JEGYZŐKÖNY

Adatkezelés XML környezetben

Féléves feladat

Privát Katonai Csoport

Készítette:
Kriston Ádám
Programtervező informatikus
SYQ7E2

Dátum:
2023.12.05

Miskolc, 2023

A feladat leírása

Az ER modell egy privát katonai csoportokat vezető cég által tárolt adatokat modellezi le.

Az adatbázis a következő fontosabb elemeket tárolja le:

- A cégnek dolgozó csapatok
- A csapatok felszerelését
- A csapatok számára elérhető járműveket
- A csapatok tagjait
- A csapatok operátorjait

Csapatok:

Minden csapatnak egyedi **csapatneve** (kulcs, szöveges típus) van, illetve tárolják a csapatokat **alapítók neveit** (szöveges típus, több értékű), akik függetlenek a jelenlegi katonáktól (az összeférhetetlenség elkerülése miatt), az **alakulás dátumát** (dátum típus), a csapat által **elvégzett küldetések számát** (szám típus) és a **jelenlegi küldetést** (szöveges típus). A csapatok operátorai egy külön nyilvántartásban találhatóak Egy időben csak egy operátorja lehet a csapatnak és egy operátor csak egy csapatot vezet. A operátorok szintén függetlenek az összeférhetetlenség elkerülése végett.

Egy csapat 4 főből áll és egy fő csak egy csapatban lehet.

Jármű:

Minden csapat rendelkezik több járművel. A járműveknek van **neve** (szöveges típus), **azonosítója** (kulcs, szám), **tipusa** (szöveg) és **páncélozottsági szintje** (szöveg),

Felszerelés:

Minden katonának rendelkezik a saját felszerelésével. Minden katonának egy felszerelése van és minden felszerelés csak egy katonához tartozik. A felszerelés **azonosítóból** (kulcs, szám), **fegyver**

(összetett típus Fő (szöveg) mellék (szöveg)), kiegészítők (több értékű szöveg) és **páncélzat** (szöveg),

Katona:

Minden katonának van **neve** (szöveges típus), **azonosítója** (kulcs, szám), **születési dátumát** (dátum), születési **országát** (szöveg), **munkatapasztalatát**(szöveg), illetve ***korát*** (származtatott, szám, jelenlegi dátum - születési dátum alapján)]. Egy katona csak egy csapat rész lehet.

Operátorok:

A operátoroknak tárolják a **nevét** (szöveg), a **születési dátumát** (dátum típus) [+***korát*** (származtatott, szám, jelenlegi dátum - születési dátum alapján)], operátor azonosítóját [**azonosító** (kulcs, szám)], Az operátor **csatlakozásának idejét**[dátum]. Ha a csapatot már nem vezeti, akkor azt is tárolják, hogy **meddig** (dátum, opcionális típus) vezette.

Kapcsolatok

A mezőket 4 kapcsolat köti össze.

Tagság: A csapatok és a csapatokba tartozó katonák egytöbb kapcsolata. Mivel egy csapatban több tag van.

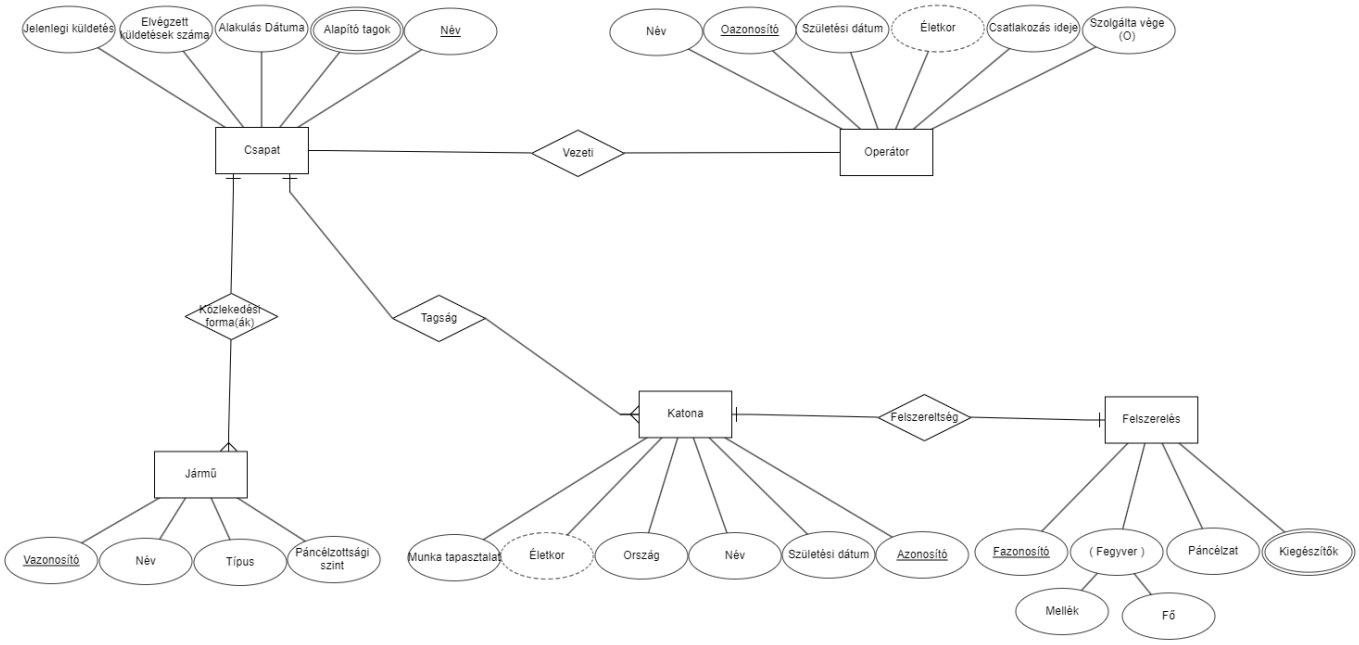
Közlekedési formák: A csapatok és a számukra elérhető járművek ami egy egytöbb kapcsolat, hiszen az adott jármű csak egy csapathoz tartozik de a csapatnak van több járműve is.

Felszereltség:A katonák által használt felszerelés ami egy egy-egy kapcsolat, hiszen minden katonának saját felszerelése van a saját specifikációival.

Vezeti: Operátorokat és a csapatokat összekötő egy-egy kapcsolat mivel az operátor csak egy csapatot vezethet, illetve egy csapatnak csak egy operátorja lehet adott időpontban.

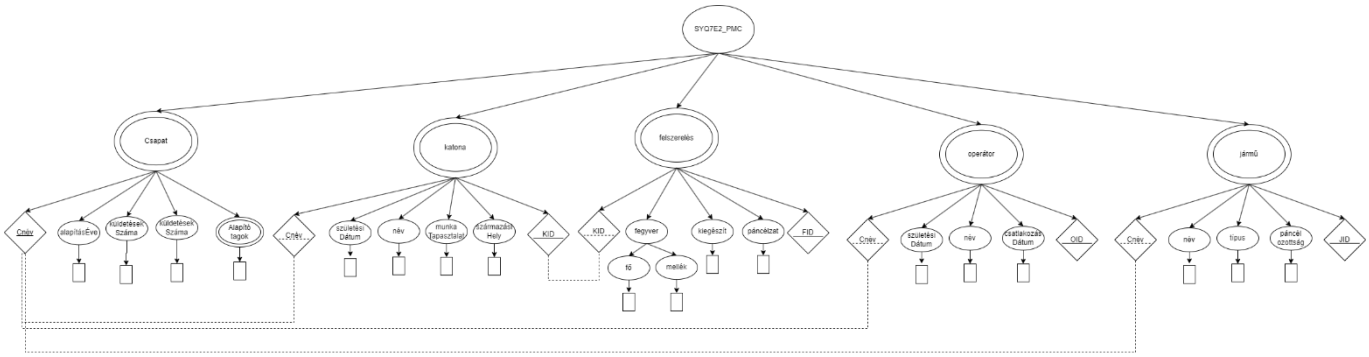
1a. Feladat

ER modell:



1b. Feladat

XDM modell: Az XDM modell gyökéreleme a SYQ7E2_PMC, 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. A kódot Visual Studio Code-ban készítettem el és másoltam ide, a formázás nem tökéletes

A gyökérelemben látható a kapcsolás az XMLSchemához

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

<XMLTaskSYQ7E2 xmlns:xs="http://www.w3.org/2001/XMLSchema-instance"
xs:noNamespaceSchemaLocation="XMLSchemaSYQ7E2.xsd">

    <!-- Csapatok -->
        <csapat Cnév="Alpha">
            <alapításÉve>1998</alapításÉve>
            <küldetésekSzáma>124</küldetésekSzáma>
            <jelenlegiküldetés>Project Black Sheep</jelenlegiküldetés>
                <alapítótagok>
                    <alapítótag>Ghost</alapítótag>
                    <alapítótag>David Mercer</alapítótag>
                </alapítótagok>
        </csapat>

        <csapat Cnév="Delta">
            <alapításÉve>1998</alapításÉve>
            <küldetésekSzáma>124</küldetésekSzáma>
            <jelenlegiküldetés>Encypted</jelenlegiküldetés>
                <alapítótagok>
                    <alapítótag>Eric Black</alapítótag>
                    <alapítótag>George Force</alapítótag>
                </alapítótagok>
        </csapat>

        <csapat Cnév="Foxtrott">
            <alapításÉve>2012</alapításÉve>
            <küldetésekSzáma>74</küldetésekSzáma>
            <jelenlegiKüldetés>Encypted</jelenlegiKüldetés>
                <alapítótagok>
                    <alapítótag>Sarah Ryder</alapítótag>
                    <alapítótag>Scott Ryder</alapítótag>
                    <alapítótag>Vetra Nyx</alapítótag>
                </alapítótagok>
        </csapat>

    <!-- Katonák -->
        <katona KID="01" Cnév="Alpha">
            <születésiDátum>2000.09.01</születésiDátum>
            <név>Alex Fey</név>
            <munkaTapasztalat>Beginner</munkaTapasztalat>
            <származásiHely>England</származásiHely>
        </katona>
```

```
<katona KID="12" Cnév="Alpha">
    <születésiDátum>1999.01.24</születésiDátum>
    <név>Ray Gunn</név>
    <munkaTapasztalat>Pro</munkaTapasztalat>
    <származásiHely>USA</származásiHely>
</katona>

<katona KID="06" Cnév="Alpha">
    <születésiDátum>1980.03.26</születésiDátum>
    <név>David Anderson</név>
    <munkaTapasztalat>Pro</munkaTapasztalat>
    <származásiHely>England</származásiHely>
</katona>

<katona KID="11" Cnév="Alpha">
    <születésiDátum>1995.04.02</születésiDátum>
    <név>Abel Monk</név>
    <munkaTapasztalat>Medium</munkaTapasztalat>
    <származásiHely>Germany</származásiHely>
</katona>

<katona KID="21" Cnév="Delta">
    <születésiDátum>2002.01.12</születésiDátum>
    <név>Fred Milk</név>
    <munkaTapasztalat>Beginner</munkaTapasztalat>
    <származásiHely>Greece</származásiHely>
</katona>

<katona KID="54" Cnév="Delta">
    <születésiDátum>1992.06.01</születésiDátum>
    <név>David Koronis</név>
    <munkaTapasztalat>Pro</munkaTapasztalat>
    <származásiHely>Hungary</származásiHely>
</katona>

<katona KID="69" Cnév="Delta">
    <születésiDátum>1994.07.03</születésiDátum>
    <név>Alberto</név>
    <munkaTapasztalat>Medium</munkaTapasztalat>
    <származásiHely>Spain</származásiHely>
</katona>

<katona KID="32" Cnév="Delta">
    <születésiDátum>1994.07.03</születésiDátum>
    <név>Liz Fey</név>
    <munkaTapasztalat>Pro</munkaTapasztalat>
    <származásiHely>USA</származásiHely>
</katona>

<katona KID="09" Cnév="Foxtrott">
    <születésiDátum>1998.03.17</születésiDátum>
    <név>Sarah Rider</név>
```

```
        <munkaTapasztalat>Pro</munkaTapasztalat>
        <származásiHely>USA</származásiHely>
    </katona>
    <katona KID="10" Cnév="Foxtrott">
        <születésiDátum>1998.03.17</születésiDátum>
        <név>Scott Rider</név>
        <munkaTapasztalat>Pro</munkaTapasztalat>
        <származásiHely>USA</származásiHely>
    </katona>
    <katona KID="19" Cnév="Foxtrott">
        <születésiDátum>1996.06.23</születésiDátum>
        <név>Vetra Nyx</név>
        <munkaTapasztalat>Pro</munkaTapasztalat>
        <származásiHely>Palaven</származásiHely>
    </katona>
    <katona KID="08" Cnév="Foxtrott">
        <születésiDátum>2000.09.17</születésiDátum>
        <név>Adam Koronis</név>
        <munkaTapasztalat>Pro</munkaTapasztalat>
        <származásiHely>Hungary</származásiHely>
    </katona>

<!-- Felszerelések -->

    <felszerelés FID="01" KID="01">
        <fegyver>
            <fő>AR</fő>
            <mellék>Pistol</mellék>
        </fegyver>
        <kiegészít>Red dot</kiegészít>
        <páncélzat>Heavy</páncélzat>
    </felszerelés>

    <felszerelés FID="02" KID="12">
        <fegyver>
            <fő>Sniper</fő>
            <mellék>Pistol</mellék>
        </fegyver>
        <kiegészít>Red dot</kiegészít>
        <páncélzat>Light</páncélzat>
    </felszerelés>

    <felszerelés FID="03" KID="06">
        <fegyver>
            <fő>AR</fő>
            <mellék>Pistol</mellék>
        </fegyver>
        <kiegészít>Explosive</kiegészít>
        <páncélzat>Light</páncélzat>
    </felszerelés>

    <felszerelés FID="04" KID="11">
        <fegyver>
```

```
            <fő>AR</fő>
            <mellék>Pistol</mellék>
        </fegyver>
        <kiegészít>Red dot</kiegészít>
        <páncélzat>Heavy</páncélzat>
    </felszerelés>

    <felszerelés FID="05" KID="21">
        <fegyver>
            <fő>AR</fő>
            <mellék>SMG</mellék>
        </fegyver>
        <kiegészít>Red dot</kiegészít>
        <páncélzat>Heavy</páncélzat>
    </felszerelés>

    <felszerelés FID="06" KID="54">
        <fegyver>
            <fő>Sniper</fő>
            <mellék>SMG</mellék>
        </fegyver>
        <kiegészít>Red dot</kiegészít>
        <páncélzat>Light</páncélzat>
    </felszerelés>

    <felszerelés FID="07" KID="67">
        <fegyver>
            <fő>AR</fő>
            <mellék>SMG</mellék>
        </fegyver>
        <kiegészít>Explosive</kiegészít>
        <páncélzat>Light</páncélzat>
    </felszerelés>

    <felszerelés FID="08" KID="32">
        <fegyver>
            <fő>AR</fő>
            <mellék>Pistol</mellék>
        </fegyver>
        <kiegészít>Red dot</kiegészít>
        <páncélzat>Heavy</páncélzat>
    </felszerelés>

    <felszerelés FID="09" KID="09">
        <fegyver>
            <fő>Sniper</fő>
            <mellék>Pistol</mellék>
        </fegyver>
        <kiegészít>Red dot</kiegészít>
        <páncélzat>Light</páncélzat>
    </felszerelés>

    <felszerelés FID="10" KID="10">
```

```
        <fegyver>
            <fő>AR</fő>
            <mellék>Pistol</mellék>
        </fegyver>
        <kiegészít>Red dot</kiegészít>
        <páncélzat>Light</páncélzat>
    </felszerelés>

    <felszerelés FID="11" KID="19">
        <fegyver>
            <fő>AR</fő>
            <mellék>SMG</mellék>
        </fegyver>
        <kiegészít>Explosive</kiegészít>
        <páncélzat>Heavy</páncélzat>
    </felszerelés>

    <felszerelés FID="12" KID="08">
        <fegyver>
            <fő>Sniper</fő>
            <mellék>Pistol</mellék>
        </fegyver>
        <kiegészít>Explosive</kiegészít>
        <páncélzat>Light</páncélzat>
    </felszerelés>


<!-- Operátorok -->
    <operátor Cnév="Alpha" OID="102">
        <születésiDátum>2000.11,01</születésiDátum>
        <név>Rose Alec</név>
        <csatlakozásDátum>2020.11.01</csatlakozásDátum>
    </operátor>

    <operátor Cnév="Alpha" OID="101">
        <születésiDátum>1965.10,25</születésiDátum>
        <név>Phantom</név>
        <csatlakozásDátum>1996.05.09</csatlakozásDátum>
    </operátor>

    <operátor Cnév="Delta" OID="103">
        <születésiDátum>1994.09.21</születésiDátum>
        <név>Ben Log</név>
        <csatlakozásDátum>2012.01.29</csatlakozásDátum>
    </operátor>

    <operátor Cnév="Foxtrott" OID="104">
        <születésiDátum>2000.11,11</születésiDátum>
        <név>SAM</név>
        <csatlakozásDátum>2020.11.12</csatlakozásDátum>
    </operátor>
```

```xml
    <!-- Járművek -->

        <jármű Cnev="Alpha" JID="01">
            <név>Jeep</név>
            <típus>Land</típus>
            <páncélozottság>Lvl 5</páncélozottság>
        </jármű>

        <jármű Cnev="Alpha" JID="02">
            <név>Ship</név>
            <típus>Water</típus>
            <páncélozottság>Lvl 2</páncélozottság>
        </jármű>

        <jármű Cnev="Delta" JID="03">
            <név>Jeep</név>
            <típus>Land</típus>
            <páncélozottság>Lvl 4</páncélozottság>
        </jármű>

        <jármű Cnev="Delta" JID="04">
            <név>Jetski</név>
            <típus>Water</típus>
            <páncélozottság>Lvl 1</páncélozottság>
        </jármű>

        <jármű Cnev="Delta" JID="05">
            <név>Ship</név>
            <típus>Water</típus>
            <páncélozottság>Lvl 4</páncélozottság>
        </jármű>

        <jármű Cnev="Foxtrott" JID="06">
            <név>Jeep</név>
            <típus>Land</típus>
            <páncélozottság>Lvl 5</páncélozottság>
        </jármű>

        <jármű Cnev="Foxtrott" JID="07">
            <név>Jet</név>
            <típus>Air</típus>
            <páncélozottság>Lvl 3</páncélozottság>
        </jármű>

</XMLTaskSYQ7E2>
```

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 típusokat, illetve 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 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="alapításÉve" type="xs:integer" />
    <xs:element name="küldetésekszáma" type="xs:integer" />
    <xs:element name="jelenlegiküldetés" type="xs:string" />
    <xs:element name="alapítótag" type="xs:string" />
    <xs:element name="születésiDátum" type="DateTípus" />
    <xs:element name="név" type="xs:string" />
    <xs:element name="munkaTapasztalat" type="xs:string" />
    <xs:element name="származásiHely" type="xs:string" />
    <xs:element name="fő" type="xs:string" />
    <xs:element name="mellék" type="xs:string" />
    <xs:element name="kiegészít" type="FelszerelésTípus" />
    <xs:element name="páncélzat" type="xs:string" />
    <xs:element name="születésiDátum" type="DateTípus" />
    <xs:element name="csatlakozásDátum" type="DateTípus" />
    <xs:element name="típus" type="xs:string" />
    <xs:element name="páncélozottság" type="xs:string" />



    <!-- Saját típusok -->
    <xs:simpleType name="DateTipus">
        <xs:restriction base="xs:date">
            <xs:minInclusive value="1900.01.01" />
            <xs:maxInclusive value="2023.12.31" />
        </xs:restriction>
    </xs:simpleType>

    <xs:simpleType name="FelszerelésTipus">
        <xs:restriction base="xs:string">
            <xs:enumeration value="Red dot"></xs:enumeration>
            <xs:enumeration value="Explosive"></xs:enumeration>
        </xs:restriction>
    </xs:simpleType>

    <!-- Komplex típusok -->
```

```
    <xs:complexType name="FegyverTipus">
        <xs:sequence>
            <xs:element ref="fő" />
            <xs:element ref="mellék" />
        </xs:sequence>
    </xs:complexType>

    <xs:complexType name="CsapatTipus">
        <xs:sequence>
            <xs:element ref="alapításÉve"/>
            <xs:element ref="küldetésekszáma"/>
            <xs:element ref="jelenlegiküldetés"/>
            <xs:element ref="alapítótag" minOccurs="1" maxOccurs="3"/>
        </xs:sequence>
        <xs:attribute name="Cnév" type="xs:string" />
    </xs:complexType>

    <xs:complexType name="KatonaTipus">
        <xs:sequence>
            <xs:element ref="születésiDátum"/>
            <xs:element ref="név"/>
            <xs:element ref="munkaTapasztalat"/>
            <xs:element ref="származásiHely"/>
        </xs:sequence>
        <xs:attribute name="Cnév" type="xs:string" />
        <xs:attribute name="KID" type="xs:integer" />
    </xs:complexType>

    <xs:complexType name="FelszerelésTipus">
        <xs:sequence>
            <xs:element name="fegyver" type="FegyverTípus"/>
            <xs:element ref="kiegészít"/>
            <xs:element ref="páncélzat"/>
        </xs:sequence>
        <xs:attribute name="FID" type="xs:integer" />
        <xs:attribute name="KID" type="xs:integer" />
    </xs:complexType>

    <xs:complexType name="OperátorTipus">
        <xs:sequence>
            <xs:element ref="születésiDátum"/>
            <xs:element ref="név"/>
            <xs:element ref="csatlakozásDátum"/>
        </xs:sequence>
        <xs:attribute name="Cnév" type="xs:string" />
        <xs:attribute name="OID" type="xs:integer" />
    </xs:complexType>

    <xs:complexType name="JárműTipus">
        <xs:sequence>
            <xs:element ref="név"/>
            <xs:element ref="típus"/>
            <xs:element ref="páncélozottság"/>
```

```
        </xs:sequence>
        <xs:attribute name="Cnév" type="xs:string" />
        <xs:attribute name="JID" type="xs:integer" />
    </xs:complexType>

     <!-- Elemek -->
    <xs:element name="SYQ7E2_ESportok">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="Csapat" type="CsapatTipus" maxOccurs="unbounded" />
                <xs:element name="Katona" type="KatonaTipus" maxOccurs="unbounded" />
                <xs:element name="Felszerelés" type="FelszerelésTipus"
maxOccurs="unbounded" />
                <xs:element name="Operátor" type="OperátorTipus" maxOccurs="unbounded"
/>
                <xs:element name="Jármű" type="JárműTipus" maxOccurs="unbounded" />
            </xs:sequence>
        </xs:complexType>

        <!--Elsődleges kulcsok-->

        <xs:key name="CsapatKulcs">
            <xs:selector xpath="Csapat" />
            <xs:field xpath="@Cnév" />
        </xs:key>

        <xs:key name="KatonaKulcs">
            <xs:selector xpath="Katona" />
            <xs:field xpath="@KID" />
        </xs:key>

        <xs:key name="FelszerelésKulcs">
            <xs:selector xpath="Felszerelés" />
            <xs:field xpath="@FID" />
        </xs:key>

        <xs:key name="OperátorKulcs">
            <xs:selector xpath="Operátor" />
            <xs:field xpath="@OID" />
        </xs:key>

        <xs:key name="JárműKulcs">
            <xs:selector xpath="Jármű" />
            <xs:field xpath="@JID" />
        </xs:key>

        <!-- Idegen kulcsok -->

        <xs:key name="KatonaKulcs">
            <xs:selector xpath="Katona" />
            <xs:field xpath="@Cnév" />
        </xs:key>
```

```
        <xs:key name="FelszerelésKulcs">
            <xs:selector xpath="Felszerelés" />
            <xs:field xpath="@KID" />
        </xs:key>

        <xs:key name="OperátorKulcs">
            <xs:selector xpath="Operátor" />
            <xs:field xpath="@Cnév" />
        </xs:key>

        <xs:key name="JárműKulcs">
            <xs:selector xpath="Jármű" />
            <xs:field xpath="@Cnév" />
        </xs:key>

        <!-- 1:1 kapcsolat-->

        <xs:unique name="FelszereltségKulcs">
            <xs:selector xpath="Felszereltség" />
            <xs:field xpath="@KID" />
        </xs:unique>


</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 jegyzőkönyvbe.

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 XML_SYQ7E2_read.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.

```java
package hu.domparse.SYQ7E2;

import org.w3c.dom.*;
import org.xml.sax.SAXException;

import javax.xml.parsers.*;
import java.io.*;
import java.util.StringJoiner;

public class DOMReadSYQ7E2 {

    public static void main(String[] args) {
        DOMReadSYQ7E2.ReadDocument("src/hu/domparse/SYQ7E2/XMLSYQ7E2.xml");
    }

    public static void ReadDocument(String filePath) {
        try {
            // Fájl beolvasása
            File inputFile = new File(filePath);
            DocumentBuilderFactory dbFactory = DocumentBuilderFactory.newInstance();
            DocumentBuilder dBuilder = dbFactory.newDocumentBuilder();
            Document doc = dBuilder.parse(inputFile);
            doc.getDocumentElement().normalize();
            printDocument(doc);
        } catch (ParserConfigurationException e) {
            e.printStackTrace();
        } catch (IOException e) {
            e.printStackTrace();
        } catch (SAXException e) {
```

```
            e.printStackTrace();
        }
    }

    public static void printDocument(Document doc) {
        try {
            File outputFile = new File("XML_SYQ7E2_read.xml");
            PrintWriter writer = new PrintWriter(new FileWriter(outputFile, true));

            // Kiírjuk az XML fõgyökér elemét a konzolra és fájlba
            // ------
            Element rootElement = doc.getDocumentElement();
            String rootName = rootElement.getTagName();
            StringJoiner rootAttributes = new StringJoiner(" ");
            NamedNodeMap rootAttributeMap = rootElement.getAttributes();

            for (int i = 0; i < rootAttributeMap.getLength(); i++) {
                Node attribute = rootAttributeMap.item(i);
                rootAttributes.add(attribute.getNodeName() + "=\"" +
attribute.getNodeValue() + "\"");
            }

            System.out.print("<?xml version=\"1.0\" encoding=\"UTF-8\"?>\n");
            writer.print("<?xml version=\"1.0\" encoding=\"UTF-8\"?>\n");

            System.out.print("<" + rootName + " " + rootAttributes.toString() + ">\n");
            writer.print("<" + rootName + " " + rootAttributes.toString() + ">\n");

            NodeList csapatList = doc.getElementsByTagName("csapat");
            NodeList katonaList = doc.getElementsByTagName("katona");
            NodeList felszerelésList = doc.getElementsByTagName("felszerelés");
            NodeList operátorList = doc.getElementsByTagName("operátor");
            NodeList járműList = doc.getElementsByTagName("jármû");

            // Kiírjuk az XML-t a konzolra megtartva az eredeti formázást
            // ------
            printNodeList(csapatList, writer);
            System.out.println("");
            writer.println("");
            printNodeList(katonaList, writer);
            System.out.println("");
            writer.println("");
            printNodeList(felszerelésList, writer);
            System.out.println("");
            writer.println("");
            printNodeList(operátorList, writer);
            System.out.println("");
            writer.println("");
            printNodeList(járműList, writer);

            // Zárjuk le az XML gyökér elemét
            // ------
            System.out.println("</" + rootName + ">");
```

```
            writer.append("</" + rootName + ">");

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

    public static void printNodeList(NodeList nodeList, PrintWriter writer) {
        for (int i = 0; i < nodeList.getLength(); i++) {
            Node node = nodeList.item(i);
            printNode(node, 1, writer);
            System.out.println("");
            writer.println("");
        }
    }

    public static void printNode(Node node, int indent, PrintWriter writer) {
        // Ha a node egy szöveg node, akkor kiírjuk a tartalmát
        // --...--
        if (node.getNodeType() == Node.ELEMENT_NODE) {
            Element element = (Element) node;
            String nodeName = element.getTagName();
            StringJoiner attributes = new StringJoiner(" ");
            NamedNodeMap attributeMap = element.getAttributes();
            // Kiírjuk az elem nevét és attribÃºtumait
            for (int i = 0; i < attributeMap.getLength(); i++) {
                Node attribute = attributeMap.item(i);
                attributes.add(attribute.getNodeName() + "=\"" +
attribute.getNodeValue() + "\"");
            }
            // Kiírjuk az elem tartalmát
            System.out.print(getIndentString(indent));
            System.out.print("<" + nodeName + " " + attributes.toString() + ">");
            writer.print(getIndentString(indent));
            writer.print("<" + nodeName + " " + attributes.toString() + ">");

            NodeList children = element.getChildNodes();
            if (children.getLength() == 1 && children.item(0).getNodeType() ==
Node.TEXT_NODE) {
                System.out.print(children.item(0).getNodeValue());
                writer.print(children.item(0).getNodeValue());
            } else {
                System.out.println();
                writer.println();
                for (int i = 0; i < children.getLength(); i++) {
                    printNode(children.item(i), indent + 1, writer);
                }
                System.out.print(getIndentString(indent));
                writer.print(getIndentString(indent));
            }
            System.out.println("</" + nodeName + ">");
            writer.println("</" + nodeName + ">");
```

```
        }

    }

    public static String getIndentString(int indent) {
        StringBuilder sb = new StringBuilder();
        for (int i = 0; i < indent; i++) {
            // A szóközök száma, amivel indentálunk
            sb.append("  ");
        }
        return sb.toString();
    }
}
```

2b. Feladat

Adatmódosítás

A DOMModifySYQ7E2 fájlban a dokumentumon végzett adatmódosítások vannak. A módosítások között találhatóak egyszerűbbek és nehezebbek is. Az osztály egyetlen függvényt tartalmaz, a Modify függvényt, ami megkapja az eredeti xml fájlból generált Document objektumot, amin ez az osztály végez módosításokat.
Az öt módosítás amik:

- Minden csapat nevét átalakítjuk arra hogy „Team x” ahol az x azt jelöli hogy hanyadik csapat a sorrendbe
- Az első katona születési dátumát kicseréljuk egy másikra. Ez egy text Node tartalmának átírását jelenti.
- Az első felszerelés azonosítójának megváltoztatása 13-ra. Ezzel nem borul fel az XML séma mert nincs másik olyan felszerelés ami 13 szonosítóval rendelkezne.
- Utolsó operátor nevát változtassuk meg. Ez is egy text Node tartalomátírással jár.
- Az utolsó csapat nevének és küldetés számának megváltoztatása.

A módosítások után a printDocumant metódusot használom,hogy kiírassam az file-t.

```
package hu.domparse.SYQ7E2;

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;

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

public class DOMModifySYQ7E2 {

    // Adatok Beolvasása
    // --...--
    public static void main(String[] args) throws TransformerException {
        Document doc = null;
```

```
        try {
        File inputFile = new File("src/hu/domparse/SYQ7E2/XMLSYQ7E2.xml");
        DocumentBuilderFactory dbFactory = DocumentBuilderFactory.newInstance();
        DocumentBuilder dBuilder = dbFactory.newDocumentBuilder();
        doc = dBuilder.parse(inputFile);
        doc.getDocumentElement().normalize();
    } catch (ParserConfigurationException e) {
        e.printStackTrace();
    } catch (IOException e) {
        e.printStackTrace();
    } catch (SAXException e) {
        e.printStackTrace();
    }
        Modify(doc);
    }

    public static void Modify(Document document) throws TransformerException {

        // #1
        // Változtassuk meg a csapatok nevét egy szám értékre az anonimitás érdekében
        // --...--
        NodeList csapatok = document.getElementsByTagName("csapat");
        for (int i = 0; i < csapatok.getLength(); i++) {
            Element csapat = (Element) csapatok.item(i);
            csapat.setAttribute("Cnév","Team: " + i);
        }
        // --...--

        // #2
        // Változtassuk meg a z egyik katonánk életkorát
        // --...--
        NodeList katonák = document.getElementsByTagName("katona");
        Element katona = (Element) katonák.item(0);
        Node szuletesiDatumNode = katona.getElementsByTagName("születésiDátum").item(0);
        szuletesiDatumNode.setTextContent("1999.12.01");
        // --...--

        // #3
        // Változtassuk meg az elsõ felszerelés azonosítóját a használó kérésére
        // --...--
        NodeList felszerelések = document.getElementsByTagName("felszerelés");
        Element felszerelés = (Element) felszerelések.item(0);
        felszerelés.setAttribute("FID", "12");
        // --...--

        // #4
        // Az utolsó operátor személyazonossága megváltozott frissítsük az információját
        // --...--
        NodeList operátorok = document.getElementsByTagName("operátor");
        Element operátor = (Element) operátorok.item(operátorok.getLength() - 1);
        operátor.getElementsByTagName("név").item(0).setNodeValue("Kevin");
        // --...--
```

```
        // #5
        // Az utolsó csapat név váltást kért a legutóbbi küldetés után
        // Frissítsük az infromációkat
        // --...--
        Element csapat = (Element) csapatok.item(csapatok.getLength() - 1);
        csapat.setAttribute("Cnév", "Death Squad");
        Node küldetések = csapat.getElementsByTagName("küldetésekSzáma").item(0);
        küldetések.setTextContent("200");
        // --...--

        printDocument(document);
    }

    public static void printDocument(Document document) throws TransformerException {
        TransformerFactory tf = TransformerFactory.newInstance();
        Transformer transformer = tf.newTransformer();
        transformer.setOutputProperty(OutputKeys.OMIT_XML_DECLARATION, "no");
        transformer.setOutputProperty(OutputKeys.INDENT, "yes");
        StringWriter writer = new StringWriter();
        transformer.transform(new DOMSource(document), new StreamResult(writer));
        String output = writer.getBuffer().toString();
        System.out.println(output);
    }
}
```

2c. Feladat

Adatlekérdezés

DOMQuerySYQ7E2 fájlban a Query metódust használjuk, ahol az eredeti xml fájlból generált Document objektumot használjuk fel. A metódban lekérdezés 5 szerepel:

- Írjuk ki, hogy összesen elvégzett küldetések száma. A lekérdezés során az összes csapaton végig kell iterálni, és kiolvasni a küldetés számot a „küldetésekSzáma” Node-ból.
- Írjuk ki azt az operátort aki a Foxtrott csapathoz tartozik
- Az összes katona közül írjuk ki azokat akik Amerikai származásúak.
- Számoljuk meg az összes járművet ami szervezetnél van. Ez egy egyszerű összeszámlálás.
- Keressük meg a legfiatalabb csapatot. Ez egy minimumkeresés

```
package hu.domparse.SYQ7E2;

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.NodeList;
import org.xml.sax.SAXException;

public class DOMQuerySYQ7E2 {

    // Adatok Beolvas?sa
    // --...--
    public static void main(String[] args) {
        Document doc = null;
        try {
        File inputFile = new
File("src/hu/domparse/SYQ7E2/XMLSYQ7E2.xml");
        DocumentBuilderFactory dbFactory =
DocumentBuilderFactory.newInstance();
        DocumentBuilder dBuilder =
dbFactory.newDocumentBuilder();
        doc = dBuilder.parse(inputFile);
        doc.getDocumentElement().normalize();
    } catch (ParserConfigurationException e) {
        e.printStackTrace();
    } catch (IOException e) {
        e.printStackTrace();
    } catch (SAXException e) {
        e.printStackTrace();
    }
        Query(doc);
    }

    public static void Query(Document document) {

        // #1
        // N?zz?k meg mennyi az eddig elv?gzett
k?ldet?sek sz?ma
        // --...--
        NodeList csapatok =
document.getElementsByTagName("csapat");
        int ?ssz = 0;
        for (int i = 0; i < csapatok.getLength();
i++) {
            Element csapat = (Element)
csapatok.item(i);
            ?ssz +=
Integer.parseInt(csapat.getElementsByTagName("k?ldet?sekSz?ma").item(0).getText
Content());
        }
        System.out.println("Elv?gzett k?ldet?sek
sz?ma: " + ?ssz);
        System.out.println("--------");
        // --...--
```

```
                                    // #2
                                    // Írjuk ki az Foxtrott csapat Oper?tor?t
                                    // --...--
                                    System.out.println("Foxtrott Oper?tora:
");
                                    NodeList oper?torok =
document.getElementsByTagName("oper?tor");
                                    for (int i = 0; i <
oper?torok.getLength(); i++) {
                                        Element oper?tor = (Element)
oper?torok.item(i);
                                        if
(oper?tor.getAttribute("Cn?v").contains( "Foxtrott")) {
                                            System.out.println(oper?tor.getEle
mentsByTagName("n?v").item(0).getTextContent());
                                        }
                                    }
                                    System.out.println("--------");
                                    // --...--

                                    // #3
                                    // Írjuk ki az amerikai katon?k nev?t
                                    // --...--
                                    System.out.println("Amerikaiak:");
                                    NodeList katon?k =
document.getElementsByTagName("katona");
                                    for (int i = 0; i < katon?k.getLength();
i++) {
                                        Element katona = (Element)
katon?k.item(i);
                                        Element orszag = (Element)
katona.getElementsByTagName("sz?rmaz?siHely").item(0);
                                        if
(orszag.getTextContent().contains("USA")) {
                                            System.out.println(katona.getElemen
tsByTagName("n?v").item(0).getTextContent());
                                        }
                                    }
                                    System.out.println("--------");
                                    // --...--
                                package hu.domparse.SYQ7E2;

                                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.NodeList;
```

```java
import org.xml.sax.SAXException;

public class DOMQuerySYQ7E2 {

    // Adatok Beolvasása
    // --...--
    public static void main(String[] args) {
        Document doc = null;
        try {
        File inputFile = new
File("src/hu/domparse/SYQ7E2/XMLSYQ7E2.xml");
        DocumentBuilderFactory dbFactory =
DocumentBuilderFactory.newInstance();
        DocumentBuilder dBuilder =
dbFactory.newDocumentBuilder();
        doc = dBuilder.parse(inputFile);
        doc.getDocumentElement().normalize();
    } catch (ParserConfigurationException e) {
        e.printStackTrace();
    } catch (IOException e) {
        e.printStackTrace();
    } catch (SAXException e) {
        e.printStackTrace();
    }
        Query(doc);
    }

    public static void Query(Document document) {

        // #1
        // Nézzük meg mennyi az eddig elvégzett
küldetések száma
        // --...--
        NodeList csapatok =
document.getElementsByTagName("csapat");
        int össz = 0;
        for (int i = 0; i < csapatok.getLength();
i++) {
            Element csapat = (Element)
csapatok.item(i);
            össz +=
Integer.parseInt(csapat.getElementsByTagName("küldetésekSzáma").item(0).getTextCon
tent());
        }
        System.out.println("Elvégzett küldetések
száma: " + össz);
        System.out.println("--------");
        // --...--

        // #2
        // Ã rjuk ki az Foxtrott csapat Operátorát
        // --...--
        System.out.println("Foxtrott Operátora: ");
```

```
                                        NodeList operátorok =
document.getElementsByTagName("operátor");
                                        for (int i = 0; i < operátorok.getLength();
i++) {
                                            Element operátor = (Element)
operátorok.item(i);
                                            if
(operátor.getAttribute("Cnév").contains( "Foxtrott")) {
                                                System.out.println(operátor.getElem
entsByTagName("név").item(0).getTextContent());
                                            }
                                        }
                                        System.out.println("--------");
                                        // --...--

                                        // #3
                                        // Ã rjuk ki az amerikai katonák nevét
                                        // --...--
                                        System.out.println("Amerikaiak:");
                                        NodeList katonák =
document.getElementsByTagName("katona");
                                        for (int i = 0; i < katonák.getLength();
i++) {
                                            Element katona = (Element)
katonák.item(i);
                                            Element orszag = (Element)
katona.getElementsByTagName("származásiHely").item(0);
                                            if
(orszag.getTextContent().contains("USA")) {
                                                System.out.println(katona.getElemen
tsByTagName("név").item(0).getTextContent());
                                            }
                                        }
                                        System.out.println("--------");
                                        // --...--

                                        // #4
                                        // Számoljuk az meg az összes használt
járművet
                                        // --...--
                                        NodeList járművek =
document.getElementsByTagName("jármû");
                                        int darab = 0;
                                        for (int i = 0; i < járművek.getLength();
i++) {
                                            darab++;
                                        }
                                        System.out.println("A járművek száma: " +
darab);
                                        System.out.println("--------");
                                        // --...--
```

```
                                    // #5
                                    // Melyik a legfiatalabb csapat keletkezési
éve
                                    // --...--
                                    System.out.println("Legfiatalabb csapat");
                                    Element csapatfirst = (Element)
csapatok.item(0);
                                    Element elsõév = (Element)
csapatfirst.getElementsByTagName("alapításÉve").item(0);
                                    int év =
Integer.parseInt(elsõév.getTextContent());
                                    for (int i = 0; i < csapatok.getLength();
i++) {
                                        Element csapat = (Element)
csapatok.item(i);
                                        Element létrehozás = (Element)
csapat.getElementsByTagName("alapításÉve").item(0);
                                        if
(Integer.parseInt(létrehozás.getTextContent()) > év) {
                                            System.out.println(csapat.getAttrib
ute("Cnév"));
                                        }
                                    }
                                    // --...--
                                }
                            }

                                    // #4
                                    // Sz?moljuk az meg az ?sszes haszn?lt
j?rm?vet
                                    // --...--
                                    NodeList j?rm?vek =
document.getElementsByTagName("j?rm?");
                                    int darab = 0;
                                    for (int i = 0; i < j?rm?vek.getLength();
i++) {
                                        darab++;
                                    }
                                    System.out.println("A j?rm?vek sz?ma: " +
darab);
                                    System.out.println("--------");
                                    // --...--


                                    // #5
                                    // Melyik a legfiatalabb csapat
keletkez?si ?ve
                                    // --...--
                                    System.out.println("Legfiatalabb csapat");
                                    Element csapatfirst = (Element)
csapatok.item(0);
                                    Element els??v = (Element)
csapatfirst.getElementsByTagName("alap?t?s?ve").item(0);
```

```
                                    int €v =
Integer.parseInt(els€€v.getTextContent());
                                    for (int i = 0; i < csapatok.getLength();
i++) {
                                        Element csapat = (Element)
csapatok.item(i);
                                        Element l€trehoz€s = (Element)
csapat.getElementsByTagName("alap€t€s€ve").item(0);
                                        if
(Integer.parseInt(l€trehoz€s.getTextContent()) > €v) {
                                            System.out.println(csapat.getAttrib
ute("Cn€v"));
                                        }
                                    }
                                    // --...--
                                }
                            }
```

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.

```java
package hu.domparse.SYQ7E2;

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

import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.DocumentBuilderFactory;
import javax.xml.parsers.ParserConfigurationException;
import javax.xml.transform.Transformer;
import javax.xml.transform.TransformerException;
import javax.xml.transform.TransformerFactory;
import javax.xml.transform.dom.DOMSource;
import javax.xml.transform.stream.StreamResult;

import org.w3c.dom.Document;
import org.xml.sax.SAXException;

public class DOMWriteSYQ7E2 {

    public static void main(String[] args) {
        Write("src/hu/domparse/SYQ7E2/XMLSYQ7E2.xml");
    }

    public static void Write(String filePath)
    {
```

```
        try
        {
            File inputFile = new File(filePath);

            //documentum elûkószõtóse
            DocumentBuilderFactory dbFactory = DocumentBuilderFactory.newInstance();
            DocumentBuilder dBuilder = dbFactory.newDocumentBuilder();
            Document document = dBuilder.parse(inputFile);
            document.getDocumentElement().normalize();

            System.out.println("Writing into the file");

            //a tranformer osztíllyal segõtsógóvel kószõtjök az 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 fíjlba az xmlt
            StreamResult result = new StreamResult(new File("SYQ7E2_1.xml"));
            transformer.transform(source, result);

        }
        catch (SAXException e)
        {
            e.printStackTrace();
        }
        catch (IOException e)
        {
            e.printStackTrace();
        }
        catch (ParserConfigurationException e)
        {
            e.printStackTrace();
        }
        catch (TransformerException e)
        {
            e.printStackTrace();
        }
    }

}
```