JEGYZŐKÖNYV

Adatkezelés XML környezetben

2022. ősz féléves feladat

Készítette: **Tőzsér Zétény**
Neptunkód: **QGNLD2**
Dátum: **2022.11.27**

Tartalomjegyzék

Feladat leírása

A feladat futball csapatokat követ. Tagjaikat, az edzőket és közös meccseiket.

Minden játékosnak van egyedi kódja: jatekosID, ami a kulcsa is. Így azonos névű játékosok megkülönböztethetőek. Ezen felül a jatekosban ott van a neve, mezszáma és hajszíne azonosíthatóság miatt. A játékos tagja egy csapatnak, ezért még a csapat kódja is ott van, idegen kulcsként.

Az edzoben az edzo kulcsa: edzoID. Ez egyedi. Az edző neve és hajszíne azonosítás végett. Az életkora tapasztalat becslésére alkalmas. Minden csapatnak egy edzője van, egy csapat kulcsa van az edzoben, mint idegen kulcs.

Már említettem, hogy a csapatoknak is van azonosítója, ami pedig a csapatID. A csapatoknak is van neve (csapatnev). Emellett a városuk (hazai pálya) és a mezük színe (pályán felismerhetőség) van feljegyezve.

A meccseknek is van kódja, a meccsID. A meccseken lehetnek gollövők, többen is. Ebben a játékos(ok) kódja(i) van(nak) feljegyezve. A gólok száma külön fel van jegyezve. A meccsekre eladott jegyek száma is tárolt. Ezzel látható, hogy mennyire népszerű egy-egy meccs.

A csapat–meccs kapcsolat több-több kapcsolat, az adott meccs nyertese kapcsolódik hozzá.

A stadionoknak már van neve, ID-je is. A stadionok címe több elemű (város, utca, szám). Még a stadion kapacitása is tárolt (férőhelyek).

1. feladat

1a) ER modell

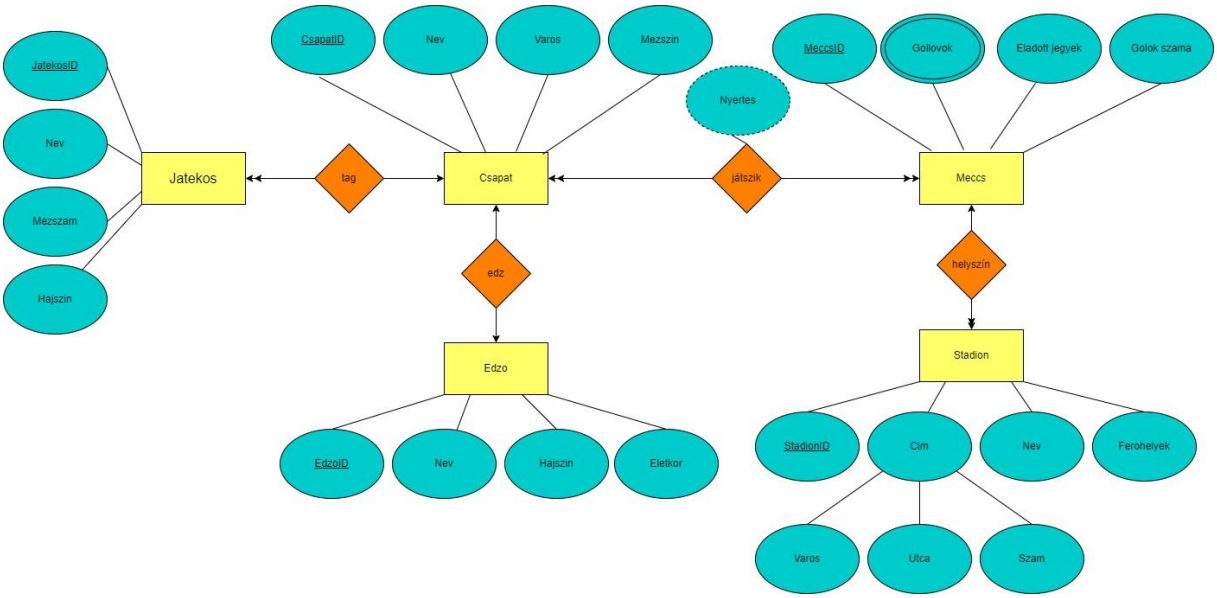
A modell a draw.io nevű szerkesztőprogrammal készült. A Csapatnak van a legtöbb kapcsolata, ezért abból indultam el.

A csapat és játékos közt 1:N kapcsolat van (tag). Egy játékos egy csapatnak lehet tagja. Egy csapatban több játékos van.

A csapat és edző közt 1:1 kapcsolat van (edz). Minden csapathoz egy edző tartozik és minden edző egy csapatot edz.

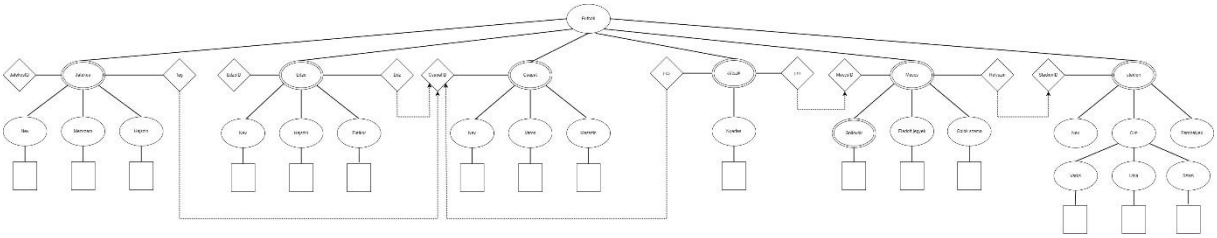
A csapat és meccs közt N:M kapcsolat van (játszik). Egy csapat több meccsen játszhat és egy meccsen két csapat játszik. Ebből a kapcsolatból származtatható a nyertes.

A meccs és stadion közt 1:N kapcsolat van. Egy meccs egy stadionban játszódik, de egy stadionban több meccs is játszódhat.



1b) XDM modell

Az XDM modell is a draw.io szerkesztőprogrammal készült. Az modell a futball összefogó nevet kapta. A csapat-meccs (játszik) kapcsolatból kapcsoló”tábla” lett.



1c) XML dokumentum

Az XML dokumentum Visual Studio Code-ban készült. A dokumentum az xml deklarációval kezdődik, mejd a gyökér elem következik (football). Ezen belül először a csapatok szerepelnek, mivel az edzők és játekosok is használják a kulcsait. A stadion pedig a meccs előtt, hasonló módon. A futball minden gyerekeleméből legalább három példány van.

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

<futball noNameSpaceSchemaLocation="XMLSchemaQGNLD2.xsd">

   <!-- csapatok-->
   <csapat csapatID="c1">
      <nev>ABCFC</nev>
      <varos>Budapest</varos>
      <mezszin>piros</mezszin>
   </csapat>

   <csapat csapatID="c2">
      <nev>MTK</nev>
      <varos>Miskolc</varos>
      <mezszin>zold</mezszin>
   </csapat>

   <csapat csapatID="c3">
      <nev>HalmajSC</nev>
      <varos>Halmaj</varos>
      <mezszin>kek</mezszin>
   </csapat>

   <!-- edzok-->
   <edzo edzoID="e1" csapat-idegen-kulcs="c1">
      <nev>Soma</nev>
      <hajszin>barna</hajszin>
      <eletkor>28</eletkor>
   </edzo>

   <edzo edzoID="e2" csapat-idegen-kulcs="c2">
      <nev>Samu</nev>
      <hajszin>fekete</hajszin>
      <eletkor>30</eletkor>
   </edzo>
```

```
<edzo edzoID="e3" csapat-idegen-kulcs="c3">
   <nev>Pista</nev>
   <hajszin>voros</hajszin>
   <eletkor>29</eletkor>
</edzo>

<!-- jatekosok-->
<jatekos jatekosID="j1" csapat-idegen-kulcs="c1">
   <nev>Jose</nev>
   <mezszam>12</mezszam>
   <hajszin>szoke</hajszin>
</jatekos>

<jatekos jatekosID="j2" csapat-idegen-kulcs="c1">
   <nev>Armando</nev>
   <mezszam>11</mezszam>
   <hajszin>barna</hajszin>
</jatekos>

<jatekos jatekosID="j3" csapat-idegen-kulcs="c2">
   <nev>Abel</nev>
   <mezszam>9</mezszam>
   <hajszin>barna</hajszin>
</jatekos>

<jatekos jatekosID="j4" csapat-idegen-kulcs="c2">
   <nev>Rokus</nev>
   <mezszam>11</mezszam>
   <hajszin>szoke</hajszin>
</jatekos>

<jatekos jatekosID="j5" csapat-idegen-kulcs="c3">
   <nev>Abel</nev>
   <mezszam>8</mezszam>
   <hajszin>voros</hajszin>
</jatekos>
```

```
<jatekos jatekosID="j6" csapat-idegen-kulcs="c3">
   <nev>Alfonz</nev>
   <mezszam>12</mezszam>
   <hajszin>szoke</hajszin>
</jatekos>

<!-- stadionok-->
<stadion stadionID="s1">
   <nev>ABC-Arena</nev>
   <cim>
      <varos>Budapest</varos>
      <utca>Vamhaz</utca>
      <szam>9</szam>
   </cim>
   <ferohelyek>20000</ferohelyek>
</stadion>

<stadion stadionID="s2">
   <nev>Duhongo</nev>
   <cim>
      <varos>Miskolc</varos>
      <utca>Kiraly</utca>
      <szam>22</szam>
   </cim>
   <ferohelyek>18000</ferohelyek>
</stadion>

<stadion stadionID="s3">
   <nev>Halmaj-Stadion</nev>
   <cim>
      <varos>Halmaj</varos>
      <utca>Danko-Pista</utca>
      <szam>37</szam>
   </cim>
   <ferohelyek>25000</ferohelyek>
```

```
</stadion>

<!-- meccsek-->
<meccs meccsID="m1" stadion-idegen-kulcs="s1">
   <gollovok>j1</gollovok>
   <gollovok>j3</gollovok>
   <eladott-jegyek>12000</eladott-jegyek>
   <golok-szama>2</golok-szama>
</meccs>

<meccs meccsID="m2" stadion-idegen-kulcs="s2">
   <gollovok>j5</gollovok>
   <eladott-jegyek>18000</eladott-jegyek>
   <golok-szama>3</golok-szama>
</meccs>

<meccs meccsID="m3" stadion-idegen-kulcs="s3">
   <gollovok>j2</gollovok>
   <eladott-jegyek>15000</eladott-jegyek>
   <golok-szama>1</golok-szama>
</meccs>

<!-- jatszik-->
<jatszik jID="01" meccs-idegen-kulcs="m1" csapat-idegen-kulcs="c1">
   <nyertes></nyertes>
</jatszik>

<jatszik jID="02" meccs-idegen-kulcs="m1" csapat-idegen-kulcs="c2">
   <nyertes></nyertes>
</jatszik>

<jatszik jID="03" meccs-idegen-kulcs="m2" csapat-idegen-kulcs="c3">
   <nyertes>c3</nyertes>
</jatszik>

<jatszik jID="04" meccs-idegen-kulcs="m2" csapat-idegen-kulcs="c2">
```

```
    <nyertes>c3</nyertes>
  </jatszik>

  <jatszik jID="05" meccs-idegen-kulcs="m3" csapat-idegen-kulcs="c1">
    <nyertes>c1</nyertes>
  </jatszik>

  <jatszik jID="06" meccs-idegen-kulcs="m3" csapat-idegen-kulcs="c3">
    <nyertes>c1</nyertes>
  </jatszik>
```

1d) XML séma

A séma szintén Visual Studio Code-ban készült. Először az egyszerű típusok szerepelnek, vagyis a példányok gyerekelemi. Utána kezdődik a „futball”, benne a komplex típusokkal. A játszik után, még a futballon belül szerepelnek a kulcsok és idegen kulcsok.

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

<xs:schema attributeFormDefault="unqualified" elementFormDefault="qualified"
xmlns:xs="http://www.w3.org/2001/XMLSchema">

  <!-- egyszeru tipusok-->

  <xs:simpleType name="nev-type">
    <xs:restriction base="xs:string"/>
  </xs:simpleType>

  <xs:simpleType name="hajszin-type">
    <xs:restriction base="xs:string"/>
  </xs:simpleType>

  <xs:simpleType name="mezszam-type">
    <xs:restriction base="xs:int">
      <xs:minExclusive value="1"/>
    </xs:restriction>
  </xs:simpleType>

  <xs:simpleType name="eletkor-type">
    <xs:restriction base="xs:int">
```

```
      <xs:minExclusive value="0"/>
   </xs:restriction>
</xs:simpleType>

<xs:simpleType name="varos-type">
   <xs:restriction base="xs:string"/>
</xs:simpleType>

<xs:simpleType name="mezszin-type">
   <xs:restriction base="xs:string"/>
</xs:simpleType>

<xs:simpleType name="eladott-jegyek-type">
   <xs:restriction base="xs:int">
      <xs:minInclusive value="0"/>
   </xs:restriction>
</xs:simpleType>

<xs:simpleType name="gollovok-type">
   <xs:restriction base="xs:string"/>
</xs:simpleType>

<xs:simpleType name="golok-szama-type">
   <xs:restriction base="xs:int">
      <xs:minExclusive value="0"/>
   </xs:restriction>
</xs:simpleType>

<xs:simpleType name="utca-type">
   <xs:restriction base="xs:string"/>
</xs:simpleType>

<xs:simpleType name="szam-type">
   <xs:restriction base="xs:int">
      <xs:minExclusive value="1"/>
   </xs:restriction>
</xs:simpleType>

<xs:simpleType name="ferohelyek-type">
   <xs:restriction base="xs:int">
      <xs:minExclusive value="0"/>
   </xs:restriction>
</xs:simpleType>
```

```
<!-- -->

<xs:element name="futball">
  <xs:complexType>
    <xs:sequence>

      <xs:element name="csapat" maxOccurs="unbounded">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="nev"/>
            <xs:element name="varos"/>
            <xs:element name="mezszin"/>
          </xs:sequence>
          <xs:attribute name="csapatID" type="xs:string" use="required"/>
        </xs:complexType>
      </xs:element>

      <xs:element name="edzo" maxOccurs="unbounded">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="nev"/>
            <xs:element name="hajszin"/>
            <xs:element name="eletkor"/>
          </xs:sequence>
          <xs:attribute name="edzoID" type="xs:string" use="required"/>
        </xs:complexType>
      </xs:element>

      <xs:element name="jatekos" maxOccurs="unbounded">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="nev"/>
            <xs:element name="mezszam"/>
            <xs:element name="hajszin"/>
          </xs:sequence>
          <xs:attribute name="jatekosID" type="xs:string" use="required"/>
        </xs:complexType>
      </xs:element>

      <xs:element name="stadion" maxOccurs="unbounded">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="nev"/>
            <xs:element name="cim">
```

```
                <xs:complexType>
                  <xs:sequence>
                    <xs:element name="varos"/>
                    <xs:element name="utca"/>
                    <xs:element name="szam"/>
                  </xs:sequence>
                </xs:complexType>
              </xs:element>
              <xs:element name="ferholyek"/>
            </xs:sequence>
            <xs:attribute name="stadionID" type="xs:string" use="required"/>
          </xs:complexType>
        </xs:element>

        <xs:element name="meccs" maxOccurs="unbounded">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="gollovok"/>
              <xs:element name="eladott-jegyek"/>
              <xs:element name="golok-szama"/>
            </xs:sequence>
            <xs:attribute name="meccsID" type="xs:string" use="required"/>
          </xs:complexType>
        </xs:element>

        <xs:element name="jatszik" maxOccurs="unbounded">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="nyertes"/>
            </xs:sequence>
            <xs:attribute name="j-cs" type="xs:string" use="required"/>
            <xs:attribute name="j-m" type="xs:string" use="required"/>
          </xs:complexType>
        </xs:element>

    </xs:sequence>
  </xs:complexType>

  <!-- kulcsok-->

  <xs:key name="csapat-kulcs">
    <xs:selector xpath="csapat"/>
    <xs:field xpath="@csapatID"/>
  </xs:key>
```

```
    <xs:key name="edzo-kulcs">
    <xs:selector xpath="edzo"/>
    <xs:field xpath="@edzoID"/>
    </xs:key>

    <xs:key name="jatekos-kulcs">
    <xs:selector xpath="jatekos"/>
    <xs:field xpath="@jatekosID"/>
    </xs:key>

    <xs:key name="stadion-kulcs">
    <xs:selector xpath="stadion"/>
    <xs:field xpath="@stadionID"/>
    </xs:key>

    <xs:key name="meccs-kulcs">
    <xs:selector xpath="meccs"/>
    <xs:field xpath="@meccsID"/>
    </xs:key>

    <!-- idegen kulcsok-->

    <xs:keyref refer="csapat-kulcs" name="csapat-idegen-kulcs">
      <xs:selector xpath="csapat"/>
      <xs:field xpath="@csapatID"/>
    </xs:keyref>

    <xs:keyref refer="meccs-kulcs" name="meccs-idegen-kulcs">
      <xs:selector xpath="meccs"/>
      <xs:field xpath="@meccsID"/>
    </xs:keyref>

    <xs:keyref refer="stadion-kulcs" name="stadion-idegen-kulcs">
      <xs:selector xpath="stadion"/>
      <xs:field xpath="@stadionID"/>
    </xs:keyref>

  </xs:element>

</xs:schema>
```

2. feladat

2a) adatolvasás

A DomRead belovassa és struktúráltan írja ki a konzolra az xml tartalmát. Közben létrehoz egy .txt fájlt és oda is írja.

```
package hu.domparse.QGNLD2;

import java.io.*;

import javax.xml.parsers.DocumentBuilderFactory;
import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.ParserConfigurationException;
import org.xml.sax.SAXException;

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

public class DomReadQGNLD2 {

	public static void main(String argv[]) throws SAXException,
	IOException, ParserConfigurationException {
		//Elemek es gyerekelemeik.
		String[] csapatFields = {
				"nev",
				"varos",
				"mezszin"
		};
```

```
String[] edzoFields = {
                    "nev",
                    "hajszin",
                    "eletkor"
};

String[] jatekosFields = {
                    "nev",
                    "mezszam",
                    "hajszin"
};

String[] stadionFields = {
                    "nev",
                    "cim",
                    "ferohelyek"
};

String[] meccsFields = {
                    "gollovok",
                    "eladott-jegyek",
                    "golok-szama"
};

String[] jatszikFields = {
                    "nyertes"
          };
```

```
String[][] fields = {
                csapatFields,
                edzoFields,
                jatekosFields,
                stadionFields,
                meccsFields,
                jatszikFields
};

String [] subRoots = {
                "csapat",
                "edzo",
                "jatekos",
                "stadion",
                "meccs",
                "jatszik"
};

String [] idList = {
                "csapatID",
                "edzoID",
                "jatekosID",
                "stadionID",
                "meccsID",
                "jID"
};
```

```
		//XML file megnyitasa.
		File xmlFile = new File("XMLQGNLD2.xml");
		//TXT file letrehozas, iras elokeszites.
		File txt = new File("DomRead.txt");
		FileWriter fw = new FileWriter(txt);
		PrintWriter pw = new PrintWriter(fw);

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

		System.out.println("Root element: " +
doc.getDocumentElement().getNodeName());
		//Txt-be is iras.
		pw.write( "Root element: " +
doc.getDocumentElement().getNodeName() + "\n" );

		int index = 0;
		//Az xml-en vegigmenni.
		for(String element : subRoots) {
			NodeList nList = doc.getElementsByTagName(element);
			for (int i = 0; i < nList.getLength(); i++){
				Node nNode = nList.item(i);
				//Uj elemnel irni konzolra es fajlba.
				System.out.println("\nCurrent Element: " +
nNode.getNodeName());
```

```
                        pw.write( "\nCurrent Element: " + nNode.getNodeName()
+ "\n");
                        //A fieldek segitsegevel gyerekelemek kiirasa.
                        if (nNode.getNodeType() == Node.ELEMENT_NODE) {
                                Element elem = (Element) nNode;
                                String uid = elem.getAttribute(idList[index]);
                                System.out.println(idList[index] + ": " + uid);
                                pw.write( idList[index] + ": " + uid  + "\n");
                                for(String field : fields[index]){
                                        Node node =
elem.getElementsByTagName(field).item(0);
                                        String data = node.getTextContent();
                                        System.out.println(field + ": " + data);
                                        pw.write( field + ": " + data + "\n");
                                }
                        }
                }
                index++;
        }
        //Fajl iro bezarasa.
        pw.close();
    }
}
```

2b) adatmódosítás

A DomModify nem menti el a változtatásokat fájlba, csupán a konzolra írja ki a megváltoztatott xml dokumentumot.

```
package hu.domparse.QGNLD2;
```

```
import java.io.File;
import java.io.IOException;

import javax.xml.transform.*;
import javax.xml.transform.dom.DOMSource;
import javax.xml.transform.stream.StreamResult;
import javax.xml.parsers.*;

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

public class DomModifyQGNLD2 {
	public static void main(String argv[]) throws ParserConfigurationException,
	TransformerException, IOException, TransformerConfigurationException {
		try {
			File inputFile = new File("XMLQGNLD2.xml");

			 DocumentBuilderFactory docFactory =
DocumentBuilderFactory.newInstance();
		DocumentBuilder docBuilder = docFactory.newDocumentBuilder();

		Document doc = docBuilder.parse(inputFile);

		//Edzo hajszin valtoztatas.
		NodeList edzoLista = doc.getElementsByTagName("edzo");
```

```
//Edzolistan vegigmenni.
for(int i = 0; i < edzoLista.getLength(); i++) {
   Node edzo = doc.getElementsByTagName("edzo").item(i);
   NamedNodeMap attr = edzo.getAttributes();
   Node nodeAttr = attr.getNamedItem("edzoID");
   //ID Változtatás
   nodeAttr.setTextContent("e" + (i+1));
   //Az edzo gyerekelemein vegigmenni
   NodeList lista = edzo.getChildNodes();
   for(int t = 0; t < lista.getLength(); t++) {
            Node node = lista.item(t);
            if (node.getNodeType() == Node.ELEMENT_NODE) {
                     Element element = (Element) node;
                     //Ha a hajszin elemre er...
                     if ("hajszin".equals(element.getNodeName())) {
              //es a hajszin erteke barna...
              if ("barna".equals(element.getTextContent())){
                 //akkor voros lesz.
                 element.setTextContent("voros");
              }
         }
            }


   }
}

//Elso meccs torlese.
```

```
            NodeList meccsLista = doc.getElementsByTagName("meccs");

                    for(int i = 0; i < meccsLista.getLength(); i++) {
                            Node meccs = meccsLista.item(i);
                            Element element = (Element) meccs;
                                    if(element.getAttribute("meccsID").equals("m1")) {

element.getParentNode().removeChild(element);
                                            break;
                                    }
                    }

                    //Meccsekhez szabalytalansagok szama szama.

        for (int i = 0; i < meccsLista.getLength(); i++)
        {
           Node meccs = meccsLista.item(i);
           //szabalytalansagok elem letrehozasa
           Element szabalytalansagok = doc.createElement("szabalytalansagok");
           meccs.appendChild(szabalytalansagok);
           //Minden meccsnek 4et allit be.
           szabalytalansagok.appendChild(doc.createTextNode("4"));

        }

       //Stadion nev valtoztatas.
            NodeList stadionLista = doc.getElementsByTagName("stadion");
            for(int i = 0; i < stadionLista.getLength(); i++) {
```

```
                Node stadion = doc.getElementsByTagName("stadion").item(i);
                NamedNodeMap attr = stadion.getAttributes();
                Node nodeAttr = attr.getNamedItem("stadionID");
                nodeAttr.setTextContent("s" + (i+1));
                NodeList lista = stadion.getChildNodes();
                for(int t = 0; t < lista.getLength(); t++) {
                        Node node = lista.item(t);
                        if (node.getNodeType() == Node.ELEMENT_NODE) {
                                Element element = (Element) node;
                                if ("ferohelyek".equals(element.getNodeName())) {
                          if ("18000".equals(element.getTextContent())){
                             element.setTextContent("18181");
                          }
                      }
                        }
                  }
              }

              //Masodik stadion eltavolitasa.
                        for(int i = 0; i < stadionLista.getLength(); i++) {
                                Node stadion = stadionLista.item(i);
                                Element element = (Element) stadion;
                                        if(element.getAttribute("stadionID").equals("s2")) {

element.getParentNode().removeChild(element);
                                                break;
                                        }
                        }
```

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


            DOMSource source = new DOMSource(doc);


            System.out.println("--Results--");
            StreamResult consoleResult = new StreamResult(System.out);
            transformer.transform(source, consoleResult);
              }catch (Exception e){
        e.printStackTrace();
    }


        }
}
```

2c) adatlekérdezés

```
package hu.domparse.QGNLD2;


import java.io.*;


import javax.xml.parsers.DocumentBuilderFactory;
import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.ParserConfigurationException;
import org.xml.sax.SAXException;


import org.w3c.dom.Document;
```

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

public class DomQueryQGNLD2 {

	public static void main(String argv[]) throws SAXException,
	IOException, ParserConfigurationException {

		File xmlFile = new File("XMLQGNLD2.xml");

		DocumentBuilderFactory factory =
DocumentBuilderFactory.newInstance();
		DocumentBuilder dBuilder = factory.newDocumentBuilder();

		String[] jatekosFields = {
			"nev",
			"mezszam",
			"hajszin"
		};

		//A file parse-olasa.
		Document doc = dBuilder.parse(xmlFile);
		doc.getDocumentElement().normalize();

		//Edzok adatai (minusz az ID).
		System.out.println("Edzok adatai (minusz az ID)\n");
```

```
		NodeList edzoLista = doc.getElementsByTagName("edzo");
		//Forral vegig az edzokon.
		for(int i = 0; i < edzoLista.getLength(); i++) {
			Node e = edzoLista.item(i);
			if(e.getNodeType()==Node.ELEMENT_NODE) {
				Element edzo = (Element) e;
				NodeList nevList = edzo.getChildNodes();
				System.out.println("Edzo: ");
				//Az edzo gyerekelemein vegigmenni.
				for(int j = 0; j < nevList.getLength(); j++) {
					Node n = nevList.item(j);
					if (n.getNodeType()==Node.ELEMENT_NODE) {
						Element nev = (Element) n;
						System.out.println(nev.getTagName() + "= " +
nev.getTextContent());

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

		//12-es mezszamu jatekosok nevei.
		System.out.println("12-es mezszamu jatekosok nevei.\n");

		NodeList jatekosLista = doc.getElementsByTagName("jatekos");

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

```
                    Node jatekos = jatekosLista.item(i);
                    if(jatekos.getNodeType() == Node.ELEMENT_NODE) {
                        Element element = (Element) jatekos;
                        Node n =
element.getElementsByTagName("mezszam").item(0);
                        String mezszam = n.getTextContent();
                        if("12".equals(mezszam)) {
                            Node node =
element.getElementsByTagName("nev").item(0);
                            String nev = node.getTextContent();

                            System.out.println("Nev: " + nev);
                        }
                    }
                }
                System.out.println("");

                //10nel kisebb mezszamu jatekosok kiirasa.
                System.out.println("10nel kisebb mezszamu jatekosok kiirasa.\n");
                //jatekosokon vegigmenni.
                for(int i = 0; i < jatekosLista.getLength(); i++) {
                    Node jatekos = jatekosLista.item(i);
                    if(jatekos.getNodeType() == Node.ELEMENT_NODE) {
                        Element element = (Element) jatekos;
                        //mezszamok ellenorzese.
                        Node n =
element.getElementsByTagName("mezszam").item(0);
                        String mezszam = n.getTextContent();
                        //A mezszambol integer kinyerese.
```

```
                        int szam = Integer.parseInt(mezszam);
                        //Ha a mezszam kisebb, mint 10, akkor nev es mezszam
kiirasa.
                        if(szam < 10) {
                                Node node =
element.getElementsByTagName("nev").item(0);
                                String nev = node.getTextContent();

                                System.out.println("Nev: " + nev);
                                System.out.println("Mezszam: " + mezszam);
                                System.out.println("");
                        }
                }
        }

                        //28 evnel idosebb edzok kiirasa.
                        System.out.println("28 evnel idosebb edzok kiirasa.\n");
                        //edzookon vegigmenni.
                        for(int i = 0; i < edzoLista.getLength(); i++) {
                                Node edzo = edzoLista.item(i);
                                if(edzo.getNodeType() == Node.ELEMENT_NODE) {
                                        Element element = (Element) edzo;
                                        //kor ellenorzese.
                                        Node n =
element.getElementsByTagName("eletkor").item(0);
                                        String eletkor = n.getTextContent();
                                        //A korbol integer kinyerese.
                                        int kor = Integer.parseInt(eletkor);
```

```
                                        //Ha a kor nagyobb, mint 28, akkor nev
kiirasa.
                                        if(kor > 28) {
                                                Node node =
element.getElementsByTagName("nev").item(0);
                                                String nev = node.getTextContent();

                                                System.out.println("Nev: " + nev);
                                                System.out.println("eletkor: " +
eletkor);
                                                System.out.println("");
                                        }
                                }
                        }

                //Egyes csapatban jatszo jatekosok.
                System.out.println("Egyes csapatban jatszo jatekosok.\n");

                //Vegigmenni a jetekosokon.
                for (int i = 0; i < jatekosLista.getLength(); i++){
                        Node jatekos = jatekosLista.item(i);
                        if (jatekos.getNodeType() == Node.ELEMENT_NODE) {
                                Element element = (Element) jatekos;
                                String jatekosID = element.getAttribute("jatekosID");
                                //A csapat kulcs ellenorzese.
                                String csapata = element.getAttribute("csapat-idegen-
kulcs");
                                if (!csapata.contains("c1")) {
                                        continue;
```

```
                                }
                                System.out.println("jatekosID" + ": " + jatekosID);
                                //A jatekos mezovel a jatekos adatainak kiirasa.
                                for(String field : jatekosFields){
                                        Node node =
element.getElementsByTagName(field).item(0);
                                        String data = node.getTextContent();
                                        System.out.println(field + ": " + data);
                                }
                                System.out.println("");
                        }
                }




        }

}
```