Jegyzőkönyv

Adatkezelés XML környezetben

Féléves feladat

Készítette: Czikó Tivadar

Neptunkód: O2IXLB

1. **Feladat**

**Egyedek:**

* Vonat
* Mozdonyvezető
* kalauz
* Menetrend
* Útvonal
* Megállót

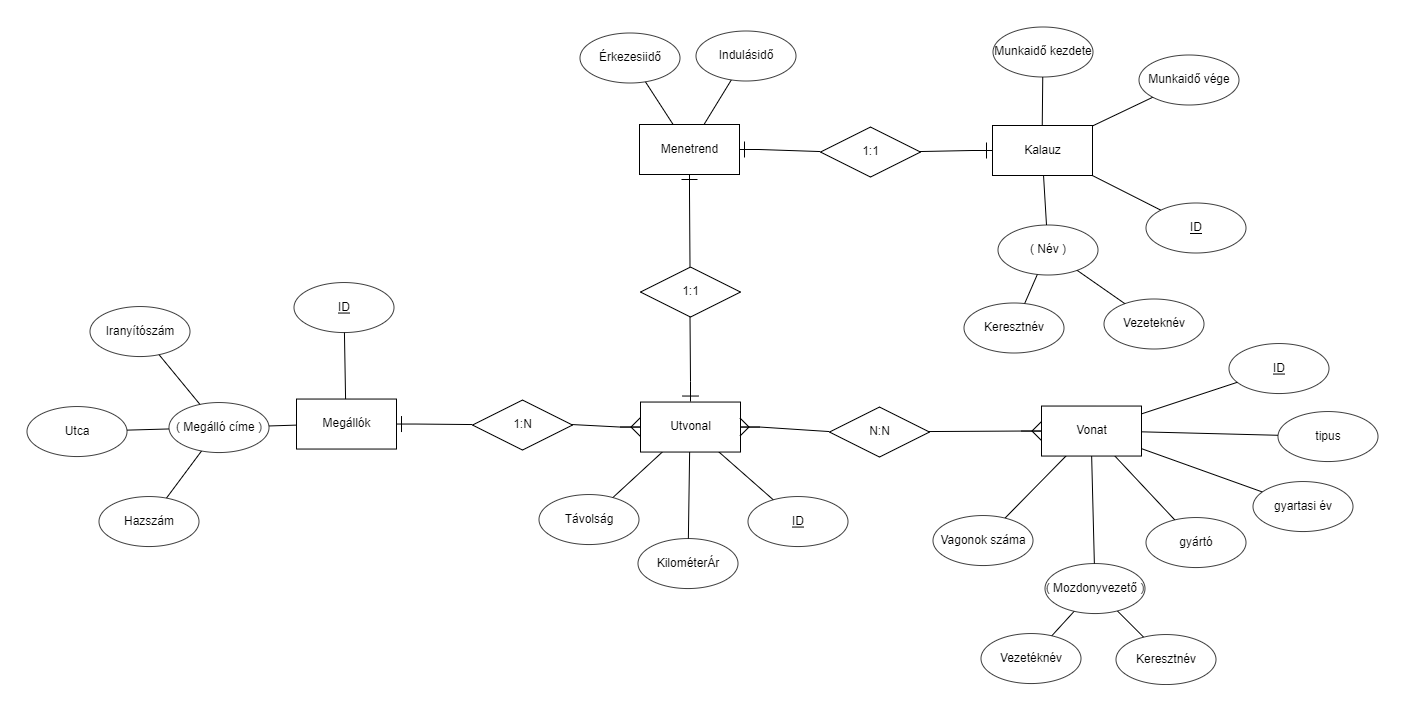
**Tulajdonságaik:**

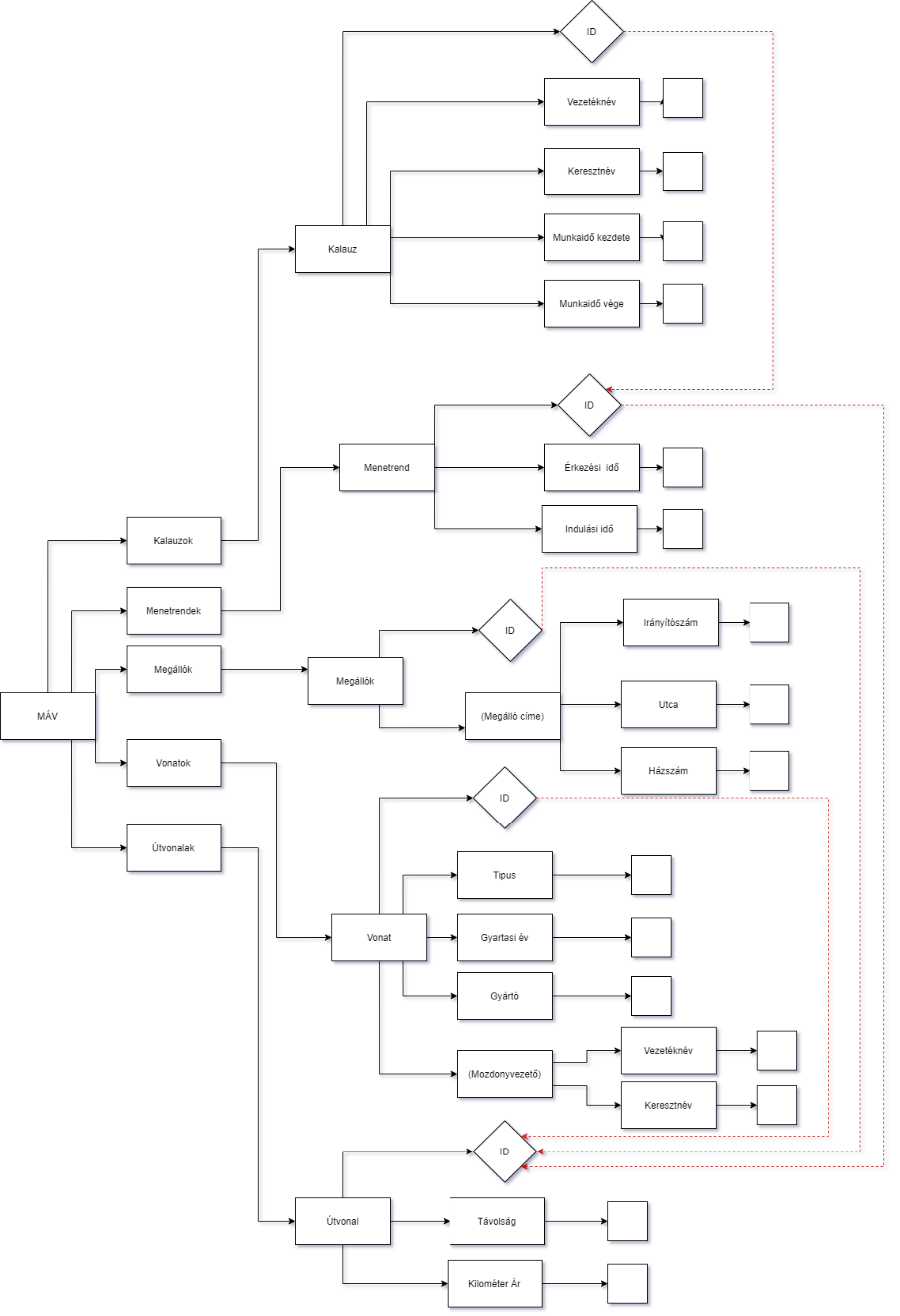
* Egy vonat rendelkezik: Vonat azonosítóval, gyártóval, vagonokszámával, gyártási évvel és típussal. A vonat azonosító azonosítja a vonatott. A vonat mozdonyvezetővel is rendelkezik, aminek van neve, amiből leszármazik a vezeték- és keresztnév.
* A kalauz egyed rendelkezik: Névvel, amiből le származik a kereszt- és a vezetéknév. A kalauznak is kell egy azonosító, amivel meg lehessen határozni melyik vonaton tartózkodik.
* A menetrend egyed rendelkezik: Indulási- és érkezési időponttal. A menetrendnek is van egy azonosítója, hogy tudjuk melyik menetrendhez, mely megállók tartoznak és milyen vonatok közlekednek az adott menetrend szerint.
* Az útvonal egyed rendelkezik: Vételdíjjal, távolsággal, kilométer árral és vonalszámmal, ami egy fajta azonosítóként működik.
* A megálló egyed rendelkezik: Megállónévvel és címmel. A megállónév mint azonosító van jelen.

**Kapcsolataik:**

* Egy vonat mehet több útvonalon is és egy útvonalon mehet több vonat is. (M: N)
* Egy kalauzhoz általában több vonat tartozik, de egy vonathoz nem szokott több kalauz tartozni. (1: N)
* Egy menetrendhez általában egy útvonal tartozik, és egy útvonalhoz is egy menetend. (1: 1)
* Egy menetrendhez általában több megálló tartozik és egy megállóhoz több menetrend tartozik. (1: N)
* Egy menetrendhez általában egy kalauz tartozik és úgy szintén egy kalauzhoz egy menetrend. (1: 1)

**1a) Az adatbázis ER modell**



**1b) Az adatbázis konvertálása XDM modellre:**

**1c) Az XDM modell alalján XML dokumentum készítése:**

<?xml version="1.0" encoding="UTF-8" standalone="no"?><mav>

<mav xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="XSD\_O2IXLB.xsd"></mav>

    <vonat id="VV1">

        <vagonokszama>15</vagonokszama>

        <mozdonyvezeto>

            <keresztnev>Lazar</keresztnev>

            <vezeteknev>Kovacs</vezeteknev>

        </mozdonyvezeto>

        <gyarto>Siemens Taurus</gyarto>

        <gyartasiev>2007</gyartasiev>

        <tipus>Teher</tipus>

    <mVezetokor>35</mVezetokor></vonat>

    <vonat id="V02">

        <vagonokszama>6</vagonokszama>

        <mozdonyvezeto>

            <keresztnev>Tibor</keresztnev>

            <vezeteknev>Schaffer</vezeteknev>

        </mozdonyvezeto>

        <gyarto>KISS</gyarto>

        <gyartasiev>2013</gyartasiev>

        <tipus>Személy vonat</tipus>

    </vonat>

    <vonat id="V03">

        <vagonokszama>3</vagonokszama>

        <mozdonyvezeto>

            <keresztnev>Meastro</keresztnev>

            <vezeteknev>Kipembe</vezeteknev>

        </mozdonyvezeto>

        <gyarto>FLirt</gyarto>

        <gyartasiev>1995</gyartasiev>

        <tipus>EC (Euro City)</tipus>

    </vonat>

    <kalauz id="k01">

        <nev>

            <keresztnev>Joe</keresztnev>

            <vezeteknev>Johnson</vezeteknev>

        </nev>

        <munkaido\_kezdete>07:30</munkaido\_kezdete>

        <munkaido\_vege>15:00</munkaido\_vege>

    </kalauz>

    <kalauz id="k02">

        <nev>

            <keresztnev>Erik</keresztnev>

            <vezeteknev>Eriksen</vezeteknev>

        </nev>

        <munkaido\_kezdete>08:30</munkaido\_kezdete>

        <munkaido\_vege>15:30</munkaido\_vege>

    </kalauz>

    <kalauz id="k03">

        <nev>

            <keresztnev>Vlad'imir</keresztnev>

            <vezeteknev>Vlad'imirevich</vezeteknev>

        </nev>

        <munkaido\_kezdete>07:50</munkaido\_kezdete>

        <munkaido\_vege>18:00</munkaido\_vege>

    </kalauz>

    <megallok id="mk01"> <!--azonosíto-->

        <megallo\_cime>

            <iranyitoszam>3535</iranyitoszam>

            <hazszam>25/A</hazszam>

            <utca>Kando Kalman</utca>

        </megallo\_cime>

    </megallok>

    <megallok id="mk02"> <!--azonosíto-->

        <megallo\_cime>

            <iranyitoszam>3738</iranyitoszam>

            <hazszam>65/C</hazszam>

            <utca>Petőfi Sándor</utca>

        </megallo\_cime>

    </megallok>

    <megallok id="mk03"> <!--azonosíto-->

        <megallo\_cime>

            <iranyitoszam>3225</iranyitoszam>

            <hazszam>75</hazszam>

            <utca>Petőfi Sandor</utca>

        </megallo\_cime>

    </megallok>

    <megallok id="mk04"> <!--azonosíto-->

        <megallo\_cime>

            <iranyitoszam>3132</iranyitoszam>

            <hazszam>47/E</hazszam>

            <utca>Berlini</utca>

        </megallo\_cime>

    </megallok>

    <megallok id="mk05"> <!--azonosíto-->

        <megallo\_cime>

            <iranyitoszam>3354</iranyitoszam>

            <hazszam>46/H</hazszam>

            <utca>Bécsi</utca>

        </megallo\_cime>

    </megallok>

    <megallok id="mk06"> <!--azonosíto-->

        <megallo\_cime>

            <iranyitoszam>3925</iranyitoszam>

            <hazszam>87/B</hazszam>

            <utca>Ballassi</utca>

        </megallo\_cime>

    </megallok>

    <megallok id="mk07"> <!--azonosíto-->

        <megallo\_cime>

            <iranyitoszam>3865</iranyitoszam>

            <hazszam>17/D</hazszam>

            <utca>Mexikó Völgyi</utca>

        </megallo\_cime>

    </megallok>

    <megallok id="mk08"> <!--azonosíto-->

        <megallo\_cime>

            <iranyitoszam>3672</iranyitoszam>

            <hazszam>76</hazszam>

            <utca>Eper</utca>

        </megallo\_cime>

    </megallok>

    <utvonal elsomegallo\_id="mk1" utolsomegallo\_id="mk2" utvonal\_id="u01" vonat\_id="V01"> <!--Vonalszam -->

        <tavolsag>75</tavolsag>

        <kilometerAr>200</kilometerAr>

    </utvonal>

    <utvonal elsomegallo\_id="mk1" utolsomegallo\_id="mk3" utvonal\_id="u02" vonat\_id="V02"> <!--Vonalszam -->

        <tavolsag>65</tavolsag>

        <kilometerAr>170</kilometerAr>

    </utvonal>

    <utvonal elsomegallo\_id="mk2" utolsomegallo\_id="mk4" utvonal\_id="u03" vonat\_id="V03"> <!--Vonalszam -->

        <tavolsag>55</tavolsag>

        <kilometerAr>120</kilometerAr>

    </utvonal>

    <utvonal elsomegallo\_id="mk8" utolsomegallo\_id="mk1" utvonal\_id="u04" vonat\_id="V01"> <!--Vonalszam -->

        <tavolsag>175</tavolsag>

        <kilometerAr>1800</kilometerAr>

    </utvonal>

    <utvonal elsomegallo\_id="mk7" utolsomegallo\_id="mk6" utvonal\_id="u05" voant\_id="V02"> <!--Vonalszam -->

        <tavolsag>125</tavolsag>

        <kilometerAr>1260</kilometerAr>

    </utvonal>

    <utvonal elsomegallo\_id="mk3" utolsomegallo\_id="mk5" utvonal\_id="u06" vonat\_id="V03"> <!--Vonalszam -->

        <tavolsag>200</tavolsag>

        <kilometerAr>2000</kilometerAr>

    </utvonal>

    <menetrend kalauz\_id="K01" utvonal\_id="u01">

        <indulasiido>07:40</indulasiido>

        <erkezesiido>08:50</erkezesiido>

    </menetrend>

    <menetrend kalauz\_id="K01" utvonal\_id="u02">

        <indulasiido>08:40</indulasiido>

        <erkezesiido>10:40</erkezesiido>

    </menetrend>

    <menetrend kalauz\_id="K01" utvonal\_id="u03">

        <indulasiido>08:10</indulasiido>

        <erkezesiido>08:50</erkezesiido>

    </menetrend>

    <menetrend kalauz\_id="K01" utvonal\_id="u04">

        <indulasiido>11:40</indulasiido>

        <erkezesiido>13:50</erkezesiido>

    </menetrend>

    <menetrend kalauz\_id="K01" utvonal\_id="u05">

        <indulasiido>12:40</indulasiido>

        <erkezesiido>13:55</erkezesiido>

    </menetrend>

    <menetrend kalauz\_id="K01" utvonal\_id="u06">

        <indulasiido>15:40</indulasiido>

        <erkezesiido>17:00</erkezesiido>

    </menetrend>

</mav>

1d) Az XML dokumentum alapján XML schema készítése:

<?xml version="1.0" encoding="utf-8"?>

<!-- MAV's xml Schema -->

<xs:schema attributeFormDefault="unqualified"

           elementFormDefault="qualified"

           xmlns:xs="http://www.w3.org/2001/XMLSchema">

  <xs:element name="mav">

    <xs:complexType>

      <xs:sequence>

        <xs:element maxOccurs="unbounded"

                    name="vonat">

          <xs:complexType>

            <xs:sequence>

              <xs:element name="vagonokszama"

                          type="xs:unsignedByte" />

              <xs:element name="mozdonyvezeto">

                <xs:complexType>

                  <xs:sequence>

                    <xs:element name="keresztnev"

                                type="xs:string" />

                    <xs:element name="vezeteknev"

                                type="xs:string" />

                  </xs:sequence>

                </xs:complexType>

              </xs:element>

              <xs:element name="gyarto"

                          type="xs:string" />

              <xs:element name="gyartasiev"

                          type="xs:unsignedShort" />

              <xs:element name="tipus"

                          type="xs:string" />

            </xs:sequence>

            <xs:attribute name="id"

                          type="xs:string"

                          use="required" />

          </xs:complexType>

        </xs:element>

        <xs:element maxOccurs="unbounded"

                    name="kalauz">

          <xs:complexType>

            <xs:sequence>

              <xs:element name="nev">

                <xs:complexType>

                  <xs:sequence>

                    <xs:element name="keresztnev"

                                type="xs:string" />

                    <xs:element name="vezeteknev"

                                type="xs:string" />

                  </xs:sequence>

                </xs:complexType>

              </xs:element>

              <xs:element name="munkaido\_kezdete"

                          type="xs:string" />

              <xs:element name="munkaido\_vege"

                          type="xs:string" />

            </xs:sequence>

            <xs:attribute name="id"

                          type="xs:string"

                          use="required" />

          </xs:complexType>

        </xs:element>

        <xs:element maxOccurs="unbounded"

                    name="megallok">

          <xs:complexType>

            <xs:sequence>

              <xs:element name="megallo\_cime">

                <xs:complexType>

                  <xs:sequence>

                    <xs:element name="iranyitoszam"

                                type="xs:unsignedShort" />

                    <xs:element name="hazszam"

                                type="xs:string" />

                    <xs:element name="utca"

                                type="xs:string" />

                  </xs:sequence>

                </xs:complexType>

              </xs:element>

            </xs:sequence>

            <xs:attribute name="id"

                          type="xs:string"

                          use="required" />

          </xs:complexType>

        </xs:element>

        <xs:element maxOccurs="unbounded"

                    name="utvonal">

          <xs:complexType>

            <xs:sequence>

              <xs:element name="tavolsag"

                          type="xs:unsignedByte" />

              <xs:element name="kilometerAr"

                          type="xs:unsignedShort" />

            </xs:sequence>

            <xs:attribute name="utvonal\_id"

                          type="xs:string"

                          use="required" />

            <xs:attribute name="elsomegallo\_id"

                          type="xs:string"

                          use="required" />

            <xs:attribute name="utolsomegallo"

                          type="xs:string"

                          use="required" />

            <xs:attribute name="vonat\_id"

                          type="xs:string"

                          use="optional" />

            <xs:attribute name="voant\_id"

                          type="xs:string"

                          use="optional" />

          </xs:complexType>

        </xs:element>

        <xs:element maxOccurs="unbounded"

                    name="menetrend">

          <xs:complexType>

            <xs:sequence>

              <xs:element name="indulasiido"

                          type="xs:string" />

              <xs:element name="erkezesiido"

                          type="xs:string" />

            </xs:sequence>

            <xs:attribute name="utvonal\_id"

                          type="xs:string"

                          use="required" />

          </xs:complexType>

        </xs:element>

      </xs:sequence>

    </xs:complexType>

    <!--ID-k meg szoritasa -->

    <xs:unique name="vonat\_id">

      <xs:selector xpath=".//utvonal" />

      <xs:field xpath="@vonat\_id" />

  </xs:unique>

  <xs:keyref name="vona\_idref" refer="vonat\_id">

    <xs:selector xpath=".//vonat" />

    <xs:field xpath="@id" />

</xs:keyref>

<xs:key name="megallok\_id">

  <xs:selector xpath=".//utvonal" />

  <xs:field xpath="@megallok\_id" />

</xs:key>

<xs:keyref name="megallok\_idref" refer="megallok\_id">

  <xs:selector xpath=".//megallok" />

  <xs:field xpath="@id" />

</xs:keyref>

<xs:key name="menetrend\_id">

  <xs:selector xpath=".//utvonal" />

  <xs:field xpath="@menetrend\_id" />

</xs:key>

<xs:keyref name="menetrend\_idref" refer="menetrend\_id">

  <xs:selector xpath=".//menetrend" />

  <xs:field xpath="@id" />

</xs:keyref>

<xs:key name="kalauz\_id">

  <xs:selector xpath=".//menetrend" />

  <xs:field xpath="@kalauz\_Id" />

</xs:key>

<xs:keyref name="kalauz\_idref" refer="kalauz\_id">

  <xs:selector xpath=".//kalauz" />

  <xs:field xpath="@id" />

</xs:keyref>

  </xs:element>

<!--Egyedi komplextipusok -->

<xs:complexType name="kalauz">

  <xs:sequence>

      <xs:element name="vezeteknev" type="xs:string" />

      <xs:element name="keresztnev" type="xs:string" />

  </xs:sequence>

  <xs:attribute name="id" type="xs:int" use="required" />

</xs:complexType>

<xs:complexType name="megallok">

  <xs:sequence>

      <xs:element name="iranyitoszam" type="xs:int"/>

      <xs:element name="utca" type="xs:string" />

      <xs:element name="hazszam" type="xs:string" />

  </xs:sequence>

  <xs:attribute name="id" type="xs:int" use="required" />

</xs:complexType>

<xs:complexType name="utvonal">

  <xs:sequence>

      <xs:element name="tavlosag" type="xs:int" />

      <xs:element name="kilometerAr" type="xs:decimal" />

  </xs:sequence>

  <xs:attribute name="id" type="xs:string" use="required" />

  <xs:attribute name="meneterndId" type="xs:int" use="required" />

</xs:complexType>

</xs:schema>

1. **Feladat**

A feladatban egy DOM program Készítése az XML dokumentum adatainak adminisztrálása alapján:

**2a) Adatolvasás**

package hu.domparse.o2ixlb;

import java.io.File;

import javax.xml.parsers.DocumentBuilder;

import javax.xml.parsers.DocumentBuilderFactory;

import org.w3c.dom.Document;

import org.w3c.dom.NamedNodeMap;

import org.w3c.dom.Node;

import org.w3c.dom.NodeList;

public class DOMReadO2IXLB {

    // Forras: https://www.javatpoint.com/how-to-read-xml-file-in-java

    public static void main(String[] args) {

        try {

            File file = new File(

                    "D:\\UniversityOfMiskolc\\University\\2021\_22\_1\\XML\\O2IXLB\_FelevesBeadando\\O2IXLBvonat.xml"); // XML

                                                                                                                        // file

                                                                                                                        // forrasa

            DocumentBuilder documentBuilder = DocumentBuilderFactory.newInstance().newDocumentBuilder();

            Document document = documentBuilder.parse(file);

            System.out.println("Root element: " + document.getDocumentElement().getNodeName());

            if (document.hasChildNodes()) {

                printNodeList(document.getChildNodes());

            }

        } catch (Exception e) {

            System.out.println(e.getMessage());

        }

    }

    private static void printNodeList(NodeList nodeList) {

        for (int count = 0; count < nodeList.getLength(); count++) {

            Node elemNode = nodeList.item(count);

            if (elemNode.getNodeType() == Node.ELEMENT\_NODE) {

// get node name and value

                System.out.println("\nNode Name =" + elemNode.getNodeName() + " [OPEN]");

                System.out.println("Node Content =" + elemNode.getTextContent());

                if (elemNode.hasAttributes()) {

                    NamedNodeMap nodeMap = elemNode.getAttributes();

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

                        Node node = nodeMap.item(i);

                        System.out.println("attr name : " + node.getNodeName());

                        System.out.println("attr value : " + node.getNodeValue());

                    }

                }

                if (elemNode.hasChildNodes()) {

//recursive call if the node has child nodes

                    printNodeList(elemNode.getChildNodes());

                }

                System.out.println("Node Name =" + elemNode.getNodeName() + " [CLOSE]");

            }

        }

    }

}

**2b) Adatmódosítás**

package hu.domparse.o2ixlb;

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.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.NamedNodeMap;

import org.w3c.dom.Node;

import org.w3c.dom.NodeList;

import org.xml.sax.SAXException;

public class DOMModifyO2IXLB {

    // Forras:

    // https://examples.javacodegeeks.com/core-java/xml/parsers/documentbuilderfactory/modify-xml-file-in-java-using-dom-parser-example/

    public static final String xmlFilePath = "D:\\UniversityOfMiskolc\\University\\2021\_22\_1\\XML\\O2IXLB\_FelevesBeadando\\O2IXLBvonat.xml";

    public static void main(String argv[]) {

        try {

            DocumentBuilderFactory documentBuilderFactory = DocumentBuilderFactory.newInstance();

            DocumentBuilder documentBuilder = documentBuilderFactory.newDocumentBuilder();

            Document document = documentBuilder.parse(xmlFilePath);

            // Get employee by tag name

            // use item(0) to get the first node with tage name "vonat"

            Node employee = document.getElementsByTagName("vonat").item(0);

            // update employee , set the id to 10

            NamedNodeMap attribute = employee.getAttributes();

            Node nodeAttr = attribute.getNamedItem("id");

            nodeAttr.setTextContent("VV1");

            // append a new node to the first vonat

            Element age = document.createElement("mVezetokor");

            age.appendChild(document.createTextNode("35"));

            employee.appendChild(age);

            // loop the vonat node and update vagonokszama, and delete a node

            NodeList nodes = employee.getChildNodes();

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

                Node element = nodes.item(i);

                if ("vagonokszama".equals(element.getNodeName())) {

                    element.setTextContent("15");

                }

                // remove keresztnev

                if ("keresztnev".equals(element.getNodeName())) {

                    employee.removeChild(element);

                }

            }

            // write the DOM object to the file

            TransformerFactory transformerFactory = TransformerFactory.newInstance();

            Transformer transformer = transformerFactory.newTransformer();

            DOMSource domSource = new DOMSource(document);

            StreamResult streamResult = new StreamResult(new File(xmlFilePath));

            transformer.transform(domSource, streamResult);

            System.out.println("The XML File was ");

        } catch (ParserConfigurationException pce) {

            pce.printStackTrace();

        } catch (TransformerException tfe) {

            tfe.printStackTrace();

        } catch (IOException ioe) {

            ioe.printStackTrace();

        } catch (SAXException sae) {

            sae.printStackTrace();

        }

    }

}

**2c) Adatlegkérés**

package hu.domparse.o2ixlb;

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 DOMQueryO2IXLB {

    // Forras: https://www.tutorialspoint.com/java\_xml/java\_dom\_query\_document.htm

    public static void main(String argv[]) {

        try {

            File inputFile = new File(

                    "D:\\UniversityOfMiskolc\\University\\2021\_22\_1\\XML\\O2IXLB\_FelevesBeadando\\O2IXLBvonat.xml");

            DocumentBuilderFactory dbFactory = DocumentBuilderFactory.newInstance();

            DocumentBuilder dBuilder = dbFactory.newDocumentBuilder();

            Document doc = dBuilder.parse(inputFile);

            doc.getDocumentElement().normalize();

            System.out.print("Root element: ");

            System.out.println(doc.getDocumentElement().getNodeName());

            NodeList nList = doc.getElementsByTagName("vonat");

            NodeList mList = doc.getElementsByTagName("megallok");

            System.out.println("----------------------------");

            for (int temp = 0; temp < nList.getLength(); temp++) {

                Node nNode = nList.item(temp);

                System.out.println("\nCurrent Element :");

                System.out.print(nNode.getNodeName());

                if (nNode.getNodeType() == Node.ELEMENT\_NODE) {

                    Element eElement = (Element) nNode;

                    System.out.print(" ID : ");

                    System.out.println(eElement.getAttribute("id"));

                    NodeList vagonokszamaelement = eElement.getElementsByTagName("vagonokszama");

                    for (int count = 0; count < vagonokszamaelement.getLength(); count++) {

                        Node node1 = vagonokszamaelement.item(count);

                        if (node1.getNodeType() == Node.ELEMENT\_NODE) {

                            Element vonatVagon = (Element) node1;

                            System.out.print("Vagonok szama : ");

                            System.out.println(vonatVagon.getTextContent());

                            System.out.print(vonatVagon.getAttribute("type"));

                        }

                    }

                }

                if (nNode.getNodeType() == Node.ELEMENT\_NODE) {

                    Element eElement = (Element) nNode;

                    eElement.getAttribute("id");

                    NodeList nodeElement = eElement.getElementsByTagName("keresztnev");

                    for (int count = 0; count < nodeElement.getLength(); count++) {

                        Node node1 = nodeElement.item(count);

                        if (node1.getNodeType() == Node.ELEMENT\_NODE) {

                            Element keresztnevelement = (Element) node1;

                            System.out.print("Mozdonyvezet :\n\tKeresztnev : ");

                            System.out.println(keresztnevelement.getTextContent());

                            System.out.print(keresztnevelement.getAttribute("type"));

                        }

                    }

                }

                if (nNode.getNodeType() == Node.ELEMENT\_NODE) {

                    Element eElement = (Element) nNode;

                    eElement.getAttribute("id");

                    NodeList nodeElement = eElement.getElementsByTagName("vezeteknev");

                    for (int count = 0; count < nodeElement.getLength(); count++) {

                        Node node1 = nodeElement.item(count);

                        if (node1.getNodeType() == Node.ELEMENT\_NODE) {

                            Element vezetekenevelement = (Element) node1;

                            System.out.print("\tVezeteknev : ");

                            System.out.println(vezetekenevelement.getTextContent());

                            System.out.print(vezetekenevelement.getAttribute("type"));

                        }

                    }

                }

            }

            System.out.println("\n----------------------------");

            for (int temp = 0; temp < mList.getLength(); temp++) {

                Node nNode = mList.item(temp);

                System.out.println("\nCurrent Element :");

                System.out.print(nNode.getNodeName());

                if (nNode.getNodeType() == Node.ELEMENT\_NODE) {

                    Element eElement = (Element) nNode;

                    System.out.print(" ID : ");

                    System.out.println(eElement.getAttribute("id"));

                    NodeList iranyoszanelement = eElement.getElementsByTagName("iranyitoszam");

                    for (int count = 0; count < iranyoszanelement.getLength(); count++) {

                        Node node1 = iranyoszanelement.item(count);

                        if (node1.getNodeType() == node1.ELEMENT\_NODE) {

                            Element iranyitoszamelement = (Element) node1;

                            System.out.print("Iranyitoszam : ");

                            System.out.println(iranyitoszamelement.getTextContent());

                            System.out.print(iranyitoszamelement.getAttribute("type"));

                        }

                    }

                }

                if (nNode.getNodeType() == Node.ELEMENT\_NODE) {

                    Element eElement = (Element) nNode;

                    eElement.getAttribute("id");

                    NodeList nodeElement = eElement.getElementsByTagName("hazszam");

                    for (int count = 0; count < nodeElement.getLength(); count++) {

                        Node node1 = nodeElement.item(count);

                        if (node1.getNodeType() == Node.ELEMENT\_NODE) {

                            Element hazszamelement = (Element) node1;

                            System.out.print("Hazszam : ");

                            System.out.println(hazszamelement.getTextContent());

                            System.out.print(hazszamelement.getAttribute("type"));

                        }

                    }

                }

                if (nNode.getNodeType() == Node.ELEMENT\_NODE) {

                    Element eElement = (Element) nNode;

                    eElement.getAttribute("id");

                    NodeList nodeElement = eElement.getElementsByTagName("utca");

                    for (int count = 0; count < nodeElement.getLength(); count++) {

                        Node node1 = nodeElement.item(count);

                        if (node1.getNodeType() == Node.ELEMENT\_NODE) {

                            Element utcaelement = (Element) node1;

                            System.out.print("Utca : ");

                            System.out.println(utcaelement.getTextContent());

                            System.out.print(utcaelement.getAttribute("type"));

                        }

                    }

                }

            }

        } catch (Exception e) {

            e.printStackTrace();

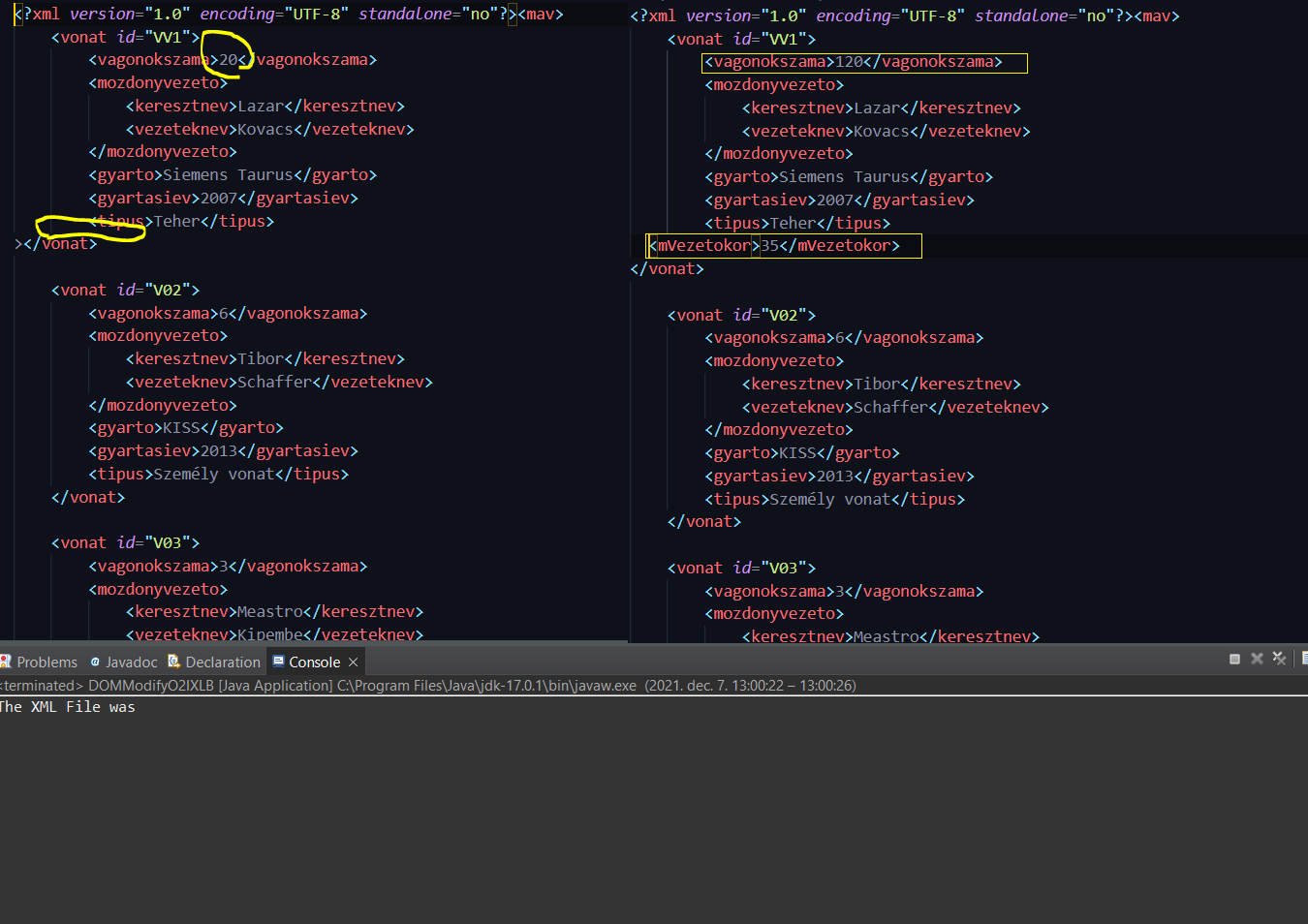
        }

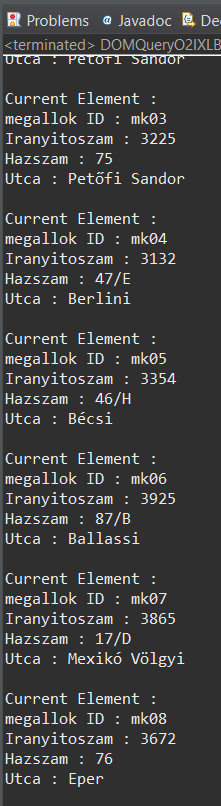
    }

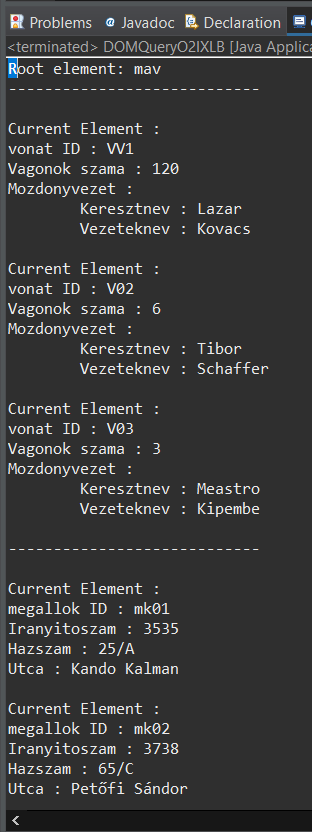
}

**Outputok:**

DOMmodify:





DOMquery:

DOMread:

