**Jegyzőkönyv**

Adatkezelés XML környezetben Féléves feladat Labdarúgó bajnokság

Tartalomjegyzék

[A feladat: 3](#_Toc121231301)

[Egyedek: 3](#_Toc121231302)

[Kapcsolatok: 3](#_Toc121231303)

[1.Feladat 4](#_Toc121231304)

[**1b)** Az ER modell konvertálása XDM modellre 4](#_Toc121231305)

[**1c)** Az XDM modell alapján validált XML dokumentum készítése 4](#_Toc121231306)

[1d) Az XML dokumentum alapján XMLSchema készítése (saját típus használata) – fájlnév: XMLSchemaNeptunkod.xsd 8](#_Toc121231307)

[2.Feladat 13](#_Toc121231308)

[2a) Adatolvasás: 13](#_Toc121231309)

[Leírás: 13](#_Toc121231310)

[Futási eredmény: 19](#_Toc121231311)

[2b) Adatlekérdezés: 19](#_Toc121231312)

[Leírás: 19](#_Toc121231313)

[Futási eredmény: 23](#_Toc121231314)

[2c) Adatmódosítás: 24](#_Toc121231315)

[Leírás: 24](#_Toc121231316)

[Futási eredmény: 26](#_Toc121231317)

# A feladat:

A feladathoz egy labdarúgó bajnokság szezon végi adatbázis rendszerét választottam.

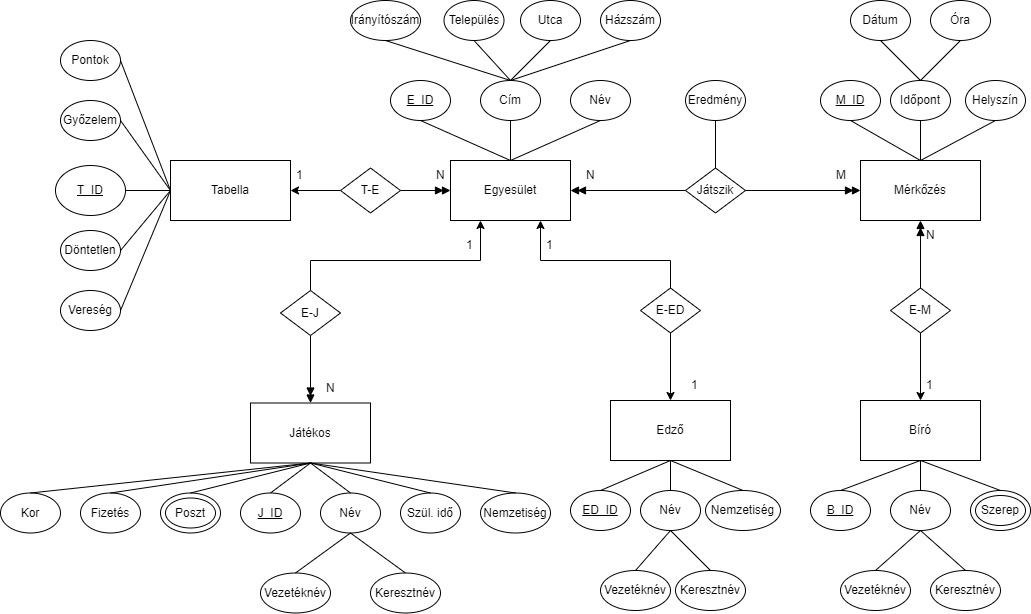
## Egyedek:

Tabella, Egyesület, Játékos, Mérkőzés, Bíró, Edző

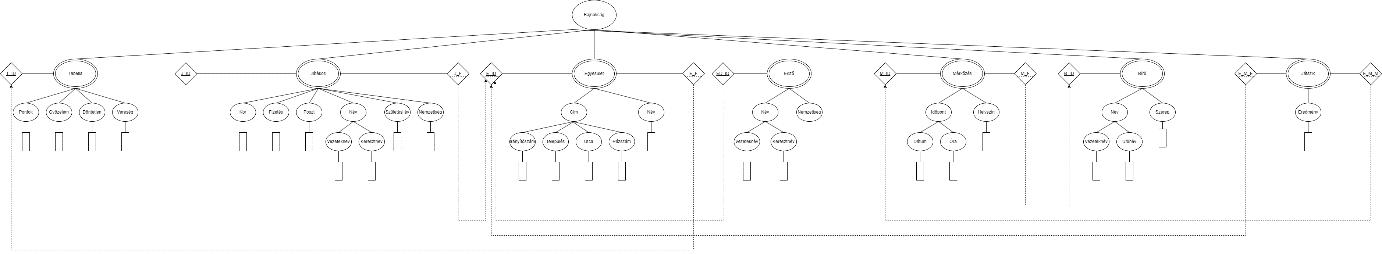
## Kapcsolatok:

* **Tabella - Egyesület:** 1-N kapcsolat, mert egy egyesület egyszer szerepelhet a tabellán, a tabellán viszont több Egyesület is szerepel (az összes egyesület szerepel a tabellán).
* **Egyesület – Játékos:** 1-N kapcsolat, mivel egy játékos egyszerre csak egy egyesület tagja lehet, viszont egy egyesülethez több játékos is tartozik.
* **Mérkőzés – Egyesület:** N-M kapcsolat, egy mérkőzéshez többegyesület tartozik (pontosan 2), egy egyesület több mérkőzést is játszik az idény során. Neve: Játszik saját adattagja a Mikor.
* **Bíró - Mérkőzés:** 1-N kapcsolat, mert egy Bíró több különbözőmérkőzésen is fújhat/dolgozhat, de egy adott mérkőzésen egy Bíró csak egyszer szerepelhet.
* **Egyesület - Edző:** 1-1 kapcsolat, mivel egy egyesületnek egyszerre egyedzője lehet és egy edző egy időben egy csapathoz tartozik.

# 1.Feladat

1 a) Az adatbázis ER modell készítése: 

## **1b)** Az ER modell konvertálása XDM modellre



## **1c)** Az XDM modell alapján validált XML dokumentum készítése

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

*<bajnoksag xsi:noNamespaceSchemaLocation="XMLSchemaWH85ZH.xsd" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">*

*<!-- Tabella -->*

*<tabella T\_ID="1">*

*<pontok>18</pontok>*

*<gyozelem>5</gyozelem>*

*<dontetlen>3</dontetlen>*

*<vereseg>2</vereseg>*

*</tabella>*

*<tabella T\_ID="2">*

*<pontok>15</pontok>*

*<gyozelem>3</gyozelem>*

*<dontetlen>6</dontetlen>*

*<vereseg>1</vereseg>*

*</tabella>*

*<tabella T\_ID="3">*

*<pontok>14</pontok>*

*<gyozelem>3</gyozelem>*

*<dontetlen>5</dontetlen>*

*<vereseg>2</vereseg>*

*</tabella>*

*<!-- Játékosok -->*

*<jatekos J\_ID="1" J\_F="1">*

*<kor>31</kor>*

*<fizetes>100</fizetes>*

*<poszt>kozeppalyas</poszt>*

*<vezeteknev>Kante</vezeteknev>*

*<keresztnev>N Golo</keresztnev>*

*<szulev>1991</szulev>*

*<nemzetiseg>Francia</nemzetiseg>*

*</jatekos>*

*<jatekos J\_ID="2" J\_F="2">*

*<kor>34</kor>*

*<fizetes>120</fizetes>*

*<poszt>tamado</poszt>*

*<vezeteknev>Lewandowski</vezeteknev>*

*<keresztnev>Kante</keresztnev>*

*<szulev>1988</szulev>*

*<nemzetiseg>Lengyel</nemzetiseg>*

*</jatekos>*

*<jatekos J\_ID="3" J\_F="3">*

*<kor>22</kor>*

*<fizetes>90</fizetes>*

*<poszt>kozeppalyas</poszt>*

*<vezeteknev>Haaland</vezeteknev>*

*<keresztnev>Erling</keresztnev>*

*<szulev>2000</szulev>*

*<nemzetiseg>Norvég</nemzetiseg>*

*</jatekos>*

*<!-- Egyesületek -->*

*<egyesulet E\_ID="1" E\_F="1">*

*<cim>*

*<iranyitoszam>1012</iranyitoszam>*

*<telepules>London</telepules>*

*<utca>Stadium</utca>*

*<hazszam>1</hazszam>*

*</cim>*

*<nev>Chelsea</nev>*

*</egyesulet>*

*<egyesulet E\_ID="2" E\_F="2">*

*<cim>*

*<iranyitoszam>1012</iranyitoszam>*

*<telepules>Barcelona</telepules>*

*<utca>Senour</utca>*

*<hazszam>33</hazszam>*

*</cim>*

*<nev>Barcelona</nev>*

*</egyesulet>*

*<egyesulet E\_ID="3" E\_F="3">*

*<cim>*

*<iranyitoszam>1012</iranyitoszam>*

*<telepules>Manchester</telepules>*

*<utca>Garden</utca>*

*<hazszam>10</hazszam>*

*</cim>*

*<nev>Manchester City</nev>*

*</egyesulet>*

*<!-- Mérkőzések -->*

*<merkozes M\_ID="1" M\_F="1">*

*<datum>2022.10.12</datum>*

*<ora>21:00</ora>*

*<helyszin>London</helyszin>*

*</merkozes>*

*<merkozes M\_ID="2" M\_F="2">*

*<datum>2022.08.05</datum>*

*<ora>18:00</ora>*

*<helyszin>Barcelona</helyszin>*

*</merkozes>*

*<merkozes M\_ID="3" M\_F="3">*

*<datum>2022.06.28</datum>*

*<ora>20:00</ora>*

*<helyszin>Paris</helyszin>*

*</merkozes>*

*<!-- Bírók -->*

*<biro B\_ID="1">*

*<szerep>Pálya</szerep>*

*<vezeteknev>Collina</vezeteknev>*

*<keresztnev>Pierluigi</keresztnev>*

*</biro>*

*<biro B\_ID="2">*

*<szerep>Pálya</szerep>*

*<vezeteknev>Kassai</vezeteknev>*

*<keresztnev>Viktor</keresztnev>*

*</biro>*

*<biro B\_ID="3">*

*<szerep>Partjelző</szerep>*

*<vezeteknev>Pedro</vezeteknev>*

*<keresztnev>Poenca</keresztnev>*

*</biro>*

*<!-- Edzők -->*

*<edzo ED\_ID="1">*

*<vezeteknev>Tuchel</vezeteknev>*

*<keresztnev>Thomsa</keresztnev>*

*<nemzetiseg>Német</nemzetiseg>*

*</edzo>*

*<edzo ED\_ID="2">*

*<vezeteknev>Alonso</vezeteknev>*

*<keresztnev>Xavi</keresztnev>*

*<nemzetiseg>Spanyol</nemzetiseg>*

*</edzo>*

*<edzo ED\_ID="3">*

*<vezeteknev>Guardiola</vezeteknev>*

*<keresztnev>Pep</keresztnev>*

*<nemzetiseg>Spanyol</nemzetiseg>*

*</edzo>*

*<!-- Ki játszik kivel -->*

*<jatszik E\_M\_E="1" E\_M\_M="1">*

*<eredmeny>2</eredmeny>*

*</jatszik>*

*<jatszik E\_M\_E="2" E\_M\_M="1">*

*<eredmeny>1</eredmeny>*

*</jatszik>*

*<jatszik E\_M\_E="3" E\_M\_M="3">*

*<eredmeny>1</eredmeny>*

*</jatszik>*

*</bajnoksag>*

## 1d) Az XML dokumentum alapján XMLSchema készítése (saját típus használata) – fájlnév: XMLSchemaNeptunkod.xsd

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

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

<!-- Elements, types -->

<xs:element name="pontok" type="xs:integer" />

<xs:element name="gyozelem" type="xs:integer" />

<xs:element name="dontetlen" type="xs:integer" />

<xs:element name="vereseg" type="xs:integer" />

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

<xs:element name="fizetes" type="xs:integer" />

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

<xs:element name="szulev" type="xs:integer" />

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

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

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

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

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

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

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

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

<xs:element name="eredmeny" type="xs:integer" />

<xs:attribute name="T\_ID" type="xs:integer" />

<xs:attribute name="J\_ID" type="xs:integer" />

<xs:attribute name="J\_F" type="xs:integer" />

<xs:attribute name="E\_ID" type="xs:integer" />

<xs:attribute name="E\_F" type="xs:integer" />

<xs:attribute name="M\_ID" type="xs:integer" />

<xs:attribute name="M\_F" type="xs:integer" />

<xs:attribute name="B\_ID" type="xs:integer" />

<xs:attribute name="ED\_ID" type="xs:integer" />

<xs:attribute name="E\_M\_E" type="xs:integer" />

<xs:attribute name="E\_M\_M" type="xs:integer" />

<!—Simple types -->

<xs:simpleType name="datum">

<xs:restriction base="xs:string">

<xs:pattern value="(19|20)\d\d.(0[1-9]|1[012]).(0[1-9]|[12][0-9]|3[01])"></xs:pattern>

</xs:restriction>

</xs:simpleType>

<xs:simpleType name="iranyitoszam">

<xs:restriction base="xs:string">

<xs:length value="4" />

<xs:pattern value="([0-9])\*" />

</xs:restriction>

</xs:simpleType>

<xs:simpleType name="vezeteknev">

<xs:restriction base="xs:string">

<xs:pattern value="[A-Z][a-zA-Z]\*( [A-Z][a-zA-Z]\*)\*" />

</xs:restriction>

</xs:simpleType>

<xs:simpleType name="keresztnev">

<xs:restriction base="xs:string">

<xs:pattern value="[A-Z][a-zA-Z]\*( [A-Z][a-zA-Z]\*)\*" />

</xs:restriction>

</xs:simpleType>

<!-- Complex types -->

<xs:complexType name="tabella\_tipus">

<xs:sequence>

<xs:element ref="pontok"/>

<xs:element ref="gyozelem"/>

<xs:element ref="dontetlen"/>

<xs:element ref="vereseg"/>

</xs:sequence>

<xs:attribute ref="T\_ID" use="required"/>

</xs:complexType>

<xs:complexType name="jatekos\_tipus">

<xs:sequence>

<xs:element ref="kor"/>

<xs:element ref="fizetes"/>

<xs:element ref="poszt"/>

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

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

<xs:element ref="szulev"/>

<xs:element ref="nemzetiseg"/>

</xs:sequence>

<xs:attribute ref="J\_ID" use="required"/>

<xs:attribute ref="J\_F" use="required"/>

</xs:complexType>

<xs:complexType name="cim\_tipus">

<xs:sequence>

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

<xs:element ref="telepules"/>

<xs:element ref="utca"/>

<xs:element ref="hazszam"/>

</xs:sequence>

</xs:complexType>

<xs:complexType name="egyesulet\_tipus">

<xs:sequence>

<xs:element name="cim" type="cim\_tipus"/>

<xs:element ref="nev" />

</xs:sequence>

<xs:attribute ref="E\_ID" use="required"/>

<xs:attribute ref="E\_F" use="required"/>

</xs:complexType>

<xs:complexType name="merkozes\_tipus">

<xs:sequence>

<xs:element name="datum" type="datum"/>

<xs:element ref="ora"/>

<xs:element ref="helyszin"/>

</xs:sequence>

<xs:attribute ref="M\_ID" use="required"/>

<xs:attribute ref="M\_F" use="required"/>

</xs:complexType>

<xs:complexType name="biro\_tipus">

<xs:sequence>

<xs:element ref="szerep"/>

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

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

</xs:sequence>

<xs:attribute ref="B\_ID" use="required"/>

</xs:complexType>

<xs:complexType name="edzo\_tipus">

<xs:sequence>

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

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

<xs:element ref="nemzetiseg"/>

</xs:sequence>

<xs:attribute ref="ED\_ID" use="required"/>

</xs:complexType>

<xs:complexType name="jatszik\_tipus">

<xs:sequence>

<xs:element ref="eredmeny"/>

</xs:sequence>

<xs:attribute ref="E\_M\_E" use="required"/>

<xs:attribute ref="E\_M\_M" use="required"/>

</xs:complexType>

<xs:element name="bajnoksag">

<xs:complexType>

<xs:sequence>

<xs:element name="tabella" type="tabella\_tipus" maxOccurs="unbounded"/>

<xs:element name="jatekos" type="jatekos\_tipus" maxOccurs="unbounded"/>

<xs:element name="egyesulet" type="egyesulet\_tipus" maxOccurs="unbounded"/>

<xs:element name="merkozes" type="merkozes\_tipus" maxOccurs="unbounded"/>

<xs:element name="biro" type="biro\_tipus" maxOccurs="unbounded"/>

<xs:element name="edzo" type="edzo\_tipus" maxOccurs="unbounded"/>

<xs:element name="jatszik" type="jatszik\_tipus" maxOccurs="unbounded"/>

</xs:sequence>

</xs:complexType>

<!-- Keys -->

<xs:key name="T\_ID">

<xs:selector xpath="tabella"/>

<xs:field xpath="@T\_ID"/>

</xs:key>

<xs:key name="J\_ID">

<xs:selector xpath="jatekos"/>

<xs:field xpath="@J\_ID"/>

</xs:key>

<xs:key name="E\_ID">

<xs:selector xpath="egyesulet"/>

<xs:field xpath="@E\_ID"/>

</xs:key>

<xs:key name="M\_ID">

<xs:selector xpath="merkozes"/>

<xs:field xpath="@M\_ID"/>

</xs:key>

<xs:key name="B\_ID">

<xs:selector xpath="biro"/>

<xs:field xpath="@B\_ID"/>

</xs:key>

<xs:key name="ED\_ID">

<xs:selector xpath="edzo"/>

<xs:field xpath="@ED\_ID"/>

</xs:key>

<!-- Foreign keys -->

<xs:keyref name="egyesulet\_FK" refer="E\_ID">

<xs:selector xpath="jatekos"></xs:selector>

<xs:field xpath="@J\_F"></xs:field>

</xs:keyref>

<xs:keyref name="tabella\_FK" refer="E\_ID">

<xs:selector xpath="egyesulet"></xs:selector>

<xs:field xpath="@E\_F"></xs:field>

</xs:keyref>

<xs:keyref name="biro\_FK" refer="B\_ID">

<xs:selector xpath="merkozes"></xs:selector>

<xs:field xpath="@M\_F"></xs:field>

</xs:keyref>

<xs:keyref name="jatszik\_egyesulet\_FK" refer="E\_ID">

<xs:selector xpath="jatszik"></xs:selector>

<xs:field xpath="@E\_M\_E"></xs:field>

</xs:keyref>

<xs:keyref name="jatszik\_merkozes\_FK" refer="M\_ID">

<xs:selector xpath="jatszik"></xs:selector>

<xs:field xpath="@E\_M\_M"></xs:field>

</xs:keyref>

</xs:element>

</xs:schema>

# 2.Feladat

## 2a) Adatolvasás:

### Leírás:

Kiolvassa a játékosok és egyesületek adatait

package hu.domparse.wh85zh;

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 DOMReadWH85ZH {

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

//Forrás file

File file = new File("XMLWH85ZH.xml");

DocumentBuilderFactory factory = DocumentBuilderFactory.newInstance();

DocumentBuilder dBuilder = factory.newDocumentBuilder();

Document doc = dBuilder.parse(file);

doc.getDocumentElement().normalize();

//Gyökérelem

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

//Gyerekelemek lementése

NodeList nList = (NodeList) doc.getDocumentElement();

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

Node node = nList.item(i);

//Ha jatekoshoz erünk

if(node.getNodeName() == "jatekos"){

if(!node.getNodeName().equals("#text")) {

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

System.out.println("Current element: " + node.getNodeName());

}

//Játékosok adatainak kiírása

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

Element elem = (Element) node;

String uid = elem.getAttribute("J\_ID");

Node node1 = elem.getElementsByTagName("vezeteknev").item(0);

String fname = node1.getTextContent();

Node node2 = elem.getElementsByTagName("keresztnev").item(0);

String lname = node2.getTextContent();

Node node3 = elem.getElementsByTagName("nemzetiseg").item(0);

String nation = node3.getTextContent();

Node node4 = elem.getElementsByTagName("szulev").item(0);

String year = node4.getTextContent();

System.out.printf("User id = %s%n", uid);

System.out.printf("Firstname = %s%n", fname);

System.out.printf("Lastname = %s%n", lname);

System.out.printf("Nation = %s%n", nation);

System.out.printf("Born in = %s%n", year);

}

}

}

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

Node node = nList.item(i);

//Ha jatekoshoz erünk

if(node.getNodeName() == "egyesulet"){

if(!node.getNodeName().equals("#text")) {

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

System.out.println("Current element: " + node.getNodeName());

}

//Egyesület adatainak kiírása

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

Element elem = (Element) node;

String uid = elem.getAttribute("J\_ID");

Node node1 = elem.getElementsByTagName("iranyitoszam").item(0);

String irsz = node1.getTextContent();

Node node2 = elem.getElementsByTagName("telepules").item(0);

String tel = node2.getTextContent();

Node node3 = elem.getElementsByTagName("utca").item(0);

String utca = node3.getTextContent();

Node node4 = elem.getElementsByTagName("hazszam").item(0);

String hszam = node4.getTextContent();

Node node5 = elem.getElementsByTagName("nev").item(0);

String name = node5.getTextContent();

System.out.printf("User id = %s%n", uid);

System.out.printf("Iranyitoszam = %s%n", irsz);

System.out.printf("Telepules = %s%n", tel);

System.out.printf("Utca = %s%n", utca);

System.out.printf("Hazszam = %s%n", hszam);

System.out.printf("Name = %s%n", name);

}

}

}

### Futási eredmény:

A képen szöveg látható

Automatikusan generált leírásA képen szöveg látható

Automatikusan generált leírás

## 

## 2b) Adatlekérdezés:

### Leírás:

Összeszámolja a csapatok összpontszámát, a játékosok akik idősebbek 30-nál, kilistázza a bírókat, azon játékosok akiknek a fizetése kevesebb mint 100, a spanyol edzők

package hu.domparse.wh85zh;

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 DOMQueryWH85ZH {

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

//Forrás file

File file = new File("XMLWH85ZH.xml");

//csapatok álltal szerzett összpontszám

int sum = 0;

DocumentBuilderFactory dbFactory = DocumentBuilderFactory.newInstance();

DocumentBuilder dBuilder = dbFactory.newDocumentBuilder();

Document doc = dBuilder.parse(file);

doc.getDocumentElement().normalize();

//Gyökér elem

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

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

//Tabella mentése

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

// összeszámolja a csapatok összpontszámát

NodeList nList = doc.getDocumentElement().getElementsByTagName("tabella");

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

NodeList query = nList.item(i).getChildNodes();

for (int j = 0; j < query.getLength(); j++) {

if (query.item(j).getNodeName().equals("pontok")){

sum += Integer.parseInt(query.item(j).getTextContent());

}

}

}

//összpontok

System.out.println("Összes pont: "+ sum);

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

//Játékosok akik idősebbek 30 nál

NodeList nList2 = doc.getDocumentElement().getElementsByTagName("jatekos");

for (int i = 0; i < nList2.getLength(); i++) {

NodeList query = nList2.item(i).getChildNodes();

for (int j = 0; j < query.getLength(); j++) {

Node node2 = query.item(j);

if (query.item(j).getNodeName().equals("kor") && Integer.parseInt(query.item(j).getTextContent()) >= 30){

NodeList nList0 = nList2.item(i).getChildNodes();

for (int k = 0; k < nList0.getLength(); k++) {

Node node3 = nList0.item(k);

if (node3.getNodeType() == Node.ELEMENT\_NODE) {

System.out.println(node3.getNodeName()+" : "+node3.getTextContent());

}

}

System.out.println();

}

}

}

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

//Bírók kilistázása

NodeList nList3 = doc.getDocumentElement().getElementsByTagName("biro");

for (int i = 0; i < nList3.getLength(); i++) {

NodeList query = nList3.item(i).getChildNodes();

for (int j = 0; j < query.getLength(); j++) {

Node node2 = query.item(j);

if (node2.getNodeType() == Node.ELEMENT\_NODE) {

System.out.println(node2.getNodeName()+" : "+node2.getTextContent());

}

}

System.out.println();

}

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

//Játékosok adatai akiknek a fizetése kevesebb mint 100

NodeList nList4 = doc.getDocumentElement().getElementsByTagName("jatekos");

for (int i = 0; i < nList4.getLength(); i++) {

NodeList query = nList4.item(i).getChildNodes();

for (int j = 0; j < query.getLength(); j++) {

Node node2 = query.item(j);

if (node2.getNodeName().equals("fizetes") && Integer.parseInt(node2.getTextContent()) <= 100){

NodeList nList0 = nList4.item(i).getChildNodes();

for (int k = 0; k < nList0.getLength(); k++) {

Node node3 = nList0.item(k);

if (node3.getNodeType() == Node.ELEMENT\_NODE) {

System.out.println(node3.getNodeName()+" : "+node3.getTextContent());

}

}

System.out.println();

}

}

}

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

//Spanyol nemzetiségű edző

NodeList nList5 = doc.getDocumentElement().getElementsByTagName("edzo");

for (int i = 0; i < nList5.getLength(); i++) {

NodeList query = nList5.item(i).getChildNodes();

for (int j = 0; j < query.getLength(); j++) {

Node node2 = query.item(j);

if (node2.getNodeName().equals("nemzetiseg") && node2.getTextContent().equals("Spanyol")){

NodeList nList0 = nList5.item(i).getChildNodes();

for (int k = 0; k < nList0.getLength(); k++) {

Node node3 = nList0.item(k);

if (node3.getNodeType() == Node.ELEMENT\_NODE) {

System.out.println(node3.getNodeName()+" : "+node3.getTextContent());

}

}

System.out.println();

}

}

}

}

}

### Futási eredmény:

A képen szöveg látható

Automatikusan generált leírás A képen szöveg látható

Automatikusan generált leírás

## 2c) Adatmódosítás:

### Leírás:

Harmadik egyesület ID megváltoztatása, Második egyesület irányítószám változás, Harmadik egyesület utca változás, Bírók törlése

package hu.domparse.wh85zh;

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.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 DOMModifyWH85ZH {

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

try {

//Forrás file

File inputFile = new File("XML2WH85ZH.xml");

DocumentBuilderFactory docFactory = DocumentBuilderFactory.newInstance();

DocumentBuilder docBuilder = docFactory.newDocumentBuilder();

Document doc = docBuilder.parse(inputFile);

//Második és harmadik egyesület mentése

Node club1 = doc.getElementsByTagName("egyesulet").item(1);

Node club2 = doc.getElementsByTagName("egyesulet").item(2);

//Gyökérelem

Node bajnoksag = doc.getFirstChild();

//Harmadik egyesület ID váltás

NamedNodeMap attr = club2.getAttributes();

Node nodeAttr = attr.getNamedItem("E\_ID");

nodeAttr.setTextContent("4");

//Második egyesület irányítószámának megváltoztatása

NodeList list = club1.getChildNodes();

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

Node node = list.item(i);

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

Element elem = (Element) node;

if ("iranyitoszam".equals(elem.getNodeName())) {

elem.setTextContent("3532");

}

}

}

//Harmadik egyesület utcanevének megváltoztatása

NodeList list1 = club2.getChildNodes();

for (int i = 0; i < list1.getLength(); i++) {

Node node1 = list1.item(i);

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

Element elem1 = (Element) node1;

if ("utca".equals(elem1.getNodeName())) {

elem1.setTextContent("Gorton");

}

}

}

//Bírók kitörlése

NodeList childNodes = bajnoksag.getChildNodes();

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

Node node = childNodes.item(i);

if("biro".equals(node.getNodeName()))

bajnoksag.removeChild(node);

}

//Konzolra kírás

TransformerFactory transformerFactory = TransformerFactory.newInstance();

Transformer transformer = transformerFactory.newTransformer();

DOMSource source = new DOMSource(doc);

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

StreamResult consoleResult = new StreamResult(System.out);

transformer.transform(source, consoleResult);

}catch(Exception e) {

e.printStackTrace();

}

}

}

### Futási eredmény:

A képen szöveg, névtábla látható

Automatikusan generált leírás