**JEGYZŐKÖNY**

Adatkezelés XML környezetben

Féléves feladat

ESport Versenyek

**Készítette:**  
Bordás Milán  
Programtervező informatikus   
WJB0DC

**Dátum:**2023.12.05

**Miskolc, 2023**

A feladat leírása

Az ER modell egy E-sport csapatversenyeket rendező cég által tárolt adatokat modellezi le.

Az adatbázis a következő fontosabb elemeket tárolja le:

* Versenyek, melyeket a cég megrendez
* A versenyeken meghirdetett játékokat
* A versenyeken résztvevő csapatokat
* A csapatok tagjait
* A csapatok vezetőit

Versenyek:

Minden versenynek tárolja a cég az azonosítóját (kulcs, szám), a nevét (szöveges típus), a megrendezés dátumát (dátum típus), illetve a helyszínét (szöveges típus). Egy versenyt csak egyszer rendeznek meg havonta, de többször is megrendezhető egy verseny egy évben. A versenyeknek a megrendezés dátuma, illetve a neve alapján egy számot (azonosítót) generálnak, hogy elkülöníthető legyen két azonos nevű verseny.  
A versenyeken több versenyszám (játék) is meg lehet hirdetve, a versenyeken a játékosok csak csapatban indulhatnak. Egy csapat több versenyen is résztvehet.

Csapatok:

Minden csapatnak egyedi csapatneve (CSAzon - kulcs, szöveges típus) van, illetve tárolják a csapatokat alapítók neveit (szöveges típus, több értékű), akik teljesen függetlenek a cégtől, illetve a játékosoktól (az összeférhetetlenség elkerülése miatt), az alapítás dátumát (dátum típus) és a csapatot szponzoráló egyetlen szponzort [összetett típus – név (szöveg) – támogatási összeg (szám)].  
A csapatoknak a vezetőit archiválják, róluk külön nyilvántartást tartanak. Egy időben csak egy vezetője lehet a csapatnak és egy vezető csak egy csapatot vezet. A vezetők szintén függetlenek az összeférhetetlenség elkerülése végett.  
Egy csapatban 3-5 játékos van, egy fő csak egy csapatban lehet.

Csapatvezetők:

A csapatvezetőknek tárolják a nevét (szöveg), a születési dátumát (dátum típus) [+*korát* (származtatott, szám, jelenlegi dátum - születési dátum alapján)], vezetői azonosítóját [azonosító (kulcs, szám)], illetve azt, hogy mikortól (dátum) volt vezetői pozícióban. Ha a csapatot már nem vezeti, akkor azt is tárolják, hogy meddig (dátum, opcionális típus) vezette.

Játékok:

Minden játéknak van neve (szöveges típus), illetve típusa (szöveg, többértékű). A típus a játék zsáneréből származik, pl. lövöldözős, stratégiai stb. Egy játéknak több típusa is lehet. Mivel a játékok neveire semmilyen megkötés nincs, így az azonosításukhoz azonosítót (kulcs, szám) generálnak.

Játékos:

A játékosoknak tárolják a nevét (szöveg), a születési dátumát (dátum típus) [+*korát* (származtatott, szám, jelenlegi dátum - születési dátum alapján)], játékos azonosítóját [JA (kulcs, szám)], illetve országát (szöveg).  
A játékosok csak egyféle játékot játszanak.

Kapcsolatok

A mezőket 4 kapcsolat köti össze.

Versenyez: a versenyző csapatokat a versenyekkel kötjük össze több-több kapcsolattal, hiszen több csapat is indulhat egy versenyen, és egy csapat több versenyen is indulhat. A kapcsolatnak egy plusz tulajdonsága az adott csapatoknak a helyezése.

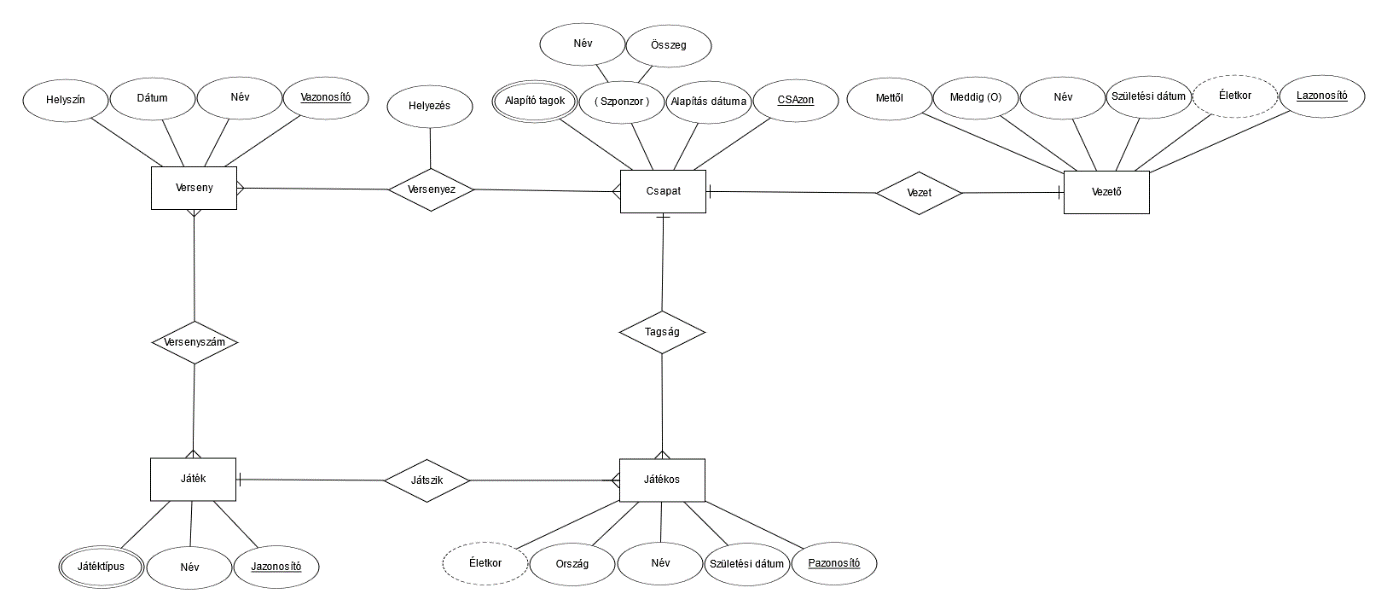
Versenyszám: a versenyek és az azon meghirdetett játékok több-több kapcsolata, hiszen több játékszám is lehet egy versenyen, és egy játékszám több versenyen is lehet.

Játszik: a játékosok és az általuk játszott játék kapcsolata. Egy-több kapcsolat, hiszen egy játékos egy játékkal játszhat, de egy játékkal több játékos.

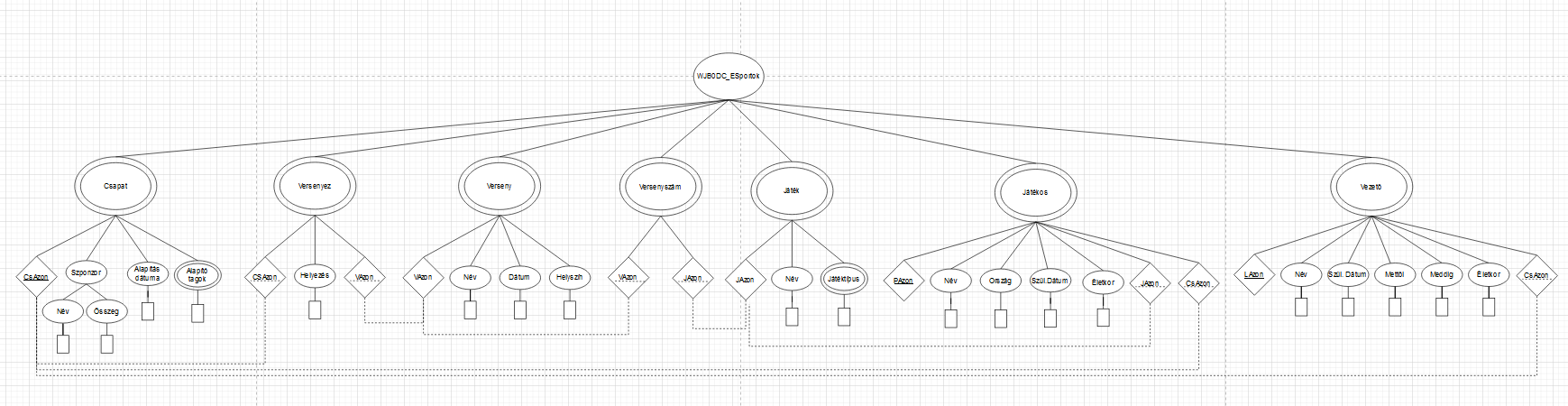
Vezet: a vezetők és a csapatokat összekötő egy-egy kapcsolat, mert egy vezető csak egy csapatot vezethet(ett), illetve egy csapatnak csak egy vezetője lehet adott időpontban.

1a. Feladat

ER modell:



1b. Feladat

XDM modell:

Az XDM modell gyökéreleme a WJB0DC\_Esportok, melynek gyerekelemei az ER modell egyedei, illetve kapcsolótáblái, ahol szükségesek. A modell szerkezete hasonló az ER modelléhez, de itt minden gyerekelemből többet hozhatunk létre, így mindegyiket egy két vonalú ellipszis reprezentálja. Az összetett típusokat. Látható, hogy a csapat sok egyeddel áll kapcsolatban, így az a feladat során is központi szerepet vesz fel.  
Az XDM modellben is szemléltetjük. A modell alapján egyszerűen elkészíthető az XML dokumentum.

1c. Feladat

XML az XDM alapján

Az XML dokumentum az XDM modell alapján készült el.  
Minden többször előforduló elemből létrehoztam legalább 3-at. Ahol szükséges volt, ott több ilyen elem is van, mint pl. a kapcsolatokat megvalósító Versenyez elem. A kódot Visual Studio Code-ban készítettem el és másoltam ide, ezért ilyen a formázása.  
A gyökérelemben látható a kapcsolás az XMLSchemához.

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

<WJB0DC\_ESportok xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

 xsi:noNamespaceSchemaLocation="XMLSchemaWJB0DC.xsd" >

    <!-- Csapatok -->

    <Csapat CSAzon="Kerge Kecskék">

        <alapitas\_datuma>2002-04-06</alapitas\_datuma>

        <Szponzor>

            <nev>Nestlé</nev>

            <osszeg>50000</osszeg>

        </Szponzor>

        <alapito\_tag>Kiss János</alapito\_tag>

        <alapito\_tag>Nagy Martin</alapito\_tag>

    </Csapat>

    <Csapat CSAzon="Zöld Békák">

        <alapitas\_datuma>2002-04-06</alapitas\_datuma>

        <Szponzor>

            <nev>Nestlé</nev>

            <osszeg>75000</osszeg>

        </Szponzor>

        <alapito\_tag>Magyar Márk</alapito\_tag>

        <alapito\_tag>Kossuth Máté</alapito\_tag>

    </Csapat>

    <Csapat CSAzon="Kutyások">

        <alapitas\_datuma>2002-05-01</alapitas\_datuma>

        <Szponzor>

            <nev>FIFA</nev>

            <osszeg>100000</osszeg>

        </Szponzor>

        <alapito\_tag>Nemes Géza</alapito\_tag>

    </Csapat>

    <!--Játékosok-->

    <Jatekos PAzon="1" CSAzon="Kerge Kecskék" JAzon="1">

        <nev>Havas Henrik</nev>

        <orszag>Magyarország</orszag>

        <szuletesi\_datum>1998-12-01</szuletesi\_datum>

        <eletkor>24</eletkor>

    </Jatekos>

    <Jatekos PAzon="2" CSAzon="Kerge Kecskék" JAzon="1">

        <nev>Arany Anna</nev>

        <orszag>Magyarország</orszag>

        <szuletesi\_datum>2003-11-18</szuletesi\_datum>

        <eletkor>20</eletkor>

    </Jatekos>

    <Jatekos PAzon="3" CSAzon="Zöld Békák" JAzon="1">

        <nev>Kiss Anita</nev>

        <orszag>Magyarország</orszag>

        <szuletesi\_datum>1999-10-11</szuletesi\_datum>

        <eletkor>24</eletkor>

    </Jatekos>

    <Jatekos PAzon="4" CSAzon="Zöld Békák" JAzon="1">

        <nev>Nagy Hugó</nev>

        <orszag>Magyarország</orszag>

        <szuletesi\_datum>1999-07-11</szuletesi\_datum>

        <eletkor>24</eletkor>

    </Jatekos>

    <Jatekos PAzon="5" CSAzon="Kutyások" JAzon="2">

        <nev>Németh Ottó</nev>

        <orszag>Magyarország</orszag>

        <szuletesi\_datum>2000-05-07</szuletesi\_datum>

        <eletkor>23</eletkor>

    </Jatekos>

    <Jatekos PAzon="6" CSAzon="Kutyások" JAzon="2">

        <nev>Ding Liren</nev>

        <orszag>Kína</orszag>

        <szuletesi\_datum>1995-12-01</szuletesi\_datum>

        <eletkor>28</eletkor>

    </Jatekos>

    <!-- Játékok -->

    <Jatek JAzon="1">

        <nev>CS-GO</nev>

        <jatektipus>Lövöldözős</jatektipus>

        <jatektipus>Több játékos</jatektipus>

        <jatektipus>Akció</jatektipus>

    </Jatek>

    <Jatek JAzon="2">

        <nev>LoL</nev>

        <jatektipus>MOBA</jatektipus>

        <jatektipus>Több játékos</jatektipus>

        <jatektipus>Stratégia</jatektipus>

    </Jatek>

    <Jatek JAzon="3">

        <nev>FIFA</nev>

        <jatektipus>Labdarúgás</jatektipus>

        <jatektipus>Több játékos</jatektipus>

        <jatektipus>Ügyességi</jatektipus>

    </Jatek>

    <!-- Versenyszámok (kapcsolótábla) -->

    <Versenyszam JAzon="1" VAzon="1"></Versenyszam>

    <Versenyszam JAzon="1" VAzon="2"></Versenyszam>

    <Versenyszam JAzon="2" VAzon="3"></Versenyszam>

    <Versenyszam JAzon="1" VAzon="3"></Versenyszam>

    <Versenyszam JAzon="2" VAzon="4"></Versenyszam>

    <!-- Vezetők -->

    <Vezeto LAzon="1" CSAzon="Kerge Kecskék">

        <nev>Alantas Balázs</nev>

        <szuletesi\_datum>1975-04-01</szuletesi\_datum>

        <mettol>2002</mettol>

        <meddig>2008</meddig>

        <eletkor>48</eletkor>

    </Vezeto>

    <Vezeto LAzon="2" CSAzon="Kerge Kecskék">

        <nev>Kovács Ágnes</nev>

        <szuletesi\_datum>1982-09-21</szuletesi\_datum>

        <mettol>2008</mettol>

        <eletkor>41</eletkor>

    </Vezeto>

    <Vezeto LAzon="3" CSAzon="Zöld Békák">

        <nev>Tóth Szilvia</nev>

        <szuletesi\_datum>1979-12-24</szuletesi\_datum>

        <mettol>2002</mettol>

        <eletkor>41</eletkor>

    </Vezeto>

    <Vezeto LAzon="4" CSAzon="Kutyások">

        <nev>Németh László</nev>

        <szuletesi\_datum>1970-01-16</szuletesi\_datum>

        <mettol>2002</mettol>

        <eletkor>53</eletkor>

    </Vezeto>

    <!-- Versenyez kapcsoló táblák -->

    <Versenyez CSAzon="Kerge Kecskék" VAzon="1">

        <helyezes>1</helyezes>

    </Versenyez>

    <Versenyez CSAzon="Zöld Békák" VAzon="1">

        <helyezes>2</helyezes>

    </Versenyez>

    <Versenyez CSAzon="Kutyások" VAzon="3">

        <helyezes>1</helyezes>

    </Versenyez>

    <Versenyez CSAzon="Kutyások" VAzon="4">

        <helyezes>1</helyezes>

    </Versenyez>

     <!-- Versenyek -->

     <Verseny VAzon="1">

        <nev>CS-GO Championship Las Vegas</nev>

        <datum>2010-10-04</datum>

        <helyszin>Las Vegas</helyszin>

     </Verseny>

     <Verseny VAzon="2">

        <nev>CS-GO Championship Dublin</nev>

        <datum>2012-10-04</datum>

        <helyszin>Dublin</helyszin>

     </Verseny>

     <Verseny VAzon="3">

        <nev>ESport Cup</nev>

        <datum>2014-05-04</datum>

        <helyszin>Berlin</helyszin>

     </Verseny>

     <Verseny VAzon="4">

        <nev>Play IT!</nev>

        <datum>2020-10-10</datum>

        <helyszin>Budapest</helyszin>

     </Verseny>

</WJB0DC\_ESportok>

1d. Feladat

XMLSchema

Az XMLSChema az XML dokumentum után került elkészítésre, tehát a validálás a kód megírása után történik. Az XML dokumentum az alábbi séma alapján megfelel a megkötéseknek, a típus egyezéseknek és a kapcsolatoknak is.  
A feladat során felhasználtam a saját típusaimat, saját összetett típusokat hoztam létre, és azokat „ref” kulcsszóval kapcsoltam össze. A kulcsoknak a kapcsolatát is leírtam, ami a gyökér element megadása végénél szerepel. Ezek tartalmazzák az 1:1, 1:N N:N kapcsolatok leírását.

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

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

    <!-- Egyszerű típusok -->

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

    <xs:element name="alapito\_tag" type="xs:string" />

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

    <xs:element name="szuletesi\_datum" type="DatumTipus" />

    <xs:element name="alapitas\_datuma" type="DatumTipus" />

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

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

    <xs:element name="jatektipus" type="JatekTipusTipus" />

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

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

    <xs:element name="helyezes" type="HelyezesTipus" />

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

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

    <!-- Saját típusok -->

    <xs:simpleType name="DatumTipus">

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

            <xs:minInclusive value="1900-01-01" />

            <xs:maxInclusive value="2023-12-31" />

        </xs:restriction>

    </xs:simpleType>

    <xs:simpleType name="JatekTipusTipus">

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

            <xs:enumeration value="Lövöldözős"></xs:enumeration>

            <xs:enumeration value="Több játékos"></xs:enumeration>

            <xs:enumeration value="Akció"></xs:enumeration>

            <xs:enumeration value="MOBA"></xs:enumeration>

            <xs:enumeration value="Stratégia"></xs:enumeration>

            <xs:enumeration value="Labdarúgás"></xs:enumeration>

            <xs:enumeration value="Ügyességi"></xs:enumeration>

        </xs:restriction>

    </xs:simpleType>

    <xs:simpleType name="HelyezesTipus">

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

            <xs:minInclusive value="1" />

            <xs:maxInclusive value="99"/>

        </xs:restriction>

    </xs:simpleType>

    <!-- Komplex típusok -->

    <xs:complexType name="SzponzorTipus">

        <xs:sequence>

            <xs:element ref="nev" />

            <xs:element ref="osszeg" />

        </xs:sequence>

    </xs:complexType>

    <xs:complexType name="CsapatTipus">

        <xs:sequence>

            <xs:element ref="alapitas\_datuma"/>

            <xs:element name="Szponzor" type="SzponzorTipus" />

            <xs:element ref="alapito\_tag" minOccurs="1" maxOccurs="10"/>

        </xs:sequence>

        <xs:attribute name="CSAzon" type="xs:string" />

    </xs:complexType>

    <xs:complexType name="JatekosTipus">

        <xs:sequence>

            <xs:element ref="nev" />

            <xs:element ref="orszag" />

            <xs:element ref="szuletesi\_datum"/>

            <xs:element ref="eletkor"/>

        </xs:sequence>

        <xs:attribute name="PAzon" type="xs:integer" />

        <xs:attribute name="CSAzon" type="xs:string" />

        <xs:attribute name="JAzon" type="xs:integer" />

    </xs:complexType>

    <xs:complexType name="JatekTipus">

        <xs:sequence>

            <xs:element ref="nev" />

            <xs:element name="jatektipus" type="JatekTipusTipus" minOccurs="1" maxOccurs="10"/>

        </xs:sequence>

        <xs:attribute name="JAzon" type="xs:integer" />

    </xs:complexType>

    <xs:complexType name="VersenyszamTipus">

        <xs:attribute name="JAzon" type="xs:integer" />

        <xs:attribute name="VAzon" type="xs:integer" />

    </xs:complexType>

    <xs:complexType name="VezetoTipus">

        <xs:sequence>

            <xs:element ref="nev" />

            <xs:element ref="szuletesi\_datum"/>

            <xs:element ref="mettol"/>

            <xs:element ref="meddig" minOccurs="0"/>

            <xs:element ref="eletkor"/>

        </xs:sequence>

        <xs:attribute name="LAzon" type="xs:integer" />

        <xs:attribute name="CSAzon" type="xs:string" />

    </xs:complexType>

    <xs:complexType name="VersenyezTipus">

        <xs:sequence>

            <xs:element ref="helyezes" />

        </xs:sequence>

        <xs:attribute name="CSAzon" type="xs:string" />

        <xs:attribute name="VAzon" type="xs:integer" />

    </xs:complexType>

    <xs:complexType name="VersenyTipus">

        <xs:sequence>

            <xs:element ref="nev" />

            <xs:element ref="datum" />

            <xs:element ref="helyszin" />

        </xs:sequence>

        <xs:attribute name="VAzon" type="xs:integer" />

    </xs:complexType>

    <!-- Elemek -->

    <xs:element name="WJB0DC\_ESportok">

        <xs:complexType>

            <xs:sequence>

                <xs:element name="Csapat" type="CsapatTipus" maxOccurs="unbounded" />

                <xs:element name="Jatekos" type="JatekosTipus" maxOccurs="unbounded" />

                <xs:element name="Jatek" type="JatekTipus" maxOccurs="unbounded" />

                <xs:element name="Versenyszam" type="VersenyszamTipus" maxOccurs="unbounded" />

                <xs:element name="Vezeto" type="VezetoTipus" maxOccurs="unbounded" />

                <xs:element name="Versenyez" type="VersenyezTipus" maxOccurs="unbounded" />

                <xs:element name="Verseny" type="VersenyTipus" maxOccurs="unbounded" />

            </xs:sequence>

        </xs:complexType>

        <!--Elsődleges kulcsok-->

        <xs:key name="CsapatKulcs">

            <xs:selector xpath="Csapat" />

            <xs:field xpath="@CSAzon" />

        </xs:key>

        <xs:key name="JatekosKulcs">

            <xs:selector xpath="Jatekos" />

            <xs:field xpath="@PAzon" />

        </xs:key>

        <xs:key name="JatekKulcs">

            <xs:selector xpath="Jatek" />

            <xs:field xpath="@JAzon" />

        </xs:key>

        <xs:key name="VezetoKulcs">

            <xs:selector xpath="Vezeto" />

            <xs:field xpath="@LAzon" />

        </xs:key>

        <xs:key name="VersenyKulcs">

            <xs:selector xpath="Verseny" />

            <xs:field xpath="@VAzon" />

        </xs:key>

        <!-- Idegen kulcsok -->

        <xs:keyref name="JatekosCsapatKulcs" refer="CsapatKulcs">

            <xs:selector xpath="Jatekos" />

            <xs:field xpath="@CSAzon" />

        </xs:keyref>

        <xs:keyref name="JatekosJatekKulcs" refer="JatekKulcs">

            <xs:selector xpath="Jatekos" />

            <xs:field xpath="@JAzon" />

        </xs:keyref>

        <xs:keyref name="VersenyszamJatekKulcs" refer="JatekKulcs">

            <xs:selector xpath="Versenyszam" />

            <xs:field xpath="@JAzon" />

        </xs:keyref>

        <xs:keyref name="VersenyszamVersenyKulcs" refer="VersenyKulcs">

            <xs:selector xpath="Versenyszam" />

            <xs:field xpath="@VAzon" />

        </xs:keyref>

        <xs:keyref name="VezetoCsapatKulcs" refer="CsapatKulcs">

            <xs:selector xpath="Vezeto" />

            <xs:field xpath="@CSAzon" />

        </xs:keyref>

        <xs:keyref name="VersenyezCsapatKulcs" refer="CsapatKulcs">

            <xs:selector xpath="Versenyez" />

            <xs:field xpath="@CSAzon" />

        </xs:keyref>

        <xs:keyref name="VersenyezVersenyKulcs" refer="VersenyKulcs">

            <xs:selector xpath="Versenyez" />

            <xs:field xpath="@VAzon" />

        </xs:keyref>

        <!-- 1:1 kapcsolat-->

        <xs:unique name="CsapatVezetoKulcs">

            <xs:selector xpath="VezetoKulcs" />

            <xs:field xpath="@CSAzon" />

        </xs:unique>

    </xs:element>

</xs:schema>

2a. Feladat

XML dokumentum beolvasása

Az kódokat Eclipse környezetben készítettem el, a jobb olvashatóság érdekében átmásoltam a Visual Studio Code környezetbe, és onnan másoltam a jegyzúkönyvbe.

Main class:

Itt hívjuk meg a DOMParse csomagban definiált osztályok metódusait. A dokumentum beolvasását a DOMReadWJB0DC osztályban implementáltam, az általa visszaadott dokumentumot kapják meg az azt felhasználó osztályok metódusai. A feladatok elválasztását a konzolra kiírt elválasztók végzik.

package hu.domparse.WJB0DC;

import org.w3c.dom.Document;

public class Main {

    public static void main(String[] args) {

        Document document = DOMReadWJB0DC.ReadDocument("src/hu/domparse/WJB0DC/XMLWJB0DC.xml","src/hu/domparse/WJB0DC/XMLOutput.txt");

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

        DOMModifyWJB0DC.ModifyFiveElements(document);

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

        DOMQueryWJB0DC.QueryFiveTimes(document);

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

        DOMWriteWJB0DC.WriteDocument("src/hu/domparse/WJB0DC/XMLWJB0DC.xml");

    }

}

DOMRead class:

Ebben az osztályban olvasom be az XML dokumentumot. A metódus létrehozza a dokumentum beolvasásához, és a DOM fa kialakításához szükséges objektumokat, amivel aztán műveletek végezhetünk. A beolvasás után a PrintDocument metódus kiírja a DOM fát a konzolra és az XMLOutput.xml fájlba olyan strukturált módon, ahogyan az eredeti dokumentum is meg van írva. A kiíratás során rekurzív megoldást használ a program.  
Megadunk egy Node objektumot. Az algoritmus kiírja a nevét, majd a gyerekelemeire újra meghívja a metódust. Ha a gyerekelem egy text Node, akkor azt kiírja, és felfelé halad tovább a rekurzió, ha nem, akkor addig halad míg egy üres Node-ot, vagy text Node-ot ér el. Majd a rekurzió végén kiírja a Node nevét újra, záró tagként. Ha a gyökérelemet adnánk meg az algoritmusnak, akkor a végeredmény nem változna, ám a DOM fa nem lenne felépítve.  
Az osztály segéd függvényeket is tartalmaz, amik a kiíráshoz szükségesek.

package hu.domparse.WJB0DC;

import java.io.File;

import java.io.FileWriter;

import java.io.IOException;

import javax.xml.parsers.DocumentBuilder;

import javax.xml.parsers.DocumentBuilderFactory;

import javax.xml.parsers.ParserConfigurationException;

import org.w3c.dom.\*;

import org.xml.sax.SAXException;

public class DOMReadWJB0DC {

    private static FileWriter writer;

    public static boolean writeToFile = true;

    public static Document ReadDocument(String filePath, String outputFilePath) {

        // létrehozzuk a szükséges adatstruktúrákat, hogy beolvashassuk a dokumentumot

        DocumentBuilderFactory dbFactory = DocumentBuilderFactory.newInstance();

        try {

            // beolvassuk a dokumentumot

            DocumentBuilder db = dbFactory.newDocumentBuilder();

            Document document = db.parse(new File(filePath));

            document.getDocumentElement().normalize();

            // kiirjuk a tartalmát úgy, ahogy eredetileg volt

            writer = new FileWriter(outputFilePath);

            PrintDocument(document);

            writer.close();

            return document;

        } catch (ParserConfigurationException | SAXException | IOException e) {

            e.printStackTrace();

        }

        return null;

    }

    public static void PrintDocument(Document document) {

        Element root = document.getDocumentElement();

        // kiirjuk a fájl tetejét ami minden xml dokumentum elején van

        Print("<?xml version=\"1.0\" encoding=\"UTF-8\"?>\n");

        Print("<" + root.getNodeName());

        NamedNodeMap rootAttributes = root.getAttributes();

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

            Print(rootAttributes.item(i).getNodeName() + "=\"" + rootAttributes.item(i).getNodeValue() + "\"");

        }

        Print(">\n");

        // DOM fa

        NodeList csapatok = document.getElementsByTagName("Csapat");

        NodeList jatekosok = document.getElementsByTagName("Jatekos");

        NodeList jatekok = document.getElementsByTagName("Jatek");

        NodeList versenyszamok = document.getElementsByTagName("Versenyszam");

        NodeList vezetok = document.getElementsByTagName("Vezeto");

        NodeList versenyezTablak = document.getElementsByTagName("Versenyez");

        NodeList versenyek = document.getElementsByTagName("Verseny");

        PrintNodeList(csapatok);

        PrintNodeList(jatekosok);

        PrintNodeList(jatekok);

        PrintNodeList(versenyszamok);

        PrintNodeList(vezetok);

        PrintNodeList(versenyezTablak);

        PrintNodeList(versenyek);

        Print("<" + root.getNodeName() + "/>");

    }

    public static void PrintNodeList(NodeList nodeList) {

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

            PrintNodes(nodeList.item(i), 1);

        }

    }

    public static void PrintNodes(Node node, int nodeRank) {

        // ha null lenne a node, akkor nem csinálunk semmit

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

            return;

        }

        // ha üres a node akkor sem csinálunk semmit

        if (node instanceof Text) {

            String value = node.getNodeValue().trim();

            if (value.equals("")) {

                return;

            }

        }

        // a sorok behúzása (3 szóköz) \* mennyire beágyazott a node (nodeRank)

        StringBuilder builder = new StringBuilder();

        for (int i = 0; i < nodeRank; i++) {

            builder.append("   ");

        }

        String formatting = builder.toString();

        // kiirjuk a node elejét, pl. <Csapat

        Print(formatting + "<" + node.getNodeName());

        NamedNodeMap nodeMap = node.getAttributes();

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

            // kiirjuk az attribútomokat, értékkel együtt -> <Csapat CSAzon="Kerge Kecskék"

            // ...

            Node attr = nodeMap.item(i);

            Print(" " + attr.getNodeName() + "=\"" + attr.getNodeValue() + "\"");

        }

        // lezárjuk az adott node-ot, pl. <Csapat CSAzon="Kerge Kecskék">

        Print(">");

        // kiirjuk a beágyazott nodeokat (gyerek nodeok)

        NodeList children = node.getChildNodes();

        if (children.getLength() == 1 && children.item(0).getNodeType() == Node.TEXT\_NODE) {

            // ha a node maga a szöveg, akkor kiírjuk pl.

            // <nev> volt idáig, majd ebből

            // <nev>Nestlé

            Print(children.item(0).getNodeValue());

        } else if (!(children.getLength() == 0)) {

            // nem csak egy szöveg, tehát rekurzívan kiirjuk a gyerek node-okat

            Println("");

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

                PrintNodes(children.item(i), nodeRank + 1);

            }

        } else {

            // ez akkor kell, ha üres volt a node

            // ekkor <node></node> íródik ki és visszatérünk

            Print("</" + node.getNodeName() + ">\n");

            return;

        }

        // ez akkor igaz, ha nem szöveg volt a node

        // ekkor eltoljuk a szöveget

        if (!(children.getLength() == 1 && children.item(0).getNodeType() == Node.TEXT\_NODE)) {

            Print(formatting);

        }

        // kiírjuk a node végét

        Println("</" + node.getNodeName() + ">");

    }

    private static void Print(String text) {

        System.out.print(text);

        if (writeToFile) {

            try {

                writer.write(text);

            } catch (IOException e) {

                // TODO Auto-generated catch block

                e.printStackTrace();

            }

        }

    }

    private static void Println(String text) {

        System.out.println(text);

        if (writeToFile) {

            try {

                writer.write(text + "\n");

            } catch (IOException e) {

                // TODO Auto-generated catch block

                e.printStackTrace();

            }

        }

    }

}

2b. Feladat

Adatmódosítás

A DOMModifyWJB0DC fájlban a dokumentumon végzett adatmódosítások találhatóak. A módosítások között találhatóak egyszerűbb és nehezebb feladatok is.  
Az osztály egyetlen függvényt tartalmaz, a ModifyFiveElements függvényt, ami megkapja a DOMReadWJB0DC fájlból a Document objektumot, amin ez az osztály végez módosításokat. Az öt módosítás, amit választottam:

1. Minden csapat nevét elrejtjük, és ehelyett a „CsapatX” nevet adjuk nekik, ahol X az adott csapat sorrendje a fájlban. A csapat neve egyben a csapat azonosítója is, tehát a CSAzon attribútomot változtatjuk meg minden csapat esetén.
2. Az első előforduló játékos születési évét megváltoztatjuk. Ez egy text Node tartalmának átírását jelenti.
3. Az első játék azonosítóját átírjuk 0-ra. Ezzel nem bontjuk meg az XML megfelelését a sémának, ugyanis egy olyan azonosítót adunk meg ami még biztosan nem szerepel, hiszen minden azonosító 1-től keződött.
4. Az utolsó vezető nevét változtatjuk meg. Ez szintén egy text Node tartalomátírással jár, viszont figyelni kell, hogy egy beágyazott gyerekelem értékét változtatjuk meg.
5. A Versenyez kapcsolótáblában megváltoztatjuk, hogy milyen csapat milyen versenyen versenyzett. Itt figyelni kell, hogy érvényes csapatnevet és verseny azonosítót adjunk meg, ami szerepel az XML dokumentumban.

A módosítások után a DOMReadWJB0DC osztályban megírt kiírató metódusokat használom, kód újra-felhasználás révén.

package hu.domparse.WJB0DC;

import org.w3c.dom.Document;

import org.w3c.dom.Element;

import org.w3c.dom.Node;

import org.w3c.dom.NodeList;

public class DOMModifyWJB0DC {

    public static void ModifyFiveElements(Document document) {

        NodeList csapatok = document.getElementsByTagName("Csapat");

        // minden csapat nevét/azonosítóját "elrejtjük" és egy indexet rendelünk hozzá

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

            Element csapat = (Element) csapatok.item(i);

            csapat.setAttribute("CSAzon", "Csapat" + i);

        }

        NodeList jatekosok = document.getElementsByTagName("Jatekos");

        // az első játékos születési évét átírjuk

        Element jatekos = (Element) jatekosok.item(0);

        Node szuletesiDatumNode = jatekos.getElementsByTagName("szuletesi\_datum").item(0);

        szuletesiDatumNode.setTextContent("1999-12-01");

        NodeList jatekok = document.getElementsByTagName("Jatek");

        // az első játék azonosítóját 0 ra állítjuk

        Element jatek = (Element) jatekok.item(0);

        jatek.setAttribute("JAzon", "0");

        NodeList vezetok = document.getElementsByTagName("Vezeto");

        // az utolsó vezető nevét átírjuk

        Element vezeto = (Element) vezetok.item(vezetok.getLength() - 1);

        vezeto.getElementsByTagName("nev").item(0).setNodeValue("Utolsó Vezető");

        NodeList versenyezTablak = document.getElementsByTagName("Versenyez");

        // az utolsó versenyez táblában átírjuk, ki versenyzett és milyen versenyen

        Element versenyezTabla = (Element) versenyezTablak.item(versenyezTablak.getLength() - 1);

        versenyezTabla.setAttribute("CSAzon", "Kerge Kecskék");

        versenyezTabla.setAttribute("VAzon", "2");

        //kiiratáshoz használjuk a már megírt metódusokat

        //ne írjunk fileba

        DOMReadWJB0DC.writeToFile = false;

        DOMReadWJB0DC.PrintNodeList(csapatok);

        DOMReadWJB0DC.PrintNodes(szuletesiDatumNode, 1);

        DOMReadWJB0DC.PrintNodes(jatek, 1);

        DOMReadWJB0DC.PrintNodes(vezeto, 1);

        DOMReadWJB0DC.PrintNodes(versenyezTabla, 1);

    }

}

2c. Feladat

Adatlekérdezés

Az XML dokumentum felhasználásával egyszerűen tudunk adatlekérdezéseket csinálni. A DOMQueryWJB0DC fájlban szintén egy metódus, a QueryFiveTimes szerepel, ahol szintén a DOMReadWJB0DC osztályból megkapott dokumentumot használjuk fel. A metódus során az alábbi 5 lekérdezést valósítottam meg:

1. Írjuk ki, hogy összesen mekkora összeget kaptak a csapatok a szponzorjaiktól. A lekérdezés során az összes csapaton végig kell iterálni, és kiolvasni a csapatok szponzorainak az összeg Node-jának a tartalmát.
2. Írjuk ki az összes olyan játékos nevét, aki nem magyar. Itt is egybeágyazott elemeket kell figyelembe venni.
3. Írjuk ki az összes olyan csapat nevét, akiknél több vezető is megfordult. A vezetők nem a csapat egész élettartama alatt vezetheti azt, változhatnak a vezetők. A megoldás során ki kell választani a vezetőkből a megfelelőket, és azoknak a csapatait redundancia nélkül kiírni (minden csapatot csak egyszer).
4. Számoljuk meg hány verseny van nyilvántartva. Ez a lekérdezés egy egyszerű megszámlálás, a verseny NodeList elemszáma.
5. Írjuk ki azon játékosok nevét, akik 24 év alattiak. Ez a lekérdezés is egyszerűbb, itt a már létrehozott játékosok NodeList-et használhatjuk.

package hu.domparse.WJB0DC;

import java.util.ArrayList;

import org.w3c.dom.Document;

import org.w3c.dom.Element;

import org.w3c.dom.NodeList;

public class DOMQueryWJB0DC {

    public static void QueryFiveTimes(Document document) {

        // írjuk ki a szponzorálások együttes összegét

        NodeList csapatok = document.getElementsByTagName("Csapat");

        int osszeg = 0;

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

            Element csapat = (Element) csapatok.item(i);

            Element szponzor = (Element) csapat.getElementsByTagName("Szponzor").item(0);

            osszeg += Integer.parseInt(szponzor.getElementsByTagName("osszeg").item(0).getTextContent());

        }

        System.out.println("Összes szponzorált összeg: " + osszeg);

        System.out.println();

        // írjuk ki azon játékosok nevét, akik nem magyarok

        System.out.println("Játékosok akik nem magyarok:");

        NodeList jatekosok = document.getElementsByTagName("Jatekos");

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

            Element jatekos = (Element) jatekosok.item(i);

            Element orszag = (Element) jatekos.getElementsByTagName("orszag").item(0);

            if (!orszag.getTextContent().equalsIgnoreCase("Magyarország")) {

                System.out.println(jatekos.getElementsByTagName("nev").item(0).getTextContent());

            }

        }

        System.out.println();

        // írjuk ki azon csapatok azonosítóját, ahol több vezető is volt már

        System.out.println("Csapatok ahol több vezető is volt:");

        NodeList vezetok = document.getElementsByTagName("Vezeto");

        ArrayList<String> csapatAzonositok = new ArrayList<String>();

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

            Element vezeto = (Element) vezetok.item(i);

            if (csapatAzonositok.contains(vezeto.getAttribute("CSAzon"))) {

                //benne volt már a listába, tehát ezen csapatnak több vezetője volt

                System.out.println(vezeto.getAttribute("CSAzon"));

            }

            else

            {

                //nem volt benne, tehát beleteszzük

                csapatAzonositok.add(vezeto.getAttribute("CSAzon"));

            }

        }

        //számoljuk meg hány verseny volt összesen

        //kifejezetten egyszerű lekérdezés

        NodeList versenyek = document.getElementsByTagName("Verseny");

        System.out.println("\nÖsszesen " + versenyek.getLength() + " versenyt rendeztek meg");

        //írjuk ki azon játékokos születési dátumát, akik 24 év alattiak

        // jatekosok nodelist már létezik

        System.out.println("\nAzon játékosok születési dátuma akik 24 éven aluliak");

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

            Element jatekos = (Element) jatekosok.item(i);

            Element eletkor = (Element) jatekos.getElementsByTagName("eletkor").item(0);

            if (Integer.parseInt(eletkor.getTextContent()) < 24) {

                System.out.println(jatekos.getElementsByTagName("szuletesi\_datum").item(0).getTextContent());

            }

        }

    }

}

2d. Feladat

Adatírás

A feladat során a Transformer osztályt használjuk fel. Az osztály segítségével egyszerűen megoldható a beolvasás és a kiírás is. A tranformer.transform függvény segítségével az egész XML dokumentumot szinte egy az egyben vissza tudjuk adni, kommentekkel és behúzásokkal együtt.

package hu.domparse.WJB0DC;

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.OutputKeys;

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.xml.sax.SAXException;

public class DOMWriteWJB0DC {

    public static void WriteDocument(String filePath)

    {

        try

        {

            File inputFile = new File(filePath);

            DocumentBuilderFactory dbFactory = DocumentBuilderFactory.newInstance();

            DocumentBuilder dBuilder = dbFactory.newDocumentBuilder();

            Document document = dBuilder.parse(inputFile);

            document.getDocumentElement().normalize();

            System.out.println("Writing to file");

            //a tranformer osztállyal készítünk egy XML filet

            TransformerFactory transformerFactory = TransformerFactory.newInstance();

            Transformer transformer = transformerFactory.newTransformer();

            DOMSource source = new DOMSource(document);

            //kiirjuk az xmlt a consolera

            StreamResult consoleResult = new StreamResult(System.out);

            transformer.transform(source, consoleResult);

            //kiirjuk fájlba az xmlt

            StreamResult result = new StreamResult(new File("WJB0DC1.xml"));

            transformer.transform(source, result);

        }

catch (SAXException | IOException | ParserConfigurationException | TransformerException e)

        {

            e.printStackTrace();

        }

    }

}