JEGYZŐKÖNYV

Adatkezelés XML környezetben

Autósiskolák nyilvántartása

Készítette: Nagy Máté

Neptunkód: U3ROFS

Dátum: 2023.11.09.

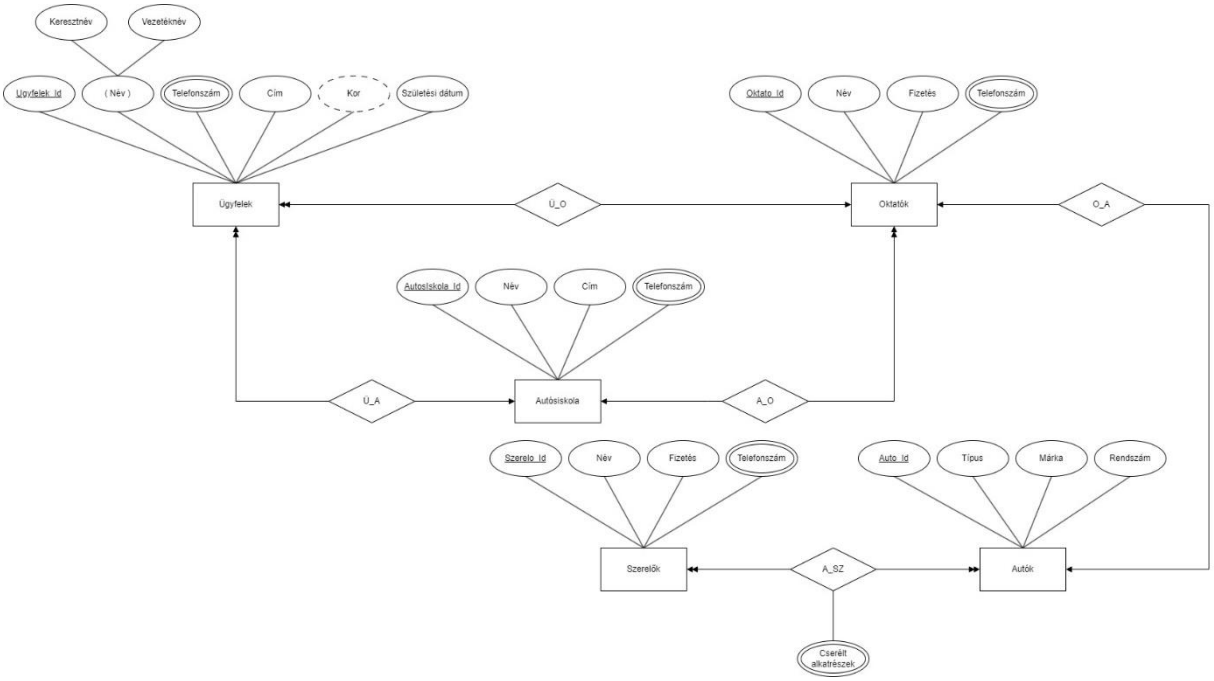
Tartalom

A feladat témája

A beadandó témája egy olyan adatbázis, amely az autósiskolákat és a hozzá kapcsolódó egyedeket tartja nyilván.

- **Autósiskola**
  - Autosiskola_Id: ez az elsődleges kulcs.
  - Név: ez az autósiskola neve.
  - Cím: ez az autósiskola címe.
  - Telefonszám: többértékű tulajdonság az autósiskola telefonszámai.
- **Ügyfelek**
  - Ugyfelek_Id: ez az elsődleges kulcs.
  - Név: több értékű tulajdonság(keresztnév, vezetéknév).
  - Cím: az ügyfelek címe.
  - Kor: az ügyfél életkora.
  - Születési dátum: születési dátum.
- **Oktatók**
  - Oktato_Id: ez az elsődleges kulcs
  - Név: oktató teljes neve.
  - Fizetés: oktató fizetése.
  - Telefonszám: többértékű tulajdonság az oktató telefonszámai.
- **Autók**
  - Auto_Id: ez az elsődleges kulcs.
  - Típus: az autó típusa.
  - Márka: az autó márkája
  - Rendszám: az autó rendszáma.
- **Szerelők**
  - Szerelo_Id: ez az elsődleges kulcs.
  - Név: a szerelő teljes neve.
  - Fizetés: szerelő fizetése.
  - Telefonszám: többértékű tulajdonság az szerelő telefonszámai.

A feladat ER modellje:



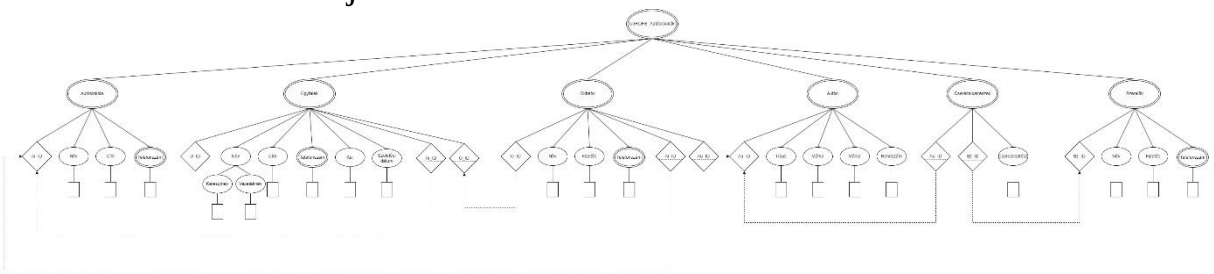
Az egyedek közötti kapcsolat

- Ü_A kapcsolat: egy autósiskolához több ügyfél tartozik 1:N.
- A_O kapcsolat: egy autósiskolához több oktató tartozik 1:N.
- Ü_O kapcsolat: egy oktatóhoz több ügyfél tartozik 1:N.
- O_A kapcsolat: egy oktatóhoz egy autó tartozik 1:1.
- A_SZ: több autó több szerelőhöz tartozik.

Az ER-modell konvertálása XDM modellre

XDM modellnél háromféle jelölést alkalmazhatunk. Ezek az ellipszis, a rombusz, illetve a téglalap. Az ellipszis jelöli az elemeket minden egyedből elem lesz, ezen felül a tulajdonságokból is. A rombusz jelöli az attribútumokat, amelyek a kulcs tulajdonságokból keletkeznek. A téglalap jelöli a szöveget, amely majd az XML dokumentumban fog megjelenni. Azoknak az elemeknek, amelyek többször is előfordulhatnak, a jelölése dupla ellipszissel történik. Az idegenkulcsok és a kulcsok közötti kapcsolatot szaggatott vonalas nyíllal jelöljük.

A feladat XDM modellje:



XML dokumentum készítése

Az XDM modell alapján az XML dokumentumot úgy készítettem el, hogy először is a root elementtel kezdtem, ami az U3ROFS_Autosiskolak volt.

A gyermek elemeiből 3-3 példányt hoztam létre, ezeknek az elemeknek az attribútumai közé tartoznak a kulcsok, illetve idegenkulcsok is, mindezek után ezeknek az elemeknek létrehoztam a többi gyermek elementet is.

XML dokumentum forráskódja

```
?xml version="1.0" encoding="utf-8"?>
<U3ROFS_Autosiskolak xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:noNamespaceSchemaLocation="XMLSchemaU3ROFS.xsd">

    <!-- Autosiskolák -->
    <Autosiskola ai_id="1">
        <nev>Go Car</nev>
        <cim>1111 Budapest, Kossuth Lajos utca 1.</cim>
        <telefon>06-70-123-4567</telefon>
        <telefon>06-30-123-4567</telefon>
    </Autosiskola>
    <Autosiskola ai_id="2">
        <nev>Guruljunk</nev>
        <cim>1111 Budapest, Petőfi utca 2.</cim>
        <telefon>06-70-123-4567</telefon>
    </Autosiskola>
    <Autosiskola ai_id="3">
        <nev>UNI</nev>
        <cim>3515, Miskolc, Egyetem út 1</cim>
        <telefon>06-70-123-4567</telefon>
        <telefon>06-30-123-4567</telefon>
    </Autosiskola>

    <!--Ügyfelek-->
    <Ugyfel u_id="1" ai_id="1" o_id="1">
        <nev>
            <vezeteknev>Nagy</vezeteknev>
```

```
            <keresztnev>Máté</keresztnev>
        </nev>
        <telefon>06-70-123-4567</telefon>
        <telefon>06-30-123-4567</telefon>
        <kor>18</kor>
        <szuletesi_datum>2003-01-01</szuletesi_datum>
    </Ugyfel>
    <Ugyfel u_id="2" ai_id="2" o_id="2">
        <nev>
            <vezeteknev>Gyáni</vezeteknev>
            <keresztnev>Kevin</keresztnev>
        </nev>
        <telefon>06-70-123-4567</telefon>
        <kor>18</kor>
        <szuletesi_datum>2003-01-01</szuletesi_datum>
    </Ugyfel>
    <Ugyfel u_id="3" ai_id="3" o_id="3">
        <nev>
            <vezeteknev>Kovács</vezeteknev>
            <keresztnev>Ádám</keresztnev>
        </nev>
        <telefon>06-70-123-4567</telefon>
        <telefon>06-30-123-4567</telefon>
        <kor>18</kor>
        <szuletesi_datum>2003-01-01</szuletesi_datum>
    </Ugyfel>

    <!-- Oktatók -->
    <Oktato o_id="1" ai_id="1">
        <nev>Kovács János</nev>
        <fizetes>300000</fizetes>
        <telefon>06-70-123-4567</telefon>
        <telefon>06-30-123-4567</telefon>
    </Oktato>
    <Oktato o_id="2" ai_id="2">
        <nev>Kiss János</nev>
        <fizetes>300000</fizetes>
        <telefon>06-70-123-4567</telefon>
    </Oktato>
    <Oktato o_id="3" ai_id="3">
        <nev>Nagy János</nev>
        <fizetes>300000</fizetes>
        <telefon>06-70-123-4567</telefon>
        <telefon>06-30-123-4567</telefon>
    </Oktato>

    <!-- Autók -->
    <Auto au_id="1" o_id="1">
        <rendszam>ABC-111</rendszam>
```

```
        <tipus>Astra</tipus>
        <marka>opel</marka>
    </Auto>
    <Auto au_id="2" o_id="2">
        <rendszam>ABC-222</rendszam>
        <tipus>Focus</tipus>
        <marka>Ford</marka>
    </Auto>
    <Auto au_id="3" o_id="3">
        <rendszam>ABC-333</rendszam>
        <tipus>Corolla</tipus>
        <marka>Toyota</marka>
    </Auto>

    <!-- Szerelők -->
    <Szerelo sz_id="1" au_id="1">
        <nev>Kovács Abdul</nev>
        <fizetes>400000</fizetes>
        <telefon>06-70-123-1111</telefon>
        <telefon>06-30-123-1112</telefon>
    </Szerelo>
    <Szerelo sz_id="2" au_id="2">
        <nev>Kiss Adorján</nev>
        <fizetes>420000</fizetes>
        <telefon>06-70-123-2222</telefon>
    </Szerelo>
    <Szerelo sz_id="3" au_id="3">
        <nev>Nagy Ferenc</nev>
        <fizetes>450000</fizetes>
        <telefon>06-70-123-3333</telefon>
        <telefon>06-30-123-3334</telefon>
    </Szerelo>

    <!-- Cserélt alkatrészek -->
    <cserealkatreszek au_id="1" sz_id="1">
        <cserealkatresz>fékbetét</cserealkatresz>
        <cserealkatresz>féktárcsa</cserealkatresz>
    </cserealkatreszek>
    <cserealkatreszek au_id="2" sz_id="2">
        <cserealkatresz>motor</cserealkatresz>
        <cserealkatresz>lengőkar</cserealkatresz>
    </cserealkatreszek>
    <cserealkatreszek au_id="3" sz_id="3">
        <cserealkatresz>szélvédő</cserealkatresz>
    </cserealkatreszek>

</U3ROFS_Autosiskolak>
```

XMLSchema készítése

Az XML Schemám meghatározza az adatokat, mint például az iskola nevét, címét, és a telefon számokat, amelyeket egy TelefonszámTípus néven definiált saját típus szerint kell formázni. A SzuletesiDatumTipus típus korlátozza a születési dátumokat 1900 és 2005 közé. Továbbá komplex típusokat is definiál, mint az AutosiskolaTipus, UgyfelTipus, OktatoTipus, AutoTipus, SzereloTipus, és CserealkatreszTipus, melyek különféle attribútumokat és elemeket tartalmaznak. Az adatbázis integritásának megőrzése érdekében elsődleges (PK) és idegen kulcsok (FK) meghatározására kerül sor, valamint egyediség biztosítása (pl. minden autónak egyedülálló oktatója lehet) az Auto elem esetében. Az XML séma így biztosítja, hogy az adatok szerkezete és kapcsolatai érvényesek és következetesek legyenek.

Az XMLSchema forráskódja:

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

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

    <!-- Egyszerű típusok, saját típusok meghatározása -->
    <xs:element name="nev" type="xs:string" />
    <xs:element name="cim" type="xs:string" />
    <xs:element name="telefon" type="TelefonszámTípus" />
    <xs:element name="kor" type="xs:integer" />
    <xs:element name="szuletesi_datum" type="SzuletesiDatumTipus" />
    <xs:element name="fizetes" type="xs:integer" />
    <xs:element name="vezeteknev" type="xs:string" />
    <xs:element name="keresztnev" type="xs:string" />
    <xs:element name="rendszam" type="xs:string" />
    <xs:element name="tipus" type="xs:string" />
    <xs:element name="marka" type="xs:string" />
    <xs:element name="cserealkatresz" type="xs:string" />


    <xs:simpleType name="TelefonszámTípus">
        <xs:restriction base="xs:string">
            <xs:pattern value="\d{2}-\d{2}-\d{3}-\d{4}" />
        </xs:restriction>
    </xs:simpleType>

    <xs:simpleType name="SzuletesiDatumTipus">
        <xs:restriction base="xs:date">
            <xs:minInclusive value="1900-01-01" />
            <xs:maxInclusive value="2005-12-31" />
        </xs:restriction>
```

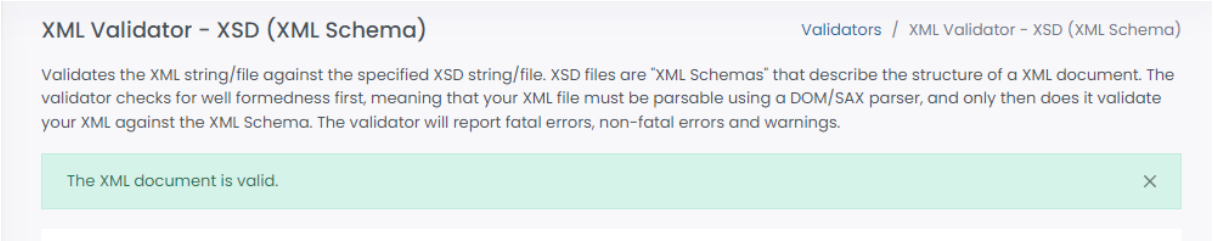
```
    </xs:simpleType>

    <!-- Komplex típusok meghatározása -->
    <xs:complexType name="AutosiskolaTipus">
        <xs:sequence>
            <xs:element name="nev" type="xs:string" />
            <xs:element name="cim" type="xs:string" />
            <xs:element ref="telefon" maxOccurs="unbounded" />
        </xs:sequence>
        <xs:attribute name="ai_id" type="xs:integer" />
    </xs:complexType>

    <xs:complexType name="UgyfelTipus">
        <xs:sequence>
            <xs:element name="nev">
                <xs:complexType>
                    <xs:sequence>
                        <xs:element ref="vezeteknev" />
                        <xs:element ref="keresztnev" />
                    </xs:sequence>
                </xs:complexType>
            </xs:element>
            <xs:element ref="telefon" maxOccurs="unbounded" />
            <xs:element ref="kor" minOccurs="1" />
            <xs:element ref="szuletesi_datum" />
        </xs:sequence>
        <xs:attribute name="u_id" type="xs:integer" />
        <xs:attribute name="ai_id" type="xs:integer" />
        <xs:attribute name="o_id" type="xs:integer" />
    </xs:complexType>

    <xs:complexType name="OktatoTipus">
        <xs:sequence>
            <xs:element ref="nev" />
            <xs:element ref="fizetes" />
            <xs:element ref="telefon" maxOccurs="unbounded" />
        </xs:sequence>
        <xs:attribute name="o_id" type="xs:integer" />
        <xs:attribute name="ai_id" type="xs:integer" />
    </xs:complexType>

    <xs:complexType name="AutoTipus">
        <xs:sequence>
            <xs:element ref="rendszam" />
            <xs:element ref="tipus" />
            <xs:element ref="marka" />
        </xs:sequence>
        <xs:attribute name="au_id" type="xs:integer" />
        <xs:attribute name="o_id" type="xs:integer" />
```

```
    </xs:complexType>

    <xs:complexType name="SzereloTipus">
        <xs:sequence>
            <xs:element ref="nev" />
            <xs:element ref="fizetes" />
            <xs:element ref="telefon" maxOccurs="unbounded" />
        </xs:sequence>
        <xs:attribute name="sz_id" type="xs:integer" />
        <xs:attribute name="au_id" type="xs:integer" />
    </xs:complexType>

    <xs:complexType name="CserealkatreszTipus">
        <xs:sequence>
            <xs:element ref="cserealkatresz" maxOccurs="unbounded" />
        </xs:sequence>
        <xs:attribute name="au_id" type="xs:integer" />
        <xs:attribute name="sz_id" type="xs:integer" />
    </xs:complexType>

    <!-- Gyökér elem meghatározása -->
    <xs:element name="U3ROFS_Autosiskolak">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="Autosiskola" type="AutosiskolaTipus"
maxOccurs="unbounded" />
                <xs:element name="Ugyfel" type="UgyfelTipus"
maxOccurs="unbounded" />
                <xs:element name="Oktato" type="OktatoTipus"
maxOccurs="unbounded" />
                <xs:element name="Auto" type="AutoTipus" maxOccurs="unbounded"
/>
                <xs:element name="Szerelo" type="SzereloTipus"
maxOccurs="unbounded" />
                <xs:element name="cserealkatreszek" type="CserealkatreszTipus"
maxOccurs="unbounded" />
            </xs:sequence>
        </xs:complexType>

        <!--Elsődleges
        kulcsok-->
        <xs:key name="AutosiskolaPK">
            <xs:selector xpath="Autosiskola" />
            <xs:field xpath="@ai_id" />
        </xs:key>

        <xs:key name="UgyfelPK">
            <xs:selector xpath="Ugyfel" />
            <xs:field xpath="@u_id" />
```

```
        </xs:key>

        <xs:key name="OktatoPK">
            <xs:selector xpath="Oktato" />
            <xs:field xpath="@o_id" />
        </xs:key>

        <xs:key name="AutoPK">
            <xs:selector xpath="Auto" />
            <xs:field xpath="@au_id" />
        </xs:key>

        <xs:key name="SzereloPK">
            <xs:selector xpath="Szerelo" />
            <xs:field xpath="@sz_id" />
        </xs:key>

        <!-- Idegen kulcsok -->
        <xs:keyref name="UgyfelAIFK" refer="AutosiskolaPK">
            <xs:selector xpath="Ugyfel" />
            <xs:field xpath="@ai_id" />
        </xs:keyref>

        <xs:keyref name="UgyfelOFK" refer="OktatoPK">
            <xs:selector xpath="Ugyfel" />
            <xs:field xpath="@o_id" />
        </xs:keyref>

        <xs:keyref name="OktatoFK" refer="AutosiskolaPK">
            <xs:selector xpath="Oktato" />
            <xs:field xpath="@ai_id" />
        </xs:keyref>

        <xs:keyref name="AutoFK" refer="OktatoPK">
            <xs:selector xpath="Auto" />
            <xs:field xpath="@o_id" />
        </xs:keyref>

        <xs:keyref name="SzereloFK" refer="AutoPK">
            <xs:selector xpath="Szerelo" />
            <xs:field xpath="@au_id" />
        </xs:keyref>

        <xs:keyref name="CserealkatreszAUFK" refer="AutoPK">
            <xs:selector xpath="Cserealkatresz" />
            <xs:field xpath="@au_id" />
        </xs:keyref>

        <xs:keyref name="CserealkatreszSFK" refer="SzereloPK">
```

```
            <xs:selector xpath="Cserealkatresz" />
            <xs:field xpath="@sz_id" />
        </xs:keyref>

        <!-- 1:1 -->
        <xs:unique name="Auto">
            <xs:selector xpath="Auto" />
            <xs:field xpath="@o_id" />
        </xs:unique>
    </xs:element>
</xs:schema>
```

Validáció sikeressége:



XML file beolvasása

Az XML file beolvasására DomReadU3ROFS osztályt hoztam létre amit a hu.domparse.u3rofs package-ben tároltam, valamint létrehoztam egy statikus osztály metódust ReadXMLDocument néven amit a Main osztályban hívtam meg, ennek a metódusnak egy string bemeneti paramétere van ami az xml fájl elérési útvonalát adja meg. Ez a függvény egy element listát ad vissza. A további módosításokhoz és lekérdezésekhez, ez az element listát használtam fel.

ReadXMLDocument függvény forráskódja:

```
public static ArrayList<Element> ReadXMLDocument(String
filePath){
    try {
        // Egy új File objektumot hozunk létre, ami a megadott
útvonalon lévő fájlra mutat.
        File inputFile = new File(filePath);

        // Új DocumentBuilderFactory példány létrehozása, ami
XML fájlok feldolgozására szolgál.
        DocumentBuilderFactory dbFactory =
DocumentBuilderFactory.newInstance();

        // DocumentBuilder példányt hozunk létre a Factory
segítségével, amit aztán használhatunk az XML fájlok
elemzésére.
        DocumentBuilder dBuilder =
dbFactory.newDocumentBuilder();
```

```
        // Az XML dokumentumot beolvassuk és egy Document
objektumba töltjük, ami lehetővé teszi a struktúrájának
kezelését.
        Document doc = dBuilder.parse(inputFile);

        // Az XML dokumentum normalizálása, ami többek között
azt jelenti, hogy eltávolítjuk a felesleges whitespace
karaktereket,
        // és egyéb normalizálási lépéseket hajtunk végre,
hogy egységes legyen a dokumentum szerkezete.
        doc.getDocumentElement().normalize();

        // Kiírjuk a konzolra az XML dokumentum gyökéreleménének
nevét.
        System.out.println("Gyökérelem: " +
doc.getDocumentElement().getNodeName());

        // Egy NodeList objektumot hozunk létre, amely az
összes "U3ROFS_Autosiskolak" tag nevű elemet tartalmazza az
XML dokumentumból.
        NodeList nList =
doc.getElementsByTagName("U3ROFS_Autosiskolak");

        // Egy új Element típusú ArrayListet hozunk létre, ami
tárolni fogja az elemeket.
        ArrayList<Element> elements = new
ArrayList<Element>();

        // Végigiterálunk a NodeList elemein.
        for (int temp = 0; temp < nList.getLength(); temp++) {
            // Kivesszük a NodeList-ből az aktuális Node
elemet.
            Node nNode = nList.item(temp);

            // Ellenőrizzük, hogy az aktuális Node elem tipusa
ELEMENT_NODE-e, ami azt jelenti, hogy egy elemről van szó.
            if (nNode.getNodeType() == Node.ELEMENT_NODE) {
                // A Node elemet Elementté konvertáljuk, hogy
hozzáférhessünk az elem specifikus metódusaihoz.
                Element eElement = (Element) nNode;

                // Hozzáadjuk az Elementet az elemeket
tartalmazó listához.
                elements.add(eElement);
            }
        }
        // Visszatérünk az elemek listájával.
        return elements;
    } catch (Exception e) {
        e.printStackTrace();
```

```
        return null;
    }
}
```

DOM adatmódosítás

Az adat módosításhoz a következő segéd metódusokat használtam:

- getElementsByTagName: ezzel kértem le a különböző elemeket egy listába amelyeket módosítottam.
- setTextContext: ezzel módosítottam az elementhez tartozó szöveget.
- setAttribute: ezzel módosítottam az attribútumokat.

Ennek létrehoztam egy osztályt DomQueryU3ROFS néven, valamint egy metódust ModifyElement néven. Ez végig iterál az elements-eken és az összes egyedre lefuttatom a ModifyPrescribedElements-et ami módosítja a tulajdonságokat.

DomModifyU3ROFS osztály ModifyPrescribedElements függvény forráskódja:

```
private static void ModifyPrescribedElements(Element element)
{
    //Megváltoztatom az autosiskolákból az első elem nevét
    NodeList autosiskolaList =
element.getElementsByTagName("Autosiskola");
    Element autosiskola = (Element) autosiskolaList.item(0);

autosiskola.getElementsByTagName("nev").item(0).setTextContent
("Fast Car");
    //Megváltoztatom az atributumot is
    autosiskola.setAttribute("ai_id", "4");

    // Például az első Ugyfel nevét változtatja meg
    NodeList ugyfelList =
element.getElementsByTagName("Ugyfel");
    Element ugyfel = (Element) ugyfelList.item(0);

ugyfel.getElementsByTagName("vezeteknev").item(0).setTextConte
nt("Kis");

ugyfel.getElementsByTagName("keresztnev").item(0).setTextConte
nt("Gábor");
    ugyfel.setAttribute("ai_id", "4");

    // Az első Oktato fizetesét változtatja meg
    NodeList oktatoList =
element.getElementsByTagName("Oktato");
    Element oktato = (Element) oktatoList.item(0);
```

```
oktato.getElementsByTagName("fizetes").item(0).setTextContent(
"350000");
    oktato.setAttribute("ai_id","4");

    // Az első Auto markáját változtatja meg
    NodeList autoList = element.getElementsByTagName("Auto");
    Element auto = (Element) autoList.item(0);

auto.getElementsByTagName("marka").item(0).setTextContent("Che
vrolet");

    // Az első Szerelo nevét változtatja meg
    NodeList szereloList =
element.getElementsByTagName("Szerelo");
    Element szerelo = (Element) szereloList.item(0);

szerelo.getElementsByTagName("nev").item(0).setTextContent("Ko
vács Béla");

    // Az első cserealkatreszek cserealkatresz elemének
tartalmát változtatja meg
    NodeList cserealkatreszekList =
element.getElementsByTagName("cserealkatreszek");
    Element cserealkatreszek = (Element)
cserealkatreszekList.item(0);

cserealkatreszek.getElementsByTagName("cserealkatresz").item(0
).setTextContent("kuplung");

    //Kiírom egy fájlba és a consolra a módosított elemeket
    DomWriteU3ROFS.WriteXMLDocumentToConsole(element);

DomWriteU3ROFS.WriteXMLDocumentToFile(element,"./src/XML_U3ROF
S1.xml");
}
```

DOM adat lekérdezés

Továbbra is a beolvasott elemeket használtam. Erre egy DomQueryU3ROFS osztályt hoztam létre. Összesen 6 lekérdezést készítettem.

- Összes autósiskola
- Összes Ugyfel adatainak kiíratása, akik egy bizonyos Autosiskola-hoz tartoznak
- Azoknak az Oktato-knak a neve és fizetése, akik bizonyos Autosiskola-ban tanítanak
- Az Auto elemek rendszam, tipus, és marka adatainak kiíratása
- Szerelők és az általuk szerelt autók, valamint a cserélt alkatrészek

QueryPrescribedDetails metódus forráskódja:

```
private static void QueryPrescribedDetails(Element element) {
    // Új szakasz kezdete a konzolon
    System.out.println();
    // Kiírja, hogy "Összes autósiskola:"
    System.out.println("Összes autósiskola:");
    // Lekéri az összes "Autosiskola" elemet az XML-ből
    NodeList autosiskolaList =
element.getElementsByTagName("Autosiskola");
    // Végigmegy az összes autósiskola elemen
    for (int i = 0; i < autosiskolaList.getLength(); i++) {
        Node node = autosiskolaList.item(i);
        // Ellenőrzi, hogy az elem tényleg elem típusú-e (nem
szöveg vagy komment)
        if (node.getNodeType() == Node.ELEMENT_NODE) {
            Element autosiskola = (Element) node;
            // Kiírja az autósiskola ID-ját és nevét
            System.out.println("Autosiskola ID: " +
autosiskola.getAttribute("ai_id"));
            System.out.println("Név: " +
autosiskola.getElementsByTagName("nev").item(0).getTextContent
());
        }
    }

    // Új szakasz kezdete a konzolon
    System.out.println();
    // Kiírja, hogy "Összes Ugyfel adatainak kiíratása, akik
egy bizonyos Autosiskola-hoz tartoznak"
    System.out.println("Összes Ugyfel adatainak kiíratása,
akik egy bizonyos Autosiskola-hoz tartoznak");
    // Lekéri az összes "Ugyfel" elemet az XML-ből
    NodeList ugyfelList =
element.getElementsByTagName("Ugyfel");
    // Végigmegy az összes ügyfél elemen
    for (int i = 0; i < ugyfelList.getLength(); i++) {
        Node node = ugyfelList.item(i);
        // Ellenőrzi, hogy az elem tényleg elem típusú-e és az
"ai_id" attribútum értéke "1"
        if (node.getNodeType() == Node.ELEMENT_NODE &&
"1".equals(((Element) node).getAttribute("ai_id"))) {
            Element ugyfel = (Element) node;
            // Kiírja az ügyfél nevét
            System.out.println("Ügyfél név: " +
ugyfel.getElementsByTagName("nev").item(0).getTextContent());
        }
    }

    // Új szakasz kezdete a konzolon
    System.out.println();
```

```
    // Kiírja, hogy "Azoknak az Oktato-knak a neve és
fizetése, akik bizonyos Autosiskola-ban tanítanak"
    System.out.println("Azoknak az Oktato-knak a neve és
fizetése, akik bizonyos Autosiskola-ban tanítanak");
    // Lekéri az összes "Oktato" elemet az XML-ből
    NodeList oktatoList =
element.getElementsByTagName("Oktato");
    // Végigmegy az összes oktató elemen
    for (int i = 0; i < oktatoList.getLength(); i++) {
        Node node = oktatoList.item(i);
        // Ellenőrzi, hogy az elem tényleg elem típusú-e és az
"ai_id" attribútum értéke "1"
        if (node.getNodeType() == Node.ELEMENT_NODE) {
            Element oktato = (Element) node;
            if ("1".equals(oktato.getAttribute("ai_id"))) {
                // Kiírja az oktató nevét és fizetését
                System.out.println("Oktató neve: " +
oktato.getElementsByTagName("nev").item(0).getTextContent());
                System.out.println("Fizetése: " +
oktato.getElementsByTagName("fizetes").item(0).getTextContent(
));
            }
        }
    }

    // Új szakasz kezdete a konzolon
    System.out.println();
    // Kiírja, hogy "Az Auto elemek rendszam, tipus, és marka
adatainak kiíratása"
    System.out.println("Az Auto elemek rendszam, tipus, és
marka adatainak kiíratása");
    // Lekéri az összes "Auto" elemet az XML-ből
    NodeList autoList = element.getElementsByTagName("Auto");
    // Végigmegy az összes auto elemen
    for (int i = 0; i < autoList.getLength(); i++) {
        Node node = autoList.item(i);
        // Ellenőrzi, hogy az elem tényleg elem típusú-e
        if (node.getNodeType() == Node.ELEMENT_NODE) {
            Element auto = (Element) node;
            // Kiírja az auto rendszámát, típusát, és márkáját
            System.out.println("Rendszám: " +
auto.getElementsByTagName("rendszam").item(0).getTextContent()
);
            System.out.println("Típus: " +
auto.getElementsByTagName("tipus").item(0).getTextContent());
            System.out.println("Márka: " +
auto.getElementsByTagName("marka").item(0).getTextContent());
        }
    }
    // Új szakasz kezdete a konzolon
    System.out.println();
```

```
    // Kiírja, hogy "Szerelők és az általuk szerelt autók,
valamint a cserélt alkatrészek:"
    System.out.println("Szerelők és az általuk szerelt autók,
valamint a cserélt alkatrészek:");

    // Lekéri az összes "Szerelo" elemet az XML-ből
    NodeList szereloList =
element.getElementsByTagName("Szerelo");
    for (int i = 0; i < szereloList.getLength(); i++) {
        Node szereloNode = szereloList.item(i);
        // Ellenőrzi, hogy az elem tényleg elem típusú-e
        if (szereloNode.getNodeType() == Node.ELEMENT_NODE) {
            Element szerelo = (Element) szereloNode;
            // Kiírja a szerelő nevét és az autó ID-ját, amit
szerelt
            String szereloNev =
szerelo.getElementsByTagName("nev").item(0).getTextContent();
            String autoId = szerelo.getAttribute("au_id");

            System.out.println("Szerelő neve: " + szereloNev);
            System.out.println("Szerelt Auto ID: " + autoId);

            // Lekéri az összes "Auto" elemet az XML-ből
            for (int j = 0; j < autoList.getLength(); j++) {
                Node autoNode = autoList.item(j);
                // Ellenőrzi, hogy az elem tényleg elem
típusú-e és az auto ID egyezik-e
                if (autoNode.getNodeType() ==
Node.ELEMENT_NODE && autoId.equals(((Element)
autoNode).getAttribute("au_id"))) {
                    Element auto = (Element) autoNode;
                    // Kiírja az autó rendszámát, típusát és
márkáját
                    System.out.println("Auto Rendszám: " +
auto.getElementsByTagName("rendszam").item(0).getTextContent()
);
                    System.out.println("Auto Típus: " +
auto.getElementsByTagName("tipus").item(0).getTextContent());
                    System.out.println("Auto Márka: " +
auto.getElementsByTagName("marka").item(0).getTextContent());
                }
            }

            // Lekéri az összes "cserealkatreszek" elemet az
XML-ből
            NodeList cserealkatreszekList =
element.getElementsByTagName("cserealkatreszek");
            for (int k = 0; k <
cserealkatreszekList.getLength(); k++) {
                Node cserealkatreszekNode =
cserealkatreszekList.item(k);
```

```
                // Ellenőrzi, hogy az elem tényleg elem
típusú-e és az auto ID egyezik-e
                if (cserealkatreszekNode.getNodeType() ==
Node.ELEMENT_NODE && autoId.equals(((Element)
cserealkatreszekNode).getAttribute("au_id"))) {
                    Element cserealkatreszek = (Element)
cserealkatreszekNode;
                    NodeList alkatresek =
cserealkatreszek.getElementsByTagName("cserealkatresz");
                    System.out.println("Cserélt
alkatrészek:");
                    // Végigmegy a cserélt alkatrészek
listáján
                    for (int l = 0; l <
alkatresek.getLength(); l++) {
                        Node alkatresz = alkatresek.item(l);
                        // Ellenőrzi, hogy az elem tényleg
elem típusú-e
                        if (alkatresz.getNodeType() ==
Node.ELEMENT_NODE) {
                            // Kiírja az alkatrész nevét
                            System.out.println(" - " +
alkatresz.getTextContent());
                        }
                    }
                }
            }
            // Új szakasz a konzolon
            System.out.println();
        }
    }
}
```

DOM adatírás

A konzolra írást és a fájlba írást a következő módon valósítottam meg.
DomWriteU3ROFS forráskódja:

```
public class DomWriteU3ROFS {
    public static void
WriteElementsToConsoleAndFile(ArrayList<Element> elements,
String filePath) {
        //megnyitjuk a file írást
        try (FileWriter writer = new FileWriter(filePath)) {
            for(var element : elements)
            {
                writeToConsoleAndFile(writer, element, "");
            }
        }
```

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

    public static void writeToConsoleAndFile(FileWriter
writer, Node node, String indent) {
        try {
            //ellenőrizzük hogy Element-e
            if (node.getNodeType() == Node.ELEMENT_NODE) {
                Element element = (Element) node;
                //Elkezdjük írni a start tag-et
                String startTag = indent  +
element.getTagName();

                // Attribútumok formázása
                String attributesText =
formatAttributes(element.getAttributes());
                if (!attributesText.isEmpty()) {
                    startTag += " " + attributesText;
                }

                startTag +=  " start";

                System.out.println(startTag);
                writer.write(startTag + "\n");
                indent += "    ";

                NodeList children = element.getChildNodes();
                for (int i = 0; i < children.getLength(); i++)
{
                    writeToConsoleAndFile(writer,
children.item(i), indent);
                }

                indent = indent.substring(0, indent.length() -
4);
                String endTag = indent  + element.getTagName()
+ " end";
                System.out.println(endTag);
                writer.write(endTag + "\n");
            } else if (node.getNodeType() == Node.TEXT_NODE) {
                String nodeValue =
node.getTextContent().trim();
                if (!nodeValue.isEmpty()) {
                    System.out.println(indent + nodeValue);
                    writer.write(indent + nodeValue + "\n");
                }
            }
        } catch (IOException e) {
```

```
            e.printStackTrace();
        }
    }

    private static String formatAttributes(NamedNodeMap
attributes) {
        //Atributumok formázása
        StringBuilder attributesBuilder = new StringBuilder();
        if (attributes != null && attributes.getLength() > 0)
{
            attributesBuilder.append("{");
            for (int i = 0; i < attributes.getLength(); i++) {
                Attr attribute = (Attr) attributes.item(i);

attributesBuilder.append(attribute.getName()).append("=").appe
nd(attribute.getValue());
                if (i < attributes.getLength() - 1) {
                    attributesBuilder.append(",");
                }
            }
            attributesBuilder.append("}");
        }
        return attributesBuilder.toString();
    }
```