JEGYZŐKÖNYV

Adatkezelés XML környezetben

Féléves feladat

Készítette: **Makó Péter**

Neptunkód: **E8HZ8D**

A feladat leírása:

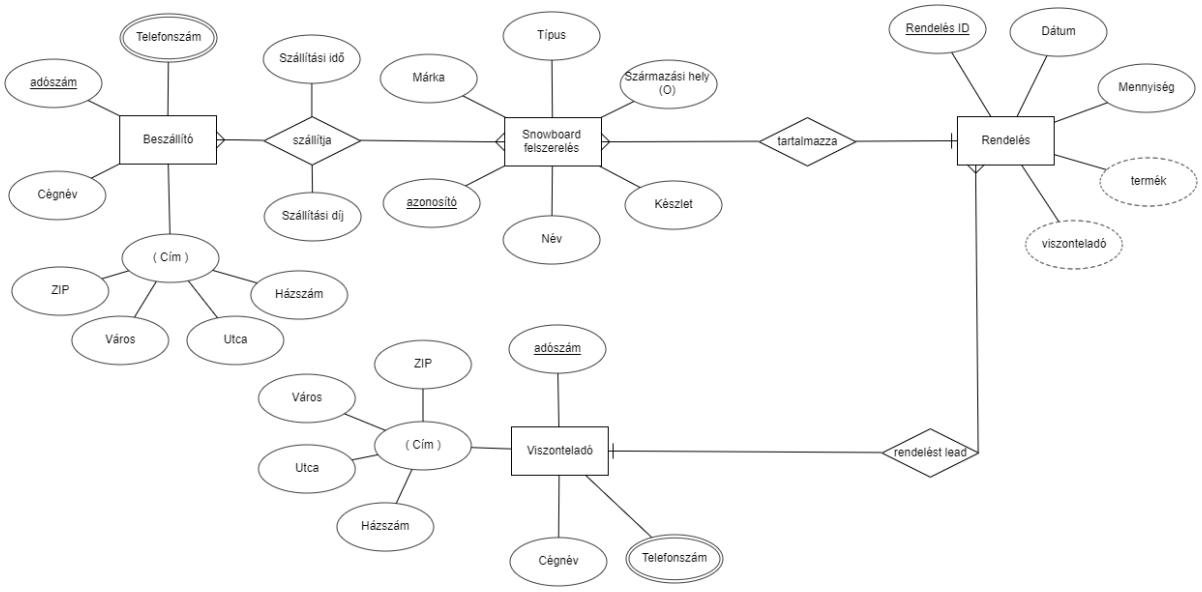
A féléves feladatom témájaként bizonyos snowboard felszerelések viszonteladói lerendeléseit prezentáló rendszert terveztem. A rendszer tartalmaz adatokat a felszerelésekről, beszállítókról, a viszonteladókról és a rendelésekről.

A rendszer alapját az adja, hogy maga a snowboardozás nem egy olcsó sport, kifejezetten széleskörű a kínálat, mind a termékek típusát tekintve, mind a termékeket gyártó cégek esetében. Az imént felsoroltak értelmében egy kis viszonteladó boltjának nincs akkora tőkéje, hogy nagy mennyiségű árut felvásároljon, majd az üzletben elhelyezzen és eladásra kínáljon. Éppen ezért csak bizonyos megrendelések esetén fogja kérni a beszállítót, hogy bizonyos termékből szállítson neki, mert így biztos a profit, és csökken a felesleges kiköltekezés veszélye.

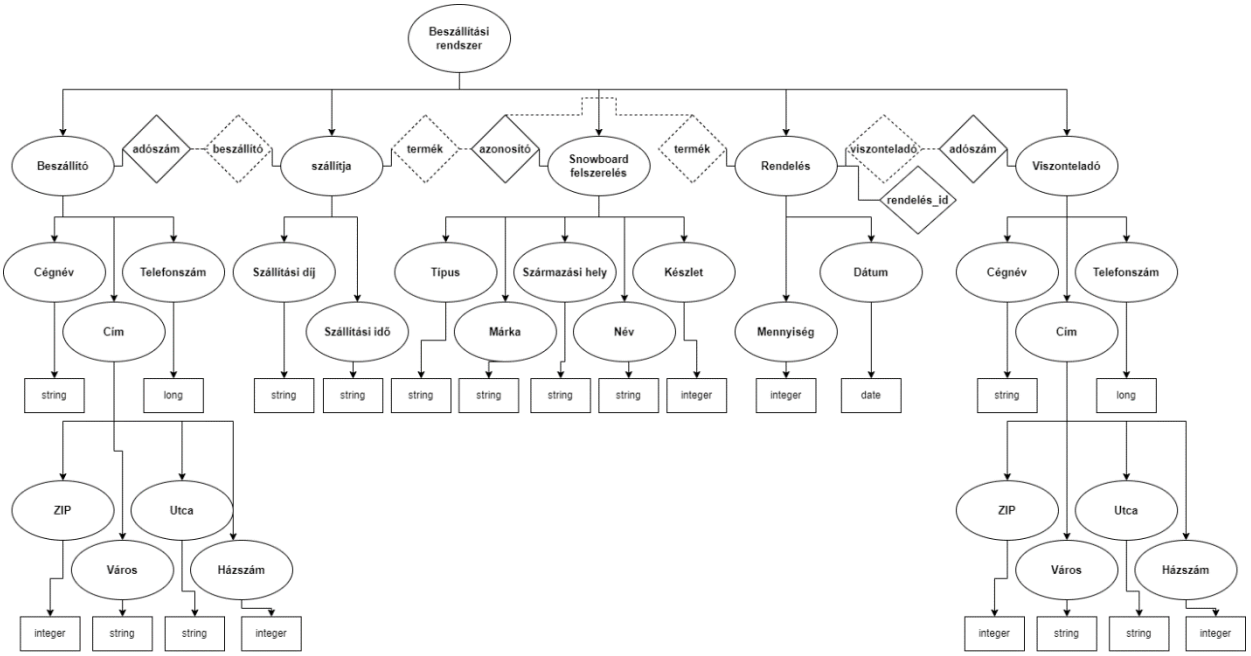
A rendszerben a következő módon fog történni az adatok tároldása. Ugyanazon felszereléshez tartozhat akár több beszállító is, illetve ebből következik, hogy egy beszállító több különböző terméket beszállítását is végzi. Egy rendelés tartalmazni fogja a snowboard felszerelés ID-ját és a viszonteladó adószámát, mint két kulcsadatot. Minden adott felszereléshez tartozik egy vagy több beszállító, amik közül lehet választani majd, attól függően, hogy melyik megfelelő szállítási díj és idő szempontjából. Minden rendeléshez megadható a mennyiség és alapértelmezett módon minden rendelés esetében 1 db terméket vesz alapul a rendszer. Egy rendelés egyszerre egy viszonteladóhoz tartozhat, illetve minden rendelés dátumát is számon fogja tartani a rendszer.

1. feladat

1a) Az adatbázis ER modell:



1b) Az adatbázis konvertálása XDM modellre:



1c) Az XDM modell alapján XML dokumentum készítése:

```
<?xml version="1.0" encoding="UTF-8"?>
<beszallitasi_rendszer xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="./XMLSchemaE8HZ8D.xsd">

    <beszallito adoszam="11111111-2-33">
        <cegnev>SnowBoardStuff Bt.</cegnev>
        <cim>
            <zip>3535</zip>
            <varos>Miskolc</varos>
            <utca>Petofi ut</utca>
            <hazszam>13</hazszam>
        </cim>
        <telefonszam>06706027422</telefonszam>
        <telefonszam>06306027422</telefonszam>
    </beszallito>
    <beszallito adoszam="11111111-3-44">
        <cegnev>Kerekgyarto es Tarsa Zrt.</cegnev>
        <cim>
            <zip>2344</zip>
            <varos>Domsod</varos>
            <utca>Kossuth Lajos utca</utca>
            <hazszam>2</hazszam>
        </cim>
        <telefonszam>06206662710</telefonszam>
    </beszallito>
    <beszallito adoszam="11111111-3-35">
        <cegnev>We Sell SnwBrds Zrt.</cegnev>
        <cim>
            <zip>3341</zip>
            <varos>Szucs</varos>
            <utca>Kiraly utca</utca>
            <hazszam>54</hazszam>
        </cim>
        <telefonszam>06708899112</telefonszam>
    </beszallito>

    <szallitja beszallito="11111111-2-33" termek="111222333">
        <szallitasi_ido>3 nap</szallitasi_ido>
```

```
    <szallitasi_dij>3000 Ft</szallitasi_dij>
</szallitja>
<szallitja beszallito="11111111-2-33" termek="111222444">
    <szallitasi_ido>2 nap</szallitasi_ido>
    <szallitasi_dij>4000 Ft</szallitasi_dij>
</szallitja>
<szallitja beszallito="11111111-3-35" termek="111222555">
    <szallitasi_ido>5 nap</szallitasi_ido>
    <szallitasi_dij>1000 Ft</szallitasi_dij>
</szallitja>

<snowboard_felszereles azonosito="111222333">
    <tipus>Deszka</tipus>
    <marka>Nitro</marka>
    <nev>Team 159</nev>
    <keszlet>24</keszlet>
    <szarmazasi_hely>Ausztria</szarmazasi_hely>
    <ar>60000</ar>
</snowboard_felszereles>
<snowboard_felszereles azonosito="111222444">
    <tipus>Bakancs</tipus>
    <marka>Burton</marka>
    <nev>Moto BOA</nev>
    <keszlet>12</keszlet>
    <ar>50000</ar>
</snowboard_felszereles>
<snowboard_felszereles azonosito="111222555">
    <tipus>Kotes</tipus>
    <marka>Ride</marka>
    <nev>Rodeo</nev>
    <keszlet>8</keszlet>
    <szarmazasi_hely>USA</szarmazasi_hely>
    <ar>49000</ar>
</snowboard_felszereles>

<viszontelado adoszam="11111111-9-13">
    <cegnev>Snowboard Shop</cegnev>
    <cim>
        <zip>3508</zip>
        <varos>Miskolc</varos>
        <utca>Testvervarosok utja</utca>
        <hazszam>22</hazszam>
    </cim>
    <telefonszam>06706110100</telefonszam>
    <telefonszam>06206110100</telefonszam>
</viszontelado>
<viszontelado adoszam="11112222-2-33">
    <cegnev>Snowboards and CO.</cegnev>
    <cim>
        <zip>3521</zip>
        <varos>Miskolc</varos>
        <utca>Arpad ut</utca>
        <hazszam>56</hazszam>
    </cim>
    <telefonszam>06709856734</telefonszam>
</viszontelado>
<viszontelado adoszam="11113333-2-33">
    <cegnev>SnowboardStuff</cegnev>
    <cim>
        <zip>3537</zip>
        <varos>Miskolc</varos>
        <utca>Szechenyi utca</utca>
        <hazszam>10</hazszam>
    </cim>
    <telefonszam>06208417632</telefonszam>
</viszontelado>
```

```
    <rendeles rendeles_id="aaa111" viszontelado="11111111-9-13"
termek="111222333">
        <datum>2021-09-15</datum>
        <mennyiseg>1</mennyiseg>
    </rendeles>
    <rendeles rendeles_id="aaa222" viszontelado="11112222-2-33"
termek="111222444">
        <datum>2021-10-22</datum>
        <mennyiseg>2</mennyiseg>
    </rendeles>
    <rendeles rendeles_id="aaa333" viszontelado="11113333-2-33"
termek="111222555">
        <datum>2021-11-09</datum>
        <mennyiseg>3</mennyiseg>
    </rendeles>

</beszallitasi_rendszer>
```

1d) Az XML dokumentum alapján XMLSchema készítése:

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

    <xs:element name="beszallitasi_rendszer"
type="beszallitasi_rendszerTipus" />

    <xs:complexType name="beszallitasi_rendszerTipus">
        <xs:sequence>
            <xs:element name="beszallito" type="beszallitoTipus"
maxOccurs="unbounded" />
            <xs:element name="szallitja" type="szallitjaTipus"
maxOccurs="unbounded" />
            <xs:element name="snowboard_felszereles"
type="snowboard_felszerelesTipus" maxOccurs="unbounded" />
            <xs:element name="viszontelado" type="viszonteladoTipus"
maxOccurs="unbounded" />
            <xs:element name="rendeles" type="rendelesTipus"
maxOccurs="unbounded" />
        </xs:sequence>
    </xs:complexType>

    <xs:complexType name="beszallitoTipus">
        <xs:sequence>
            <xs:element name="cegnev" type="xs:string" />
            <xs:element name="cim" type="cimTipus" />
            <xs:element name="telefonszam" type="telefonszamTipus"
maxOccurs="unbounded" />
        </xs:sequence>
        <xs:attribute name="adoszam" type="adoszamTipus" use="required" />
    </xs:complexType>

    <xs:complexType name="szallitjaTipus">
        <xs:sequence>
            <xs:element name="szallitasi_ido" type="xs:string" />
            <xs:element name="szallitasi_dij" type="xs:string" />
        </xs:sequence>
        <xs:attribute name="beszallito" type="adoszamTipus" use="required" />
        <xs:attribute name="termek" type="azonositoTipus" use="required" />
    </xs:complexType>

    <xs:complexType name="snowboard_felszerelesTipus">
        <xs:sequence>
            <xs:element name="tipus" type="xs:string" />
            <xs:element name="marka" type="xs:string" />
            <xs:element name="nev" type="xs:string" />
            <xs:element name="keszlet" type="xs:unsignedByte" />
```

```
            <xs:element minOccurs="0" name="szarmazasi_hely" type="xs:string"
/>
            <xs:element name="ar" type="xs:integer" />
        </xs:sequence>
        <xs:attribute name="azonosito" type="azonositoTipus" use="required"
/>
    </xs:complexType>

    <xs:complexType name="viszonteladoTipus">
        <xs:sequence>
            <xs:element name="cegnev" type="xs:string" />
            <xs:element name="cim" type="cimTipus" />
            <xs:element name="telefonszam" type="telefonszamTipus"
maxOccurs="unbounded" />
        </xs:sequence>
        <xs:attribute name="adoszam" type="adoszamTipus" use="required" />
    </xs:complexType>

    <xs:complexType name="rendelesTipus">
        <xs:sequence>
            <xs:element name="datum" type="xs:date" />
            <xs:element name="mennyiseg" type="xs:integer" />
        </xs:sequence>
        <xs:attribute name="rendeles_id" type="idTipus" use="required" />
        <xs:attribute name="viszontelado" type="adoszamTipus" use="required"
/>
        <xs:attribute name="termek" type="azonositoTipus" use="required" />
    </xs:complexType>

    <xs:simpleType name="adoszamTipus">
        <xs:restriction base="xs:string">
            <xs:pattern value="[0-9]{8}[-]{1}[0-9]{1}[-]{1}[0-9]{2}" />
        </xs:restriction>
    </xs:simpleType>

    <xs:simpleType name="azonositoTipus">
        <xs:restriction base="xs:string">
            <xs:pattern value="[0-9]{9}" />
        </xs:restriction>
    </xs:simpleType>

    <xs:simpleType name="idTipus">
        <xs:restriction base="xs:string">
            <xs:pattern value="[a-z]{3}[0-9]{3}" />
        </xs:restriction>
    </xs:simpleType>

    <xs:complexType name="cimTipus">
        <xs:sequence>
            <xs:element name="zip" type="zipTipus" />
            <xs:element name="varos" type="xs:string" />
            <xs:element name="utca" type="xs:string" />
            <xs:element name="hazszam" type="hazszamTipus" />
        </xs:sequence>
    </xs:complexType>

    <xs:simpleType name="zipTipus">
        <xs:restriction base="xs:integer">
            <xs:minInclusive value="1000" />
            <xs:maxInclusive value="9999" />
        </xs:restriction>
    </xs:simpleType>

    <xs:simpleType name="hazszamTipus">
        <xs:restriction base="xs:integer">
            <xs:minInclusive value="1" />
        </xs:restriction>
```

```
    </xs:simpleType>

    <xs:simpleType name="telefonszamTipus">
        <xs:restriction base="xs:unsignedLong">
            <xs:pattern value="[0-9]{11}" />
        </xs:restriction>
    </xs:simpleType>

 </xs:schema>
```

2. feladat

2a) adatolvasás – DOMReadE8HZ8D

```
package hu.domparse.e8hz8d;

import java.io.File;
import javax.xml.parsers.DocumentBuilderFactory;
import javax.xml.parsers.DocumentBuilder;

import org.w3c.dom.Attr;
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;

//This class goes through any xml file using a recursive function. Idea and
implementation exclusively by: Peter Mako, E8HZ8D.
public class DOMReadE8HZ8D {

      public static void main (String[] args) {

            try {
                  File inputFile = new File("../XMLE8HZ8D.xml");
                  DocumentBuilderFactory dbFactory =
DocumentBuilderFactory.newInstance();
                  DocumentBuilder dBuilder = dbFactory.newDocumentBuilder();
                  Document doc = dBuilder.parse(inputFile);
                  doc.getDocumentElement().normalize();

                  System.out.println("Root element: " +
doc.getDocumentElement().getNodeName());
                  System.out.println("----------------------------------
-----------------");
                  printNodes(doc.getDocumentElement().getChildNodes(), 0);

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

      private static void printNodes (NodeList nodes, int depth) {
            for (int index = 0; index < nodes.getLength(); index++) {
                  if (nodes.item(index).getNodeType() == Node.ELEMENT_NODE) {
                        Element eElement = (Element) nodes.item(index);
                        String eName = eElement.getTagName().substring(0,
1).toUpperCase() + eElement.getTagName().substring(1);
                        String indent = "";

                        for (int depthIndex = 0; depthIndex<depth;
depthIndex++) {
                              indent += "\t";
                        }

                        if (eElement.getChildNodes().getLength() > 1) {
```

```
                              System.out.println(indent + eName + ": ");
                        }
                        else {
                              System.out.println(indent + eName + ": " +
eElement.getTextContent());
                        }

                        checkForAttributes(eElement.getAttributes(),
indent+"\t");
                        printNodes(nodes.item(index).getChildNodes(),
depth+1);
                  }
            }
      }

      private static void checkForAttributes (NamedNodeMap attributes, String
indent) {
            for (int index = 0; index<attributes.getLength(); index++) {
                  Attr attribute = (Attr) attributes.item(index);
                  String aName = attribute.getNodeName().substring(0,
1).toUpperCase() + attribute.getNodeName().substring(1);
                  System.out.println(indent + aName + ": " +
attribute.getNodeValue());
            }
      }

 }
```

2b) adatmódosítás – DOMModifyE8HZ8D

```
package hu.domparse.e8hz8d;

import java.io.File;
import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.DocumentBuilderFactory;
import javax.xml.transform.OutputKeys;
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.Element;
import org.w3c.dom.NamedNodeMap;
import org.w3c.dom.Node;
import org.w3c.dom.NodeList;
import org.w3c.dom.html.HTMLObjectElement;

public class DOMModifyE8HZ8D {
      public static void main(String argv[]) {

            try {
                  File inputFile = new File("../XMLE8HZ8D.xml");
                  DocumentBuilderFactory dbFactory =
DocumentBuilderFactory.newInstance();
                  DocumentBuilder dBuilder = dbFactory.newDocumentBuilder();
                  Document doc = dBuilder.parse(inputFile);
                  doc.getDocumentElement().normalize();

                  NodeList equipments =
doc.getElementsByTagName("snowboard_felszereles");
                  NodeList suppliers =
doc.getElementsByTagName("beszallito");
                  NodeList deliveries =
doc.getElementsByTagName("szallitja");
                  NodeList orders = doc.getElementsByTagName("rendeles");
                  NodeList resellers =
doc.getElementsByTagName("viszontelado");
```

```
                  //Modification no.1: SnowboardStuff viszonteladó
címváltozása
                  for (int rIndex = 0; rIndex < resellers.getLength();
rIndex++) {
                        Node reseller = resellers.item(rIndex);

                        if (reseller.getNodeType() == Node.ELEMENT_NODE) {
                              Element rElement = (Element) reseller;


      if("SnowboardStuff".equals(rElement.getElementsByTagName("cegnev").item
(0).getTextContent())) {
                                    NodeList address =
rElement.getElementsByTagName("cim").item(0).getChildNodes();

                                    for (int aIndex = 0; aIndex <
address.getLength(); aIndex++) {
                                          Node addressPart =
address.item(aIndex);

                                          switch(addressPart.getNodeName()) {
                                          case "zip":

      addressPart.setTextContent("3522");
                                                break;
                                          case "utca":

      addressPart.setTextContent("Karolyi utca");
                                                break;
                                          case "hazszam":

      addressPart.setTextContent("35");
                                                break;
                                          default:
                                          }
                                    }
                              }
                        }

                  //Modification no.2: Snowboard Shop "aaa111" rendelésének
változtatása, 2 db Ride Rodeo kötésre
                  for (int oIndex = 0; oIndex < orders.getLength(); oIndex++)
{
                        Node order = orders.item(oIndex);

                        if (order.getNodeType() == Node.ELEMENT_NODE) {
                              Element oElement = (Element) order;

                              if
("aaa111".equals(oElement.getAttribute("rendeles_id"))) {
                                    for (int rIndex = 0; rIndex <
resellers.getLength(); rIndex++) {
                                          Node reseller =
resellers.item(rIndex);

                                          if (reseller.getNodeType() ==
Node.ELEMENT_NODE) {
                                                Element rElement = (Element)
reseller;


      if(rElement.getAttribute("adoszam").equals(oElement.getAttribute("viszo
ntelado"))) {
                                                      for (int eIndex = 0;
eIndex < equipments.getLength(); eIndex++) {
```

```
                                                        Node equipment =
equipments.item(eIndex);

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


      if(eElement.getElementsByTagName("marka").item(0).getTextContent().equa
ls("Ride") &&

      eElement.getElementsByTagName("nev").item(0).getTextContent().equals("R
odeo")){


      oElement.setAttribute("termek", eElement.getAttribute("azonosito"));

      oElement.getElementsByTagName("mennyiseg").item(0).setTextContent("2");
                                                              }


      if(eElement.getAttribute("azonosito").equals((oElement).getAttribute("t
ermek"))) {

                                                              }
                                                        }
                                                  }
                                            }
                                      }
                                }
                          }
                    }

                    // write to file
                    TransformerFactory transformerFactory =
TransformerFactory.newInstance();
                    Transformer transformer =
transformerFactory.newTransformer();
                    transformer.setOutputProperty(OutputKeys.INDENT, "no");
                    transformer.setOutputProperty(OutputKeys.METHOD, "xml");

      transformer.setOutputProperty("{http://xml.apache.org/xslt}indent-
amount", "4");
                    DOMSource source = new DOMSource(doc);
                    StreamResult result = new StreamResult(new
File("../XMLE8HZ8D_Modified.xml"));
                    transformer.transform(source, result);

                    // Output to console for testing
                    StreamResult consoleResult = new StreamResult(System.out);
                    transformer.transform(source, consoleResult);
              } catch (Exception e) {
                    e.printStackTrace();
              }
      }
}
```

3c) adatlekérdezés – DOMQueryE8HZ8D

```
package hu.domparse.e8hz8d;

import javax.xml.parsers.DocumentBuilderFactory;
import javax.xml.parsers.DocumentBuilder;
import org.w3c.dom.Document;
import org.w3c.dom.NodeList;
```

```
import org.w3c.dom.Node;
import org.w3c.dom.Element;
import java.io.File;

public class DOMQueryE8HZ8D {

	public static void main(String argv[]) {

		try {
			File inputFile = new File("../XMLE8HZ8D.xml");
			DocumentBuilderFactory dbFactory =
DocumentBuilderFactory.newInstance();
			DocumentBuilder dBuilder = dbFactory.newDocumentBuilder();
			Document doc = dBuilder.parse(inputFile);
			doc.getDocumentElement().normalize();
			System.out.println("-----------------------------------------
-----------------------------------------");

			NodeList equipments =
doc.getElementsByTagName("snowboard_felszereles");
			NodeList suppliers =
doc.getElementsByTagName("beszallito");
			NodeList deliveries =
doc.getElementsByTagName("szallitja");
			NodeList orders = doc.getElementsByTagName("rendeles");
			NodeList resellers =
doc.getElementsByTagName("viszontelado");

			//Query no.1: Termékekhez tartozó beszállítók
			System.out.println("Termékekhez tartozó beszállítók:\n");

			for (int eIndex = 0; eIndex < equipments.getLength();
eIndex++) {
				Node equipment = equipments.item(eIndex);

				if (equipment.getNodeType() == Node.ELEMENT_NODE) {
					Element eElement = (Element) equipment;
					System.out.println("Felszerelés:\n "
							+ eElement
							.getElementsByTagName("marka")
							.item(0)
							.getTextContent() + " "
							+ eElement
							.getElementsByTagName("nev")
							.item(0)
							.getTextContent() + ": ");

					for (int dIndex = 0; dIndex <
deliveries.getLength(); dIndex++) {
						Node delivery = deliveries.item(dIndex);

						if (delivery.getNodeType() ==
Node.ELEMENT_NODE) {
							Element dElement = (Element)
delivery;

							for (int sIndex = 0; sIndex <
suppliers.getLength(); sIndex++) {
								Node supplier =
suppliers.item(sIndex);

								if (supplier.getNodeType() ==
Node.ELEMENT_NODE) {
									Element sElement =
(Element) supplier;
```

```
                                                        if
(eElement.getAttribute("azonosito").equals(dElement.getAttribute("termek"))
&&

      sElement.getAttribute("adoszam").equals(dElement.getAttribute("beszalli
to"))) {


      System.out.println(" -> Beszállító: "
                                                                              +
sElement

      .getElementsByTagName("cegnev")

      .item(0)

      .getTextContent()
                                                                              +
"\n");
                                                        }
                                                    }
                                                }
                                            }
                                        }
                                    }
                                }

                                //Query no.2: 60 ezer forintnál olcsóbb termékek
                                System.out.println("----------------------------------------
----------------------------------------");
                                System.out.println("60 ezer forintnál olcsóbb
termékek:\n");

                                for (int eIndex = 0; eIndex < equipments.getLength();
eIndex++) {
                                    Node equipment = equipments.item(eIndex);

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

                                        if
(Double.valueOf(eElement.getElementsByTagName("ar").item(0).getTextContent())
<60000) {
                                            System.out.println("Felszerelés:\n "
                                                    + eElement

      .getElementsByTagName("marka")
                                                    .item(0)
                                                    .getTextContent() + " "
                                                    + eElement
                                                    .getElementsByTagName("nev")
                                                    .item(0)
                                                    .getTextContent() + ": ");
                                            System.out.println("Ár:\n "
                                                    + eElement
                                                    .getElementsByTagName("ar")
                                                    .item(0)
                                                    .getTextContent()
                                                    + "\n");
                                        }
                                    }
                                }

                                //Query no.3: 60 ezer forintnál drágább rendelésekhez
tartozó viszonteladó és termék(ek)
                                System.out.println("----------------------------------------
----------------------------------------");
```

```
                  System.out.println("60 ezer forintnál drágább rendelésekhez
tartozó viszonteladó és termék(ek):\n");

                  for (int oIndex = 0; oIndex < orders.getLength(); oIndex++)
{
                        Node order = orders.item(oIndex);

                        if (order.getNodeType() == Node.ELEMENT_NODE) {
                              Element oElement = (Element) order;

                              for (int eIndex = 0; eIndex <
equipments.getLength(); eIndex++) {
                                    Node equipment = equipments.item(eIndex);

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

                                          for (int rIndex = 0; rIndex <
resellers.getLength(); rIndex++) {
                                                Node reseller =
resellers.item(rIndex);

                                                if (reseller.getNodeType() ==
Node.ELEMENT_NODE) {
                                                      Element rElement =
(Element) reseller;

                                                if
(oElement.getAttribute("termek").equals(eElement.getAttribute("azonosito"))
&&
      oElement.getAttribute("viszonteladó").equals(rElement.getAttribute("ado
szam"))) {

                                                      double  oSize =
Double.valueOf(oElement.getElementsByTagName("mennyiseg").item(0).getTextCont
ent());
                                                      double ePrice =
Double.valueOf(eElement.getElementsByTagName("ar").item(0).getTextContent());

                                                      if (ePrice *
oSize > 60000) {
      System.out.println("Rendelés (no. "
      + oElement
      .getAttribute("rendeles_id")
      + ") ("
      + ePrice*oSize
      +" Ft)");
      System.out.println(" -> Viszonteladó: "
      + rElement
      .getElementsByTagName("cegnev")
      .item(0)
      .getTextContent());
```

```
        System.out.println(" -> Felszerelés: "
        + oElement
        .getElementsByTagName("mennyiseg")
        .item(0)
        .getTextContent() + "x "
        + eElement
        .getElementsByTagName("marka")
        .item(0)
        .getTextContent() + " "
        + eElement
        .getElementsByTagName("nev")
        .item(0)
        .getTextContent()
        + "\n");
                                                                    }
                                                                }
                                                            }
                                                        }
                                                    }
                                                }
                                            }
                                        }
                                    } catch (Exception e) {
                                        e.printStackTrace();
                                    }
                                }
                            }
```