JEGYZŐKÖNYV

Adatkezelés XML környezetben

Féléves feladat

Autóbusz Pályaudvar adatbázis

Készítette: **Honti Dániel**
Neptunkód: **HR6121**
Dátum: **2023. 11. 27.**

Tartalomjegyzék

1. A feladat leírása

Az általam kidolgozott feladat egy autóbusz állomás adatbázisát mutatja be. Az ER modellben megtalálhatóak az autóbuszok általános adatai, a sofőrök, karbantartók és felügyelők adatai, valamint a karbantartók által használt eszközök tulajdonságai és az autóbuszok által bejárt útvonalak adatai.

Az ER modellhez tartozik egy XDM modell, amely megadja a működéshez szükséges XML keretet.

Az XML-be fel kell vinni az autóbuszok, sofőrök, karbantartók, karbantartó eszközök, felügyelők, valamint az útvonalak adatait.

Az XML-hez továbbá tartozik egy XMLSchema, amely pontosan megadja hogy milyen típusú adatok vihetők fel az XML-be és hogy ezekből mennyit vihetünk fel.

Ezek mellé készült egy DOM program, amely módosításokkal és lekérdezésekkel foglalkozik, épp azzal amelyikre az adott helyzetben szükségünk van.

Az adatbázis egyedei:

- Autóbusz
- Autóbuszok által bejárt út
- Karbantartó
- Karbantartó eszköz
- Sofőr
- Felügyelő

Autóbusz:

- BID – Az autóbusz 3 számjegyű azonosítója
- Rendszám – Az autóbusz rendszámát adja meg
- Típus – Az autóbusz típusát adja meg
- Max_hely – Az autóbuszon található max. helyeket adja meg

Útvonal:

- UID – Az útvonal 3 számjegyű azonosítója
- Kezdeti állomás – Az útvonal kezdeti állomását adja meg
- Végállomás – Az útvonal végállomását adja meg
- km – Az útvonal hosszát adja meg

Karbantartó:

- KID – A karbantartó 3 számjegyű azonosítója
- Javítói szám – A karbantartó javítói számát adja meg
- Név – A karbantartó nevét adja meg
- BID – Az autóbusz ID-jét adja meg (idegen kulcs)

Karbantartó felszerelés:

- EID – Az felszerelés 3 számjegyű azonosítóját adja meg
- Gyártó – A felszerelés gyártóját adja meg.
- Használt évek – A felszerelés használatban töltött éveit adja meg
- Termék szám – A felszerelés termékszámát adja meg
- KID – A Karbantartó ID-t adja meg (Idegen kulcs)

Sofőr:

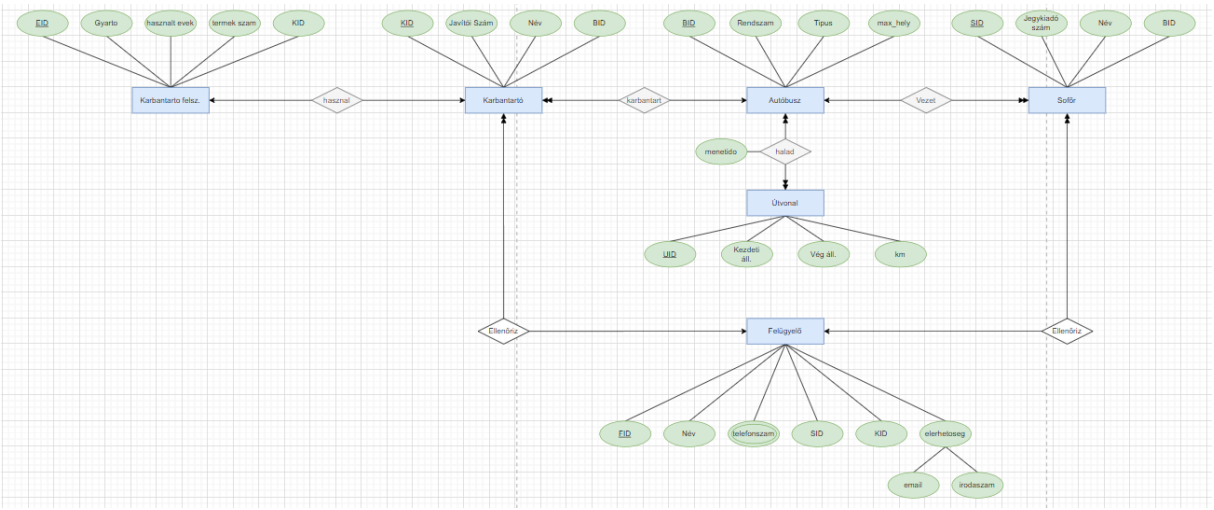
- SID – A sofőr 3 számjegyű azonosítóját adja meg
- Jegykiadó szám – A sofőr jegykiadó számát adja meg
- Név – A sofőr nevét adja meg
- BID – Az autóbusz ID-t adja meg (idegen kulcs)

Felügyelő:

- FID – A felügyelő 3 számjegyű ID-jét adja meg
- Név – A felügyelő nevét adja meg
- Telefonszám – A felügyelő telefonszám(ait) adja meg
- SID – A sofőr ID-t adja meg (Idegen kulcs)
- KID – A karbantartó ID-t adja meg (Idegen kulcs)
- Elérhetőség
    - email – A felügyelő email-ját adja meg
    - irodaszám A felügyelő irodaszámát adja meg

1a) Az adatbázis ER modell tervezése

Az ER modell elkészítésének szempontjai voltak az átláthatóság és az esztétika, hogy könnyen érthető legyen az elkészített modell. Az egyedek, kapcsolatok és a tulajdonságokat megpróbáltam egyvonalba elhelyezni, követelményeknek megfelelően szerkeszteni, egyedeket és kapcsolatokat létrehozni.



Egyedek:

Autóbusz:

Ez az egyed tartalmazza az üzemeltetett autóbuszok adatait. A BID (Busz ID) az azonosítója, elsődleges kulcs. Emellett rendelkezik rendszámmal, gyártóval, ezek szöveges típusok, és egy max férőhellyel, ami integer típusú.

Útvonal:

Ez az egyed tartalmazza az autóbuszok által bejárt útvonalakat. A UID (Útvonal ID) az azonosítója, elsődleges kulcs. Rendelkezik kezdeti és végállomás szöveges adattal, és egy km számmal, ami a két állomás közötti távolság.

Sofőr:

Ez az egyed az autóbuszvezető adatait tartalmazza. Rendelkezik egy SID (Sofőr ID) azonosítóval, ami az elsődleges kulcs. Emellett rendelkezik névvel, ami szöveges adat, jegykiadó számmal, ami integer típusú. A BID a vezető által irányított busz egyedi azonosítója, foreign key.

Karbantartó:

Ez az egyed az autóbusz karbantartó adatait tartalmazza. Rendelkezik egy KID (Karbantartó ID) azonosítóval, ami az elsődleges kulcs. Rendelkezik névvel, ami szöveges adat, javítási azonosítóval, ami integer típusú. A BID a karbantartó által ellenőrzött autóbusz azonosítóját tartalmazza, foreign key.

Karbantartó-felszerelés:
Ez az egyed a karbantartókra kiosztott eszközök adatait tartalmazza. Rendelkezik egy EID (Eszköz ID) azonosítóval, ami az elsődleges kulcs. Rendelkezik Gyártóval, ami szöveges adat, használt évekkel, termék számmal, amik integer típusúak. A KID az eszközhöz tartozó Karbantartó azonosítóját tartalmazza, foreign key.

Felügyelő:
Ez az egyed a karbantartók és sofőröket felügyelő egyed adatait tartalmazza. Rendelkezik FID (Felügyelő ID) azonosítóval, ami elsődleges kulcs. Rendelkezik névvel, ami szöveges adat, és elérhetőséggel, ami összetett tulajdonság. Ezen belül van egy irodaszám integer típusú, és egy email cím szöveges típusú adattal. Van egy Telefonszám tulajdonsága, ami többértékű, mivel egy Felügyelőhöz több Telefonszám is tartozhat. A SID és KID a felügyelő által megfigyelt sofőr és karbantartó azonosítóját tartalmazza, foreign key-ek.

Kapcsolatok:

Autóbusz-Útvonal:
Az Autóbusz és az Útvonal között több a többhöz kapcsolat áll, mivel egy útvonalon több autóbusz közlekedhet, valamint egy autóbusz több útvonalon is járhat. Ennek van egy menetidő tulajdonsága is.

Autóbusz-Sofőr:
Az Autóbusz és a Sofőr között egy a többhöz kapcsolat áll fent, mivel az autóbusz vezetője egyetlen járművet vezet, de egy autóbusz rendelkezhet több sofőrrel is.

Autóbusz-Karbantartó:
Az Autóbusz és a Karbantartó között egy a többhöz kapcsolat áll fent, mivel a karbantartó egy járművet javít, de egy autóbusz rendelkezhet több karbantartóval is.

Karbantartó-Felszerelés:
A Karbantartó és a Felszerelés között egy az egyhez kapcsolat áll fent, mivel egy Karbantartó csak egy felszereléssel rendelkezik.

Felügyelő-Karbantartó:
A Felügyelő és a Karbantartó között egy a többhöz kapcsolat áll fent, mivel egy Felügyelő több Karbantartót ellenőrizhet, azonban egy Karbantartónak csak egy Felügyelője van.

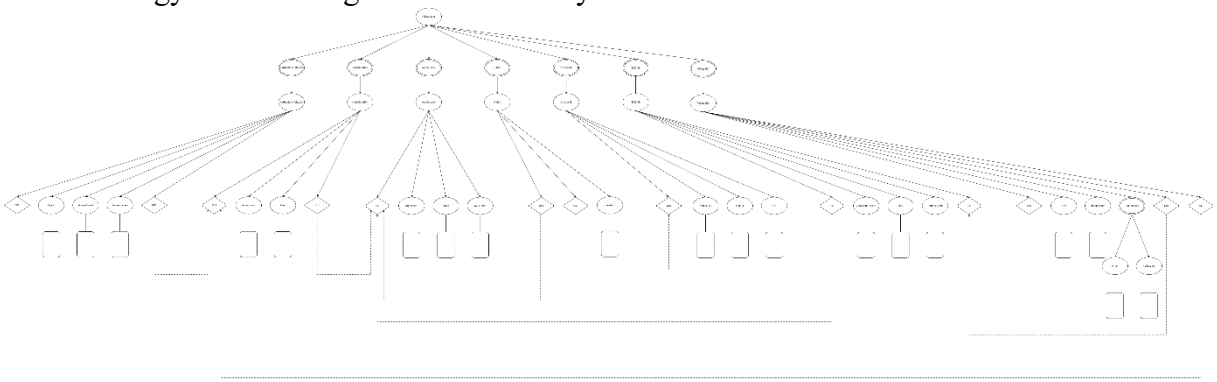
Felügyelő-Sofőr:
A Felügyelő és a Sofőr között egy a többhöz kapcsolat áll fent, mivel egy Felügyelő több Sofőrt ellenőrizhet, azonban egy Sofőrnek csak egy Felügyelője van.

1 b.) Az adatbázis konvertálása XDM modellre

Mivel az XML dokumentum készítése során egy egyedből több példányt is létre szeretnék hozni. Ez okból az összes egyedet egy szülő egyedbe vezetni. A szülő egyedek elnevezése az alap egyed neve csak többesszámban ezzel utalva hogy a szülő egyedben több egyed is van.

A feladat készítése során szükség volt egy új osztály létrehozására ez a Halad, mivel az Autóbusz és az Útvonal közt több-több kapcsolat van. Ez az osztály továbbá tartalmazza a menetidőt is.

Az összes egyedet összefogó root elem a Pályaudvar.



1.c) Az XDM model alapján XML dokumentum készítése

Az XDM modell segítségével létrehoztam az XML kódot.

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

<Palyaudvar xsi:noNameSpaceSchemaLocation="./XMLSchemaHR6121.xsd"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">

    <!--Autóbuszok pédányosítása-->

    <Autobuszok>

        <Autobusz AutobuszId="001">
            <Rendszam> AAA-111 </Rendszam>
            <Tipus> Ikarus </Tipus>
            <max_hely> 40 </max_hely>
        </Autobusz>

        <Autobusz AutobuszId="002">
            <Rendszam> BBB-222 </Rendszam>
            <Tipus> MAN </Tipus>
            <max_hely> 45 </max_hely>
        </Autobusz>

        <Autobusz AutobuszId="003">
            <Rendszam> CCC-333 </Rendszam>
            <Tipus> Volvo </Tipus>
            <max_hely> 50 </max_hely>
        </Autobusz>
```

<Autobusz AutobuszId="004">
<Rendszam> DDD-444 </Rendszam>
<Tipus> Ikarus </Tipus>
<max_hely> 40 </max_hely>
</Autobusz>

<Autobusz AutobuszId="005">
<Rendszam> EEE-555 </Rendszam>
<Tipus> Ikarus </Tipus>
<max_hely> 40 </max_hely>
</Autobusz>

<Autobusz AutobuszId="006">
<Rendszam> FFF-666 </Rendszam>
<Tipus> Ikarus </Tipus>
<max_hely> 40 </max_hely>
</Autobusz>

<Autobusz AutobuszId="007">
<Rendszam> GGG-777 </Rendszam>
<Tipus> Volvo </Tipus>
<max_hely> 50 </max_hely>
</Autobusz>

<Autobusz AutobuszId="008">
<Rendszam> HHH-888 </Rendszam>
<Tipus> Ikarus </Tipus>
<max_hely> 40 </max_hely>
</Autobusz>

```
        <Autobusz AutobuszId="009">
            <Rendszam> III-999 </Rendszam>
            <Tipus> MAN </Tipus>
            <max_hely> 45 </max_hely>
        </Autobusz>

</Autobuszok>

<Soforok>

        <Sofor SoforId="101">
            <Jegykiadoszam> A54321 </Jegykiadoszam>
            <Nev> Major Daniel </Nev>
            <AutobuszId_FK> 009 </AutobuszId_FK>
        </Sofor>

        <Sofor SoforId="102">
            <Jegykiadoszam> B12345 </Jegykiadoszam>
            <Nev> Tancos Judit </Nev>
            <AutobuszId_FK> 005 </AutobuszId_FK>
        </Sofor>

        <Sofor SoforId="103">
            <Jegykiadoszam> C98765 </Jegykiadoszam>
            <Nev> Beno Rezso </Nev>
            <AutobuszId_FK> 003 </AutobuszId_FK>
        </Sofor>

        <Sofor SoforId="104">
            <Jegykiadoszam> D56789 </Jegykiadoszam>
```

```
        <Nev> Prezs Peter </Nev>
        <AutobuszId_FK> 008 </AutobuszId_FK>
</Sofor>

<Sofor SoforId="105">
        <Jegykiadoszam> E23456 </Jegykiadoszam>
        <Nev> Kont Fanni </Nev>
        <AutobuszId_FK> 001 </AutobuszId_FK>
</Sofor>

<Sofor SoforId="106">
        <Jegykiadoszam> F65432 </Jegykiadoszam>
        <Nev> Kann András </Nev>
        <AutobuszId_FK> 002 </AutobuszId_FK>
</Sofor>

<Sofor SoforId="107">
        <Jegykiadoszam> G34567 </Jegykiadoszam>
        <Nev> Furko Evelin </Nev>
        <AutobuszId_FK> 003 </AutobuszId_FK>
</Sofor>

<Sofor SoforId="108">
        <Jegykiadoszam> H76543 </Jegykiadoszam>
        <Nev> Labas Lajos </Nev>
        <AutobuszId_FK> 004 </AutobuszId_FK>
</Sofor>

<Sofor SoforId="109">
        <Jegykiadoszam> I45678 </Jegykiadoszam>
```

```
            <Nev> Bence Nandor </Nev>
            <AutobuszId_FK> 006 </AutobuszId_FK>
        </Sofor>

</Soforok>

<Karbantartok>

        <Karbantarto KarbantartoId="201">
            <Javitoszam> A0001 </Javitoszam>
            <Nev> Ant Istvan </Nev>
            <AutobuszId_FK> 004 </AutobuszId_FK>
        </Karbantarto>

        <Karbantarto KarbantartoId="202">
            <Javitoszam> B0002 </Javitoszam>
            <Nev> Várti Ádám </Nev>
            <AutobuszId_FK> 003 </AutobuszId_FK>
        </Karbantarto>

        <Karbantarto KarbantartoId="203">
            <Javitoszam> C0003 </Javitoszam>
            <Nev> Ács Viktória </Nev>
            <AutobuszId_FK> 001 </AutobuszId_FK>
        </Karbantarto>

        <Karbantarto KarbantartoId="204">
            <Javitoszam> D0004 </Javitoszam>
            <Nev> Csik Zoltán </Nev>
            <AutobuszId_FK> 008 </AutobuszId_FK>
```

```
        </Karbantarto>

        <Karbantarto KarbantartoId="205">
            <Javitoszam> E0005 </Javitoszam>
            <Nev> Szabó Imre </Nev>
            <AutobuszId_FK> 009 </AutobuszId_FK>
        </Karbantarto>
</Karbantartok>

<Eszkozok>
        <Eszkoz EszkozId="301">
            <Termekszam> 97 </Termekszam>
            <Gyarto> Makita </Gyarto>
            <Ev> 1 </Ev>
            <KarbantartoId_FK> 201 </KarbantartoId_FK>
        </Eszkoz>

        <Eszkoz EszkozId="302">
            <Termekszam> 42 </Termekszam>
            <Gyarto> Bosch </Gyarto>
            <Ev> 3 </Ev>
            <KarbantartoId_FK> 202 </KarbantartoId_FK>
        </Eszkoz>

        <Eszkoz EszkozId="303">
            <Termekszam> 55 </Termekszam>
            <Gyarto> Bosch </Gyarto>
            <Ev> 5 </Ev>
            <KarbantartoId_FK> 203 </KarbantartoId_FK>
        </Eszkoz>
```

```
        <Eszkoz EszkozId="304">
            <Termekszam> 12 </Termekszam>
            <Gyarto> Makita </Gyarto>
            <Ev> 2 </Ev>
            <KarbantartoId_FK> 204 </KarbantartoId_FK>
        </Eszkoz>

        <Eszkoz EszkozId="305">
            <Termekszam> 08 </Termekszam>
            <Gyarto> Bosch </Gyarto>
            <Ev> 7 </Ev>
            <KarbantartoId_FK> 205 </KarbantartoId_FK>
        </Eszkoz>

</Eszkozok>

<Utvonalak>
        <Utvonal UtvonailId="401">
            <KezdAll> Buza Ter </KezdAll>
            <VegAll> Tapolca </VegAll>
            <Km> 8 </Km>
        </Utvonal>

        <Utvonal UtvonalId="402">
            <KezdAll> Buza Ter </KezdAll>
            <VegAll> Szirma </VegAll>
            <km> 9 </km>
        </Utvonal>
```

```
        <Utvonal UtvonalId="403">
            <KezdAll> Felso-Majlath </KezdAll>
            <VegAll> Lillafured </VegAll>
            <km> 6 </km>
        </Utvonal>

        <Utvonal UtvonalId="404">
            <KezdAll> Ujgyori Foter </KezdAll>
            <VegAll> Pereces </VegAll>
            <km> 7 </km>
        </Utvonal>

        <Utvonal UtvonalId="405">
            <KezdAll> Avas Kilato </KezdAll>
            <VegAll> Centrum </VegAll>
            <km> 5 </km>
        </Utvonal>

</Utvonalak>

<Haladok>
        <Halad>
            <UtvonalId_FK> 401 </UtvonalId_FK>
            <AutobuszID_FK> 003 </AutobuszID_FK>
            <Menetido> 25 </Menetido>
        </Halad>

        <Halad>
            <UtvonalId_FK> 404 </UtvonalId_FK>
            <AutobuszID_FK> 001 </AutobuszID_FK>
```

<Menetido> 26 </Menetido>
</Halad>

<Halad>
<UtvonalId_FK> 405 </UtvonalId_FK>
<AutobuszID_FK> 007 </AutobuszID_FK>
<Menetido> 19 </Menetido>
</Halad>

<Halad>
<UtvonalId_FK> 402 </UtvonalId_FK>
<AutobuszID_FK> 002 </AutobuszID_FK>
<Menetido> 24 </Menetido>
</Halad>

<Halad>
<UtvonalId_FK> 404 </UtvonalId_FK>
<AutobuszID_FK> 003 </AutobuszID_FK>
<Menetido> 21 </Menetido>
</Halad>

<Halad>
<UtvonalId_FK> 403 </UtvonalId_FK>
<AutobuszID_FK> 009 </AutobuszID_FK>
<Menetido> 20 </Menetido>
</Halad>

<Halad>
<UtvonalId_FK>406 </UtvonalId_FK>
<AutobuszID_FK> 001 </AutobuszID_FK>

```
            <Menetido> 23 </Menetido>
        </Halad>

</Haladok>

<Felugyelok>
        <Felugyelo FelugyeloId="501">
            <Nev> Pataki Csanad </Nev>
            <SoforId_FK> 102 </SoforId_FK>
            <KarbantartoId_FK> 204 </KarbantartoId_FK>
            <Elerhetoseg>
                <Irodaszam> 11 </Irodaszam>
                <Email> pat.csan@gmail.com </Email>
            </Elerhetoseg>
            <Telefonszam> 06 55 584 888 </Telefonszam>
            <Telefonszam> +36 55 563 979 </Telefonszam>
        </Felugyelo>

        <Felugyelo FelugyeloId="502">
            <Nev> Gabor Borbala </Nev>
            <SoforId_FK> 104 </SoforId_FK>
            <KarbantartoId_FK> 202 </KarbantartoId_FK>
            <Elerhetoseg>
                <Irodaszam> 12 </Irodaszam>
                <Email> gab.bor@gmail.com </Email>
            </Elerhetoseg>
            <Telefonszam> 06 55 010 226 </Telefonszam>
            <Telefonszam> +36 55 686 186 </Telefonszam>
        </Felugyelo>
```

<Felugyelo FelugyeloId="503">
<Nev> Szabó Oszkár </Nev>
<SoforId_FK> 103 </SoforId_FK>
<KarbantartoId_FK> 201 </KarbantartoId_FK>
<Elerhetoseg>
<Irodaszam> 13 </Irodaszam>
<Email> szab.osz@gmail.com </Email>
</Elerhetoseg>
<Telefonszam> 06 55 686 186 </Telefonszam>
<Telefonszam> +36 55 010 226 </Telefonszam>
</Felugyelo>

<Felugyelo FelugyeloId="504">
<Nev> Hegyi Borbala </Nev>
<SoforId_FK> 108 </SoforId_FK>
<KarbantartoId_FK> 203 </KarbantartoId_FK>
<Elerhetoseg>
<Irodaszam> 14 </Irodaszam>
<Email> hegy.bor@gmail.com </Email>
</Elerhetoseg>
<Telefonszam> 06 55 563 979 </Telefonszam>
<Telefonszam> +36 55 584 888 </Telefonszam>
</Felugyelo>

<Felugyelo FelugyeloId="505">
<Nev> Nagy Andrea </Nev>
<SoforId_FK> 109 </SoforId_FK>
<KarbantartoId_FK> 205 </KarbantartoId_FK>
<Elerhetoseg>
<Irodaszam> 15 </Irodaszam>

<Email> nagy.and@gmail.com </Email>
</Elerhetoseg>
<Telefonszam> 06 55 923 742 </Telefonszam>
<Telefonszam> +36 55 923 742 </Telefonszam>
</Felugyelo>

</Felugyelok>

</Palyaudvar>

1.d, Az XML dokumentum alapján XMLSchema készítése:

Az XMLSchemaban minden egyedhez saját típust hoztam létre.

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

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

    <xs:element name="Palyaudvar">
        <xs:complexType>
            <xs:sequence>

                <xs:element name="KarbantartoFelsz-ek">
                    <xs:complexType>
                        <xs:sequence>
                            <xs:element name="KarbantartoFelsz"
type="KarbantartoFelszTip" minOccurs="0" maxOccurs="unbounded" />
                        </xs:sequence>
                    </xs:complexType>
                </xs:element>

                <xs:element name="Karbantartok">
                    <xs:complexType>
                        <xs:sequence>
                            <xs:element name="karbantarto" type="KarbantartoTip"
minOccurs="0" maxOccurs="unbounded"/>
                        </xs:sequence>
                    </xs:complexType>
                </xs:element>

                <xs:element name="Autobuszok">
                    <xs:complexType>
                        <xs:sequence>
                            <xs:element name="Autobusz" type="AutobuszTip"
minOccurs="0" maxOccurs="unbounded"/>
                        </xs:sequence>
                    </xs:complexType>
                </xs:element>

                <xs:element name="Soforok">
                    <xs:complexType>
                        <xs:sequence>
                            <xs:element name="Sofor" type="SoforTip"
minOccurs="0" maxOccurs="unbounded"/>
                        </xs:sequence>
                    </xs:complexType>
                </xs:element>

                <xs:element name="Utvonalak">
                    <xs:complexType>
                        <xs:sequence>
                            <xs:element name="Utvonal" type="UtvonalTip"
minOccurs="0" maxOccurs="unbounded"/>
                        </xs:sequence>
                    </xs:complexType>
                </xs:element>

                <xs:element name="Felugyelok">
                    <xs:complexType>
                        <xs:sequence>
                            <xs:element name="Felugyelo" type="FelugyeloTip"
minOccurs="0" maxOccurs="unbounded"/>
```

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

                <xs:element name="Haladok">
                    <xs:complexType>
                        <xs:sequence>
                            <xs:element name="Halad" type="HaladTip"
minOccurs="0" maxOccurs="unbounded"/>
                        </xs:sequence>
                    </xs:complexType>
                </xs:element>
            </xs:sequence>
        </xs:complexType>

        <xs:unique name="KarbantartoFelszId_FK">
            <xs:selector xpath="KarbantartoFelszek/KarbantartoFelsz"/>
            <xs:field xpath="@KarbantartoFelszId"/>
        </xs:unique>

          <xs:keyref name="AutobuszID_FK-Karbantarto" refer="AutobuszID_FK">
               <xs:selector xpath="Karbantartok/Karbantarto/AutobuszId_FK"/>
               <xs:field xpath="."/>
          </xs:keyref>

          <xs:keyref name="AutobuszID_FK-Sofor" refer="AutobuszID_FK">
               <xs:selector xpath="Soforok/Sofor/AutobuszId_FK"/>
               <xs:field xpath="."/>
          </xs:keyref>

          <xs:keyref name="AutobuszID_FK-Halad" refer="AutobuszID_FK">
               <xs:selector xpath="Haladas/Halad/AutobuszId_FK"/>
               <xs:field xpath="."/>
          </xs:keyref>

          <xs:keyref name="KarbantartoID_FK-Felugyelo" refer="KarbantartoID_FK">
               <xs:selector xpath="Felugyelok/Felugyelo/KarbantartoId_FK"/>
               <xs:field xpath="."/>
          </xs:keyref>

          <xs:keyref name="SoforID_FK-Felugyelo" refer="SoforID_FK">
               <xs:selector xpath="Felugyelok/Felugyelo/SoforId_FK"/>
               <xs:field xpath="."/>
          </xs:keyref>
    </xs:element>

    <xs:complexType name="KarbantartoFelszTipus">
        <xs:sequence>
            <xs:element name="Gyarto" type="xs:string"/>
            <xs:element name="HasznaltEvek" type="xs:Integer"/>
            <xs:element name="TermekSzam" type="xs:integer"/>
            <xs:element name="KarbantartoId_FK" type="xs:integer"/>
        </xs:sequence>
        <xs:attribute name="KarbantartoFelszId" use="required">
            <xs:simpleType>
                <xs:restriction base="xs:integer">
                    <xs:pattern value="[0-9][0-9][0-9]" />
                </xs:restriction>
            </xs:simpleType>
        </xs:attribute>
    </xs:complexType>

    <xs:complexType name="KarbantartoTipus">
```

```
    <xs:sequence>
        <xs:element name="Nev" type="xs:string"/>
        <xs:element name="JavitoiSzam" type="xs:integer"/>
        <xs:element name="AutobuszId_FK" type="xs:integer"/>
    </xs:sequence>
    <xs:attribute name="KarbantartoId" use="required">
        <xs:simpleType>
            <xs:restriction base="xs:integer">
                <xs:pattern value="[0-9][0-9][0-9]" />
            </xs:restriction>
        </xs:simpleType>
    </xs:attribute>
</xs:complexType>

<xs:complexType name="AutobuszTipus">
    <xs:sequence>
        <xs:element name="Max_hely" type="xs:integer"/>
        <xs:element name="Tipus" type="xs:string"/>
        <xs:element name="Rendszam" type="xs:string"/>
    </xs:sequence>
    <xs:attribute name="AutobuszId" use="required">
        <xs:simpleType>
            <xs:restriction base="xs:integer">
                <xs:pattern value="[0-9][0-9][0-9]" />
            </xs:restriction>
        </xs:simpleType>
    </xs:attribute>
</xs:complexType>


<xs:complexType name="SoforTipus">
    <xs:sequence>
        <xs:element name="JegykiadoSzam " type="xs:integer"/>
        <xs:element name="Nev" type="xs:string"/>
        <xs:element name="TermekSzam" type="xs:string"/>
        <xs:element name="AutobuszId_FK" type="xs:integer"/>
    </xs:sequence>
    <xs:attribute name="SoforId" use="required">
        <xs:simpleType>
            <xs:restriction base="xs:integer">
                <xs:pattern value="[0-9][0-9][0-9]" />
            </xs:restriction>
        </xs:simpleType>
    </xs:attribute>
</xs:complexType>

<xs:complexType name="UtvonalTipus">
    <xs:sequence>
        <xs:element name="KezdetiAll" type="xs:string"/>
        <xs:element name="VegAll" type="xs:string"/>
        <xs:element name="km" type="xs:integer"/>
    </xs:sequence>
    <xs:attribute name="UtvonalId" use="required">
        <xs:simpleType>
            <xs:restriction base="xs:integer">
                <xs:pattern value="[0-9][0-9][0-9]" />
            </xs:restriction>
        </xs:simpleType>
    </xs:attribute>
</xs:complexType>

 <xs:complexType name="FelugyeloTipus">
      <xs:sequence>
```

```
                <xs:element name="Nev " type="xs:String"/>
                <xs:element name="Telefonszam" type="xs:integer"/>
                <xs:element name="email" type="xs:string"/>
                <xs:element name="irodaszam" type="xs:integer"/>
                <xs:element name="SoforId_FK" type="xs:integer"/>
                <xs:element name="KarbantartoId_FK" type="xs:integer"/>
            </xs:sequence>
            <xs:attribute name="KarbantartoId" use="required">
                <xs:simpleType>
                    <xs:restriction base="xs:integer">
                        <xs:pattern value="[0-9][0-9][0-9]" />
                    </xs:restriction>
                </xs:simpleType>
            </xs:attribute>
        </xs:complexType>

    <xs:complexType name="HaladTipus">
        <xs:sequence>
            <xs:element name="AutobuszId_FK" type="xs:integer"/>
            <xs:element name="UtvonalId_FK" type="xs:integer"/>
                <xs:element name="menetido" type="xs:integer"/>
        </xs:sequence>
    </xs:complexType>

</xs:schema>
```

2. Feladat

Egy DOM program készítése az XML dokumentum adatai adminisztrálása alapján

2,a Adatolvasás

Az adatolvasás első lépése az XML File elérése. Ezután a következő lépés a parse műveletek elvégzése. Ezek után tudjuk az adatokat kezelni.

```
package hudomparseHR6121;

import java.io.File;
import java.io.IOException;
import java.util.ArrayList;
import java.util.List;
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.Comment;
import org.w3c.dom.Document;
import org.w3c.dom.Element;
import org.w3c.dom.Node;
import org.w3c.dom.NodeList;
import org.xml.sax.SAXException;

public class DomReadHR6121 {

    public static void main(String[] args) {
        try {
            File outputFile = new File("XMLReadHR6121.xml");
            StreamResult output = new StreamResult(outputFile);
            Document xmlDocument = parseXmlFile("XMLHR6121.xml");
            writeXmlDocument(xmlDocument, output);
            System.out.println(formatXml(xmlDocument));
        } catch (IOException | SAXException | ParserConfigurationException | TransformerException e) {
            e.printStackTrace();
        }
    }

    // Parse XML file
    public static Document parseXmlFile(String fileName) throws SAXException, IOException,
ParserConfigurationException {
        DocumentBuilderFactory factory = DocumentBuilderFactory.newInstance();
        DocumentBuilder builder = factory.newDocumentBuilder();
        Document document = builder.parse(new File(fileName));
        cleanXmlDocument(document.getDocumentElement());
        return document;
    }

    // Remove empty text nodes
    private static void cleanXmlDocument(Node root) {
        NodeList nodes = root.getChildNodes();
        List<Node> toDelete = new ArrayList<>();
        for (int i = 0; i < nodes.getLength(); i++) {
            if (nodes.item(i).getNodeType() == Node.TEXT_NODE &&
nodes.item(i).getTextContent().strip().isEmpty()) {
                toDelete.add(nodes.item(i));
            } else {
                cleanXmlDocument(nodes.item(i));
            }
        }
        for (Node node : toDelete) {
```

```
            root.removeChild(node);
        }
    }

    // Write XML document
    public static void writeXmlDocument(Document document, StreamResult output) throws
TransformerException {
        TransformerFactory transformerFactory = TransformerFactory.newInstance();
        Transformer transformer = transformerFactory.newTransformer();
        transformer.setOutputProperty(OutputKeys.ENCODING, "UTF-8");
        transformer.setOutputProperty(OutputKeys.INDENT, "yes");
        DOMSource source = new DOMSource(document);
        transformer.transform(source, output);
    }

    // Format XML text
    public static String formatXml(Document document) {
        return "<?xml version=\"" + document.getXmlVersion() + "\" encoding=\"" +
document.getXmlEncoding() + "\" ?>\n" +
                formatXmlElement(document.getDocumentElement(), 0);
    }

    // Recursive function to format XML elements
    public static String formatXmlElement(Node node, int indent) {
        if (node.getNodeType() != Node.ELEMENT_NODE && node.getNodeType() != Node.COMMENT_NODE) {
            return "";
        }
        StringBuilder output = new StringBuilder();

        if (node.getNodeType() == Node.COMMENT_NODE) {
            output.append(getIndentation(indent)).append("<!--").append(((Comment)
node).getTextContent()).append("-->\n");
            return output.toString();
        }

        output.append(getIndentation(indent)).append("<").append(((Element) node).getTagName());

        if (node.hasAttributes()) {
            for (int i = 0; i < node.getAttributes().getLength(); i++) {
                Node attribute = node.getAttributes().item(i);
                output.append("
").append(attribute.getNodeName()).append("=\"").append(attribute.getNodeValue()).append("\"");
            }
        }

        NodeList children = node.getChildNodes();

        if (children.getLength() == 1 && children.item(0).getNodeType() == Node.TEXT_NODE) {

output.append(">").append(children.item(0).getTextContent().trim()).append("</").append(((Element)
node).getTagName()).append(">\n");
        } else {
            output.append(">\n");

            for (int i = 0; i < children.getLength(); i++) {
                output.append(formatXmlElement(children.item(i), indent + 1));
            }

            output.append(getIndentation(indent)).append("</").append(((Element)
node).getTagName()).append(">\n");
        }

        return output.toString();
    }

    private static String getIndentation(int indent) {
        StringBuilder indentation = new StringBuilder();
        for (int i = 0; i < indent; i++) {
            indentation.append("    ");
        }
        return indentation.toString();
    }
}
```

2,b Adatlekérdezés DomQueryHR6121.java

```
package hudomparseHR6121;

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

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

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

public class DomQueryHR6121 {

	public static void main(String[] args) throws SAXException, IOException,
ParserConfigurationException {
		// TODO Auto-generated method stub

		//XML fájl meghívása
		File xmlFile = new File("XMLHR6121.xml");

		//Document builder definiálása
		DocumentBuilderFactory factory = DocumentBuilderFactory.newInstance();
		DocumentBuilder dBuilder = factory.newDocumentBuilder();

		//Fájl betöltése a documentum builderbe
		Document doc = dBuilder.parse(xmlFile);
		doc.getDocumentElement().normalize();

		//Soforok kilistázása

		System.out.println("");
		System.out.println("Soforok kilistazasa: ");
		System.out.println("");

		NodeList soforList = doc.getElementsByTagName("Sofor");
		for(int i=0;i<soforList.getLength();i++) {
			Node nNode = soforList.item(i);
			printSofor(nNode);
		}



		//Ikarus Tipusu autobuszok kilistazasa
		System.out.println("");
		System.out.println("Ikarus tipusu autobuszok: ");
		System.out.println("");

		NodeList feederList = doc.getElementsByTagName("Autobusz");
		for(int i=0;i<feederList.getLength();i++) {
			Node nNode = feederList.item(i);
			printAutobusz(nNode, "Ikarus");
		}


		//Karbantartok kilistázása másik megoldással

		NodeList KarbantartoList =doc.getElementsByTagName("Karbantarto");

		for(int i=0;i<KarbantartoList.getLength();i++) {
			Node nNode = KarbantartoList.item(i);
			System.out.println("");
			printKarbantarto(nNode);
		}
```

```
		//Felugyelok kilistázása

		System.out.println("");
		System.out.println("Felugyelok kilistázása");

		NodeList FelugyeloList = doc.getElementsByTagName("Felugyelo");
		for(int i=0;i<FelugyeloList.getLength();i++) {
			Node nNode = FelugyeloList.item(i);
			System.out.println("");
			printFelugyelo(nNode);
		}

	}



	//Soforok kilistázása
	private static void printSofor(Node nNode) {
		if(nNode.getNodeType() == Node.ELEMENT_NODE) {
			Element elem = (Element) nNode;
			String SoforId = elem.getAttribute("SoforId");

			Node nNode1 = elem.getElementsByTagName("Nev").item(0);
			String Nev = nNode1.getTextContent();

			System.out.printf("SoforID: %s%n", SoforId);
			System.out.printf("Nev: %s%n", Nev);
		}
	}

	//Ikarus tipusu autobuszok kilistazasa
	private static void printAutobusz(Node nNode, String Tipus) {
		if(nNode.getNodeType() == Node.ELEMENT_NODE) {
			Element elem = (Element) nNode;
			String AutobuszId = elem.getAttribute("AutobuszId");

			Node nNode1 = elem.getElementsByTagName("Rendszam").item(0);
			String Rendszam = nNode1.getTextContent();

			Node nNode2 = elem.getElementsByTagName("Tipus").item(0);
			String Tipus2 = nNode2.getTextContent();

			Node nNode3 = elem.getElementsByTagName("max_hely").item(0);
			String max_hely = nNode3.getTextContent();

			if(Tipus.equals(Tipus2)) {
				System.out.printf("AutobuszID: %s%n", AutobuszId);
				System.out.printf("Rendszam: %s%n", Rendszam);
				System.out.printf("Tipus %s%n", Tipus2);
				System.out.printf("max_hely %s%n", max_hely);
				System.out.println("");
			}
		}
	}



	//Karbantartok kilistázása másik megoldással
	private static void printKarbantarto(Node nNode) {
		if(nNode.getNodeType()==Node.ELEMENT_NODE) {
			Element element =(Element) nNode;
			NodeList nList = element.getChildNodes();
			for (int j = 0; j < nList.getLength(); j++) {
		   Node node2 = nList.item(j);
		   if (node2.getNodeType() == Node.ELEMENT_NODE) {
		       Element element2 = (Element) node2;
		       System.out.println(node2.getNodeName()+" : "+node2.getTextContent());
		   }
			}
		}
	}
```

```
	//Felugyelok kilistázása
	private static void printFelugyelo(Node nNode) {
		if(nNode.getNodeType()==Node.ELEMENT_NODE) {
			Element element =(Element) nNode;
			NodeList nList = element.getChildNodes();
			for (int j = 0; j < nList.getLength(); j++) {
		   Node node2 = nList.item(j);
		   if (node2.getNodeType() == Node.ELEMENT_NODE) {
		       Element element2 = (Element) node2;
		       System.out.println(node2.getNodeName()+" : "+node2.getTextContent());


		   }
		       }
		}
	}
}
```

2,c Adatmódosítás (DomModifyHR6121.java)

Ebben az osztályban képesek vagyunk módosítani az elemek attribútumát, és az elemek tartalmát.

```
package hudomparseHR6121;

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

public class DomModifyHR6121 {

	public static void main(String[] args) throws SAXException, IOException,
ParserConfigurationException, TransformerException {
		// TODO Auto-generated method stub

		//XML fájl meghívása
		File xmlFile = new File("XMLHR6121.xml");

		//Document builder definiálása
		DocumentBuilderFactory factory = DocumentBuilderFactory.newInstance();
		DocumentBuilder dBuilder = factory.newDocumentBuilder();

		//Fájl betöltése a DocumentBuilderbe
		Document doc = dBuilder.parse(xmlFile);
		doc.getDocumentElement().normalize();

		//A 002-es id-val rendelkező autobusz rendszamat ABC-123-ra valtoztatjuk
		NodeList nodes = doc.getElementsByTagName("Autobusz");
		for(int i=0;i<nodes.getLength();i++) {
			Node node = nodes.item(i);
			if(node.getNodeType() == Node.ELEMENT_NODE) {
```

```
	if(node.getAttributes().getNamedItem("AutobuszId").getTextContent().equals("002")) {
				NodeList childNodes = node.getChildNodes();
				for(int j=0;j<childNodes.getLength();j++) {
					Node childNode = childNodes.item(j);
					if(childNode.getNodeName().equals("Rendszam")) {
						childNode.setTextContent("ABC-123");
					}

				}
			}
		}
	}

		//Újabb Autobuszt adunk hozzá
		Element Autobusz = (Element)doc.getElementsByTagName("Autobuszok").item(0);
		Autobusz.appendChild(createAutobusz(doc,"010", "LKC-154", "Ikarus", "40"));

		//Újabb Sofort adunk hozzá
		Element Sofor = (Element)doc.getElementsByTagName("Soforok").item(0);
		Sofor.appendChild(createSofor(doc,"L93751", "Korti Gyula", "086", "005"));

		//Módosítjuk a 504-es id-val rendelkező felugyelot 601-re
		modifyId(doc, "Felugyelo", "FelugyeloId", "504", " 601");

		//Módosítjuk a 303-as id-val rendelkező eszkoz id-ját 366ra
		modifyId(doc, "Szervezo", "SzervezoId", "303", "366");

		//Lementjük a módosított dokumentumot
		SaveAsDoc(doc, "teszt.xml");
	}

	//Lementjük a módosított xml dokumentumot
	public static void SaveAsDoc(Document doc, String filename) throws TransformerException {
		TransformerFactory transformerFactory = TransformerFactory.newInstance();
		Transformer transf = transformerFactory.newTransformer();

		transf.setOutputProperty(OutputKeys.ENCODING, "UTF-8");
		transf.setOutputProperty(OutputKeys.INDENT, "yes");
		transf.setOutputProperty("{https://xml.apache.org/.xslt}indent-amount", "2");

		DOMSource source = new DOMSource(doc);

		File myFile = new File(filename);

		StreamResult console = new StreamResult(System.out);
		StreamResult file = new StreamResult(myFile);

		transf.transform(source, console);
		transf.transform(source, file);
	}

	//Autobuszt hozzunk létre
	private static Node createAutobusz(Document doc, String AutobuszId, String Rendszam, String
Tipus, String max_hely) {

		Element user = doc.createElement("Autobusz");

		user.setAttribute("AutobuszId", AutobuszId);
		user.appendChild(createElement(doc, "Rendszam", Rendszam));
		user.appendChild(createElement(doc, "Tipus", Tipus));
		user.appendChild(createElement(doc, "max_hely", max_hely));

		return user;
	}


	//Sofort hozzunk létre
	private static Node createSofor(Document doc, String SoforId, String Jegykiadoszam, String Nev,
String AutobuszId_fk) {

		Element user = doc.createElement("Szektor");

		user.setAttribute("SoforId", SoforId);
```

```
            user.appendChild(createElement(doc, "Jegykiadoszam", Jegykiadoszam));
            user.appendChild(createElement(doc, "Nev", Nev));
            user.appendChild(createElement(doc, "AutobuszId_fk", AutobuszId_fk));


            return user;
      }

      //Elementet hozzunk létre
      private static Node createElement(Document doc, String name, String value) {

            Element node = doc.createElement(name);
            node.appendChild(doc.createTextNode(value));

            return node;
      }
      //Módosítjuk az id-t

      public static void modifyId(Document doc, String tagName, String attrName, String oldData,
String newData) {

            NodeList nodes = doc.getElementsByTagName(tagName);
            for (int i = 0; i < nodes.getLength(); i++) {
                  Node node = nodes.item(i);
                  if (node.getNodeType() == Node.ELEMENT_NODE) {
                        if
(node.getAttributes().getNamedItem(attrName).getTextContent().equals(oldData)) {
                              node.getAttributes().getNamedItem(attrName).setTextContent(newData);
                        }
                  }
            }
      }
```

2,d Adatírás (DomWriteHR6121.java)

Ebben az osztályban kiirathatjuk az xml filet konzolra, valamint egy xml fileba.

```
package hudomparseHR6121;

import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.DocumentBuilderFactory;
import javax.xml.transform.OutputKeys;
import javax.xml.transform.Transformer;
import javax.xml.transform.TransformerFactory;
import javax.xml.transform.dom.DOMSource;
import javax.xml.transform.stream.StreamResult;
import java.io.File;
import java.io.FileOutputStream;
import java.io.OutputStream;
import java.io.PrintWriter;
import java.io.File;
import java.io.FileWriter;
import java.io.PrintWriter;
import java.util.Arrays;
import java.util.List;
import java.util.StringJoiner;

import org.w3c.dom.*;

public class DomWriteHR6121 {

      public static void main(String[] args) {
        try {
            DocumentBuilderFactory factory = DocumentBuilderFactory.newInstance();
            DocumentBuilder builder = factory.newDocumentBuilder();
            Document document = builder.newDocument();
            Element rootElement = document.createElement("Palyaudvar_HR6121");
            rootElement.setAttribute("xmlns:xs", "http://www.w3.org/2001/XMLSchema-instance");
            rootElement.setAttribute("xs:noNamespaceSchemaLocation", "XMLSchemaHR6121.xsd");
            document.appendChild(rootElement);
```

```
            generateAutobuszElements(document, rootElement);
            generateSoforElements(document, rootElement);
            generateKarbantartoElements(document, rootElement);
            generateEszkozElements(document, rootElement);
            generateUtvonalElements(document, rootElement);
            generateHaladElements(document, rootElement);
            generateFelugyeloElements(document, rootElement);

            TransformerFactory transformerFactory = TransformerFactory.newInstance();
            Transformer transformer = transformerFactory.newTransformer();
            transformer.setOutputProperty(OutputKeys.ENCODING, "UTF-8");
            transformer.setOutputProperty(OutputKeys.INDENT, "yes");
            transformer.setOutputProperty("{https://xml.apache.org/xslt}indent-amount", "4");

            printDocument(document, "HR6121.xml");
        } catch (Exception e) {
            e.printStackTrace();
        }
    }

    private static void generateAutobuszElements(Document document, Element rootElement) {
        List<List<String>> autobuszData = Arrays.asList(
                Arrays.asList("001", "AAA-111", "Ikarus", "40"),
                Arrays.asList("002", "BBB-222", "MAN", "45"),
                Arrays.asList("003", "CCC-333", "Volvo", "50"),
                Arrays.asList("004", "DDD-444", "Ikarus", "40"),
                Arrays.asList("005", "EEE-555", "Ikarus", "40"),
                Arrays.asList("006", "FFF-666", "Ikarus", "40"),
                Arrays.asList("007", "GGG-777", "Volvo", "50"),
                Arrays.asList("008", "HHH-888", "Ikarus", "40"),
                Arrays.asList("009", "III-999", "MAN", "45")
        );

        for (List<String> autobusz : autobuszData) {
            Element autobuszElement = document.createElement("Autobusz");
            autobuszElement.setAttribute("AutobuszId", autobusz.get(0));

            addChildElement(document, autobuszElement, "Rendszam", autobusz.get(1));
            addChildElement(document, autobuszElement, "Tipus", autobusz.get(2));
            addChildElement(document, autobuszElement, "Max_Hely", autobusz.get(3));
            rootElement.appendChild(autobuszElement);
        }
    }

    private static void generateSoforElements(Document document, Element rootElement) {
        List<List<String>> soforData = Arrays.asList(
                Arrays.asList("101", "009", "Major Daniel", "A54321"),
                Arrays.asList("102", "005", "Tancos Judit", "B12345"),
                Arrays.asList("103", "003", "Beno Rezso", "C98765"),
                Arrays.asList("104", "008", "Prezs Peter", "D56789"),
                Arrays.asList("105", "001", "Kont Fanni", "E23456"),
                Arrays.asList("106", "002", "Kann András", "F65432"),
                Arrays.asList("107", "003", "Furko Evelin", "G34567"),
                Arrays.asList("108", "004", "Labas Lajos", "H76543"),
                Arrays.asList("109", "006", "Bence Nandor", "I45678")
        );

        for (List<String> sofor : soforData) {
            Element soforElement = document.createElement("Sofor");
            soforElement.setAttribute("SoforId", sofor.get(0));
            soforElement.setAttribute("AutobuszId", sofor.get(1));

            addChildElement(document, soforElement, "Nev", sofor.get(2));
            addChildElement(document, soforElement, "Jegykiadoszam", sofor.get(3));

            rootElement.appendChild(soforElement);
        }
    }

    private static void generateKarbantartoElements(Document document, Element rootElement) {
        List<List<String>> karbantartoData = Arrays.asList(
                Arrays.asList("201", "004", "Ant Istvan", "A0001"),
                Arrays.asList("202", "003", "Várti Ádám", "B0002"),
```

```
            Arrays.asList("203", "001", "Ács Viktória", "C0003"),
            Arrays.asList("204", "008", "Csik Zoltán", "D0004"),
            Arrays.asList("205", "009", "Szabó Imre", "E0005")
        );

    for (List<String> karbantarto : karbantartoData) {
        Element karbantartoElement = document.createElement("Karbantarto");
        karbantartoElement.setAttribute("KarbantartoId", karbantarto.get(0));
        karbantartoElement.setAttribute("AutobuszId", karbantarto.get(1));

        addChildElement(document, karbantartoElement, "Nev", karbantarto.get(2));
        addChildElement(document, karbantartoElement, "Javitoszam", karbantarto.get(3));

        rootElement.appendChild(karbantartoElement);
    }
}
private static void generateEszkozElements(Document document, Element rootElement) {
    List<List<String>> eszkozData = Arrays.asList(
            Arrays.asList("301", "201", "97", "Makita", "1"),
            Arrays.asList("302", "202", "42", "Bosch", "3"),
            Arrays.asList("303", "203", "55", "Bosch", "5"),
            Arrays.asList("304", "204", "12", "Makita", "2"),
            Arrays.asList("305", "205", "08", "Bosch", "7")
    );

    for (List<String> eszkoz : eszkozData) {
        Element eszkozElement = document.createElement("Eszkoz");
        eszkozElement.setAttribute("EszkozId", eszkoz.get(0));
        eszkozElement.setAttribute("KarbantartoId", eszkoz.get(1));

        addChildElement(document, eszkozElement, "Termekszam", eszkoz.get(2));
        addChildElement(document, eszkozElement, "Gyarto", eszkoz.get(3));
        addChildElement(document, eszkozElement, "Ev", eszkoz.get(4));

        rootElement.appendChild(eszkozElement);
    }
}

private static void generateUtvonalElements(Document document, Element rootElement) {
    List<List<String>> utvonalData = Arrays.asList(
            Arrays.asList("401", "Buza Ter", "Tapolca", "8"),
            Arrays.asList("402", "Buza Ter", "Szirma", "9"),
            Arrays.asList("403", "Felso-Majlath", "Lillafured", "6"),
            Arrays.asList("404", "Ujgyori Foter", "Pereces", "7"),
            Arrays.asList("405", "Avas Kilato", "Centrum", "5")
    );

    for (List<String> utvonal : utvonalData) {
        Element utvonalElement = document.createElement("Utvonal");
        utvonalElement.setAttribute("UtvonalId", utvonal.get(0));

        addChildElement(document, utvonalElement, "KezdAll", utvonal.get(1));
        addChildElement(document, utvonalElement, "VegAll", utvonal.get(2));
        addChildElement(document, utvonalElement, "km", utvonal.get(3));

        rootElement.appendChild(utvonalElement);
    }
}

private static void generateHaladElements(Document document, Element rootElement) {
    List<List<String>> haladData = Arrays.asList(
            Arrays.asList("123", "1736", "32", "2023.09.12"),
            Arrays.asList("124", "1755", "28", "2023.09.18"),
            Arrays.asList("125", "6354", "30", "2023.09.20"),
            Arrays.asList("126", "2756", "25", "2023.09.25"),
            Arrays.asList("127", "6322", "20", "2023.09.28"),
            Arrays.asList("128", "8751", "18", "2023.09.30")
    );

    for (List<String> halad : haladData) {
        Element haladElement = document.createElement("halad");
        haladElement.setAttribute("UtvonalId", halad.get(0));
        haladElement.setAttribute("AutobuszId", halad.get(1));
```

```
            addChildElement(document, haladElement, "menetido", halad.get(2));

            rootElement.appendChild(haladElement);
        }
    }

    private static void generateFelugyeloElements(Document document, Element rootElement) {
        List<List<String>> felugyeloData = Arrays.asList(
                Arrays.asList("501", "Pataki Csanad", "102", "204", "11", "pat.csan@gmail.com", "06 55
584 888", "36 55 563 979"),
                Arrays.asList("502", "Gabor Borbala", "104", "202", "12", "15", "gab.bor@gmail.com",
"06 55 010 226", "36 55 686 186"),
                Arrays.asList("503", "Szabó Oszkár", "103", "201", "13", "10", "szab.osz@gmail.com",
"06 55 686 186", "36 55 010 226"),
                Arrays.asList("504", "Hegyi Borbala", "108", "203", "14", "5", "hegy.bor@gmail.com",
"06 55 563 979", "36 55 584 888"),
                Arrays.asList("505", "Nagy Andrea", "109", "205", "15", "nagy.and@gmail.com", "06 55
923 742", "36 55 923 742")
        );

        for (List<String> felugyelo : felugyeloData) {
            Element felugyeloElement = document.createElement("Felugyelo");
            felugyeloElement.setAttribute("FelugyeloId", felugyelo.get(0));
            felugyeloElement.setAttribute("SoforId", felugyelo.get(2));
            felugyeloElement.setAttribute("KarbantartoId", felugyelo.get(3));

            addChildElement(document, felugyeloElement, "Név", felugyelo.get(1));
            addChildElement(document, felugyeloElement, "Telefonszam", felugyelo.get(6));
            addChildElement(document, felugyeloElement, "Telefonszam", felugyelo.get(7));

            Element ElerehetosegElement = document.createElement("Elerhetoseg");
            addChildElement(document, ElerehetosegElement, "Irodaszam", felugyelo.get(4));
            addChildElement(document, ElerehetosegElement, "email", felugyelo.get(5));



            felugyeloElement.appendChild(ElerehetosegElement);
            rootElement.appendChild(felugyeloElement);
        }
    }


    private static void addChildElement(Document document, Element parentElement, String tagName,
String textContent) {
        Element childElement = document.createElement(tagName);
        childElement.setTextContent(textContent);
        parentElement.appendChild(childElement);
    }
    private static void printDocument(Document document, String fileName) throws Exception {
        TransformerFactory transformerFactory = TransformerFactory.newInstance();
        Transformer transformer = transformerFactory.newTransformer();
        transformer.setOutputProperty(OutputKeys.ENCODING, "UTF-8");
        transformer.setOutputProperty(OutputKeys.INDENT, "yes");
        transformer.setOutputProperty("{https://xml.apache.org/xslt}indent-amount", "4");


        transformer.transform(new DOMSource(document), new StreamResult(System.out));


        saveToFile(document, fileName);
    }
    private static void saveToFile(Document document, String fileName) throws Exception {
        try (OutputStream os = new FileOutputStream(new File(fileName));
             PrintWriter writer = new PrintWriter(os)) {

            TransformerFactory transformerFactory = TransformerFactory.newInstance();
            Transformer transformer = transformerFactory.newTransformer();
            transformer.setOutputProperty(OutputKeys.ENCODING, "UTF-8");
            transformer.setOutputProperty(OutputKeys.INDENT, "yes");
            transformer.setOutputProperty("{https://xml.apache.org/xslt}indent-amount", "4");


            transformer.transform(new DOMSource(document), new StreamResult(writer));
        }
```

```
    }

}
```