JEGYZŐKÖNYV

Adatkezelés XML környezetben

Féléves feladat

A feladat címe

Készítette: **Kovács Krisztián**

Neptunkód: **WIQPM2**

Dátum: 2023.12.04.

Tartalom

Bevezetés

A feladat leírása

Ez a jegyzőkönyv a féléves feladat dokumentációjára készült az adatkezelés XML-ben című tárgyból. A feladatot a Visual Studio Code környezetben készítettem.

A féléves feladatom során a League of Legends nevezetű játék e-sport részének csapatainak adatnyilvántartó rendszerét hoztam létre. Különböző csapatok adatait tartalmazza, mint például maga a csapat és attribútumai, játékosok és attribútumai, csapatok edzői, megnyert versenyei, illetve szponzorjai. A modell öt egyedet tartalmaz, 1:1, 1:N, N:N kapcsolatokat, valamint összetett és többértékű tulajdonságokat is.

Az ER modell tervezésénél pontosítom az egyedeket, kapcsolatokat, illetve a tulajdonságokat.

1. feladat

1a) Az adatbázis ER modell tervezése

Egyedek: Csapatok, Játékosok, Szponzorok, Események, Edzők

Kapcsolatok: Csapat 1:N Játékosok, Csapat 1:1 Edzők, Csapat 1:N Események, Csapat N:N Szponzorok.

Tulajdonságok:

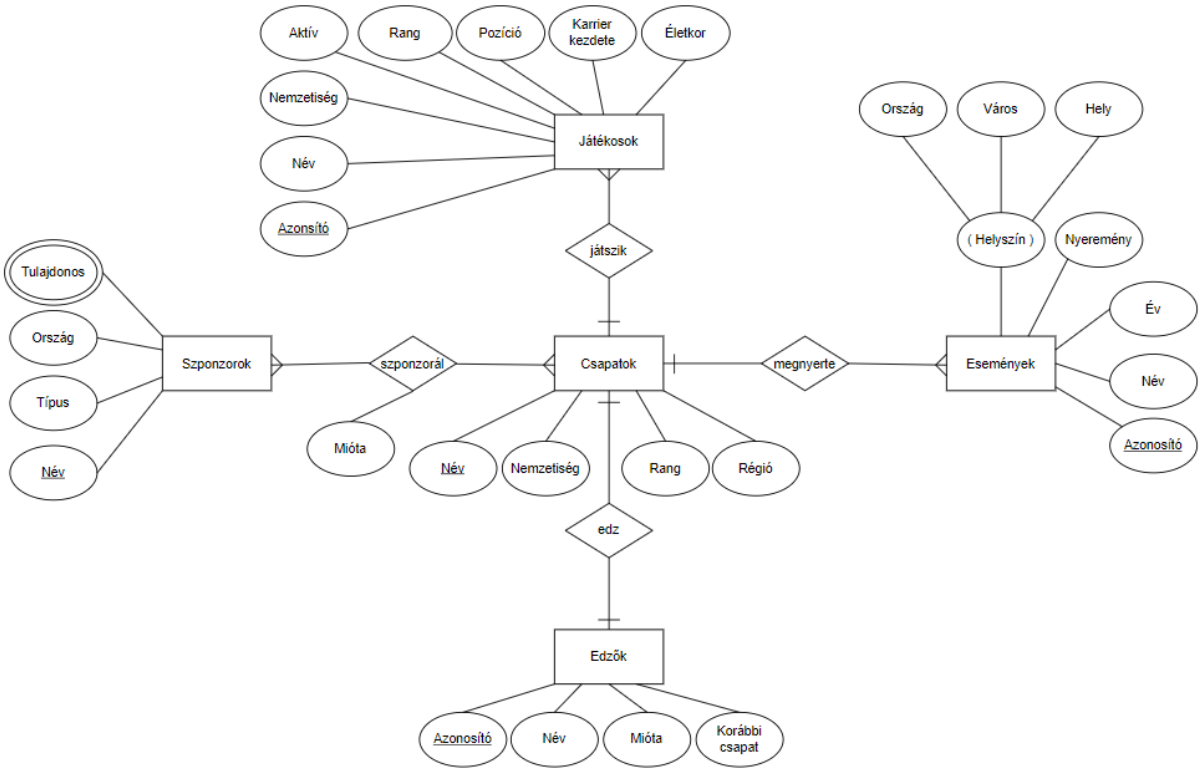
Csapatok: név (kulcs), nemzetiség, rang, régió

Játékosok: azonosító (kulcs), név, nemzetiség, aktív, rang, pozíció, karrier kezdete, életkor

Szponzorok: név (kulcs), típus, ország, tulajdonos (többértékű)

Edzők: azonosító (kulcs), név, mióta edző, korábbi csapat

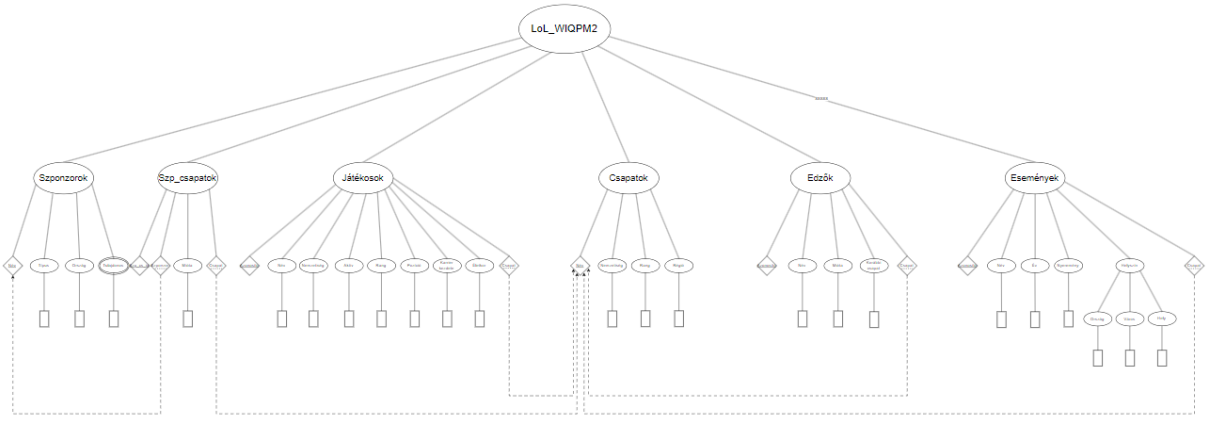
Események: azonosító (kulcs), név, év, nyeremény, helyszín (összetett: ország, város, hely)



1b) Az adatbázis konvertálása XDM modellre

Az ER modell alapján az XDM modellt szerkesztettem. Gyökérelemként megjelenik a LoL_WIQPM2, egyedeket és azok tulajdonságait ellipszissel ábrázoljuk, a többértékű tulajdonságot dupla körvonalúval. Téglalap jelöli a szöveg tartalmazását, a kulcsokat rombusz jelöli, a szöveg aláhúzva, az idegen kulcsok pedig szaggatott vonallal vannak jelölve. Az idegenkulcsok mutatnak a hivatkozott kulcsra.

Az N:N kapcsolat miatt létrejön a Mióta szponzor elem.



1c) Az XDM modell alapján XML dokumentum készítése

Az XDM alapján létrejön az XML. Minden többszörösen előforduló példányból létrehoztam minimum hármat. A kulcsok és idegen kulcsok attribútumok lesznek. A gyerekelemek lesznek a tulajdonságok.

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

<LoL_WIQPM2 xmlns:xs="http://www.w3.org/2001/XMLSchema-instance"
xs:noNamespaceSchemaLocation="XMLSchemaWIQPM2.xsd">

    <!-- Szponzorok -->
    <szponzorok>
        <szponzor sz_nev = "Hana Bank">
            <tipus>Bank</tipus>
            <sz_orszag>Dél-Korea</sz_orszag>
            <tulajdonos>Park Sung-ho</tulajdonos>
        </szponzor>
        <szponzor sz_nev = "Twitch">
            <tipus>Streamingszolgáltatás</tipus>
            <sz_orszag>USA</sz_orszag>
            <tulajdonos>Twitch Interactive</tulajdonos>
        </szponzor>
        <szponzor sz_nev = "BMW">
            <tipus>Járműipar</tipus>
            <sz_orszag>Németország</sz_orszag>
            <tulajdonos>Stefan Quandt</tulajdonos>
            <tulajdonos>Susanne Klatten</tulajdonos>
            <tulajdonos>BlackRock</tulajdonos>
        </szponzor>
    </szponzorok>

    <!-- Szp csapat-->
    <szp_csapatok>
        <szp_csapat szp_csapat_azonosito = "1" szp_nev = "BMW" csapat_nev =
"G2">
            <miota_szp>2018</miota_szp>
        </szp_csapat>
        <szp_csapat szp_csapat_azonosito = "2" szp_nev = "Twitch" csapat_nev =
"G2">
            <miota_szp>2020</miota_szp>
        </szp_csapat>
        <szp_csapat szp_csapat_azonosito = "3" szp_nev = "Hana Bank"
csapat_nev = "T1">
            <miota_szp>2022</miota_szp>
        </szp_csapat>
    </szp_csapatok>
```

```xml
<!-- Játékosok -->
<jatekosok>
    <jatekos j_azonosito = "1" csapat_nev = "G2">
        <j_nev>BrokenBlade</j_nev>
        <j_nemzetiseg>német</j_nemzetiseg>
        <aktiv>Igen</aktiv>
        <j_rang>77</j_rang>
        <pozicio>Top</pozicio>
        <karrier_kezdete>2016</karrier_kezdete>
        <eletkor>23</eletkor>
    </jatekos>
    <jatekos j_azonosito = "2" csapat_nev = "G2">
        <j_nev>Yike</j_nev>
        <j_nemzetiseg>svéd</j_nemzetiseg>
        <aktiv>Igen</aktiv>
        <j_rang>203</j_rang>
        <pozicio>Jungle</pozicio>
        <karrier_kezdete>2019</karrier_kezdete>
        <eletkor>23</eletkor>
    </jatekos>
    <jatekos j_azonosito = "3" csapat_nev = "G2">
        <j_nev>Caps</j_nev>
        <j_nemzetiseg>norvég</j_nemzetiseg>
        <aktiv>Igen</aktiv>
        <j_rang>21</j_rang>
        <pozicio>Mid</pozicio>
        <karrier_kezdete>2015</karrier_kezdete>
        <eletkor>24</eletkor>
    </jatekos>
    <jatekos j_azonosito = "4" csapat_nev = "G2">
        <j_nev>Hans Sama</j_nev>
        <j_nemzetiseg>francia</j_nemzetiseg>
        <aktiv>Igen</aktiv>
        <j_rang>83</j_rang>
        <pozicio>Bottom</pozicio>
        <karrier_kezdete>2014</karrier_kezdete>
        <eletkor>24</eletkor>
    </jatekos>
    <jatekos j_azonosito = "5" csapat_nev = "G2">
        <j_nev>Mikyx</j_nev>
        <j_nemzetiseg>szlovén</j_nemzetiseg>
        <aktiv>Igen</aktiv>
        <j_rang>31</j_rang>
        <pozicio>Support</pozicio>
        <karrier_kezdete>2014</karrier_kezdete>
        <eletkor>25</eletkor>
    </jatekos>
    <jatekos j_azonosito = "6" csapat_nev = "T1">
        <j_nev>Faker</j_nev>
```

```
            <j_nemzetiseg>dél koreai</j_nemzetiseg>
            <aktiv>Igen</aktiv>
            <j_rang>2</j_rang>
            <pozicio>Mid</pozicio>
            <karrier_kezdete>2013</karrier_kezdete>
            <eletkor>27</eletkor>
        </jatekos>
    </jatekosok>

    <!-- Csapatok -->
    <csapatok>
        <csapat cs_nev = "T1">
            <cs_nemzetiseg>Dél-Korea</cs_nemzetiseg>
            <cs_rang>1</cs_rang>
            <regio>LCK</regio>
        </csapat>
        <csapat cs_nev = "G2">
            <cs_nemzetiseg>Németország</cs_nemzetiseg>
            <cs_rang>6</cs_rang>
            <regio>LEC</regio>
        </csapat>
        <csapat cs_nev = "Mad Lions">
            <cs_nemzetiseg>Spanyolország</cs_nemzetiseg>
            <cs_rang>16</cs_rang>
            <regio>LEC</regio>
        </csapat>
        <csapat cs_nev = "JD Gaming">
            <cs_nemzetiseg>Kína</cs_nemzetiseg>
            <cs_rang>4</cs_rang>
            <regio>LPL</regio>
        </csapat>
    </csapatok>

    <!-- Edzők -->
    <edzok>
        <edzo e_azonosito = "1" csapat_nev = "G2">
            <e_nev>Dylan Falco</e_nev>
            <miota>2021</miota>
            <korabbi_csapat>Schalke 04</korabbi_csapat>
        </edzo>
        <edzo e_azonosito = "2" csapat_nev = "T1">
            <e_nev>kK0ma</e_nev>
            <miota>2023</miota>
            <korabbi_csapat>DWG Kia</korabbi_csapat>
        </edzo>
        <edzo e_azonosito = "3" csapat_nev = "Mad Lions">
            <e_nev>Mac</e_nev>
            <miota>2019</miota>
            <korabbi_csapat>Splyce</korabbi_csapat>
```

```xml
        </edzo>
        <edzo e_azonosito = "4" csapat_nev = "JD Gaming">
            <e_nev>Mafa</e_nev>
            <miota>2023</miota>
            <korabbi_csapat>Gen.G</korabbi_csapat>
        </edzo>
    </edzok>

    <!-- Események -->
    <esemenyek>
        <esemeny es_azonosito = "1" csapat_nev = "G2">
            <es_nev>LEC 2023 Season Finals</es_nev>
            <ev>2023</ev>
            <nyeremeny>65000</nyeremeny>
            <helyszin>
                <es_orszag>Franciaország</es_orszag>
                <varos>Montpellier</varos>
                <hely>Sud de France Arena</hely>
            </helyszin>
        </esemeny>
        <esemeny es_azonosito = "2" csapat_nev = "JD Gaming">
            <es_nev>MSI 2023</es_nev>
            <ev>2023</ev>
            <nyeremeny>50000</nyeremeny>
            <helyszin>
                <es_orszag>Anglia</es_orszag>
                <varos>London</varos>
                <hely>Queen Elizabeth Olympic Park</hely>
            </helyszin>
        </esemeny>
        <esemeny es_azonosito = "3" csapat_nev = "T1">
            <es_nev>Worlds 2023</es_nev>
            <ev>2023</ev>
            <nyeremeny>445000</nyeremeny>
            <helyszin>
                <es_orszag>Dél-Korea</es_orszag>
                <varos>Seul</varos>
                <hely>Gocheok Sky Dome</hely>
            </helyszin>
        </esemeny>
    </esemenyek>

</LoL_WIQPM2>
```

1d) Az XML dokumentum alapján XMLSchema készítése

Az XML Schema validálta a létrehozott XML dokumentumot. Kigyűjtöttem az egyszerű típusokat, melyekre később hivatkozni fogok, létrehoztam saját típusokat, amelyben megszorítások vannak az adatra, mint például a pozíció 5 féle lehet: top, mid, bottom, support, jungle. Komplex típusokat létrehoztam az egyedekre, felépítettem a sorrendiséget. Ezután következett az elsődleges kulcsok, idegen kulcsok, illetve az 1:1 kapcsolat megvalósítása.

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

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

<!-- Egyszerű típusok -->
<xs:element name="tipus" type="xs:string" />
<xs:element name="sz_orszag" type="xs:string" />
<xs:element name="tulajdonos" type="xs:string" />

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

<xs:element name="j_nev" type="xs:string" />
<xs:element name="j_nemzetiseg" type="xs:string" />
<xs:element name="aktiv" type="aktivTipus" />
<xs:element name="j_rang" type="xs:string" />
<xs:element name="pozicio" type="pozicioTipus" />
<xs:element name="karrier_kezdete" type="xs:gYear" />
<xs:element name="eletkor" type="xs:integer" />

<xs:element name="cs_nemzetiseg" type="xs:string" />
<xs:element name="cs_rang" type="xs:integer" />
<xs:element name="regio" type="xs:string" />

<xs:element name="e_nev" type="xs:string" />
<xs:element name="miota" type="xs:gYear" />
<xs:element name="korabbi_csapat" type="xs:string" />

<xs:element name="es_nev" type="xs:string" />
<xs:element name="ev" type="xs:gYear" />
<xs:element name="nyeremeny" type="xs:integer" />
<xs:element name="es_orszag" type="xs:string" />
<xs:element name="varos" type="xs:string" />
<xs:element name="hely" type="xs:string" />


 <!-- Saját típusok létrehozása  -->
<xs:simpleType name="aktivTipus">
    <xs:restriction base="xs:string">
```

```
        <xs:enumeration value="Igen" />
        <xs:enumeration value="Nem" />
    </xs:restriction>
</xs:simpleType>

<xs:simpleType name="pozicioTipus">
    <xs:restriction base="xs:string">
        <xs:enumeration value="Top" />
        <xs:enumeration value="Mid" />
        <xs:enumeration value="Bottom" />
        <xs:enumeration value="Support" />
        <xs:enumeration value="Jungle" />
    </xs:restriction>
</xs:simpleType>


<!-- Komplex típushoz sajat típus létrehozása-->
<xs:complexType name="szponzorTipus">
    <xs:sequence>
        <xs:element ref="tipus" />
        <xs:element ref="sz_orszag" />
        <xs:element name="tulajdonos" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute name="sz_nev" type="xs:string" use="required" />
</xs:complexType>

<xs:complexType name="szp_csapatTipus">
    <xs:sequence>
        <xs:element ref="miota_szp" />
    </xs:sequence>
    <xs:attribute name="szp_csapat_azonosito" type="xs:integer" use="required"
/>
    <xs:attribute name="szp_nev" type="xs:string" use="required" />
    <xs:attribute name="csapat_nev" type="xs:string" use="required" />
</xs:complexType>

<xs:complexType name="jatekosTipus">
    <xs:sequence>
        <xs:element ref="j_nev" />
        <xs:element ref="j_nemzetiseg" />
        <xs:element ref="aktiv" />
        <xs:element ref="j_rang" />
        <xs:element ref="pozicio" />
        <xs:element ref="karrier_kezdete" />
        <xs:element ref="eletkor" />
    </xs:sequence>
    <xs:attribute name="j_azonosito" type="xs:integer" use="required" />
    <xs:attribute name="csapat_nev" type="xs:string" use="required" />
</xs:complexType>
```

```xml
<xs:complexType name="csapatTipus">
    <xs:sequence>
        <xs:element ref="cs_nemzetiseg" />
        <xs:element ref="cs_rang" />
        <xs:element ref="regio" />
    </xs:sequence>
    <xs:attribute name="cs_nev" type="xs:string" use="required" />
</xs:complexType>

<xs:complexType name="edzoTipus">
    <xs:sequence>
        <xs:element ref="e_nev" />
        <xs:element ref="miota" />
        <xs:element ref="korabbi_csapat" />
    </xs:sequence>
    <xs:attribute name="e_azonosito" type="xs:integer" use="required" />
    <xs:attribute name="csapat_nev" type="xs:string" use="required" />
</xs:complexType>

<xs:complexType name="esemenyTipus">
    <xs:sequence>
        <xs:element ref="es_nev" />
        <xs:element ref="ev" />
        <xs:element ref="nyeremeny" />
        <xs:element name = "helyszin">
            <xs:complexType mixed = "true">
                <xs:sequence>
                    <xs:element name="es_orszag" type="xs:string" />
                    <xs:element name="varos" type="xs:string" />
                    <xs:element name="hely" type="xs:string" />
                </xs:sequence>
            </xs:complexType>
        </xs:element>
    </xs:sequence>
    <xs:attribute name="es_azonosito" type="xs:integer" use="required" />
    <xs:attribute name="csapat_nev" type="xs:string" use="required" />
</xs:complexType>


<!-- Az elemek sorrendisége a gyökérelemtől -->
<xs:element name="LoL_WIQPM2">
    <xs:complexType>
        <xs:sequence>

            <xs:element name="szponzorok">
                <xs:complexType mixed = "true">
                    <xs:sequence>
```

```
                    <xs:element name="szponzor" type="szponzorTipus"
minOccurs="0" maxOccurs="unbounded"/>
                </xs:sequence>
            </xs:complexType>
        </xs:element>
        <xs:element name="szp_csapatok">
            <xs:complexType mixed = "true">
                <xs:sequence>
                    <xs:element name="szp_csapat" type="szp_csapatTipus"
minOccurs="0" maxOccurs="unbounded"/>
                </xs:sequence>
            </xs:complexType>
        </xs:element>
        <xs:element name="jatekosok">
            <xs:complexType mixed = "true">
                <xs:sequence>
                    <xs:element name="jatekos" type="jatekosTipus"
minOccurs="0" maxOccurs="unbounded"/>
                </xs:sequence>
            </xs:complexType>
        </xs:element>
        <xs:element name="csapatok">
            <xs:complexType mixed = "true">
                <xs:sequence>
                    <xs:element name="csapat" type="csapatTipus"
minOccurs="0" maxOccurs="unbounded"/>
                </xs:sequence>
            </xs:complexType>
        </xs:element>
        <xs:element name="edzok">
            <xs:complexType mixed = "true">
                <xs:sequence>
                    <xs:element name="edzo" type="edzoTipus" minOccurs="0"
maxOccurs="unbounded"/>
                </xs:sequence>
            </xs:complexType>
        </xs:element>
        <xs:element name="esemenyek">
            <xs:complexType mixed = "true">
                <xs:sequence>
                    <xs:element name="esemeny" type="esemenyTipus"
minOccurs="0" maxOccurs="unbounded"/>
                </xs:sequence>
            </xs:complexType>
        </xs:element>

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

```
    <!-- Elsődleges kulcsok létrehozása-->
    <xs:key name="szponzor_kulcs">
        <xs:selector xpath="szponzor"/>
        <xs:field xpath="@sz_nev"/>
    </xs:key>
    <xs:key name="szp_csapat_kulcs">
        <xs:selector xpath="szp_csapat"/>
        <xs:field xpath="@szp_csapat_azonosito"/>
    </xs:key>
    <xs:key name="jatekos_kulcs">
        <xs:selector xpath="jatekos"/>
        <xs:field xpath="@j_azonosito"/>
    </xs:key>
    <xs:key name="csapat_kulcs">
        <xs:selector xpath="csapat"/>
        <xs:field xpath="@cs_nev"/>
    </xs:key>
    <xs:key name="edzo_kulcs">
        <xs:selector xpath="edzo"/>
        <xs:field xpath="@e_azonosito"/>
    </xs:key>
    <xs:key name="esemeny_kulcs">
        <xs:selector xpath="esemeny"/>
        <xs:field xpath="@es_azonosito"/>
    </xs:key>


    <!-- Idegen kulcsok létrehozása -->
    <xs:keyref name="szp_csapat_szp_ikulcs" refer="szponzor_kulcs">
        <xs:selector xpath="szp_csapat"/>
        <xs:field xpath="@szponzor"/>
    </xs:keyref>
    <xs:keyref name="szp_csapat_csapat_ikulcs" refer="csapat_kulcs">
        <xs:selector xpath="szp_csapat"/>
        <xs:field xpath="@csapat"/>
    </xs:keyref>
    <xs:keyref name="jatekos_csapat_ikulcs" refer="csapat_kulcs">
        <xs:selector xpath="jatekos"/>
        <xs:field xpath="@csapat"/>
    </xs:keyref>
    <xs:keyref name="edzo_csapat_ikulcs" refer="csapat_kulcs">
        <xs:selector xpath="edzo"/>
        <xs:field xpath="@csapat"/>
    </xs:keyref>
    <xs:keyref name="esemeny_csapat_ikulcs" refer="esemeny_kulcs">
        <xs:selector xpath="esemeny"/>
        <xs:field xpath="@csapat"/>
    </xs:keyref>
```

```
    <!-- 1:1 kapcsolat -->
    <xs:unique name="csapat_edzo_egyedi">
        <xs:selector xpath="edzo"/>
        <xs:field xpath="@csapat"/>
    </xs:unique>

</xs:element>


</xs:schema>
```

2. feladat

2a) adatolvasás

Beolvasom a fájl tartalmát és a printNode metódus segítségével kiírom konzolra formázva azt.

```
package hu.domparse.wiqpm2;

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

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

public class DomReadWIQPM2 {
    public static void main(String[] args) {
        try {
                File xmlf= new File("XMLWIQPM2.xml");
                DocumentBuilderFactory dbFactory =
DocumentBuilderFactory.newInstance();
                DocumentBuilder dBuilder = dbFactory.newDocumentBuilder();
                Document document = dBuilder.parse(xmlf);
                document.getDocumentElement().normalize();

                //Fa struktúra kiírása
                printNode(document.getDocumentElement(), 0);

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

```
    private static void printNode(Node node, int depth)
    {
        String indent = " ".repeat(depth * 5);
        if (node.getNodeType() == Node.ELEMENT_NODE)
        {
            System.out.println(indent + "<" + node.getNodeName() + ">");
            NodeList nodeL = node.getChildNodes();
            for (int i = 0; i < nodeL.getLength(); i++)
            {
                printNode(nodeL.item(i), depth + 1);
            }
            System.out.println(indent + "</" + node.getNodeName() + ">");
        }
        else if (node.getNodeType() == Node.TEXT_NODE &&
!node.getTextContent().trim().isEmpty())
        {
            System.out.println(indent + node.getTextContent().trim());
        }
    }
}
```

2b) adatmódosítás

Öt adatmódosítást végeztem, kommentben leírva, hogy mit módosít. A végén konzolra kiíratva leellenőriztem a végeredményt.

```
package hu.domparse.wiqpm2;

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


public class DomModifyWIQPM2 {
    public static void main(String[] args) {

        try {
```

```
            //xml file megnyitása
            File xmlf = new File("XMLWIQPM2.xml");

            //dokumentum létrehozasa és normalizálása
            DocumentBuilder documentBuilder =
DocumentBuilderFactory.newInstance().newDocumentBuilder();
            Document document = documentBuilder.parse(xmlf);
            document.getDocumentElement().normalize();

            //metodusok
            query1(document);
            query2(document);
            query3(document);
            query4(document);
            query5(document);

            //a módosított adatok kiírasa a konzolra
            TransformerFactory transformerFactory =
TransformerFactory.newInstance();
            Transformer transformer = transformerFactory.newTransformer();
            DOMSource source = new DOMSource(document);
            System.out.println("Módosítás után:\n");
            StreamResult consoleResult = new StreamResult(System.out);
            transformer.transform(source, consoleResult);

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

    public static void query1(Document document){
        // 1. Caps nevű játékos rangjának a módosítása
        NodeList nList = document.getElementsByTagName("jatekos");
        for (int i = 0; i < nList.getLength(); i++) {
            Node nNode = nList.item(i);
            if (nNode.getNodeType()==Node.ELEMENT_NODE) {
                Element elem = (Element) nNode;
                Node node1 = elem.getElementsByTagName("j_nev").item(0);
                String text1 = node1.getTextContent();
                if("Caps".equals(text1)){
                    Node node2 = elem.getElementsByTagName("j_rang").item(0);
                    node2.setTextContent("5");
                }
            }
        }
    }

    public static void query2(Document document){
```

```java
        // 2. 3. id-jű játékos csapatának változtatása
        NodeList nList = document.getElementsByTagName("jatekos");
        for (int i = 0; i < nList.getLength(); i++) {
            Node nNode = nList.item(i);
            if (nNode.getNodeType() == Node.ELEMENT_NODE) {
                NamedNodeMap attr = nNode.getAttributes();
                Node nodeAttrID = attr.getNamedItem("j_azonosito");
                Node nodeAttrCsapat = attr.getNamedItem("csapat_nev");
                if (nodeAttrID.getTextContent().equals("3")) {
                    nodeAttrCsapat.setTextContent("Mad Lions");
                }
            }
        }
    }

    public static void query3(Document document){
        // 3. Szponzorált csapat modosítása
        NodeList nList = document.getElementsByTagName("szp_csapat");
        for (int i = 0; i < nList.getLength(); i++) {
            Node nNode = nList.item(i);
            if (nNode.getNodeType()==Node.ELEMENT_NODE) {
                NamedNodeMap attribute = nNode.getAttributes();
                Node nodeAttrCsapat = attribute.getNamedItem("csapat_nev");
                if (nodeAttrCsapat.getTextContent().equals("G2")) {
                    nodeAttrCsapat.setTextContent("T1");
                    Element elem = (Element) nNode;
                    Node node1 =
elem.getElementsByTagName("miota_szp").item(0);
                    node1.setTextContent("2023");
                }
            }
        }
    }

    public static void query4(Document document){
        // 4. Hana Bank nevű szponzor nevének változtatása
        NodeList nList = document.getElementsByTagName("szponzor");
        for (int i = 0; i < nList.getLength(); i++) {
            Node nNode = nList.item(i);
            if (nNode.getNodeType() == Node.ELEMENT_NODE) {
                Element elem = (Element) nNode;
                String sz_nev = elem.getAttribute("sz_nev");
                if (sz_nev.equals("Hana Bank")) {
                    elem.setAttribute("sz_nev", "Anah Bank");
                }
            }
        }
    }
```

```
    public static void query5(Document document){
        // 5. LEC régiót képviselő csapatok régiójának változtatása LCL-re
        NodeList nList = document.getElementsByTagName("csapat");
        for (int i = 0; i < nList.getLength(); i++) {
            Node nNode = nList.item(i);
            if (nNode.getNodeType() == Node.ELEMENT_NODE) {
                Element elem = (Element) nNode;
                Node regio = elem.getElementsByTagName("regio").item(0);
                String text1 = regio.getTextContent();
                if (text1.equals("LEC")) {
                    regio.setTextContent("LCL");
                }
            }
        }
    }

}
```

2c) adatlekérdezés

Öt lekérdezést készítettem, kommentben a lekérdezés magyarázata megtalálható. A konzolra írja ki a lekérdezések eredményét.

```
package hu.domparse.wiqpm2;

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

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

import org.w3c.dom.Document;
import org.w3c.dom.Element;
import org.w3c.dom.Node;
import org.w3c.dom.NodeList;
import org.xml.sax.SAXException;

public class DomQueryWIQPM2 {

    public static void main(String[] args) throws IOException,
ParserConfigurationException, SAXException {
        try {
            //xml file megnyitasa
            File xmlf = new File("XMLWIQPM2.xml");

            //dokumentum létrehozása és normalizálása
```

```
            DocumentBuilder documentBuilder =
DocumentBuilderFactory.newInstance().newDocumentBuilder();
            Document document = documentBuilder.parse(xmlf);
            document.getDocumentElement().normalize();

            //metodusok
            lekerdezes1(document);
            lekerdezes2(document);
            lekerdezes3(document);
            lekerdezes4(document);
            lekerdezes5(document);
        } catch (IOException e) {
            e.printStackTrace();
        } catch (ParserConfigurationException e) {
            e.printStackTrace();
        } catch (SAXException e) {
            e.printStackTrace();
        }
    }

    public static void lekerdezes1(Document document) {
        try {
            System.out.println("\n1. A LEC régiót képviselő csapatok:\n");
            NodeList nList = document.getElementsByTagName("csapat");
            for (int i = 0; i < nList.getLength(); i++) {
                Node nNode = nList.item(i);
                if (nNode.getNodeType()==Node.ELEMENT_NODE) {
                    Element elem = (Element) nNode;
                    Node node1 = elem.getElementsByTagName("regio").item(0);
                    String text1 = node1.getTextContent();
                    if("LEC".equals(text1)){
                        String csapat_nev = elem.getAttribute("cs_nev");
                        System.out.println("Csapat neve: " + csapat_nev);
                        System.out.println("");
                    }
                }
            }
        } catch (NullPointerException e) {
            e.printStackTrace();
        }
        System.out.println("");
        System.out.println("");
    }

    public static void lekerdezes2(Document document) {
        try {
            System.out.println("\n2. Azon játékosok nevei és csapata, akinek
az életkora 23 vagy kevesebb:\n");
            NodeList nList = document.getElementsByTagName("jatekos");
```

```
            for (int i = 0; i < nList.getLength(); i++) {
                Node nNode = nList.item(i);
                if (nNode.getNodeType()==Node.ELEMENT_NODE) {
                    Element elem = (Element) nNode;
                    Node node1 = elem.getElementsByTagName("eletkor").item(0);
                    String text1 = node1.getTextContent();
                    if(Integer.valueOf(text1) <= 23){
                        Node node2 =
elem.getElementsByTagName("j_nev").item(0);
                        String text2 = node2.getTextContent();
                        String csapat_nev = elem.getAttribute("csapat_nev");
                        System.out.println("Játékos neve: " + text2);
                        System.out.println("Csapat neve: " + csapat_nev);
                        System.out.println("");
                    }
                }
            }
        } catch (NullPointerException e) {
            e.printStackTrace();
        }
        System.out.println("");
        System.out.println("");
    }

    public static void lekerdezes3(Document document) {
        try {
            System.out.println("\n3. Azon csapatok neve és rangja, amelyek
megnyertek egy 100.000-nál nagyobb díjazású versenyt, illetve az esemény
neve:\n");
            NodeList nList = document.getElementsByTagName("esemeny");
            NodeList nList2 = document.getElementsByTagName("csapat");

            for (int i = 0; i < nList.getLength(); i++) {
                Node nNode = nList.item(i);
                if (nNode.getNodeType()==Node.ELEMENT_NODE) {
                    Element elem = (Element) nNode;
                    Node nyeremeny =
elem.getElementsByTagName("nyeremeny").item(0);
                    String text1 = nyeremeny.getTextContent();
                    System.out.println(text1);
                    if(Integer.valueOf(text1) >= 100000){
                        String csapat_nev = elem.getAttribute("csapat_nev");
                        Node esemeny_neve =
elem.getElementsByTagName("es_nev").item(0);
                        String text2 = esemeny_neve.getTextContent();
                        for(int j = 0; j < nList2.getLength(); j++){
                            Node nNode2 = nList2.item(j);
                            if (nNode2.getNodeType()==Node.ELEMENT_NODE) {
```

```
                                Element elem2 = (Element) nNode2;
                                if(csapat_nev.equals(elem2.getAttribute("cs_ne
v"))){
                                    Node rang =
elem2.getElementsByTagName("cs_rang").item(0);
                                    String text3 = rang.getTextContent();
                                    System.out.println("Esemény: " + text2);
                                    System.out.println("Csapat neve: " +
csapat_nev);
                                    System.out.println("Rang: " + text3);
                                    System.out.println("");
                                }
                            }
                        }
                    }
                }
            } catch (NullPointerException e) {
                e.printStackTrace();
            }
            System.out.println("");
            System.out.println("");
        }

        public static void lekerdezes4(Document document) {
            try {
                System.out.println("\n4. T1 csapat által nyert események:\n");
                NodeList nList = document.getElementsByTagName("esemeny");
                for (int i = 0; i < nList.getLength(); i++) {
                    Node nNode = nList.item(i);
                    if (nNode.getNodeType()==Node.ELEMENT_NODE) {
                        Element elem = (Element) nNode;
                        String csapat_nev = elem.getAttribute("csapat_nev");
                        if("T1".equals(csapat_nev)){
                            Node esemeny_neve =
elem.getElementsByTagName("es_nev").item(0);
                            String text1 = esemeny_neve.getTextContent();
                            System.out.println("Esemény neve: " + text1);
                            System.out.println("");
                        }
                    }
                }
            } catch (NullPointerException e) {
                e.printStackTrace();
            }
            System.out.println("");
            System.out.println("");
        }
```

```
    public static void lekerdezes5(Document document) {
        try {
            System.out.println("\n5. A G2 csapat játékosainak adatai:\n");
            NodeList nList = document.getElementsByTagName("jatekos");
            for (int i = 0; i < nList.getLength(); i++) {
                Node nNode = nList.item(i);
                if (nNode.getNodeType()==Node.ELEMENT_NODE) {
                    Element elem = (Element) nNode;
                    String csapat_nev = elem.getAttribute("csapat_nev");
                    if("G2".equals(csapat_nev)){
                        String azonosito = elem.getAttribute("j_azonosito");
                        System.out.println("Azonosító: " + azonosito);
                        System.out.println("Csapat neve: " + csapat_nev);
                        NodeList nList2 = nNode.getChildNodes();
                        for(int j = 1; j < nList2.getLength(); j++){
                            Node nNode2 = nList2.item(j);
                            if (nNode2.getNodeType()==Node.ELEMENT_NODE) {
                                Element elem2 = (Element) nNode2;
                                String text2 = elem2.getTextContent();
                                System.out.print(elem2.getNodeName() + ": " +
text2 + ", ");
                            }
                        }
                        System.out.println("");
                    }
                }
            }
        } catch (NullPointerException e) {
            e.printStackTrace();
        }
        System.out.println("");
        System.out.println("");
    }
}
```

2d) adatírás

Kiírom konzolra az xml fájl tartalmát és utána egy új fájlba lementem azt.

```
package hu.domparse.wiqpm2;

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

public class DomWriteWIQPM2 {
    public static void main(String[] args) {
        try {
                File xmlf= new File("XMLWIQPM2.xml");
                DocumentBuilderFactory dbFactory =
DocumentBuilderFactory.newInstance();
                DocumentBuilder dBuilder = dbFactory.newDocumentBuilder();
                Document document = dBuilder.parse(xmlf);
                document.getDocumentElement().normalize();

                //Fa struktúra kiírása
                printNode(document.getDocumentElement(), 0);

                TransformerFactory transformerFactory =
TransformerFactory.newInstance();
                Transformer transformer = transformerFactory.newTransformer();
                transformer.setOutputProperty(OutputKeys.INDENT, "yes"); //
Bekezdések hozzáadása
                transformer.setOutputProperty("{http://xml.apache.org/xslt}ind
ent-amount", "5"); // Indentálási mélység

                DOMSource source = new DOMSource(document);
                StreamResult result = new StreamResult(new
File("XMLWIQPM21.xml"));
                transformer.transform(source, result);
            }
            catch (Exception e)
            {
                e.printStackTrace();
            }
        }

    private static void printNode(Node node, int depth)
    {
        String indent = " ".repeat(depth * 5);
        if (node.getNodeType() == Node.ELEMENT_NODE)
        {
            System.out.println(indent + "<" + node.getNodeName() + ">");
            NodeList nodeL = node.getChildNodes();
            for (int i = 0; i < nodeL.getLength(); i++)
            {
                printNode(nodeL.item(i), depth + 1);
            }
```

```
            System.out.println(indent + "</" + node.getNodeName() + ">");
        }
        else if (node.getNodeType() == Node.TEXT_NODE &&
!node.getTextContent().trim().isEmpty())
        {
            System.out.println(indent + node.getTextContent().trim());
        }
    }
}
```