# JEGYZŐKÖNYV

## Adatkezelés XML környezetben

## Féléves feladat

## Webshop

Készítette: Kacsir András

Neptunkód: VSG9L4

Dátum: 2024.12.10

# Tartalomjegyzék

A feladat leírása: A feladatban egy webshop adatbázisát hozom létre, az alábbi jellemzőkkel:

A feladatom ötletét az Adatbázisrendszerek I. című tárgy féléves feladata adta, kisebb-nagyobb módosításokkal.

E-mailes megegyezés alapján, a feladat 5 helyett 6 egyedet tartalmaz, viszont nem mindenhol van 4 tulajdonság.

Az Uzlet egyed a webshop által bérelt üzlethelyiségeket tartalmazza, a Raktar egyed a bérelt

raktárépületeket, a Termek egyedben szerepelnek a termékek, a Rendeles a webshopon elküldött

rendeléseknek felel meg, a Vevo pedig az a személy, aki a webshoptól vásárol. A Tulajdonos egyed a

raktár, vagy üzlethelyiség tulajdonosa.

Az Uzlet egyed Uzletid tulajdonsága magától értetődik, az Elerhetoseg pedig egy összetett

tulajdonság, amely a Cim, Telefonszam, és az Email részekből épül fel.

A Raktar egyed Raktarid tulajdonsága magától értetődő, az Elerhetoseg pedig egy összetett

tulajdonság, amely a Cim és Telefonszam részekből épül fel.

A Termek egyed Termekid, Nev, és Ar tulajdonságokkal rendelkezik.

A Rendeles egyed Rendelesid, Szallitasi koltseg, Fizetesi mod, Datum tulajdonságai magától

értetődőek, a Fizetendo osszeg pedig egy származtatott tulajdonság, amely a rendelésben szereplő

termékek árának összegéből, valamint a szállítási költségből számítódik ki.

A Vevo egyed Vevoid, Nev tulajdonságokkal rendelkezik, valamint egy Szallitasi cim többértékű

tulajdonsággal.

A Tulajdonos gyenge egyed rendelkezik egy Nev tulajdonsággal, az egyedet az Uzlettel való

Tulajdona, vagy a Raktar-al való Birtokolja kapcsolata határozza meg.

Egy üzlet több termékkel is kapcsolatban állhat, valamint egy termék több üzlethez tartozhat, ezért az

Üzletben van egy N:M kapcsolat.

Egy raktár több termékkel is kapcsolatban állhat, valamint egy termék több raktárhoz tartozhat, ezért a

Raktáron van egy N:M kapcsolat.

Egy rendelésben több termék is szerepelhet, valamint egy termék több rendelés része lehet, ezért a

Rendelik a terméket egy N:M kapcsolat.

Egy vevő több rendelést is feladhat, de egy rendelést csak 1 vevő adhat fel, ezért a Feladja a rendelést

egy 1:N kapcsolat.

Egy tulajdonosnak több üzlete is lehet, de egy üzlet csak egy tulajdonoshoz tartozhat, ezért a

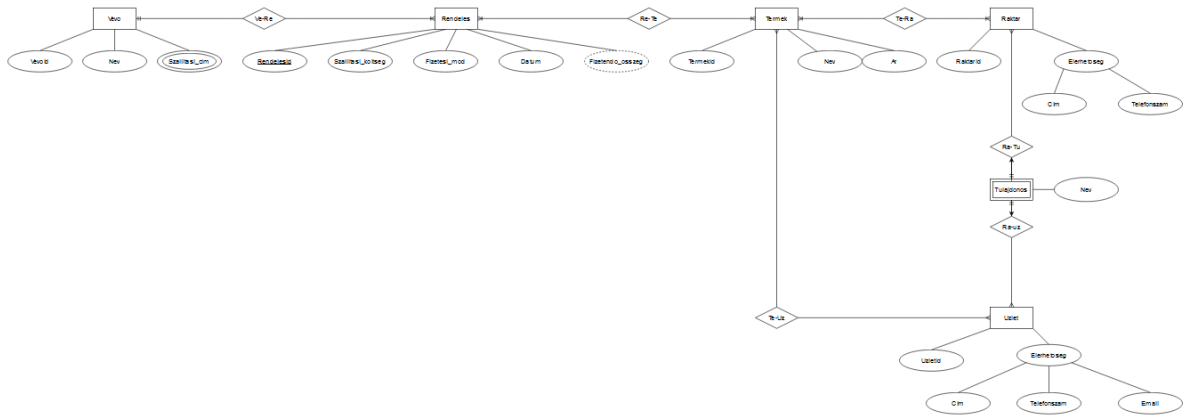
Tulajdona egy 1:N meghatározó kapcsolat.

Egy tulajdonosnak több raktára is lehet, de egy raktár csak egy tulajdonoshoz tartozhat, ezért a

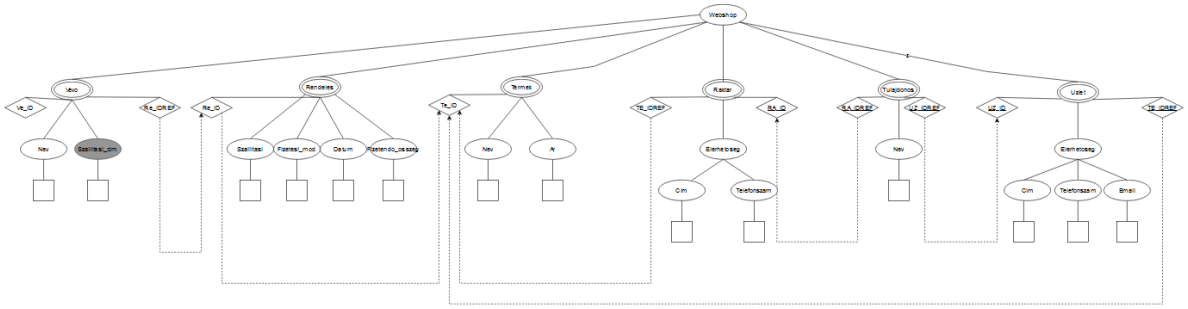
Birtokolja egy 1:N meghatározó kapcsolat.

# 1. feladat

## 1a) Az adatbázis ER modellje:



## 1b) Az adatbázis konvertálása XDM modellre:



## 1c) Az XDM modell alapján XML dokumentum készítése:

```
<?xml version="1.0" encoding="UTF-8"?>
<webshop_adatbazis xmlns:xs="http://www.w3.org/2001/XMLSchema-instance"
xs:noNamespaceSchemaLocation="XMLSchemaVSG9L4.xsd">
        <!-- Vevo -->
        <vevo vevoid="1">
                <nev>Junina Ansill</nev>
                <szallitasi_cim>42 Hanson Avenue</szallitasi_cim>
        </vevo>

        <vevo vevoid="2">
                <nev>Terrence Carlisso</nev>
                <szallitasi_cim>85908 Northport Court</szallitasi_cim>
        </vevo>

        <vevo vevoid="3">
                <nev>Stefan Roke</nev>
                <szallitasi_cim>83 Eggendart Alley</szallitasi_cim>
        </vevo>

        <vevo vevoid="4">
```

```
        <nev>Audrie Pealing</nev>
        <szallitasi_cim>7909 Rowland Trail</szallitasi_cim>
        <szallitasi_cim>71 Oriole Trai</szallitasi_cim>
</vevo>

<vevo vevoid="5">
        <nev>Elana Marousek</nev>
        <szallitasi_cim>9082 Nancy Pass</szallitasi_cim>
        <szallitasi_cim>890 Eastlawn Drive</szallitasi_cim>
</vevo>

<vevo vevoid="6">
        <nev>Tandi Poure</nev>
        <szallitasi_cim>497 Coolidge Junction</szallitasi_cim>
</vevo>

<!-- Rendeles -->

<rendeles vevoid="1" rendelesid= "1">
        <szallitasi_koltseg>4378</szallitasi_koltseg>
        <fizetesi_mod>jcb</fizetesi_mod>
        <datum>2017-10-29</datum>
        <fizetendo_osszeg>85000</fizetendo_osszeg>
</rendeles>
<rendeles vevoid="2" rendelesid= "2">
        <szallitasi_koltseg>4002</szallitasi_koltseg>
        <fizetesi_mod>jcb</fizetesi_mod>
        <datum>2017-06-04</datum>
        <fizetendo_osszeg>90000</fizetendo_osszeg>
</rendeles>
<rendeles vevoid="3" rendelesid= "3">
        <szallitasi_koltseg>6215</szallitasi_koltseg>
        <fizetesi_mod>visa-electron</fizetesi_mod>
        <datum>2017-06-26</datum>
        <fizetendo_osszeg>95000</fizetendo_osszeg>
</rendeles>
<rendeles vevoid="4" rendelesid= "4">
        <szallitasi_koltseg>5412</szallitasi_koltseg>
        <fizetesi_mod>switch</fizetesi_mod>
        <datum>2017-08-24</datum>
        <fizetendo_osszeg>100000</fizetendo_osszeg>
</rendeles>
<rendeles vevoid="5" rendelesid= "5">
        <szallitasi_koltseg>6875</szallitasi_koltseg>
        <fizetesi_mod>visa-electron</fizetesi_mod>
        <datum>2016-12-15</datum>
        <fizetendo_osszeg>110000</fizetendo_osszeg>
</rendeles>
<rendeles vevoid="6" rendelesid= "6">
        <szallitasi_koltseg>4335</szallitasi_koltseg>
        <fizetesi_mod>jcb</fizetesi_mod>
        <datum>2017-02-25</datum>
        <fizetendo_osszeg>115000</fizetendo_osszeg>
```

```
</rendeles>

<!-- Termek -->

<termek termekid="1">
        <nev>Wine - Beaujolais Villages</nev>
        <ar>80036</ar>
</termek>

<termek termekid="2">
        <nev>Shiro Miso;</nev>
        <ar>49760</ar>
</termek>

<termek termekid="3">
        <nev>Milk - 2%</nev>
        <ar>19946</ar>
</termek>

<termek termekid="4">
        <nev>'Salmon Steak</nev>
        <ar>24013</ar>
</termek>

<termek termekid="5">
        <nev>Tomatoes</nev>
        <ar>80288</ar>
</termek>

<termek termekid="6">
        <nev>'Almonds Ground Blanched',</nev>
        <ar>11229</ar>
</termek>


<!-- Raktar -->

<raktar raktarid="1">
        <elerhetoseg>
                <cim>01964 Gale Plaza</cim>
                <telefonszam>156 239 0968</telefonszam>
        </elerhetoseg>
</raktar>

<raktar raktarid="2">
        <elerhetoseg>
                <cim>4317 Buell Pass</cim>
                <telefonszam>262 138 4714</telefonszam>
        </elerhetoseg>
</raktar>

<raktar raktarid="3">
        <elerhetoseg>
```

```
                <cim>24070 Dayton Hill</cim>
                <telefonszam>834 712 9185</telefonszam>
        </elerhetoseg>
</raktar>

<raktar raktarid="4">
        <elerhetoseg>
                <cim>81299 Northport Park</cim>
                <telefonszam>771 770 1993</telefonszam>
        </elerhetoseg>
</raktar>

<raktar raktarid="5">
        <elerhetoseg>
                <cim>023 Lunder Point</cim>
                <telefonszam>919 104 8245</telefonszam>
        </elerhetoseg>
</raktar>

<raktar raktarid="6">
        <elerhetoseg>
                <cim>05 Waubesa Junction</cim>
                <telefonszam>580 230 7788</telefonszam>
        </elerhetoseg>
</raktar>

<!-- Tulajdonos -->

<tulajdonos uzletid="1" raktarid="1">
        <nev>Gill Penna</nev>
</tulajdonos>

<tulajdonos uzletid="1" raktarid="2">
        <nev>Bing Garahan</nev>
</tulajdonos>

<tulajdonos uzletid="2" raktarid="1">
        <nev>Olwen Elcoux</nev>
</tulajdonos>

<tulajdonos uzletid="3" raktarid="6">
        <nev>Brig Ollerton</nev>
</tulajdonos>

<tulajdonos uzletid="4" raktarid="4">
        <nev>Nedda Goodred</nev>
</tulajdonos>

<tulajdonos uzletid="5" raktarid="5">
        <nev>Sherilyn Stubs</nev>
</tulajdonos>

<!-- Uzlet -->
```

```
        <uzlet uzletid="1">
                <elerhetoseg>
                        <cim>65 Gerald Place</cim>
                        <telefonszam>358 639 6452'</telefonszam>
                        <email>jlosano0@posterous.com</email>
                </elerhetoseg>
        </uzlet>

        <uzlet uzletid="2">
                <elerhetoseg>
                        <cim>6650 Alpine Terrace</cim>
                        <telefonszam>449 245 9016</telefonszam>
                        <email></email>
                </elerhetoseg>
        </uzlet>

        <uzlet uzletid="3">
                <elerhetoseg>
                        <cim>314 Cardinal Point</cim>
                        <telefonszam>357 758 0118</telefonszam>
                        <email>gsanson2@drupal.org</email>
                </elerhetoseg>
        </uzlet>

        <uzlet  uzletid="4">
                <elerhetoseg>
                        <cim>7597 Thackeray Way</cim>
                        <telefonszam>333 727 0857</telefonszam>
                        <email>mcuttles3@squidoo.com</email>
                </elerhetoseg>
        </uzlet>

        <uzlet  uzletid="5">
                <elerhetoseg>
                        <cim>05038 Acker Alley</cim>
                        <telefonszam>342 796 3543</telefonszam>
                        <email>gbaukham4@networksolutions.com</email>
                </elerhetoseg>
        </uzlet>

        <uzlet  uzletid="6">
                <elerhetoseg>
                        <cim>58 Ryan Junction</cim>
                        <telefonszam>688 739 6577</telefonszam>
                        <email>okunzel5@elpais.com</email>
                </elerhetoseg>
        </uzlet>

</webshop_adatbazis>
```

## *1d) Az XML dokumentum alapján XMLSchema készítése*

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified"
attributeFormDefault="qualified">
  <!-- Simple Types -->

  <xs:simpleType name="datum_type">
    <xs:restriction base="xs:string">
      <xs:pattern value="(19|20)\d\d-(0[1-9]|1[012])-(0[1-9]|[12][0-9]|3[01])"/>
    </xs:restriction>
  </xs:simpleType>

  <xs:simpleType name="nev_type">
    <xs:restriction base="xs:string">
      <xs:pattern value="[A-Z][a-zA-Z]*( [A-Z][a-zA-Z]*)*"/>
    </xs:restriction>
  </xs:simpleType>
  <!-- Complex Types -->

  <xs:complexType name="vevo_tipus">
    <xs:sequence>
      <xs:element name="nev" type="nev_type"/>
      <xs:element name="szallitasi_cim" type="xs:string"/>
    </xs:sequence>
    <xs:attribute name="vevoid" type="xs:integer" use="required"/>
  </xs:complexType>

  <xs:complexType name="rendeles_tipus">
    <xs:sequence>
      <xs:element name="szallitasi_koltseg" type="xs:integer"/>
      <xs:element name="fizetesi_mod" type="xs:string"/>
      <xs:element name="datum" type="datum_type"/>
      <xs:element name="fizetendo_osszeg" type="xs:integer"/>
    </xs:sequence>
    <xs:attribute name="vevoid" type="xs:integer" use="required"/>
    <xs:attribute name="rendelesid" type="xs:integer" use="required"/>
  </xs:complexType>

  <xs:complexType name="termek_tipus">
    <xs:sequence>
      <xs:element name="nev" type="xs:string"/>
      <xs:element name="ar" type="xs:integer"/>
    </xs:sequence>
    <xs:attribute name="termekid" type="xs:integer" use="required"/>
  </xs:complexType>

  <xs:complexType name="raktar_tipus">
    <xs:sequence>
      <xs:element name="elerhetoseg">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="cim" type="xs:string"/>
            <xs:element name="telefonszam" type="xs:string"/>
```

```
            </xs:sequence>
          </xs:complexType>
        </xs:element>
    </xs:sequence>
    <xs:attribute name="raktarid" type="xs:integer" use="required"/>
</xs:complexType>

<xs:complexType name="tulajdonos_tipus">
    <xs:sequence>
        <xs:element name="nev" type="nev_type"/>
    </xs:sequence>
    <xs:attribute name="uzletid" type="xs:integer" use="required"/>
    <xs:attribute name="raktarid" type="xs:integer" use="required"/>
</xs:complexType>

<xs:complexType name="uzlet_tipus">
    <xs:sequence>
        <xs:element name="elerhetoseg">
            <xs:complexType>
                <xs:sequence>
                    <xs:element name="cim" type="xs:string"/>
                    <xs:element name="telefonszam" type="xs:string"/>
                    <xs:element name="email" type="xs:string"/>
                </xs:sequence>
            </xs:complexType>
        </xs:element>
    </xs:sequence>
    <xs:attribute name="uzletid" type="xs:integer" use="required"/>
</xs:complexType>
<!-- Root Element -->
<xs:element name="webshop_adatbazis">
 <xs:complexType>
    <xs:sequence>
        <xs:element name="vevo" type="vevo_tipus" maxOccurs="unbounded"/>
        <xs:element name="rendeles" type="rendeles_tipus" maxOccurs="unbounded"/>
        <xs:element name="termek" type="termek_tipus" maxOccurs="unbounded"/>
        <xs:element name="raktar" type="raktar_tipus" maxOccurs="unbounded"/>
        <xs:element name="tulajdonos" type="tulajdonos_tipus" maxOccurs="unbounded"/>
        <xs:element name="uzlet" type="uzlet_tipus" maxOccurs="unbounded"/>
    </xs:sequence>
    <!-- Keys -->
    <xs:key name="raktarKey">
        <xs:selector xpath="raktar"/>
        <xs:field xpath="@raktarid"/>
    </xs:key>
    <xs:key name="vevoKey">
        <xs:selector xpath="vevo"/>
        <xs:field xpath="@vevoid"/>
    </xs:key>
    <!-- Keyrefs -->
    <xs:keyref name="vevoRef" refer="vevoKey">
        <xs:selector xpath="rendeles"/>
        <xs:field xpath="@vevoid"/>
```

```
    </xs:keyref>
    <xs:keyref name="raktarRef" refer="raktarKey">
      <xs:selector xpath="tulajdonos"/>
      <xs:field xpath="@raktarid"/>
    </xs:keyref>
  </xs:complexType>
</xs:element>
```

# 2. feladat

## *2a) Adatolvasás:*

```
package hu.domparse.vsg9l4;

import java.io.File;

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

import org.w3c.dom.Document;
import org.w3c.dom.Element;
import org.w3c.dom.Node;
import org.w3c.dom.NodeList;

public class DomReadVSG9L4 {

        public static void main(String[] args) {
        try {
        // XML fájl betöltése
        File xmlFile = new File("XMLVSG9L4.xml.");
        DocumentBuilderFactory dbFactory = DocumentBuilderFactory.newInstance();
        DocumentBuilder dBuilder = dbFactory.newDocumentBuilder();
        Document doc = dBuilder.parse(xmlFile);
```

```
        // Normalizálás
        doc.getDocumentElement().normalize();

        // Gyökérelem neve
        System.out.println("Gyökérelem: " + doc.getDocumentElement().getNodeName());

        // Vevők kiolvasása
        NodeList vevok = doc.getElementsByTagName("vevo");
        System.out.println("\nVevők:");
        for (int i = 0; i < vevok.getLength(); i++) {
        Element vevo = (Element) vevok.item(i);
        System.out.println("  Vevő ID: " + vevo.getAttribute("vevoid"));
        System.out.println("  Név: " + vevo.getElementsByTagName("nev").item(0).getTextContent());
        NodeList cimek = vevo.getElementsByTagName("szallitasi_cim");
        for (int j = 0; j < cimek.getLength(); j++) {
                System.out.println("  Cím: " + cimek.item(j).getTextContent());
        }
        System.out.println();
        }

        // Rendelések kiolvasása
        NodeList rendelesek = doc.getElementsByTagName("rendeles");
        System.out.println("\nRendelések:");
        for (int i = 0; i < rendelesek.getLength(); i++) {
        Element rendeles = (Element) rendelesek.item(i);
        System.out.println("  Rendelés ID: " + rendeles.getAttribute("rendelesid"));
        System.out.println("  Vevő ID: " + rendeles.getAttribute("vevoid"));
        System.out.println("  Szállítási költség: " +
rendeles.getElementsByTagName("szallitasi_koltseg").item(0).getTextContent());
        System.out.println("  Fizetési mód: " +
rendeles.getElementsByTagName("fizetesi_mod").item(0).getTextContent());
```

```
        System.out.println("  Dátum: " +
rendeles.getElementsByTagName("datum").item(0).getTextContent());

        System.out.println("  Fizetendő összeg: " +
rendeles.getElementsByTagName("fizetendo_osszeg").item(0).getTextContent());

        }


        // Termékek kiolvasása

        NodeList termekek = doc.getElementsByTagName("termek");

        System.out.println("\nTermékek:");

        for (int i = 0; i < termekek.getLength(); i++) {

        Element termek = (Element) termekek.item(i);

        System.out.println("  Termék ID: " + termek.getAttribute("termekid"));

        System.out.println("  Név: " +
termek.getElementsByTagName("nev").item(0).getTextContent());

        System.out.println("  Ár: " + termek.getElementsByTagName("ar").item(0).getTextContent());

        }


        // Raktárak kiolvasása

        NodeList raktarak = doc.getElementsByTagName("raktar");

        System.out.println("\nRaktárak:");

        for (int i = 0; i < raktarak.getLength(); i++) {

        Element raktar = (Element) raktarak.item(i);

        System.out.println("  Raktár ID: " + raktar.getAttribute("raktarid"));

        Element elerhetoseg = (Element) raktar.getElementsByTagName("elerhetoseg").item(0);

        System.out.println("       Cím: " +
elerhetoseg.getElementsByTagName("cim").item(0).getTextContent());

        System.out.println("       Telefonszám: " +
elerhetoseg.getElementsByTagName("telefonszam").item(0).getTextContent());

        }


        // Tulajdonosok kiolvasása

        NodeList tulajdonosok = doc.getElementsByTagName("tulajdonos");

        System.out.println("\nTulajdonosok:");
```

```
        for (int i = 0; i < tulajdonosok.getLength(); i++) {
        Element tulajdonos = (Element) tulajdonosok.item(i);
        System.out.println("  Üzlet ID: " + tulajdonos.getAttribute("uzletid"));
        System.out.println("  Raktár ID: " + tulajdonos.getAttribute("raktarid"));
        System.out.println("  Név: " +
tulajdonos.getElementsByTagName("nev").item(0).getTextContent());
        }

        // Üzletek kiolvasása
        NodeList uzletek = doc.getElementsByTagName("uzlet");
        System.out.println("\nÜzletek:");
        for (int i = 0; i < uzletek.getLength(); i++) {
        Element uzlet = (Element) uzletek.item(i);
        System.out.println("  Üzlet ID: " + uzlet.getAttribute("uzletid"));
        Element elerhetoseg = (Element) uzlet.getElementsByTagName("elerhetoseg").item(0);
        System.out.println("       Cím: " +
elerhetoseg.getElementsByTagName("cim").item(0).getTextContent());
        System.out.println("       Telefonszám: " +
elerhetoseg.getElementsByTagName("telefonszam").item(0).getTextContent());
        Node emailNode = elerhetoseg.getElementsByTagName("email").item(0);
        System.out.println("       Email: " + (emailNode != null ? emailNode.getTextContent() : "Nincs
megadva"));
        }

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

## *2b) Adatlekérdezés:*

```
package hu.domparse.vsg9l4;
```

```
import java.io.File;

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

import org.w3c.dom.Document;
import org.w3c.dom.Element;
import org.w3c.dom.NodeList;

public class DOMQueryVSG9L4 {
	public static void main(String[] args) {
	try {
	// XML fájl betöltése
	File xmlFile = new File("XMLVSG9L4.xml");
	DocumentBuilderFactory dbFactory = DocumentBuilderFactory.newInstance();
	DocumentBuilder dBuilder = dbFactory.newDocumentBuilder();
	Document doc = dBuilder.parse(xmlFile);

	// Normalizálás
	doc.getDocumentElement().normalize();

	System.out.println("Rendelések, ahol a fizetendő összeg >= 100000:");

	// Rendelések lekérdezése
	NodeList rendelesek = doc.getElementsByTagName("rendeles");

	for (int i = 0; i < rendelesek.getLength(); i++) {
		Element rendeles = (Element) rendelesek.item(i);
		int fizetendoOsszeg =
Integer.parseInt(rendeles.getElementsByTagName("fizetendo_osszeg").item(0).getTextContent());
```

```
                if (fizetendoOsszeg >= 100000) {
                String rendelesId = rendeles.getAttribute("rendelesid");
                String vevoId = rendeles.getAttribute("vevoid");
                String fizetesiMod =
rendeles.getElementsByTagName("fizetesi_mod").item(0).getTextContent();
                String datum = rendeles.getElementsByTagName("datum").item(0).getTextContent();

                System.out.println("Rendelés ID: " + rendelesId);
                System.out.println("Vevő ID: " + vevoId);
                System.out.println("Fizetési mód: " + fizetesiMod);
                System.out.println("Dátum: " + datum);
                System.out.println("Fizetendő összeg: " + fizetendoOsszeg);
                System.out.println();
                }
        }

        System.out.println("Termékek listája:");

        // Termékek lekérdezése
        NodeList termekek = doc.getElementsByTagName("termek");

        for (int i = 0; i < termekek.getLength(); i++) {
                Element termek = (Element) termekek.item(i);
                String termekId = termek.getAttribute("termekid");
                String nev = termek.getElementsByTagName("nev").item(0).getTextContent();
                int ar =
Integer.parseInt(termek.getElementsByTagName("ar").item(0).getTextContent());

                System.out.println("Termék ID: " + termekId);
                System.out.println("Név: " + nev);
                System.out.println("Ár: " + ar);
```

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

        System.out.println("Rendelések 2017-06-01 előtt:");
        for (int i = 0; i < rendelesek.getLength(); i++) {
            Element rendeles = (Element) rendelesek.item(i);
            String datum = rendeles.getElementsByTagName("datum").item(0).getTextContent();

            if (datum.compareTo("2017-06-01") < 0) {
            String vevoId = rendeles.getAttribute("vevoid");
            String rendelesId = rendeles.getAttribute("rendelesid");
            System.out.println("  Rendelés ID: " + rendelesId + ", Vevő ID: " + vevoId + ", Dátum: "
+ datum);
            }
        }

        System.out.println("\nVevők több mint 1 szállítási címmel:");
        NodeList vevok = doc.getElementsByTagName("vevo");
        for (int i = 0; i < vevok.getLength(); i++) {
            Element vevo = (Element) vevok.item(i);
            NodeList szallitasiCimek = vevo.getElementsByTagName("szallitasi_cim");

            if (szallitasiCimek.getLength() > 1) {
            String nev = vevo.getElementsByTagName("nev").item(0).getTextContent();
            System.out.println("  Név: " + nev);
            }
        }

        System.out.println("\nTermékek, amelyek ára több mint 20000:");
        for (int i = 0; i < termekek.getLength(); i++) {
            Element termek = (Element) termekek.item(i);
```

```
                int ar =
Integer.parseInt(termek.getElementsByTagName("ar").item(0).getTextContent());

                if (ar > 20000) {
                String nev = termek.getElementsByTagName("nev").item(0).getTextContent();
                System.out.println("  Termék neve: " + nev + ", Ár: " + ar);
                }
        }

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

### *2c) Adatmódosítás:*

```
package hu.domparse.vsg9l4;

import org.w3c.dom.*;
import javax.xml.parsers.*;
import javax.xml.transform.*;
import javax.xml.transform.dom.DOMSource;
import javax.xml.transform.stream.StreamResult;
import java.io.File;

public class DOMModifyVSG9L4 {
        public static void main(String[] args) {
        try {
        // XML fájl betöltése
        File xmlFile = new File("XMLVSG9L4.xml");
```

```
DocumentBuilderFactory dbFactory = DocumentBuilderFactory.newInstance();
DocumentBuilder dBuilder = dbFactory.newDocumentBuilder();
Document doc = dBuilder.parse(xmlFile);
doc.getDocumentElement().normalize();

// Új termék hozzáadása
Element newTermek = doc.createElement("termek");
newTermek.setAttribute("termekid", "7");

Element nev = doc.createElement("nev");
nev.appendChild(doc.createTextNode("Steak"));
newTermek.appendChild(nev);

Element ar = doc.createElement("ar");
ar.appendChild(doc.createTextNode("20000"));
newTermek.appendChild(ar);

// Hozzáadás a gyökérelemhez
doc.getDocumentElement().appendChild(newTermek);
System.out.println("Új termék hozzáadva!");

// Új vevő hozzáadása
Element newVevo = doc.createElement("vevo");
newVevo.setAttribute("vevoid", "7");

Element vevoNev = doc.createElement("nev");
vevoNev.appendChild(doc.createTextNode("Steve Brad"));
newVevo.appendChild(vevoNev);

Element szallitasiCim = doc.createElement("szallitasi_cim");
szallitasiCim.appendChild(doc.createTextNode("123 Avenue"));
```

```
        newVevo.appendChild(szallitasiCim);

        // Hozzáadás a gyökérelemhez
        doc.getDocumentElement().appendChild(newVevo);
        System.out.println("Új vevő hozzáadva!");

        // Egy üzlet ID módosítása
        NodeList uzletek = doc.getElementsByTagName("uzlet");
        for (int i = 0; i < uzletek.getLength(); i++) {
        Element uzlet = (Element) uzletek.item(i);
        if (uzlet.getAttribute("uzletid").equals("1")) { // Eredeti ID
                uzlet.setAttribute("uzletid", "7"); // Új ID
                System.out.println("Az üzlet ID módosítva: 1 -> 10");
                break;
                }
        }
        // XML fájl mentése
        TransformerFactory transformerFactory = TransformerFactory.newInstance();
        Transformer transformer = transformerFactory.newTransformer();
        DOMSource source = new DOMSource(doc);
        StreamResult result = new StreamResult(new File("XMLVSG9L4_modified.xml"));
        transformer.transform(source, result);

        System.out.println("Az XML fájl sikeresen módosítva!");
        } catch (Exception e) {
        e.printStackTrace();
        }
        }
}
```

### *2d) Adatírás*

```
package hu.domparse.vsg9l4;

import org.w3c.dom.*;

import javax.xml.parsers.*;

import javax.xml.transform.*;

import javax.xml.transform.dom.DOMSource;

import javax.xml.transform.stream.StreamResult;

import java.io.File;


public class DOMWriteVSG9L4 {

        public static void main(String[] args) {

        try {

        // Az eredeti XML fájl betöltése

        File inputFile = new File("XMLVSG9L4.xml");


        // DOM parser inicializálása

        DocumentBuilderFactory dbFactory = DocumentBuilderFactory.newInstance();

        DocumentBuilder dBuilder = dbFactory.newDocumentBuilder();


        // XML fájl betöltése Document objektumba

        Document doc = dBuilder.parse(inputFile);


        // Gyökérelem kiírása

        doc.getDocumentElement().normalize();

        System.out.println("Gyökérelem: " + doc.getDocumentElement().getNodeName());


        // Fa struktúra kiíratása a konzolra

        printNode(doc.getDocumentElement(), 0);


        // Új fájlba írás (XMLNeptunkod1.xml)

        TransformerFactory transformerFactory = TransformerFactory.newInstance();
```

```
Transformer transformer = transformerFactory.newTransformer();

// Indítási paraméterek beállítása (pl. behúzás)
transformer.setOutputProperty(OutputKeys.INDENT, "yes");
transformer.setOutputProperty("{http://xml.apache.org/xslt}indent-amount", "2");

// Dokumentum mentése
DOMSource source = new DOMSource(doc);
StreamResult result = new StreamResult(new File("XMLNeptunkod1.xml"));
transformer.transform(source, result);

System.out.println("Az XML fájlt sikeresen mentettük: XMLNeptunkod1.xml");

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

/**
* Rekurzív metódus a fa struktúra kiírására a konzolra.
*
* @param node   Az aktuális elem vagy attribútum.
* @param indent Behúzások száma (vizuális megjelenítés).
*/
private static void printNode(Node node, int indent) {
// Behúzás készítése
for (int i = 0; i < indent; i++) {
System.out.print("  ");
}

// Aktuális csomópont neve és értéke
```

```
if (node.getNodeType() == Node.ELEMENT_NODE) {
System.out.print("<" + node.getNodeName());

// Attribútumok kiírása
NamedNodeMap attributes = node.getAttributes();
for (int i = 0; i < attributes.getLength(); i++) {
Node attribute = attributes.item(i);
System.out.print(" " + attribute.getNodeName() + "=\"" + attribute.getNodeValue() + "\"");
}
System.out.println(">");

// Gyermek csomópontok rekurzív feldolgozása
NodeList children = node.getChildNodes();
for (int i = 0; i < children.getLength(); i++) {
printNode(children.item(i), indent + 1);
}

// Záróelem kiírása
for (int i = 0; i < indent; i++) {
System.out.print("  ");
}
System.out.println("</" + node.getNodeName() + ">");

} else if (node.getNodeType() == Node.TEXT_NODE) {
// Szöveges csomópontok kezelése
String textContent = node.getTextContent().trim();
if (!textContent.isEmpty()) {
System.out.println(textContent);
}
}
}
```

```
}
```