexType name="konyvRendelesTipus">
        <xs:sequence>
            <xs:element name="rendelesID" type="xs:string"/>
            <xs:element name="konyvID" type="xs:string"/>
            <xs:element name="darab" type="xs:unsignedInt"/>
        </xs:sequence>
        <xs:attribute name="id" type="xs:ID" use="required"/>
    </xs:complexType>
</xs:schema>
```

2. feladat

2a) adatolvasás – DOMReadNLVHSR.java

DOMReadNLVHSR.java

```java
package hu.domparse.nlvhsr;

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 DOMReadNLVHSR {

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

            // A DOM objektum létrehozása az XML dokumentumból
            File xmlFile = new File("XMLNLVHSR.xml");
            DocumentBuilderFactory factory = DocumentBuilderFactory.newInstance();
            DocumentBuilder dBuilder = factory.newDocumentBuilder();
            Document doc = dBuilder.parse(xmlFile);
            doc.getDocumentElement().normalize();

            // Gyökér elem kiírása (adatbazis)
            System.out.println("Root element: " + doc.getDocumentElement().getNodeName() + "\n");

            // Az adatbázis elemeinek kiiratása tulajdonságaikkal együtt formázva
            String[] tagNames = { "vevo", "konyv", "kiado", "rendeles", "konyvRendeles" };
            for (String tagName : tagNames) {
                NodeList nodeList = doc.getElementsByTagName(tagName);
                String message = tagName + " elemek:";
                System.out.println("\n" + message);
                System.out.println("-".repeat(message.length()) + "\n");

                // Elemek tulajdonságainak és azonosítójának kiírása
                for (int i = 0; i < nodeList.getLength(); i++) {
                    Node nNode = nodeList.item(i);
                    if (nNode.getNodeType() == Node.ELEMENT_NODE) {
                        Element elem = (Element)nNode;

                        // Azonosító kiírása
                        String id = elem.getAttribute("id");
                        System.out.println(">>> ID: " + id);

                        // Tulajdonságok (gyermek elemek) kiírása
                        String nodeContent = "";
                        NodeList childNodes = elem.getChildNodes();
                        for (int j = 0; j < childNodes.getLength(); j++) {
                            if (childNodes.item(j).getTextContent().trim() != "") {
                                nodeContent = normalizeText(childNodes.item(j).getTextContent().trim());
                                System.out.println(childNodes.item(j).getNodeName() + ": " + nodeContent);
                            }
                        }
                    }
                    System.out.println();
                }
            }

        // Esetleges hibák kezelése
        } catch (SAXException | IOException | ParserConfigurationException ex) {
            System.out.println("Some error occured\nDescription:\n" + ex.getMessage());
            ex.printStackTrace();
        }
    }

    // Szöveg formazása a szép megjelenés érdekében
    private static String normalizeText(String text) {
        text = text.replaceAll("\\n", ", ");
        text = text.replaceAll("\\s+", " ");
        return text;
    }

}
```

2b) adatmódosítás – DOMModifyNLVHSR.java

DOMModifyNLVHSR.java

```java
package hu.domparse.nlvhsr;

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

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.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 DOMModifyNLVHSR {

    public static void main(String[] args) {
        try {
            // A DOM objektum létrehozása az XML dokumentumból
            File xmlFile = new File("XMLNLVHSR.xml");
            DocumentBuilderFactory factory = DocumentBuilderFactory.newInstance();
            DocumentBuilder dBuilder = factory.newDocumentBuilder();
            Document document = dBuilder.parse(xmlFile);
            document.getDocumentElement().normalize();

            // A v1-es azonosítójú vásárló korának módosítása 19-re
            NodeList nodes = document.getElementsByTagName("vevo");
            for (int i = 0; i < nodes.getLength(); i++) {
                Node node = nodes.item(i);
                if (node.getNodeType() == Node.ELEMENT_NODE) {
                    if (node.getAttributes().getNamedItem("id").getTextContent().equals("v1")) {
                        NodeList childNodes = node.getChildNodes();
                        for (int j = 0; j < childNodes.getLength(); j++) {
                            Node childNode = childNodes.item(j);
                            if (childNode.getNodeName().equals("kor")) {
                                childNode.setTextContent("19");
                            }
                        }
                    }
                }
            }

            // Az k1-es azonosítójú könyv árának módosítása 5000-re
            nodes = document.getElementsByTagName("konyv");
            for (int i = 0; i < nodes.getLength(); i++) {
                Node node = nodes.item(i);
                if (node.getNodeType() == Node.ELEMENT_NODE) {
                    if (node.getAttributes().getNamedItem("id").getTextContent().equals("k1")) {
                        NodeList childNodes = node.getChildNodes();
                        for (int j = 0; j < childNodes.getLength(); j++) {
                            Node childNode = childNodes.item(j);
                            if (childNode.getNodeName().equals("ar")) {
                                childNode.setTextContent("5000");
                            }
                        }
                    }
                }
            }

            //A kia1-es kiadó elérhetőségeinek a kibővítése a valami@gmail.com email címmel
            nodes = document.getElementsByTagName("kiado");
            for (int i = 0; i < nodes.getLength(); i++) {
                Node node = nodes.item(i);
                if (node.getNodeType() == Node.ELEMENT_NODE) {
                    if (node.getAttributes().getNamedItem("id").getTextContent().equals("kia1")) {
                        NodeList childNodes = node.getChildNodes();
                        for (int j = 0; j < childNodes.getLength(); j++) {
                            Node childNode = childNodes.item(j);
                            if (childNode.getNodeName().equals("elerhetoseg")) {
                                Element newElement = document.createElement("email");
                                newElement.setTextContent("valami@gmail.com");
                                childNode.appendChild(newElement);
                            }
                        }
                    }
                }
            }

            // Kiiratás
            File myFile = new File("XMLNLVHSR.xml");
            writeXml(document, myFile);
        }
        // Esetleges hibák kezelése
        catch (ParserConfigurationException | SAXException | IOException | TransformerException ex) {
```

```
            System.out.println("Some error occured\nDescription:\n" + ex.getMessage());
            ex.printStackTrace();
        }
    }

    // A módosított XML dokumentum kiiratása fájlba és a konzolra
    private static void writeXml(Document doc, File output) throws TransformerException, UnsupportedEncodingException {
        TransformerFactory transformerFactory = TransformerFactory.newInstance();
        Transformer transf = transformerFactory.newTransformer();
        transf.setOutputProperty(OutputKeys.ENCODING, "UTF-8");
        transf.setOutputProperty(OutputKeys.INDENT, "yes");
        transf.setOutputProperty("{http://xml.apache.org/xslt}indent-amunt", "2");

        DOMSource source = new DOMSource(doc);

        StreamResult console = new StreamResult(System.out);
        StreamResult file = new StreamResult(output);

        transf.transform(source, console);
        transf.transform(source, file);
    }
}
```

2c) adatlekérdezés – DOMQueryNLVHSR.java

DOMQueryNLVHSR.java

```
package hu.domparse.nlvhsr;

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.xpath.XPath;
import javax.xml.xpath.XPathConstants;
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 DOMQueryNLVHSR {

    public static void main(String[] args) {
        try {
            // A DOM objektum létrehozása az XML dokumentumból
            File xmlFile = new File("XMLNLVHSR.xml");
            DocumentBuilderFactory factory = DocumentBuilderFactory.newInstance();
            DocumentBuilder dBuilder = factory.newDocumentBuilder();
            Document document = dBuilder.parse(xmlFile);
            document.getDocumentElement().normalize();

            // XPath objektum létrehozása
            XPath xPath = XPathFactory.newInstance().newXPath();

            //-------------------LEKÉRDEZÉSEK-------------------

            // 3000 Ft-nál nagyobb értékű rendelések
            String expression = "adatbazis/rendeles[fizetendo>3000]";

            // v2 azonosítójú vásárló
            //String expression = "adatbazis/vevo[@id='v2']";

            // 20 és 30 év közötti vásárlók
            //String expression = "adatbazis/vevo[kor>20 and kor<30]";

            // Az összes adatbázis bejegyzése
            //String expression = "adatbazis/*";

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

            // A query expression kiértékelése
            NodeList nodeList = (NodeList) xPath.compile(expression).evaluate(document, XPathConstants.NODESET);

            // Az eredményül kapott elemek kiírása
            for (int j = 0; j < nodeList.getLength(); j++) {
                Node node = nodeList.item(j);
                System.out.println("\n>>> Elem típusa: " + node.getNodeName());
                Node nNode = nodeList.item(j);
                if (nNode.getNodeType() == Node.ELEMENT_NODE) {
                    Element elem = (Element)nNode;

                    // Azonosító kiírása
                    String id = elem.getAttribute("id");
```

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

                    // Tulajdonságok (gyermek elemek) kiírása
                    String nodeContent = "";
                    NodeList childNodes = elem.getChildNodes();
                    for (int k = 0; k < childNodes.getLength(); k++) {
                        if (childNodes.item(k).getTextContent().trim() != "") {
                            nodeContent = normalizeText(childNodes.item(k).getTextContent().trim());
                            System.out.println(childNodes.item(k).getNodeName() + ": " + nodeContent);
                        }
                    }
                }
                System.out.println();
            }
        }
        // Esetleges hibák kezelése
        catch(ParserConfigurationException | IOException | SAXException | XPathExpressionException ex) {
            System.out.println("Some error occured\nDescription:\n" + ex.getMessage());
            ex.printStackTrace();
        }
    }

    // Szöveg formazása a szép megjelenés érdekében
    private static String normalizeText(String text) {
        text = text.replaceAll("\\n", ", ");
        text = text.replaceAll("\\s+", " ");
        return text;
    }
}
```