JEGYZŐKÖNYV

Adatkezelés XML-ben

Féléves feladat

Iskola

Készítette: **Gerőcs Gergő**

Neptunkód: **FEU2E5**

Dátum: **2024.11.21**

Tartalomjegyzék

Bevezetés

A feladat leírása

Az iskola című feladattal egy kisebb magán általános iskola adatainak a nyilvántartását végzem el XML dokumentumban. Az iskolába diákok járnak, akiknek van nevük, IQ-juk, születési dátumuk, illetve nemük, a nemük lány vagy fiú lehet, továbbá a gyermekek IQ szintje egy 60-120 közötti szám.

Minden diákhoz egyértelműen tartozik egy tagság adatlap, amelyben eltároljuk a következő adatokat: A tagság kezdetét, ekkor iratták be a diákot az iskolába, hogy aktív-e jelenleg a tagsága, illetve, hogy részesül-e valamilyen kedvezményben a gyermek. kedvezmény jár például, ha a gyermek árva.

Tárolásra kerülnek az egyik szülő adatai is, fontos kiemelni, hogy csak az egyik szülő adatai kerülnek eltárolásra. Ezen adatok közé tartozik, a szülő neve, születési éve, illetve elérhetősége, amely telefon és E-mail cím, melyeken értesíteni tudja az iskola a szülőt baj esetén, illetve E-mailben tájékoztatókat, és fontos híreket/információt tud eljuttatni az iskola a szülőhöz, egy szülő , minimum 1, maximum 5db E-mail címmel rendelkezhet.

A nyilvántartásban a tanárok is szerepet kapnak, ők tartják az órákat. A tanárok nevét, szakterületét, órabérét, illetve szintén az elérhetőségüket tároljuk. Az órabér egy 2000-10000 közötti szám, mely Ft-ban értendő, továbbá egy tanár minimum 1, maximum 5db E-mail címmel rendelkezhet.

Nyilvántartjuk a gyermekek számára megtartott órákat is. Minden órához tartozik egy vagy több téma, illetve egy helyszín, amely lehet például tanterem, udvar, tornaterem. Tartozik az órához egy időpont, amely tartalmazza az adott napot, az óra kezdetét és végét, minden óra 8 óra után kezdődik és 15 óra előtt ér véget.

Az órákon jól teljesítő tanulók kaphatnak jutalmakat, amelyeket szintén eltárolunk az XML dokumentumban, illetve a tanár az óra végén értékeli a gyermekek, órán való teljesítményét egy 1-től 5-ig tartó skálán.

1. XML adatkezelő rendszer

1.1 Az adatbázis ER modell tervezése

Az ER modellben 5 db egyed található, ezek az alábbiak: Tanár, Óra, Diák, Szülő, Tagság. Minden egyed kulccsal rendelkezik, ezt a kulcsot aláhúzás jelöli. A kulcsok elnevezései az *egyednév kezdőbetű + kód* szabály alapján jöttek létre, kivéve a Tanár és Tagság esetén, ahol a szabály szerint mindkét egyednek Tkód kellene, hogy legyen a kulcsa, ezt a problémát megoldva a Tanár egyed kulcsát *Tankód*-nak, a Tagság egyed kulcsát *Tagkód*-nak neveztem el.

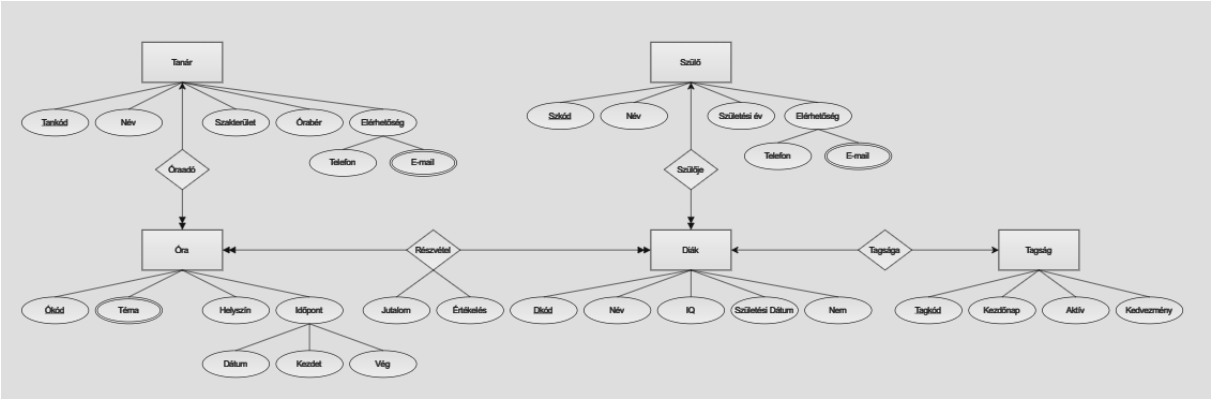
A Diák és Tagság között 1:1, Diák és Óra között N:M, Diák és Szülő, valamint Tanár és Óra között 1:N kapcsolat van. Ez azt jelenti, hogy egy diáknak pontosan egy tagsága lehet, és egy tagsághoz pontosan egy diák tartozik. Továbbá egy órán több diák is részt vehet, és egy diáknak több órája is lehet. Egy diáknak jelenleg egy szülője lehet(a korábbi leírásban leírtak miatt), de egy szülőnek több gyermeke, azaz diákja is lehet. Egy órát, csak egy tanár tarthat, de egy tanárnak több órája lehet.

Az ER modellben több összetett tulajdonság szerepel, ilyen a szülő és tanár esetén az elérhetőség, illetve az órák időpontja. Vannak többértékű tulajdonságok is, például az E-mail, amelyből egy egyednek több is lehet, továbbá az órák témái. Az óra-diák kapcsolat tulajdonságai a jutalom, amely megadja, hogy az adott diák az adott órán milyen jutalmat kapott, továbbá értékelés, ami az órát tartó tanár által adott pontszám a diák órai teljesítménye.

Egyedek és tulajdonságaik:

- Tanár
    - Tankód: egyedi azonosító, numerikus
    - Név: a tanár teljes neve
    - Szakterület: a tanár szakterülete
    - Órabér: a tanárnak fizetett órabér
    - Elérhetőség: a tanár telefonszáma és e-mail címei
- Diák
    - Dkód: egyedi azonosító, numerikus
    - Név: diák teljes neve
    - IQ: diák IQ szintje, 60-120 közötti szám
    - Születési dátum: a diák születési dátuma
    - Nem: diák neme, lány vagy fiú lehet

- Szülő
    - Szkód: egyedi azonosító, numerikus
    - Név: a szülő teljes neve
    - Születési év: a szülő születési éve
    - Elérhetőség: a szülő telefonszáma és e-mail címei
- Óra
    - Ókód: egyedi azonosító, numerikus
    - Téma: az óra témájának rövid szöveges leírása
    - Helyszín: az óra helyszíne, például tornaterem
    - Időpont: az óra dátuma, kezdete és vége (8:00 és 15:00 közötti egész órák)
- Tagság
    - Tagkód: egyedi azonosító, numerikus
    - Kezdőnap: a beiratkozás dátuma
    - Aktív: aktív-e a diák tagsága, igen vagy nem lehet
    - Kedvezmény: normál vagy kedvezményes lehet
- Részvétel kapcsolat
    - Jutalom: az órán kapott jutalom rövid szöveges leírása
    - Értékelés: a diák órai teljesítményének értékelése 1-5-ig



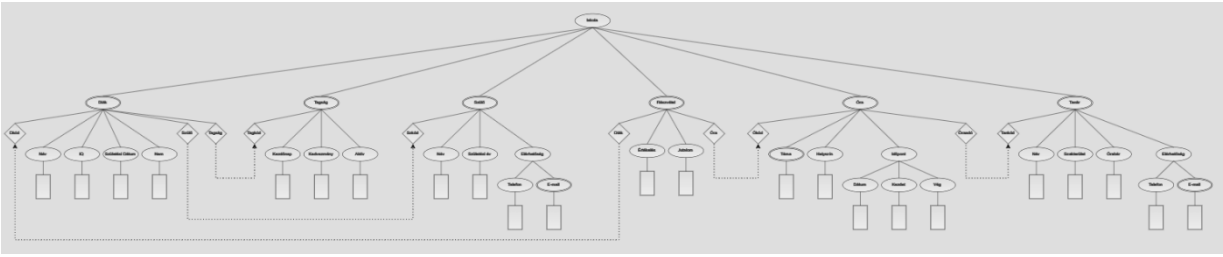
1. ábra

1.2 Az adatbázis konvertálása XDM modellre

Az adatbázis XDM modellje az ER modell alapján került megvalósításra. A gyökérelem neve iskola lett, melynek 6 db többpéldányos gyerekeleme van. Minden gyerekelemnek van kulcs attribútuma, melyeket az ER modellben szereplő kulcs nevekkel láttam el.

A diák-tagság 1:1 kapcsolat jelölésére a diák elemnek lett egy tagság nevű idegenkulcs attribútuma, ami a tagság elem elsődleges kulcsára(tagkód) mutat. A diák-óra N:M kapcsolatot egy részvétel elnevezésű új elem jelöli, melyben található a diák és óra idegenkulcs is. A diák-szülő 1:N kapcsolat ábrázolására a diák elemnek lett egy szülő idegenkulcs attribútuma, ami a szülő elem szkód kulcsértékére mutat. A tanár-óra 1:N kapcsolat reprezentálása is hasonlóan történt, ebben az esetben az órához került óraadó nevű, tanár értékre mutató idegenkulcs.

Többpéldányos elemek lettek azok az elemek is, melyek az ER modellben többértékű tulajdonságok voltak: az e-mail cím és a téma. A hierarchiában legalul elhelyezkedő gyerekelemeknek szövegtartalma van, ezt téglalapok jelölik.



2. ábra

1.3 Az XDM modell alapján XML dokumentum készítése

Az XML dokumentumot az XDM modell alapján készítettem el. A gyökérelem így az iskola lett, a főbb egyedek pedig ennek többpéldányos gyermekelemei, melyekből típusonként legalább 3 különbözőt létrehoztam. Felvettem olyan szülőket és tanárokat, akiknek egynél több e-mail címük van, illetve olyan órákat, amiken több téma is előkerült. Két diák részvételénél nem adtam meg jutalmat, mivel ez egy opcionális gyerekelem.

Az XML dokumentum tartalma:

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

<iskola xmlns:xs="http://www.w3.org/2001/XMLSchema-instance"
 xs:noNamespaceSchemaLocation="XMLSchemaFeu2e5.xsd">

<!--Diákok-->
<diák Dkód="10" szülő="100" tagság="1000">
   <név>Kiss Attila</név>
   <IQ>90</IQ>
   <születési_dátum>2016-12-17</születési_dátum>
   <nem>fiú</nem>
</diák>
<diák Dkód="11" szülő="110" tagság="1001">
   <név>Nagy Mária</név>
   <IQ>65</IQ>
   <születési_dátum>2016-10-15</születési_dátum>
   <nem>lány</nem>
</diák>
<diák Dkód="12" szülő="120" tagság="1002">
   <név>Szép Tamás</név>
   <IQ>110</IQ>
   <születési_dátum>2016-06-06</születési_dátum>
   <nem>fiú</nem>
</diák>
<diák Dkód="13" szülő="120" tagság="1003">
   <név>Szép Anita</név>
   <IQ>115</IQ>
   <születési_dátum>2016-01-05</születési_dátum>
   <nem>lány</nem>
</diák>

<!--tagság adatok-->
<tagság Tagkód="1000">
   <kezdőnap>2016-09-01</kezdőnap>
   <kedvezmény>normál</kedvezmény>
   <aktív>igen</aktív>
</tagság>
<tagság Tagkód="1001">
```

```
    <kezdőnap>2015-09-01</kezdőnap>
    <kedvezmény>kedvezményes</kedvezmény>
    <aktív>igen</aktív>
</tagság>
<tagság Tagkód="1002">
    <kezdőnap>2013-09-01</kezdőnap>
    <kedvezmény>normál</kedvezmény>
    <aktív>igen</aktív>
</tagság>
<tagság Tagkód="1003">
    <kezdőnap>2013-09-01</kezdőnap>
    <kedvezmény>kedvezményes</kedvezmény>
    <aktív>nem</aktív>
</tagság>

<!--szülők-->
<szülő Szkód="100">
    <név>Kiss Béla</név>
    <születési_év>1985</születési_év>
    <elérhetőség>
        <telefonszám>+36506060768</telefonszám>
        <email>belakiss@gmail.com</email>
        <email>kissbela@citromail.hu</email>
    </elérhetőség>
</szülő>
<szülő Szkód="110">
    <név>Nagy Adrienn</név>
    <születési_év>1991</születési_év>
    <elérhetőség>
        <telefonszám>+36701234987</telefonszám>
        <email>n.adrienn@gmail.com</email>
    </elérhetőség>
</szülő>
<szülő Szkód="120">
    <név>Szép János</név>
    <születési_év>1990</születési_év>
    <elérhetőség>
        <telefonszám>+36208877665</telefonszám>
        <email>szepjancsi@szepmail.hu</email>
        <email>jancsi.szep@gmail.com</email>
        <email>jancsi.szep@freemail.com</email>
    </elérhetőség>
</szülő>

<!--Órák-->
<óra Ókód="1200" tanár="1100">
    <téma>Matematika alapjai</téma>
    <helyszín>tanterem 101.</helyszín>
    <időpont>
        <dátum>2024-10-30</dátum>
```

```
      <kezdet>13</kezdet>
      <vég>14</vég>
   </időpont>
</óra>
<óra Ókód="1201" tanár="1100">
   <téma>osztás</téma>
   <téma>szorzás</téma>
   <helyszín>tanterem 2.</helyszín>
   <időpont>
      <dátum>2024-10-15</dátum>
      <kezdet>9</kezdet>
      <vég>10</vég>
   </időpont>
</óra>
<óra Ókód="1202" tanár="1101">
   <téma>Kidobós</téma>
   <téma>cooper teszt</téma>
   <helyszín>tornaterem</helyszín>
   <időpont>
      <dátum>2024-09-02</dátum>
      <kezdet>12</kezdet>
      <vég>13</vég>
   </időpont>
</óra>
<óra Ókód="1203" tanár="1102">
   <téma>fogócska</téma>
   <helyszín>udvar</helyszín>
   <időpont>
      <dátum>2024-11-25</dátum>
      <kezdet>11</kezdet>
      <vég>12</vég>
   </időpont>
</óra>

<!--Részvételi adatok-->
<részvétel diák="10" óra="1200">
   <értékelés>3</értékelés>
</részvétel>
<részvétel diák="10" óra="1201">
   <értékelés>4</értékelés>
   <jutalom>pirospont</jutalom>
</részvétel>
<részvétel diák="11" óra="1201">
   <értékelés>5</értékelés>
   <jutalom>matrica</jutalom>
</részvétel>
<részvétel diák="12" óra="1200">
   <értékelés>1</értékelés>
   <jutalom>feketepont</jutalom>
</részvétel>
```

```
<részvétel diák="12" óra="1202">
   <értékelés>3</értékelés>
</részvétel>
<részvétel diák="13" óra="1203">
   <értékelés>5</értékelés>
   <jutalom>pirospont</jutalom>
</részvétel>

<!--tanárok-->
<tanár Tankód="1100">
   <név>Takács Péter</név>
   <szakterület>Matematika</szakterület>
   <órabér>5000</órabér>
   <elérhetőség>
      <telefonszám>+36702525625</telefonszám>
      <email>t.p@gmail.com</email>
   </elérhetőség>
</tanár>
<tanár Tankód="1101">
   <név>Kovács Zoltán</név>
   <szakterület>Testnevelés</szakterület>
   <órabér>2500</órabér>
   <elérhetőség>
      <telefonszám>+36301233456</telefonszám>
      <email>kovizol@gmail.com</email>
      <email>kovacs.z@freemail.hu</email>
   </elérhetőség>
</tanár>
<tanár Tankód="1102">
   <név>Lédig Anna</név>
   <szakterület>Viselkedés Pszichológia</szakterület>
   <órabér>9500</órabér>
   <elérhetőség>
      <telefonszám>+36708996574</telefonszám>
      <email>ledig@gmail.com</email>
   </elérhetőség>
</tanár>


 </iskola>
```

1.4 Az XML dokumentum alapján XMLSchema készítése – saját típusok, ref, key, keyref, speciális elemek

Az XML dokumentum validálására XMLSchema sémaleíró dokumentumot hoztam létre.

A név és az elérhetőség elemeket referenciával illesztettem be a megfelelő helyekre. Létrehoztam több saját egyszerű típust, például a diákok nemének, a kedvezmény típusának megadásához. Az óra típusnál minimum és maximum értékeket jelöltem ki, figyelembe véve azt, hogy az órák csak 8 és 15 óra között lehetnek megtartva. Az értékelés típus lehetséges értékeit szintén korlátoztam, hogy azok 1 és 5 közé essenek. Az e-mail címekre és a telefonszámokra vonatkozó korlátozások megadásához egyszerű reguláris kifejezéseket használtam. Telefonszámok esetén a +36-tal és a 06-tal kezdődő formátum is megengedett.

Ezt követően komplex típusokat hoztam létre, majd ezeket felhasználva megadtam az iskola gyökérelem szerkezetét az elsődleges kulcsokkal és idegen kulcsokkal együtt. A tagság-diák 1:1 kapcsolathoz unique megszorítást vezettem be.

Az így létrejött séma alapján az XML dokumentumot Visual Studio Code fejlesztőkörnyezetben sikeresen validáltam.

Az XMLSchema dokumentum tartalma:

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

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

    <!--Elemek ref-hez-->
  <xs:element name="név" type="xs:string" />
  <xs:element name="elérhetőség" type="elérhetőségTípus" />

  <!-- Saját egyszerű típusok meghatározása, megszorítás -->
  <xs:simpleType name="nemTípus">
    <xs:restriction base="xs:string">
      <xs:enumeration value="lány" />
      <xs:enumeration value="fiú" />
    </xs:restriction>
  </xs:simpleType>
  <xs:simpleType name="kedvezményTípus">
    <xs:restriction base="xs:string">
      <xs:enumeration value="normál" />
      <xs:enumeration value="kedvezményes" />
    </xs:restriction>
  </xs:simpleType>
  <xs:simpleType name="aktívTípus">
    <xs:restriction base="xs:string">
```

```
      <xs:enumeration value="igen" />
      <xs:enumeration value="nem" />
    </xs:restriction>
  </xs:simpleType>
  <xs:simpleType name="óraIdőTípus">
    <xs:restriction base="xs:nonNegativeInteger">
      <xs:minExclusive value="8" />
      <xs:maxExclusive value="15" />
    </xs:restriction>
  </xs:simpleType>
  <xs:simpleType name="emailTípus">
    <xs:restriction base="xs:string">
      <xs:pattern value="[a-z]([a-z0-9_.-]*)@[a-z]([a-z0-9_-]*).[a-z]{2,4}" />
    </xs:restriction>
  </xs:simpleType>
  <xs:simpleType name="telefonTípus">
    <xs:restriction base="xs:string">
      <xs:pattern value="([+]?)\d{11}" />
    </xs:restriction>
  </xs:simpleType>
  <xs:simpleType name="értékelésTípus">
    <xs:restriction base="xs:int">
      <xs:minInclusive value="1"/>
      <xs:maxInclusive value="5"/>
    </xs:restriction>
  </xs:simpleType>

  <!--Komplex típusokhoz saját típus meghatározása, sorrendiség, számosság etc. -->
  <xs:complexType name="elérhetőségTípus">
    <xs:sequence>
      <xs:element name="telefonszám" type="telefonTípus" />
      <xs:element name="email" type="emailTípus" minOccurs="1" maxOccurs="5" />
    </xs:sequence>
  </xs:complexType>
  <xs:complexType name="időpontTípus">
    <xs:sequence>
      <xs:element name="dátum" type="xs:date" />
      <xs:element name="kezdet" type="óraIdőTípus" />
      <xs:element name="vég" type="óraIdőTípus" />
    </xs:sequence>
  </xs:complexType>
  <xs:complexType name="diákTípus">
    <xs:sequence>
      <xs:element ref="név" />
      <xs:element name="IQ" type="xs:integer" />
      <xs:element name="születési_dátum" type="xs:date" />
      <xs:element name="nem" type="nemTípus" />
    </xs:sequence>
    <xs:attribute name="Dkód" type="xs:integer" use="required" />
    <xs:attribute name="szülő" type="xs:integer" use="required" />
```

```
      <xs:attribute name="tagság" type="xs:integer" use="required" />
   </xs:complexType>
   <xs:complexType name="tagságTípus">
      <xs:all>
         <xs:element name="kezdőnap" type="xs:date" />
         <xs:element name="kedvezmény" type="kedvezményTípus" />
         <xs:element name="aktív" type="aktívTípus" />
      </xs:all>
      <xs:attribute name="Tagkód" type="xs:integer" use="required" />
   </xs:complexType>
   <xs:complexType name="szülőTípus">
      <xs:sequence>
         <xs:element ref="név" />
         <xs:element name="születési_év" type="xs:gYear" />
         <xs:element ref="elérhetőség" />
      </xs:sequence>
      <xs:attribute name="Szkód" type="xs:integer" use="required" />
   </xs:complexType>
   <xs:complexType name="óraTípus">
      <xs:sequence>
         <xs:element name="téma" type="xs:string" minOccurs="1"
maxOccurs="unbounded"/>
         <xs:element name="helyszín" type="xs:string" />
         <xs:element name="időpont" type="időpontTípus" />
      </xs:sequence>
      <xs:attribute name="Ókód" type="xs:integer" use="required" />
      <xs:attribute name="tanár" type="xs:integer" use="required" />
   </xs:complexType>
   <xs:complexType name="részvételTípus">
      <xs:sequence>
         <xs:element name="értékelés" type="értékelésTípus"  />
         <xs:element name="jutalom" type="xs:string" minOccurs="0" maxOccurs="1" />
      </xs:sequence>
      <xs:attribute name="diák" type="xs:integer" use="required" />
      <xs:attribute name="óra" type="xs:integer" use="required" />
   </xs:complexType>
   <xs:complexType name="tanárTípus">
      <xs:sequence>
         <xs:element ref="név" />
         <xs:element name="szakterület" type="xs:string" minOccurs="0" maxOccurs="1" />
         <xs:element name="órabér" type="xs:positiveInteger" />
         <xs:element ref="elérhetőség"/>
      </xs:sequence>
      <xs:attribute name="Tankód" type="xs:integer" use="required" />
   </xs:complexType>

   <!-- Gyökérelemtől az elemek felhasználása -->
   <xs:element name="iskola">
      <xs:complexType>
         <xs:sequence>
```

```
            <xs:element name="diák" type="diákTípus" minOccurs="0" maxOccurs="100" />
            <xs:element name="tagság" type="tagságTípus" minOccurs="0" maxOccurs="100"
/>
            <xs:element name="szülő" type="szülőTípus" minOccurs="0" maxOccurs="100" />
            <xs:element name="óra" type="óraTípus" minOccurs="0" maxOccurs="unbounded"
/>
            <xs:element name="részvétel" type="részvételTípus" minOccurs="0"
maxOccurs="unbounded" />
            <xs:element name="tanár" type="tanárTípus" minOccurs="1" maxOccurs="100" />
         </xs:sequence>
      </xs:complexType>

      <!-- Elsődleges kulcsok -->
      <xs:key name="diák_kulcs">
         <xs:selector xpath="diák" />
         <xs:field xpath="@Dkód" />
      </xs:key>
      <xs:key name="tagság_kulcs">
         <xs:selector xpath="tagság" />
         <xs:field xpath="@Tagkód" />
      </xs:key>
      <xs:key name="szülő_kulcs">
         <xs:selector xpath="szülő" />
         <xs:field xpath="@Szkód" />
      </xs:key>
      <xs:key name="óra_kulcs">
         <xs:selector xpath="óra" />
         <xs:field xpath="@Ókód" />
      </xs:key>
      <xs:key name="tanár_kulcs">
         <xs:selector xpath="tanár" />
         <xs:field xpath="@Tankód" />
      </xs:key>

      <!-- Idegen kulcsok -->
      <xs:keyref name="diák_tagság_kulcs" refer="tagság_kulcs">
         <xs:selector xpath="diák" />
         <xs:field xpath="@tagság" />
      </xs:keyref>
      <xs:keyref name="diák_szülő_kulcs" refer="szülő_kulcs">
         <xs:selector xpath="diák" />
         <xs:field xpath="@szülő" />
      </xs:keyref>
      <xs:keyref name="óra_tanár_kulcs" refer="tanár_kulcs">
         <xs:selector xpath="óra" />
         <xs:field xpath="@tanár" />
      </xs:keyref>
      <xs:keyref name="részvétel_diák_kulcs" refer="diák_kulcs">
         <xs:selector xpath="részvétel" />
         <xs:field xpath="@diák" />
```

```
        </xs:keyref>
        <xs:keyref name="részvétel_óra_kulcs" refer="óra_kulcs">
           <xs:selector xpath="részvétel" />
           <xs:field xpath="@óra" />
        </xs:keyref>

        <!-- Az 1:1 kapcsolat megvalósítása -->
        <xs:unique name="diák_tagság_1_1">
           <xs:selector xpath="diák" />
           <xs:field xpath="@tagság" />
        </xs:unique>

    </xs:element>

</xs:schema>
```

2. DOM program

2.1 Adatolvasás

A DOMReadFeu2e5.java osztály beolvassa az iskola XML dokumentumot, majd kiírja a konzolra és egy fájlba (XMLReadFeu2e5.xml) blokk formában. Ennek az osztálynak a public metódusait hívja a többi osztály is, amikor kiírásra vagy beolvasásra van szükség. A fájlba íráshoz transformert használok, míg a konzolra íráskor lépésenként haladok végig a DOM fán, csomópontról csomópontra lépve.

Kiemelt kód:

```
//Adott nevű XML dokumentumból Document készítése

public static Document parseXML(String filename) throws SAXException, IOException,
ParserConfigurationException {

File xmlFile = new File(filename);

DocumentBuilderFactory factory = DocumentBuilderFactory.newInstance();

DocumentBuilder builder = factory.newDocumentBuilder();

Document document = builder.parse(xmlFile);

Node root = document.getDocumentElement();

root.normalize();

cleanDocument(root);

return document;

}
```

Magyarázat:

A metódus egy String típusú paramétert fogad, amely az XML fájl nevét tartalmazza, ezután létrehoz egy File objektumot az adott fájlnévvel. Ezt a fájlt később a dokumentum parser fogja feldolgozni. A builder.parse(xmlFile) megadott fájlt elemzi, majd egy DOM struktúrát hoz létre belőle. Az eredmény egy Document objektum, amely az XML teljes struktúráját képviseli, és amelyet további feldolgozásokra használhatunk. A root.normalize() művelet biztosítja, hogy a dokumentum szöveges tartalma szabványosított legyen. Például, ha vannak szöveges csomópontok, azokat egyesíti.

2.2 Adatírás

A DOMWriteFeu2e5.java osztály createIskola() metódusa felépíti az iskolához tartozó DOM fát, melyet aztán a DOMReadFeu2e5.java osztályt felhasználva ír ki a konzolra és az XML fájlba (XMLFeu25e51.xml). A különböző típusú elemek létrehozására paraméterezhető metódusokat írtam, például: createDiak(), createTagsag(), stb.

Kiemelt kód:

```
//Új diak elem készítése

private static Element createDiak(Document document, String diakId, String
szuloId, String tagsag, String nev, String IQ, String szuldatum, String nem)
throws ParseException {

Element diakElement = document.createElement("diák");

diakElement.setAttribute("Dkód", diakId);

diakElement.setAttribute("szülő", szuloId);

diakElement.setAttribute("tagság", tagsag);

diakElement.appendChild(createTextElement(document, "név", nev));

diakElement.appendChild(createTextElement(document, "IQ", IQ));

diakElement.appendChild(createTextElement(document, "születési_dátum",
szuldatum));

diakElement.appendChild(createTextElement(document, "nem", nem));

return diakElement;

}

//Új szöveges elem készítése

private static Element createTextElement(Document document, String tagName, String
textContent) {

Element element = document.createElement(tagName);

element.appendChild(document.createTextNode(textContent));

return element;

}

}
```

Magyarázat:

A createTextElement() metódus egyetlen név-érték párosból álló XML elemet hoz létre.

A createDiak() metódus egy összetettebb diák elemet hoz létre, amely attribútumokat és több gyerekelemet tartalmaz. Az gyerekelemek létrehozásához a createTextElement() metódust használja.

2.3 Adatlekérdezés

A DOMQueryFeu2e5.java osztály előre meghatározott lekérdezéseket hajt végre a beolvasott XML dokumentumra vonatkozóan, melyek eredményeit a konzolra írja ki.

Lekérdezések:

1. A 2016-ban vagy utána született diákok adatai
2. Az aktív tagságok száma
3. A tanárok órabéreinek átlaga
4. Az összes kiosztott jutalom típusa

Kiemelt kód:

```
//Adott évben/évet követően született diákok adatai strukturáltan

private static String getDiakNamesBornAfter(Document document, int year) {

String output = "";

Element root = document.getDocumentElement();

//Ciklus az összes diákra

NodeList diakok = root.getElementsByTagName("diák");

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

Element diak = (Element) diakok.item(i);

Element birthDateEl = (Element)
diak.getElementsByTagName("születési_dátum").item(0);

//A születési év kivétele a diák születési dátumából

int birthYear = Integer.parseInt(birthDateEl.getTextContent().split("-
")[0].trim());

//Ha a születési évnél nem nagyobb a paraméterként megadott év, akkor a diák
adatai a kimenetbe kerülnek

if (birthYear >= year) {

output+=DOMReadFeu2e5.formatElement(diak, 0);

}

}

return output;
```

```
}
```

Magyarázat:

A getDocumentElement() metódussal lekéri az XML gyökérelemét, a továbbiakban ezen belül keresi a diák elemeket. A getElementsByTagName("diák") lekéri az összes diák nevű elemet az XML-ben. A diákok száma az diakok.getLength() értékkel kérdezhető le.

Az iteráció során minden diák elemhez külön-külön végrehajtja a következő lépéseket. Lekéri az aktuális diák elem születési_dátum al-elemét. Lekéri az aktuális diák elem születési_dátum gyerekelemét. A birthDateEl.getTextContent(): Kinyeri a szöveges tartalmat (pl. 2005-04-15). A .split("-")[0] az első kötőjel alapján szétbontja a dátumot, és az év (pl. 2005) kerül kiválasztásra. Az Integer.parseInt(...) számértékké alakítja az év szöveges formáját.

Ha a birthYear nagyobb vagy egyenlő a megadott year értékkel, akkor az adott diák adatai hozzáadódnak a kimeneti szöveghez.

2.4 Adatmódosítás

A DOMModifyFeu2e5.java osztály előre meghatározott módosításokat hajt végre a beolvasott DOM fában, egyenként kiírja az előtte-utána állapotokat, és az eredményül kapott módosult DOM fát is kiírja a konzolra.

Végrehajtott módosítások:

1. A tanterem 2. helyszín átírása tanterem 102.-re
2. A Kiss Attila nevű diák tagságának passzívra állítása.
3. Kiss Béla átírása Kiss Andrásra és születési évének átírása 1980-ra
4. A 2. tanár 1. email címének törlése.

Kiemelt kód:

```
//Egy adott nevű szülő nevének és születési évének megváltoztatása

private static void modifyOwnerNameAndBirth(Document document, String oldName,
String newName, String newBirthYear) {

//Végigmegyünk az összes szülőn, akinek a neve oldName, azt newName-re
változtatjuk

Element root = document.getDocumentElement();

NodeList szulok = root.getElementsByTagName("szülő");

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

Element szulo = (Element)szulok.item(i);

Element name = (Element)szulo.getElementsByTagName("név").item(0);

if(name.getTextContent().equals(oldName)) {

//A név és a születési év átírása és előtte-utána adatok kiírása

System.out.println("\nELŐTTE:"+DOMReadFeu2e5.formatElement(szulo, 0));

szulo.insertBefore(document.createComment("Átírt név és születési év"), name);

name.setTextContent(newName);

Element birthYear = (Element)szulo.getElementsByTagName("születési_év").item(0);

birthYear.setTextContent(newBirthYear);

System.out.println("\nUTÁNA:"+DOMReadFeu2e5.formatElement(szulo, 0));

}
```

```
}
}
```

Magyarázat:

getDocumentElement(): Lekéri a dokumentum gyökérelemét.

getElementsByTagName("szülő"): Lekéri az összes szülőt a dokumentumból.

Végigmegy minden szülő elemen a szulok listából. Az adott szülő elem első név nevű gyerekelemét lekéri. Ellenőrzi, hogy a név gyerekelem szöveges tartalma (getTextContent()) megegyezik-e az oldName-mal. Ezután módosítja a nevet és születésiévet, illetve módosítás előtt és után is kiírja a konzolra