

JEGYZŐKÖNYV

Adatkezelés XML környezetben

Féléves feladat

Börtön nyilvántartás

Készítette: **Miliczki József**

Neptunkód: **Y86I0I**

Dátum: **2022. 11. 19**

Tartalomjegyzék

1. A feladat leírása	3
1.1 Adatbázisban szereplő adatok	3
1.2 Kapcsolatok leírása	5
1.3 Az adatbázisban ER modellje	5
1.4 Az XDM Modell	5
2. Megvalósítások	6
XML	6
XML Schema	12
Java XML DOM olvasó program	18
Java XML Lekérdező program	21
Java XML Módosító program	25

1. A feladat leírása

Számos börtön adatait és annak dolgozóinak személyes információit tároló XML dokumentum létrehozása XML Schema validálással, majd ezen állományok különböző nemű manipulálása és olvasása Java DOM programokkal. Maga az ötlet az Adatbázis 1 tantárgyból származik, ám az XML beadandó érdekében lényeges tartalmi és szerkezeti változtatások történtek a börtön adatbázis logikai leírásában. A börtönőr egyed eltűnt, ehelyett részletesebben definiáltam a dolgozókat, megjelent a műszak, valamint egyéb dolgozókkal kapcsolatos tulajdonság. Az egyedek tulajdonságainál ügyeltem a változatosságra, többértékű elemek, különböző adattípusok teszik színessé az adatkezelést.

1.1 Adatbázisban szereplő adatok

Személyzet:

A börtönben dolgozó emberek.

- Szem_id (Személyzet id), 3 jegyű egész szám. Egy személy pontos beazonosítására szolgáló szám. **KULCS**.
- Pozíció, szöveg. A dolgozó posztja (takarító, konyhás, sofőr, börtönőr stb).
- Szolgálati hely, szöveg: A dolgozó azon helye, ahol a munkavégzést végrehajtja (csak oda engedik be a dolgozót, ahol munkája van!)
- Családi állapot, többértékű mező: A dolgozó családi állapota

Alkalmazott:

Az összes alkalmazottat tároló tábla. Itt általános információkat tárolunk, amely minden alkalmazottnál azonos attribútumokkal rendelkezik.

- Életkor, 3 jegyű egész. Az alkalmazott életkora.
- Név, szöveg. Az alkalmazott neve.

- Alk_id (Alkalmazotti id) 3 jegyű egész szám. Minden alkalmazott rendelkezik egy ilyen számmal. **KULCS.**
- Lakhely
 - Ir. szám (irányítószám), 4 jegyű egész. Az alkalmazott lakhelyének irányítószáma.
 - Város, szöveg. Azon város/település helye, ami az alkalmazott lakhelye.
 - Utca, szöveg. Az alkalmazott lakhelyének utca neve.
 - Házszám, 4 jegyű egész. Az alkalmazott lakhelyének házszáma.

Börtön épület:

A börtön épületek tulajdonságai, ez a munkahely az alkalmazottak számára

- Cellák száma, számadat. Megadja mely börtön épületekben mennyi cella található.
- Kapacitás, számadat. Visszaadja az épület teljes kapacitását.
- Börtön_id, 1 jegyű nagybetű. A börtön egyedi, betű alapú azonosítója. **KULCS**
- Épület neve, szöveg. A börtön neve.
- Építés, többértékű elem. A börtön építésének stílusa.

Cellák:

- Cella_id, betű + szám. A cella egyedi azonosító száma, ahol a betű az épületet, a szám a cella számát írja le. **KULCS.**
- Cella kapacitás, 1 jegyű egész. A kérdéses cella emberben mért kapacitása.
- Emelet, egyjegyű egész. A cella melyik emeleten van.
- Ablak, többértékű elem. A cella tartalmaz ablakot?

Rab:

- Rab_id szöveg + szám. A rab azonosítója, "rab" prefix + szám. **KULCS.**
- Életkor, 3 jegyű egész. A rab életkora.
- Név, szöveg. A rab neve.

- Letöltendő, dátum. 2 adat:
 - Kezdet, a börtönbüntetés kezdete.
 - Vég, a börtönbüntetés vége.

1.2 Kapcsolatok leírása

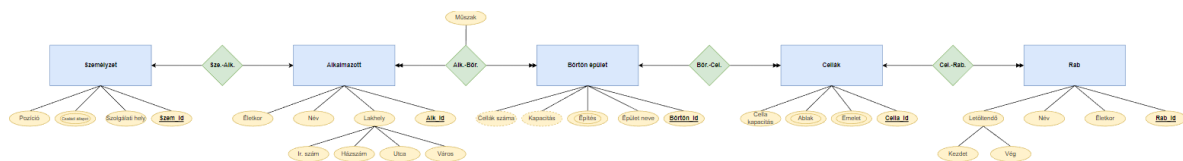
Személyzet-Alkalmazott: **1:1 kapcsolat**. Az Alkalmazott tábla a dolgozók összefoglalása, így egy Személyzet bejegyzés megegyezik 1 Alkalmazott bejegyzéssel.

Alkalmazott-Börtön épület: **N:N kapcsolat**. Egy épületben dolgozhat több alkalmazott is, és egy alkalmazott dolgozhat több épületben.

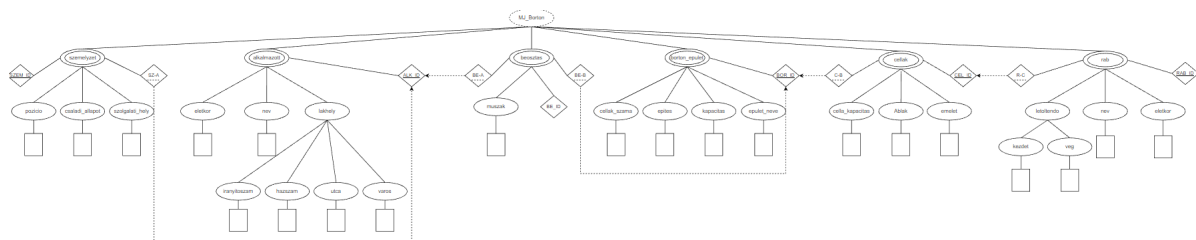
Börtön épület - Cellák: **1:N kapcsolat**. Egy börtön épületben több cella lehet.

Cellák-Rab: **1:N kapcsolat**. Egy cellában több rab is tartózkodhat.

1.3 Az adatbázisban ER modellje



1.4 Az XDM Modell



2. Megvalósítások

2.1 XML

Az **XMLY86I0I.xml** dokumentumot Visual Code-ban hoztam létre, és XML 1.0 szabvány szerint készült el. A dokumentumhoz hozzá kötöttem az **XMLSchemaY86I0I.xsd** séma file-t, és definiáltam az egyedeket az XML szabályainak megfelelően. Ahol szükséges volt, gyermek elemeket, valamint attribútumokat használtam a tagok azonosításához.

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

```
<MJ_Borton
```

```
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
```

```
xsi:noNamespaceSchemaLocation="XMLSchemaY86I0I.xsd">
```

```
  <!-- Személyzet -->
```

```
  <szemelyzet SZ-A="01" SZEM_ID="100">
```

```
    <pozicio>Takarito</pozicio>
```

```
    <szolgalmati_hely>Folyosok</szolgalmati_hely>
```

```
    <csaladi_allapot>Házás</csaladi_allapot>
```

```
  </szemelyzet>
```

```
  <szemelyzet SZ-A="02" SZEM_ID="101">
```

```
    <pozicio>Szakacs</pozicio>
```

```
    <szolgalmati_hely>Konyha</szolgalmati_hely>
```

```
    <csaladi_allapot>Egyedülálló</csaladi_allapot>
```

```
  </szemelyzet>
```

```
  <szemelyzet SZ-A="03" SZEM_ID="102">
```

```
    <pozicio>Bortonor</pozicio>
```

```
    <szolgalmati_hely>Borton
```

```
epuletek</szolgalmati_hely>
```

```
    <csaladi_allapot>Elvált</csaladi_allapot>
```

```
  </szemelyzet>
```

```
  <szemelyzet SZ-A="04" SZEM_ID="103">
```

```
    <pozicio>Sofor</pozicio>
```

```
    <szolgalmati_hely>Kocsibejaro</szolgalmati_hely>
```

```

    <csaladi_allapot>Egyedülálló</csaladi_allapot>
</szemelyzet>
<szemelyzet SZ-A="05" SZEM_ID="104">
    <pozicio>Orvos</pozicio>
    <szolgalmati_hely>Borton korhaz</szolgalmati_hely>
    <csaladi_allapot>Házás</csaladi_allapot>
</szemelyzet>

<!-- Alkalmazottak általános adatai -->
<alkalmazott ALK_ID = "01">
    <eletkor>34</eletkor>
    <nev>Kis Karoly</nev>
    <lakhely>
        <iranyitoszam>3529</iranyitoszam>
        <hazszam>23</hazszam>
        <utca>Afonyas</utca>
        <varos>Miskolc</varos>
    </lakhely>
</alkalmazott>
<alkalmazott ALK_ID = "02">
    <eletkor>55</eletkor>
    <nev>Mucsi Zoltan</nev>
    <lakhely>
        <iranyitoszam>1032</iranyitoszam>
        <hazszam>102</hazszam>
        <utca>Kossuth Lajos</utca>
        <varos>Budapest</varos>
    </lakhely>
</alkalmazott>
<alkalmazott ALK_ID = "03">
    <eletkor>57</eletkor>
    <nev>Miliczki Jozsef</nev>
    <lakhely>
        <iranyitoszam>3012</iranyitoszam>
        <hazszam>141</hazszam>
        <utca>Beke</utca>
        <varos>Győr</varos>
    </lakhely>

```

```

</alkalmazott>
<alkalmazott ALK_ID = "04">
  <eletkor>25</eletkor>
  <nev>Turing Ferenc</nev>
  <lakhely>
    <iranyitoszam>2525</iranyitoszam>
    <hazszam>26</hazszam>
    <utca>Saros</utca>
    <varos>Kecskemét</varos>
  </lakhely>
</alkalmazott>
<alkalmazott ALK_ID = "05">
  <eletkor>44</eletkor>
  <nev>Eros Antonia</nev>
  <lakhely>
    <iranyitoszam>4412</iranyitoszam>
    <hazszam>56</hazszam>
    <utca>Szabadsag Ter</utca>
    <varos>Szeged</varos>
  </lakhely>
</alkalmazott>
<!-- Alkalmazottak összekapcsolása Börtön épületekkel
-->
  <beosztas BE-A="01" BE-B="A" BE_ID="1">
    <muszak>Esti</muszak>
  </beosztas>
  <beosztas BE-A="01" BE-B="B" BE_ID="2">
    <muszak>Nappali</muszak>
  </beosztas>
  <beosztas BE-A="02" BE-B="C" BE_ID="3">
    <muszak>Esti</muszak>
  </beosztas>
  <beosztas BE-A="03" BE-B="C" BE_ID="4">
    <muszak>Delutanos</muszak>
  </beosztas>
  <beosztas BE-A="04" BE-B="A" BE_ID="5">
    <muszak>Nappali</muszak>
  </beosztas>

```



```

<beosztas BE-A="04" BE-B="C" BE_ID="6">
  <muszak>Delutanos</muszak>
</beosztas>
<beosztas BE-A="05" BE-B="B" BE_ID="7">
  <muszak>Esti</muszak>
</beosztas>
<!-- Börtön épületek -->
<borton_epulet BOR_ID = "A">
  <cellak_szama>30</cellak_szama>
  <kapacitas>120</kapacitas>
  <epulet_neve>Szecheny borton</epulet_neve>
  <epites>Újépítésű Téglá</epites>
</borton_epulet>
<borton_epulet BOR_ID = "B">
  <cellak_szama>45</cellak_szama>
  <kapacitas>220</kapacitas>
  <epulet_neve>Janos borton</epulet_neve>
  <epites>Újépítésű Beton</epites>
</borton_epulet>
<borton_epulet BOR_ID = "C">
  <cellak_szama>50</cellak_szama>
  <kapacitas>240</kapacitas>
  <epulet_neve>Solyom borton</epulet_neve>
  <epites>Régi Téglá</epites>
</borton_epulet>
<!-- "A" börtön cellái-->
<cellak CEL_ID = "a1" C-B="A">
  <cella_kapacitas>4</cella_kapacitas>
  <emelet>1</emelet>
  <ablak>Van</ablak>
</cellak>
<cellak CEL_ID = "a2" C-B="A">
  <cella_kapacitas>4</cella_kapacitas>
  <emelet>1</emelet>
  <ablak>Van</ablak>
</cellak>
<cellak CEL_ID = "a3" C-B="A">
  <cella_kapacitas>2</cella_kapacitas>

```

```

        <emelet>2</emelet>
        <ablak>Van</ablak>
    </cellak>
    <cellak CEL_ID = "a4" C-B="A">
        <cella_kapacitas>1</cella_kapacitas>
        <emelet>2</emelet>
        <ablak>Nincs</ablak>
    </cellak>
    <cellak CEL_ID = "a5" C-B="A">
        <cella_kapacitas>5</cella_kapacitas>
        <emelet>2</emelet>
        <ablak>Van</ablak>
    </cellak>
<!-- "B" börtön cellái-->
    <cellak CEL_ID = "b1" C-B="B">
        <cella_kapacitas>3</cella_kapacitas>
        <emelet>1</emelet>
        <ablak>Nincs</ablak>
    </cellak>
    <cellak CEL_ID = "b2" C-B="B">
        <cella_kapacitas>3</cella_kapacitas>
        <emelet>2</emelet>
        <ablak>Van</ablak>
    </cellak>
    <cellak CEL_ID = "b3" C-B="B">
        <cella_kapacitas>3</cella_kapacitas>
        <emelet>3</emelet>
        <ablak>Van</ablak>
    </cellak>
    <cellak CEL_ID = "b4" C-B="B">
        <cella_kapacitas>2</cella_kapacitas>
        <emelet>3</emelet>
        <ablak>Van</ablak>
    </cellak>
    <cellak CEL_ID = "b5" C-B="B">
        <cella_kapacitas>5</cella_kapacitas>
        <emelet>3</emelet>
        <ablak>Nincs</ablak>

```

```

</cellak>

<!-- "C" börtön cellái-->
<cellak CEL_ID = "c1" C-B="C">
  <cella_kapacitas>4</cella_kapacitas>
  <emelet>1</emelet>
  <ablak>Van</ablak>
</cellak>
<cellak CEL_ID = "c2" C-B="C">
  <cella_kapacitas>3</cella_kapacitas>
  <emelet>1</emelet>
  <ablak>Nincs</ablak>
</cellak>
<cellak CEL_ID = "c3" C-B="C">
  <cella_kapacitas>3</cella_kapacitas>
  <emelet>1</emelet>
  <ablak>Van</ablak>
</cellak>
<cellak CEL_ID = "c4" C-B="C">
  <cella_kapacitas>4</cella_kapacitas>
  <emelet>2</emelet>
  <ablak>Van</ablak>
</cellak>
<cellak CEL_ID = "c5" C-B="C">
  <cella_kapacitas>2</cella_kapacitas>
  <emelet>3</emelet>
  <ablak>Van</ablak>
</cellak>
<!-- Rabok cellákra osztva -->
<rab RAB_ID = "rab1" R-C="a1">
  <nev>Gengszter Lajos</nev>
  <eletkor>38</eletkor>
  <letoltendo>
    <kezdet>2000.10.12</kezdet>
    <veg>2010.10.12</veg>
  </letoltendo>
</rab>
<rab RAB_ID = "rab2" R-C="a3">

```

```
<nev>Bűnöző Ferenc</nev>
<eletkor>54</eletkor>
<letoltendo>
  <kezdet>1992.03.22</kezdet>
  <veg>2003.03.22</veg>
</letoltendo>
</rab>
<rab RAB_ID = "rab3" R-C="b1">
  <nev>Lopó Benedek</nev>
  <eletkor>22</eletkor>
  <letoltendo>
    <kezdet>2001.06.01</kezdet>
    <veg>2002.05.15</veg>
  </letoltendo>
</rab>
<rab RAB_ID = "rab4" R-C="b2">
  <nev>Hamiskártyás Ernő</nev>
  <eletkor>32</eletkor>
  <letoltendo>
    <kezdet>2010.01.01</kezdet>
    <veg>2015.01.01</veg>
  </letoltendo>
</rab>
<rab RAB_ID = "rab5" R-C="b2">
  <nev>Gonosz Gergő</nev>
  <eletkor>47</eletkor>
  <letoltendo>
    <kezdet>2006.04.30</kezdet>
    <veg>2010.07.15</veg>
  </letoltendo>
</rab>
<rab RAB_ID = "rab6" R-C="c1">
  <nev>Ártatlan Áron</nev>
  <eletkor>51</eletkor>
  <letoltendo>
    <kezdet>1995.10.12</kezdet>
    <veg>2020.01.01</veg>
  </letoltendo>
```

```
</rab>
</MJ_Borton>
```

2.2 XML Schema

Az **XMLSchemaY86I0I.xsd** séma file leírja mindazon megkötéseket, amelyeknek az XML dokumentumnak meg kell felelnie. Itt definiálunk minden típust, amit az XML file-ban használni szeretnénk, valamint az adatbázis kapcsolatait `xs:unique` és `xs:keyref` bejegyzésekkel hozom létre.

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema attributeFormDefault="unqualified"
  elementFormDefault="qualified"
  xmlns:xs="http://www.w3.org/2001/XMLSchema">

  <!-- Egyszerű típusok -->
  <xs:simpleType name="csaladi_allapot_type"
    final="restriction">
    <xs:restriction base="xs:string">
      <xs:enumeration value="Házas" />
      <xs:enumeration value="Egyedülálló" />
      <xs:enumeration value="Elvált" />
    </xs:restriction>
  </xs:simpleType>

  <xs:simpleType name="epites_type" final="restriction">
    <xs:restriction base="xs:string">
      <xs:enumeration value="Újépítésű Téglá" />
      <xs:enumeration value="Újépítésű Beton" />
      <xs:enumeration value="Régi Téglá" />
      <xs:enumeration value="Régi Beton" />
    </xs:restriction>
  </xs:simpleType>

  <xs:simpleType name="ablak_type" final="restriction">
    <xs:restriction base="xs:string">
      <xs:enumeration value="Van" />
      <xs:enumeration value="Nincs" />
    </xs:restriction>
```

```

</xs:simpleType>
<xs:simpleType name="emelet_type" final="restriction">
  <xs:restriction base="xs:byte">
    <xs:enumeration value="1" />
    <xs:enumeration value="2" />
    <xs:enumeration value="3" />
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name="pozicio_type">
  <xs:restriction base="xs:string">
    <xs:minLength value="1" />
    <xs:pattern value="([a-zA-Z])*" />
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name="szolgalati_hely_type">
  <xs:restriction base="xs:string">
    <xs:minLength value="5" />
    <xs:pattern value="([a-z\s*A-Z])*" />
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name="nev_type">
  <xs:restriction base="xs:string">
    <xs:pattern value="[A-Z][a-zA-Z]*([A-Z][a-zA-Z])*" />
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name="iranyitoszam_type">
  <xs:restriction base="xs:string">
    <xs:length value="4" />
    <xs:pattern value="([0-9])*" />
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name="hazszam_type">
  <xs:restriction base="xs:string">
    <xs:maxLength value="3" />
    <xs:pattern value="([0-9])*" />
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name="utca_type">

```

```

    <xs:restriction base="xs:string">
        <xs:pattern value="[A-Z][a-zA-Z]*( [A-Z][a-zA-Z]*)*" />
    </xs:restriction>
</xs:simpleType>

<xs:simpleType name="muszak_type">
    <xs:restriction base="xs:string">
        <xs:enumeration value="Nappali" />
        <xs:enumeration value="Delutanos" />
        <xs:enumeration value="Esti" />
    </xs:restriction>
</xs:simpleType>

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

<xs:element name="MJ_Borton">
    <xs:complexType>
        <xs:sequence>
            <xs:element maxOccurs="unbounded" name="szemelyzet">
                <xs:complexType>
                    <xs:sequence>
                        <xs:element name="pozicio" type="pozicio_type"/>
                        <xs:element name="szolgalati_hely"
type="szolgalati_hely_type" />
                        <xs:element name="csaladi_allapot"
type="csaladi_allapot_type" />
                    </xs:sequence>
                    <xs:attribute name="SZ-A" type="xs:unsignedByte"
use="required" />
                    <xs:attribute name="SZEM_ID"
type="xs:unsignedByte" use="required" />
                </xs:complexType>
            </xs:element>
            <xs:element maxOccurs="unbounded" name="alkalmazott">
                <xs:complexType>
                    <xs:sequence>
                        <xs:element name="eletkor"
type="xs:unsignedByte" />

```

```

        <xs:element name="nev" type="nev_type" />
        <xs:element name="lakhely">
            <xs:complexType>
                <xs:sequence>
                    <xs:element name="iranyitoszam"
type="iranyitoszam_type" />
                    <xs:element name="hazszam"
type="xs:unsignedByte" />
                    <xs:element name="utca" type="utca_type"
/>
                    <xs:element name="varos" type="xs:string"
/>
                </xs:sequence>
            </xs:complexType>
        </xs:element>
    </xs:sequence>
    <xs:attribute name="ALK_ID" type="xs:unsignedByte"
use="required" />
</xs:complexType>
</xs:element>
<xs:element maxOccurs="unbounded" name="beosztas">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="muszak" type="muszak_type" />
        </xs:sequence>
        <xs:attribute name="BE-A" type="xs:unsignedByte"
use="required" />
        <xs:attribute name="BE-B" type="xs:string"
use="required" />
        <xs:attribute name="BE_ID" type="xs:string"
use="required" />
    </xs:complexType>
</xs:element>
<xs:element maxOccurs="unbounded"
name="borton_epulet">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="cellak_szama"
type="xs:unsignedByte" />
            <xs:element name="kapacitas"
type="xs:unsignedByte" />
            <xs:element name="epulet_neve" type="xs:string"
/>
            <xs:element name="epites" type="epites_type" />

```



```

        </xs:sequence>
        <xs:attribute name="BOR_ID" type="xs:string"
use="required" />
    </xs:complexType>
</xs:element>
<xs:element maxOccurs="unbounded" name="cellak">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="cella_kapacitas"
type="xs:unsignedByte" />
            <xs:element name="emelet" type="emelet_type" />
            <xs:element name="ablak" type="ablak_type" />
        </xs:sequence>
        <xs:attribute name="CEL_ID" type="xs:string"
use="required" />
        <xs:attribute name="C-B" type="xs:string"
use="required" />
    </xs:complexType>
</xs:element>
<xs:element maxOccurs="unbounded" name="rab">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="nev" type="xs:string" />
            <xs:element name="eletkor"
type="xs:unsignedByte" />
            <xs:element name="letoltendo">
                <xs:complexType>
                    <xs:sequence>
                        <xs:element name="kezdet"
type="datum_type" />
                        <xs:element name="veg" type="datum_type"
/>
                    </xs:sequence>
                </xs:complexType>
            </xs:element>
        </xs:sequence>
        <xs:attribute name="RAB_ID" type="xs:string"
use="required" />
        <xs:attribute name="R-C" type="xs:string"
use="required" />
    </xs:complexType>
</xs:element>
</xs:sequence>
</xs:complexType>

```

```

<!-- Elsődleges kulcsok -->
<xs:unique name="szem_id">
  <xs:selector xpath="szemelyzet" />
  <xs:field xpath="@SZEM_ID" />
</xs:unique>
<xs:unique name="alk_id">
  <xs:selector xpath="alkalmazott" />
  <xs:field xpath="@ALK_ID" />
</xs:unique>
<xs:unique name="be_id">
  <xs:selector xpath="beosztas" />
  <xs:field xpath="@BE_ID" />
</xs:unique>
<xs:unique name="bor_id">
  <xs:selector xpath="borton_epulet" />
  <xs:field xpath="@BOR_ID" />
</xs:unique>
<xs:unique name="cel_id">
  <xs:selector xpath="cellak" />
  <xs:field xpath="@CEL_ID" />
</xs:unique>
<xs:unique name="rab_id">
  <xs:selector xpath="rab" />
  <xs:field xpath="@RAB_ID" />
</xs:unique>

<!-- Idegen kulcsok -->
<xs:keyref refer="alk_id"
name="alkalmazott_szemelyzet_idegen_kulcs">
  <xs:selector xpath="szemelyzet" />
  <xs:field xpath="@SZ-A" />
</xs:keyref>
<xs:keyref refer="alk_id" name="alkalmazott_idegen_kulcs">
  <xs:selector xpath="beosztas" />
  <xs:field xpath="@BE-A" />
</xs:keyref>
<xs:keyref refer="bor_id" name="borton_idegen_kulcs">
  <xs:selector xpath="beosztas" />
  <xs:field xpath="@BE-B" />
</xs:keyref>
<xs:keyref refer="bor_id"
name="borton_cellai_idegen_kulcs">
  <xs:selector xpath="cellak" />
  <xs:field xpath="@C-B" />

```

```

</xs:keyref>
<xs:keyref refer="cel_id" name="cella_rab_idegen_kulcs">
  <xs:selector xpath="rab" />
  <xs:field xpath="@R-C" />
</xs:keyref>
</xs:element>

</xs:schema>

```

2.3 Java XML DOM olvasó program

Az alábbi program kiolvassa az XML dokumentumból az adatokat, majd megjeleníti őket a konzolban. A dinamikus természet miatt a különböző egyedeket külső string tömbökben definiáltam, ezeket kedvünkre szerkeztethetjük attól függően, mit szeretnénk lekérdezni.

```

package hu.domparsing.y86i0i;

import java.io.*;

import javax.xml.parsers.DocumentBuilderFactory;
import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.ParserConfigurationException;
import org.xml.sax.SAXException;

import org.w3c.dom.Document;
import org.w3c.dom.NodeList;
import org.w3c.dom.Node;
import org.w3c.dom.Element;

public class DomReadY86I0I {
    public static void main(String argv[]) throws SAXException,
    IOException, ParserConfigurationException {
        // XML File opening
        String[] személyzet_fields = {
            "pozicio",
            "szolgalmati_hely",
            "csaladi_allapot"
        };
        String[] alkalmazott_fields = {
            "eletkor",

```

```

        "nev",
        "lakhely"
    };
    String[] beosztas_fields = {
        "muszak"
    };
    String[] borton_epulet_fields = {
        "cellak_szama",
        "kapacitas",
        "epulet_neve",
        "epites"
    };
    String[] cellak_fields = {
        "cella_kapacitas",
        "emelet",
        "ablak"
    };
    String[] rab_fields = {
        "nev",
        "eletkor",
        "letoltendo"
    };
    String[][] fields = {
        szemelyzet_fields,
        alkalmazott_fields,
        beosztas_fields,
        borton_epulet_fields,
        cellak_fields,
        rab_fields
    };
    String[] sub_roots = {
        "szemelyzet",
        "alkalmazott",
        "beosztas",
        "borton_epulet",
        "cellak",
        "rab"
    };
    String[] id_list = {
        "SZEM_ID",
        "ALK_ID",
        "BE_ID",
        "BOR_ID",
        "CEL_ID",

```

```

        "RAB_ID"
    };
    File xmlFile = new File("XMLY86I0I.xml");

    DocumentBuilderFactory factory =
DocumentBuilderFactory.newInstance();

    DocumentBuilder dBuilder = factory.newDocumentBuilder();

    Document doc = dBuilder.parse(xmlFile);

    doc.getDocumentElement().normalize();

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

    int index = 0;
    for (String element : sub_roots) {
        NodeList nList = doc.getElementsByTagName(element);
        for (int i = 0; i < nList.getLength(); i++) {
            Node nNode = nList.item(i);

            System.out.println("\nCurrent Element: " +
nNode.getNodeName());

            if (nNode.getNodeType() == Node.ELEMENT_NODE) {
                Element elem = (Element) nNode;
                String uid = elem.getAttribute(id_list[index]);
                System.out.println(id_list[index] + ": " + uid);
                for (String field : fields[index]) {
                    Node node =
elem.getElementsByTagName(field).item(0);
                    String data = node.getTextContent();
                    System.out.println(field + ": " + data);
                }
            }
        }
        index++;
    }
}
}

```

2.4 Java XML Lekérdező program

Az alábbi program különböző kritériumok alapján kérdez le információkat az XML programból, attribútumok és mezőértékek alapján. Az alábbi lekérdezések valósultak meg:

- Azon börtön épületek kiírása, amelyek cella száma nagyobb mint 45
- Olyan rabok lekérdezése, akik az "A" id-jű épületben vannak
- Azon cellák, amelyek kapacitása 2-nél több
- A 30 évnél fiatalabb alkalmazottak listája
- Azon emberek szakmai információi, akik "Házas" családi állapottal rendelkeznek

```
package hu.dompars.y86i0i;
import java.io.*;

import javax.xml.parsers.DocumentBuilderFactory;
import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.ParserConfigurationException;
import org.xml.sax.SAXException;

import org.w3c.dom.Document;
import org.w3c.dom.NodeList;
import org.w3c.dom.Node;
import org.w3c.dom.Element;
public class DomQueryY86I0I {
    public static void main(String argv[]) throws SAXException,
    IOException, ParserConfigurationException {

        String[] borton_epulet_fields = {
            "cellak_szama",
            "kapacitas",
            "epulet_neve",
            "epites"
        };

        String[] rab_fields = {
            "nev",
            "eletkor",
            "letoltendo"
        };

        String[] cellak_fields = {
            "cella_kapacitas",
            "emelet",
            "ablak"
        }
    }
}
```

```

        };
        String[] alkalmazott_fields = {
            "eletkor",
            "nev",
            "lakhely"
        };
        String[] személyzet_fields = {
            "pozicio",
            "szolgalmati_hely",
            "csaladi_allapot"
        };
        File xmlFile = new File("XMLY86I0I.xml");

        DocumentBuilderFactory factory =
DocumentBuilderFactory.newInstance();

        DocumentBuilder dBuilder = factory.newDocumentBuilder();

        //Át parse-oljuk az XML file-unkat
        Document doc = dBuilder.parse(xmlFile);

        doc.getDocumentElement().normalize();
        System.out.println("---Ha a cellak szama <= 45 akkor
annak a bortonnek nem irjuk ki az adatait---");
        //A borton épület mezőkből kérdezzük le, megkötések
alapján
        NodeList nList =
doc.getElementsByTagName("borton_epulet");
        for (int i=0; i<nList.getLength(); i++){
            Node nNode = nList.item(i);

            if (nNode.getNodeType() == Node.ELEMENT_NODE)
{
                Element elem = (Element) nNode;
                String uid = elem.getAttribute("BOR_ID");
                //Az A ID-vel rendelkező borton adatai
nem kerülnek kiírásra
                if (uid.equals("A"))
                    continue;
                System.out.println("BOR_ID" + ": " +
uid);

                for(String field : borton_epulet_fields){
                    Node node =
elem.getElementsByTagName(field).item(0);
                    String data =
node.getTextContent();

                    try {

```

```

        Integer.parseInt(data);

        int d =
        //Nem írjuk ki a cellák
        számát, ha mennyiségük <=45

        if (d > 45)
            System.out.println(field
+ ": " + data);

        } catch (NumberFormatException ex){
            System.out.println(field + ":
" + data);
        }
    }
}

System.out.println();
System.out.println("----Lekerdezzük a Rab mezobol
minden rabot, aki az A epuletben van----");
//Lekérdezünk a Rab mezőből minden rabot, aki az A
épületben van

nList = doc.getElementsByTagName("rab");
for (int i=0; i<nList.getLength(); i++){
    Node nNode = nList.item(i);
    if (nNode.getNodeType() == Node.ELEMENT_NODE)
    {
        Element elem = (Element) nNode;
        String uid = elem.getAttribute("RAB_ID");
        String connectid =
elem.getAttribute("R-C");

        if (connectid.contains("a") == false)
            continue;
        System.out.println("RAB_ID" + ": " +
uid);

        System.out.println("R-C" + ": " +
connectid);

        for(String field : rab_fields){
            Node node =
elem.getElementsByTagName(field).item(0);
            String data =
node.getTextContent();

            System.out.println(field + ": " +
data);
        }
    }
}

System.out.println("----Minden cella, aminek a
kapacitasa tobb, mint 2----");
nList = doc.getElementsByTagName("cellak");
for (int i=0; i<nList.getLength(); i++){

```



```

        Node nNode = nList.item(i);
        if (nNode.getNodeType() == Node.ELEMENT_NODE)
        {
            Element elem = (Element) nNode;
            Node node =
elem.getElementsByTagName("cella_kapacitas").item(0);
            String kapacitas = node.getTextContent();
            int kap = Integer.parseInt(kapacitas);
            String uid = elem.getAttribute("CEL_ID");
            System.out.println("CEL_ID" + ": " +
uid);

            if (kap > 2)
                for(String field : cellak_fields){
                    node =
elem.getElementsByTagName(field).item(0);
                    String data =
node.getTextContent();

                    System.out.println(field + ": " +
data);
                }
        }

        System.out.println();
        System.out.println("---Lekerdezunk azon
alkalmazottak nevét, akik 30 évnel fiatalabbak---");
        nList = doc.getElementsByTagName("alkalmazott");
        for (int i=0; i<nList.getLength(); i++){
            Node nNode = nList.item(i);
            if (nNode.getNodeType() == Node.ELEMENT_NODE)
            {
                Element elem = (Element) nNode;
                Node node =
elem.getElementsByTagName("eletkor").item(0);
                String eletkor = node.getTextContent();
                int ev = Integer.parseInt(eletkor);
                if (ev < 30)
                    for(String field :
alkalmazott_fields){
                        node =
elem.getElementsByTagName(field).item(0);
                        String data =
node.getTextContent();

                        System.out.println(field + ":
" + data);
                    }
            }
        }

```

```

        System.out.println();
        System.out.println("---Irjuk ki a hazas csaladi
allapotu emberek szakmai információit---");
        nList = doc.getElementsByTagName("szemelyzet");
        for (int i=0; i<nList.getLength(); i++){
            Node nNode = nList.item(i);
            if (nNode.getNodeType() == Node.ELEMENT_NODE)
            {
                Element elem = (Element) nNode;
                Node node =
elem.getElementsByTagName("csaladi_allapot").item(0);
                String csaladi_allapot =
node.getTextContent();
                if(csaladi_allapot.equals("Házás"))
                    for(String field :
szemelyzet_fields){
                        node =
elem.getElementsByTagName(field).item(0);
                        String data =
node.getTextContent();
                        System.out.println(field + ":
" + data);
                    }
            }
        }
    }
}

```

2.5 Java XML Módosító program

A következő program megváltoztat egyes információkat az XML dokumentumban. Az alábbi változtatásokat hajtottam végre:

- Minden rabnak változzon az elsődleges kulcsának felépítése
- “Gengszter Lajos” és “Gonosz Gergely” új neveket kapnak
- Töröljük ki az összes börtön épület kapacitását
- Ha a lakóhely irányítószáma 3529 akkor változzon 3000-re, ha 1032 akkor változzon 1000-re
- Adjunk hozzá a cellákhoz egy bizonyos “van_wc” mezőt, amely különböző épületek celláiban mondja meg, hogy van-e wc vagy nincs

```

package hu.domparsed.y86i0i;

import javax.xml.transform.*;
import javax.xml.transform.dom.DOMSource;
import javax.xml.transform.stream.StreamResult;

import java.io.File;
import java.io.IOException;
import java.security.KeyStore.Builder;

import javax.xml.*;
import javax.xml.parsers.*;
import org.xml.sax.SAXException;

import org.w3c.dom.Document;
import org.w3c.dom.NodeList;
import org.w3c.dom.Node;
import org.w3c.dom.Element;
import org.w3c.dom.NamedNodeMap;

public class DOMModifyY86I0I {
    public static void main(String argv[])
        throws ParserConfigurationException,
TransformerException, IOException,
TransformerConfigurationException {
        try {
            File inputFile = new File("XMLY86I0I.xml");

            DocumentBuilderFactory docFactory =
DocumentBuilderFactory.newInstance();
            DocumentBuilder docBuilder =
docFactory.newDocumentBuilder();

            Document doc = docBuilder.parse(inputFile);
            NodeList rabList =
doc.getElementsByTagName("rab");
            // Átmegyünk az összes Rab-on
            for (int i = 0; i < rabList.getLength(); i++) {

```

```

        // Node változtatás
        Node rab =
doc.getElementsByTagName("rab").item(i);
        // Attributomok lementése
        NamedNodeMap attr = rab.getAttributes();
        Node nodeAttr = attr.getNamedItem("RAB_ID");
        // ID Változtatás
        nodeAttr.setTextContent("RAB_" + (i));

        NodeList list = rab.getChildNodes();
        for (int t = 0; t < list.getLength(); t++) {
            Node node = list.item(t);
            if (node.getNodeType() ==
Node.ELEMENT_NODE) {
                Element eElement = (Element) node;
                if ("nev".equals(eElement.getNodeName()))
{
                    // Ha a név egyenlő ezzel
                    if ("Gengszter
Lajos".equals(eElement.getTextContent())) {
                        // Változtassa meg erre
                        eElement.setTextContent("Gengszter
Lajos Junior (GL)");
                    }
                    if ("Gonosz
Gergő".equals(eElement.getTextContent())) {
                        eElement.setTextContent("Már Nem
Gonosz Gergő");
                    }
                }
            }
        }
        NodeList bortonList =
doc.getElementsByTagName("borton_epulet");
        for (int i = 0; i < bortonList.getLength();
i++) {

```

```

        Node borton =
doc.getElementsByTagName("epulet_neve").item(i);
        NodeList childNodes = borton.getChildNodes();
        for (int count = 0; count <
childNodes.getLength(); count++) {
            Node node = childNodes.item(count);
            // Az összes börtön épületnek kitöröljük a
kapacitását
            if ("kapacitas".equals(node.getNodeName()))
{
                borton.removeChild(node);
            }
        }
    }
}

```

```

        NodeList lakhelyList =
doc.getElementsByTagName("lakhely");
        for (int i = 0; i < lakhelyList.getLength();
i++) {
            Node lakhely =
doc.getElementsByTagName("lakhely").item(i);
            NodeList childNodes =
lakhely.getChildNodes();
            for (int t = 0; t < childNodes.getLength();
t++) {
                Node node = childNodes.item(t);
                if (node.getNodeType() ==
Node.ELEMENT_NODE) {
                    Element eElement = (Element) node;
                    if
("iranyitoszam".equals(eElement.getNodeName())) {
                        if
("3529".equals(eElement.getTextContent())) {
                            eElement.setTextContent("3000");
                        }
                        if
("1032".equals(eElement.getTextContent())) {
                            eElement.setTextContent("1000");
                        }
                    }
                }
            }
        }
    }
}

```

```

        }
    }
}

NodeList cellakList =
doc.getElementsByTagName("cellak");
    for (int i = 0; i < cellakList.getLength();
i++) {
        Node cella = cellakList.item(i);

        String id =
cella.getAttribute().getNamedItem("CEL_ID").getTextC
ontent();

        Element wc = doc.createElement("van_wc");
        cella.appendChild(wc);

        if (id.contains("a")) {
            wc.appendChild(doc.createTextNode("Van"));
        }
        if (id.contains("b")) {

wc.appendChild(doc.createTextNode("Nincs"));
        }
        if (id.contains("c")) {
            wc.appendChild(doc.createTextNode("Van"));
        }
    }

    TransformerFactory transformerFactory =
TransformerFactory.newInstance();
    Transformer transformer =
transformerFactory.newTransformer();

    DOMSource source = new DOMSource(doc);

```

```
        System.out.println("--Results--");
        StreamResult consoleResult = new
StreamResult(System.out);
        transformer.transform(source, consoleResult);
    } catch (Exception e) {
        e.printStackTrace();
    }

}

}
```