

JEGYZŐKÖNYV

Adatkezelés XML környezetben

Féléves feladat

Szalonok

Készítette: Gáspár Fanni

Neptunkód: VFX06G

Dátum: 2023.12.05

Tartalom

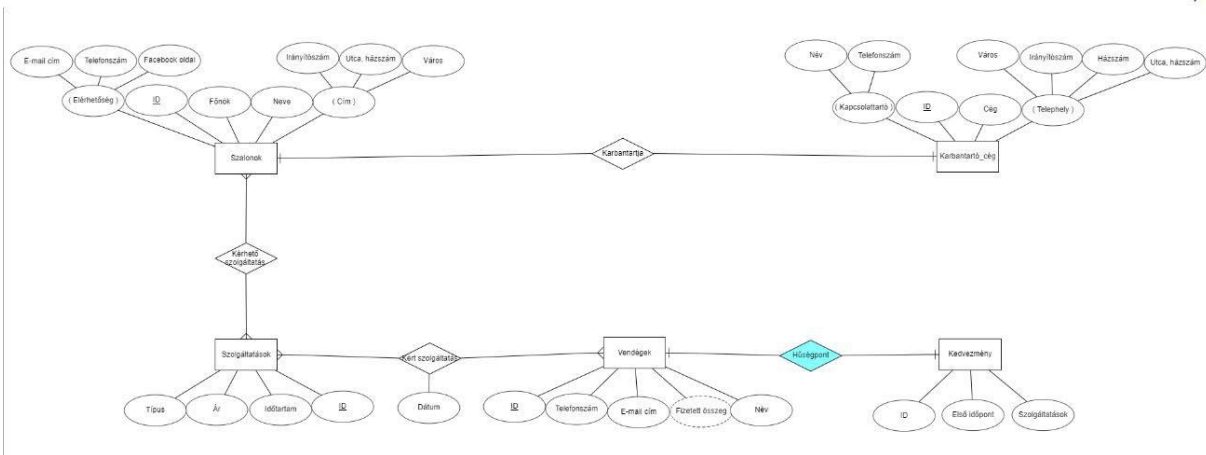
1.	Feladat leírása.....	3
2.	Az ER modell konvertálása XDM modellé.....	4
3.	XML dokumentum készítése.....	5
4.	XMLSchema készítése	12
5.	DOM adatolvasás	12
6.	DOM adatmódosítás	18
7.	DOM adatlekérdezés.....	28
8.	DOM adatírás	34

1. Feladat leírása

A beadandó témája egy olyan adatbázis, amely több cég által kezelt szalonok különböző adatait tartja számon. Lehetőség van egyes szolgáltatások, illetve vendégek, vagy akár az adott helyszín adatainak lekérdezésére.

- Karbantartó cég egyed tulajdonságai
 - o id: a cég kulcsa
 - o cég: a cég neve
 - o kapcsolattartó: összetett tulajdonság, mely a név és telefonszámból áll
 - o telephely: összetett tulajdonság, mely a város, irányítószám és utca, házszámból áll
- Szalon egyed tulajdonságai
 - o id: a szalon egyedi kulcsa
 - o főnök: a főnök neve
 - o név: a szalon neve
 - o elérhetőség: összetett tulajdonság, mely a facebook, telefonszám és email címből áll
 - o cím: összetett tulajdonság, mely a város, irányítószám és utca, házszámból áll
- Kedvezmény egyed tulajdonságai
 - o id: az kedvezmény egyed kulcsa
 - o elsőIdőpont: a kedvezmény megkapásának időpontja
 - o szolgáltatás: az szolgáltatás neve
- Szolgáltatás egyed tulajdonságai
 - o id: szolgáltatás egyedi azonosítója
 - o időtartam: milyen hosszú a kezelés
 - o ár: mennyibe kerül a szolgáltatás
 - o típus: a szolgáltatás típusa
- Vendég egyed tulajdonságai
 - o id: vendég kulcsa
 - o név: a vendég neve
 - o telefonszám: a vendég telefonszáma
 - o email-cím: a vendég email-címe
 - o fizetett összeg: opcionális mező, amelyben meg lehet adni a vendég által fizetett összeget

A feladat ER modellje:



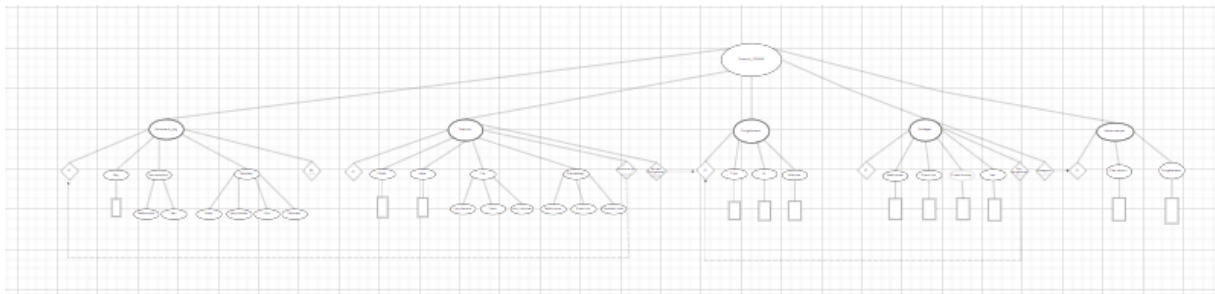
Az egyedek közötti kapcsolatok:

- Karbantartó cég és Szalon közötti kapcsolat: Karbantartja
 - o Egy-egy kapcsolat van közöttük, mivel egy szalont csak egy cég tarthat karban.
- Kedvezmény és Vendég közötti kapcsolat: Hűségpont
 - o Egy-egy kapcsolat van közöttük, mivel egy kedvezményt egy vendég fog megkapni.
- Szalon és Szolgáltatás közötti kapcsolat: Kérhető szolgáltatás
 - o Több-több kapcsolat van közöttük, mivel egy szalon több szolgáltatást is nyújthat, és ezeknek a szolgáltatásoknak a fajtája nem korlátozódhat le csak egy szalonra.
- Szolgáltatás és Vendég közötti kapcsolat: Kért szolgáltatás
 - o Több-több kapcsolat van közöttük, mivel egy vendég több szolgáltatást is kérhet, illetve egy szolgáltatásfajtát több vendég is kérhet.

2. Az ER modell konvertálása XDM modellé

Az XDM modell használatakor háromféle jelölést alkalmazunk: ellipszis, rombusz és téglalap. Az ellipszis szimbolizálja az elemeket, minden egyedi entitás elemként jelenik meg, beleértve a tulajdonságokat is. A rombusz az attribútumokat ábrázolja, melyek a kulcs tulajdonságokból keletkeznek. A téglalap a szöveget reprezentálja, amely később megjelenik az XML dokumentumban. Amennyiben egy elem többször is előfordulhat, a jelölése dupla ellipszissel történik. Az idegenkulcsok és a kulcsok közötti kapcsolatot szaggatott vonalas nyíllal jelöljük.

A feladat XDM modellje:



3. XML dokumentum készítése

Az XML dokumentumot az XDM modell alapján készítettem el, kezdve a gyökérelem létrehozásával, amely a "Szalonok_VFX06G" nevet kapta. Ezt követően minden gyökérelemhez legalább három példányt hoztam létre. Az egyes elemek attribútumaiban kulcsok és idegenkulcsok is megtalálhatók. Emellett a szülőelemekhez is elkészítettem az őket alkotó elemeket. A többértékű elemek esetén több példányt hoztam létre, és előfordult, hogy ezeknek az elemeknek is voltak saját gyerekelemeik.

Az XML dokumentum forráskódja:

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

```
<Szalonok_VFX06G xmlns:xs="http://www.w3.org/2001/XMLSchema-instance"
xs:noNamespaceSchemaLocation="XMLSchemaVFX06G.xsd">
```

```
<!--Karbantartó cégek-->
```

```
<karbantarto_ceg ceg_id="1">
```

```
<ceg>Ceg 1</ceg>
```

```
<kapcsolattarto>
```

```
<telefonszam>06705684589</telefonszam>
```

```
<nev>Kiss István</nev>
```

```
</kapcsolattarto>
```

```
<telephely>
```

```
<varos>Miskolc</varos>
```

```
<iranyitoszam>3515</iranyitoszam>
```

```
<utca>Eper</utca>
```

```
<hazszam>2</hazszam>
```

```
</telephely>
```

```
</karbantarto_ceg>
```

```
<karbantarto_ceg ceg_id="2">
```

```
<ceg>Ceg 2</ceg>
```

```
<kapcsolattarto>
```

```
<telefonszam>06705157589</telefonszam>
```

```
<nev>Nagy Dóra</nev>
```

```

    </kapcsolattarto>
    <telephely>
        <varos>Debrecen</varos>
        <iranyitoszam>4030</iranyitoszam>
        <utca>Petőfi</utca>
        <hazszam>30</hazszam>
    </telephely>
</karbantarto_ceg>

<karbantarto_ceg ceg_id="3">
    <ceg>Ceg 3</ceg>
    <kapcsolattarto>
        <telefonszam>06205684511</telefonszam>
        <nev>Tóth Lilla</nev>
    </kapcsolattarto>
    <telephely>
        <varos>Miskolc</varos>
        <iranyitoszam>3515</iranyitoszam>
        <utca>Kossuth</utca>
        <hazszam>14</hazszam>
    </telephely>
</karbantarto_ceg>

<karbantarto_ceg ceg_id="4">
    <ceg>Ceg 4</ceg>
    <kapcsolattarto>
        <telefonszam>06205622222</telefonszam>
        <nev>Tóth István</nev>
    </kapcsolattarto>
    <telephely>
        <varos>Miskolc</varos>
        <iranyitoszam>3515</iranyitoszam>
        <utca>Búza tér</utca>
        <hazszam>2</hazszam>
    </telephely>
</karbantarto_ceg>

<!--Szalonok-->

<szalon szalon_id="1" karbantartja="1">
    <fonok>Nagy Éliás</fonok>
    <neve>Szalon 1</neve>
    <cim>
        <iranyitoszam>3515</iranyitoszam>
        <varos>Miskolc</varos>
        <utca_hazszam>Petőfi utca 1</utca_hazszam>
    </cim>

```

```
</cim>
<elerhetoseg>
  <telefonszam>06705214787</telefonszam>
  <email>nagyelias@gmail.com</email>
  <facebook>nagyelias32</facebook>
</elerhetoseg>
</szalon>

<szalon szalon_id="2" karbantartja="2">
  <fonok>Szegedi Dóra</fonok>
  <neve>Szalon 2</neve>
  <cim>
    <iranyitoszam>4020</iranyitoszam>
    <varos>Debrecen</varos>
    <utca_hazszam>Petőfi utca 38</utca_hazszam>
  </cim>
  <elerhetoseg>
    <telefonszam>06701112547</telefonszam>
    <email>szegedidora@gmail.com</email>
    <facebook>dorasz02</facebook>
  </elerhetoseg>
</szalon>

<szalon szalon_id="3" karbantartja="3">
  <fonok>Kovács Adrienn</fonok>
  <neve>Szalon 3</neve>
  <cim>
    <iranyitoszam>1024</iranyitoszam>
    <varos>Budapest</varos>
    <utca_hazszam>Arany János utca 15</utca_hazszam>
  </cim>
  <elerhetoseg>
    <telefonszam>06702345678</telefonszam>
    <email>kovacsadrienn@gmail.com</email>
    <facebook>szepsegvarazsszalon</facebook>
  </elerhetoseg>
</szalon>

<szalon szalon_id="4" karbantartja="4">
  <fonok>Szabó Péter</fonok>
  <neve>Szalon 4</neve>
  <cim>
    <iranyitoszam>4400</iranyitoszam>
    <varos>Nyíregyháza</varos>
    <utca_hazszam>Kossuth utca 10</utca_hazszam>
  </cim>
```

```
<elerhetoseg>
  <telefonszam>06301234567</telefonszam>
  <email>szabopeter@gmail.com</email>
  <facebook>szepitokucko.nyiregyhaza</facebook>
</elerhetoseg>
</szalon>
```

```
<!--Kedvezmény-->
```

```
<kedvezmeny kedvezmeny_id="1">
  <elsoIdopont>2023.04.12</elsoIdopont>
  <szolgaltatasok>Manikűr</szolgaltatasok>
</kedvezmeny>
```

```
<kedvezmeny kedvezmeny_id="2">
  <elsoIdopont>2023.07.02</elsoIdopont>
  <szolgaltatasok>Smink</szolgaltatasok>
</kedvezmeny>
```

```
<kedvezmeny kedvezmeny_id="3">
  <elsoIdopont>2023.08.22</elsoIdopont>
  <szolgaltatasok>Fodrászat</szolgaltatasok>
</kedvezmeny>
```

```
<kedvezmeny kedvezmeny_id="4">
  <elsoIdopont>2023.08.22</elsoIdopont>
  <szolgaltatasok>Fodrászat</szolgaltatasok>
</kedvezmeny>
```

```
<kedvezmeny kedvezmeny_id="5">
  <elsoIdopont>2023.08.22</elsoIdopont>
  <szolgaltatasok>Fodrászat</szolgaltatasok>
</kedvezmeny>
```

```
<!--Szolgáltatások-->
```

```
<szolgaltatas szolgaltatas_id="1" vendeg_id="1">
  <tipus>Smink</tipus>
  <ar>15000</ar>
  <idotartam>60</idotartam>
</szolgaltatas>
```

```
<szolgaltatas szolgaltatas_id="2" vendeg_id="3">
  <tipus>Fodrászat</tipus>
  <ar>25000</ar>
  <idotartam>90</idotartam>
```


</szolgaltatas>

<szolgaltatas szolgaltatas_id="3" vendeg_id="1">
 <tipus>Manikűr</tipus>
 <ar>8000</ar>
 <idotartam>45</idotartam>
</szolgaltatas>

<szolgaltatas szolgaltatas_id="4" vendeg_id="4">
 <tipus>Masszázs</tipus>
 <ar>20000</ar>
 <idotartam>75</idotartam>
</szolgaltatas>

<szolgaltatas szolgaltatas_id="5" vendeg_id="5">
 <tipus>Kozmetikai kezelés</tipus>
 <ar>18000</ar>
 <idotartam>60</idotartam>
</szolgaltatas>

<szolgaltatas szolgaltatas_id="6" vendeg_id="2">
 <tipus>Manikűr-pedikűr</tipus>
 <ar>12000</ar>
 <idotartam>90</idotartam>
</szolgaltatas>

<szolgaltatas szolgaltatas_id="7" vendeg_id="1">
 <tipus>Arckezelés</tipus>
 <ar>30000</ar>
 <idotartam>120</idotartam>
</szolgaltatas>

<szolgaltatas szolgaltatas_id="8" vendeg_id="3">
 <tipus>Hajfestés</tipus>
 <ar>18000</ar>
 <idotartam>90</idotartam>
</szolgaltatas>

<szolgaltatas szolgaltatas_id="9" vendeg_id="3">
 <tipus>Szárítás és formázás</tipus>
 <ar>10000</ar>
 <idotartam>45</idotartam>
</szolgaltatas>

<szolgaltatas szolgaltatas_id="10" vendeg_id="5">
 <tipus>Pedikűr</tipus>

```
<ar>12000</ar>
<idotartam>60</idotartam>
</szolgaltatas>

<!--Vendégek-->

<vendeg vendeg_id="1" kedvezmeny_id="1">
  <telefonszam>06301234567</telefonszam>
  <email>kovacsнора@gmail.com</email>
  <fizetett>25000</fizetett>
  <nev>Kovács Nóra</nev>
</vendeg>

<vendeg vendeg_id="2" kedvezmeny_id="2">
  <telefonszam>06705555555</telefonszam>
  <email>szabo.andras@gmail.com</email>
  <fizetett>18000</fizetett>
  <nev>Szabó András</nev>
</vendeg>

<vendeg vendeg_id="3" kedvezmeny_id="4">
  <telefonszam>06708887777</telefonszam>
  <email>molnar.rita@gmail.com</email>
  <fizetett>12000</fizetett>
  <nev>Molnár Rita</nev>
</vendeg>

<vendeg vendeg_id="4" kedvezmeny_id="3">
  <telefonszam>06707654321</telefonszam>
  <email>nagyferenc@gmail.com</email>
  <fizetett>30000</fizetett>
  <nev>Nagy Ferenc</nev>
</vendeg>

<vendeg vendeg_id="5" kedvezmeny_id="5">
  <telefonszam>06701112233</telefonszam>
  <email>kissmari@gmail.com</email>
  <fizetett>15000</fizetett>
  <nev>Kiss Mária</nev>
</vendeg>

<!--Kért szolgáltatás több-több kapcsolat-->

<kert_szolgaltatas ksz_id="1" vevo_id="2" szolgaltatas_id="3">
  <datum>2023-05-15</datum>
</kert_szolgaltatas>
```

```

<kert_szolgaltatas ksz_id="2" vevo_id="4" szolgaltatas_id="6">
  <datum>2023-06-02</datum>
</kert_szolgaltatas>

<kert_szolgaltatas ksz_id="3" vevo_id="1" szolgaltatas_id="8">
  <datum>2023-04-20</datum>
</kert_szolgaltatas>

<kert_szolgaltatas ksz_id="4" vevo_id="3" szolgaltatas_id="2">
  <datum>2023-05-28</datum>
</kert_szolgaltatas>

<kert_szolgaltatas ksz_id="5" vevo_id="5" szolgaltatas_id="10">
  <datum>2023-06-10</datum>
</kert_szolgaltatas>

<kert_szolgaltatas ksz_id="6" vevo_id="2" szolgaltatas_id="1">
  <datum>2023-04-15</datum>
</kert_szolgaltatas>

<kert_szolgaltatas ksz_id="7" vevo_id="3" szolgaltatas_id="5">
  <datum>2023-06-05</datum>
</kert_szolgaltatas>

<kert_szolgaltatas ksz_id="8" vevo_id="4" szolgaltatas_id="9">
  <datum>2023-05-02</datum>
</kert_szolgaltatas>

<!--Kérhető szolgáltatások több-több kapcsolat-->

<kerheto_szolgaltatas szalon_id="1" szolgaltatas_id="6" />
<kerheto_szolgaltatas szalon_id="1" szolgaltatas_id="2" />
<kerheto_szolgaltatas szalon_id="1" szolgaltatas_id="3" />
<kerheto_szolgaltatas szalon_id="1" szolgaltatas_id="8" />
<kerheto_szolgaltatas szalon_id="1" szolgaltatas_id="10" />
<kerheto_szolgaltatas szalon_id="2" szolgaltatas_id="1" />
<kerheto_szolgaltatas szalon_id="2" szolgaltatas_id="4" />
<kerheto_szolgaltatas szalon_id="2" szolgaltatas_id="5" />
<kerheto_szolgaltatas szalon_id="3" szolgaltatas_id="7" />
<kerheto_szolgaltatas szalon_id="3" szolgaltatas_id="8" />
<kerheto_szolgaltatas szalon_id="3" szolgaltatas_id="9" />
<kerheto_szolgaltatas szalon_id="3" szolgaltatas_id="2" />
<kerheto_szolgaltatas szalon_id="4" szolgaltatas_id="3" />
<kerheto_szolgaltatas szalon_id="4" szolgaltatas_id="10" />
<kerheto_szolgaltatas szalon_id="4" szolgaltatas_id="4" />

```

```
<kerheto_szolgaltatas szalon_id="4" szolgaltatas_id="7" />

</Szalonok_VFX06G>
```

4. XMLSchema készítése

Az XML dokumentum validációjának támogatása érdekében először létrehoztam az egyszerű elemeket, amelyekre később hivatkozhatok a sémában. Ezt követően különböző saját típusokat definiáltam, köztük olyanokat is, ahol a típus korlátozása regex vagy enumeration segítségével történt. Ezt követően kidolgoztam magát a sémát, ahol az egyszerű típusokra történő hivatkozások mellett meghatároztam a minimális és maximális előfordulást is adott elemeknél. A complexType-ok után részleteztem az attribútumokat, majd ezekhez létrehoztam a kulcsokat, idegenkulcsokat és speciális kapcsolat referenciákat.

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

```
<!--Egyszerű elemek-->
```

```
<xs:element name="ceg" type="xs:string" />
<xs:element name="telefonszam" type="telefonszamTipus" />
<xs:element name="nev" type="xs:string" />
<xs:element name="varos" type="xs:string" />
<xs:element name="iranyitoszam" type="xs:int" />
<xs:element name="utca" type="xs:string" />
<xs:element name="hazszam" type="xs:int" />

<xs:element name="fonok" type="xs:string" />
<xs:element name="neve" type="xs:string" />
<xs:element name="utca_hazszam" type="xs:string" />
<xs:element name="email" type="emailTipus" />
<xs:element name="facebook" type="xs:string" />

<xs:element name="elsoIdopont" type="idoTipus" />
<xs:element name="szolgaltatasok" type="szolgaltatasTipus" />

<xs:element name="tipus" type="szolgaltatasTipus" />
<xs:element name="ar" type="xs:int" />
<xs:element name="idotartam" type="xs:int" />

<xs:element name="fizetett" type="xs:int" />

<xs:element name="datum" type="idoTipus" />

<!--Saját típusok-->
```

```

<xs:simpleType name="szolgáltatatasTipus">
  <xs:restriction base="xs:string">
    <xs:enumeration value="Smink" />
    <xs:enumeration value="Fodrászat" />
    <xs:enumeration value="Manikűr" />
    <xs:enumeration value="Masszázs" />
    <xs:enumeration value="Kozmetikai kezelés" />
    <xs:enumeration value="Manikűr-pedikűr" />
    <xs:enumeration value="Arckezelés" />
    <xs:enumeration value="Hajfestés" />
    <xs:enumeration value="Szárítás és formázás" />
    <xs:enumeration value="Pedikűr" />
    <xs:enumeration value="Szempillafestés és szemöldök formázás" />
    <xs:enumeration value="Testradírozás" />
    <xs:enumeration value="Spa kezelések" />
    <xs:enumeration value="Körömdíszítés " />
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name="telefonszamTipus">
  <xs:restriction base="xs:string">
    <xs:pattern value="(06(20|30|31|50|60|70)\d{7})" />
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name="idoTipus">
  <xs:restriction base="xs:string">
    <xs:pattern value="([12]\d{3}.(0[1-9]|1[0-2]).(0[1-9]|12)\d{3}[01]))" />
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name="emailTipus">
  <xs:restriction base="xs:string">
    <xs:pattern value="[\w\.\.]+@([\w]+\.\.)+[\w]{2,4}" />
  </xs:restriction>
</xs:simpleType>

<!--Felépítés-->

<xs:element name="Szalonok_VFX06G">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="karbantarto_ceg" minOccurs="1"
maxOccurs="unbounded">

```

```

<xs:complexType>
  <xs:sequence>
    <xs:element ref="ceg" />
    <xs:element name="kapcsolattarto" minOccurs="1"
maxOccurs="1">

      <xs:complexType>
        <xs:sequence>
          <xs:element ref="telefonszam" />
          <xs:element ref="nev" />
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="telephely" minOccurs="1"
maxOccurs="1">

      <xs:complexType>
        <xs:sequence>
          <xs:element ref="varos" />
          <xs:element ref="iranyitoszam" />
          <xs:element ref="utca" />
          <xs:element ref="hazszam" />
        </xs:sequence>
      </xs:complexType>
    </xs:element>
  </xs:sequence>
  <xs:attribute name="ceg_id" type="xs:int" />
</xs:complexType>
</xs:element>
<xs:element name="szalon" minOccurs="1" maxOccurs="unbounded">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="fonok" />
      <xs:element ref="neve" />
      <xs:element name="cim" minOccurs="1" maxOccurs="1">
        <xs:complexType>
          <xs:sequence>
            <xs:element ref="iranyitoszam" />
            <xs:element ref="varos" />
            <xs:element ref="utca_hazszam" />
          </xs:sequence>
        </xs:complexType>
      </xs:element>
      <xs:element name="elerhetoseg" minOccurs="1"
maxOccurs="1">

        <xs:complexType>
          <xs:sequence>
            <xs:element ref="telefonszam" />

```

```

                <xs:element ref="email" />
                <xs:element ref="facebook" />
            </xs:sequence>
        </xs:complexType>
    </xs:element>
</xs:sequence>
<xs:attribute name="szalon_id" type="xs:int" />
<xs:attribute name="karbantartja" type="xs:int" />
</xs:complexType>
</xs:element>
        <xs:element name="kedvezmeny" minOccurs="1"
maxOccurs="unbounded">
            <xs:complexType>
                <xs:sequence>
                    <xs:element ref="elsoIdopont" />
                    <xs:element ref="szolgaltatasok" />
                </xs:sequence>
                <xs:attribute name="kedvezmeny_id" type="xs:int" />
            </xs:complexType>
        </xs:element>
        <xs:element name="szolgaltatas" minOccurs="1"
maxOccurs="unbounded">
            <xs:complexType>
                <xs:sequence>
                    <xs:element ref="tipus" />
                    <xs:element ref="ar" />
                    <xs:element ref="idotartam" />
                </xs:sequence>
                <xs:attribute name="szolgaltatas_id" type="xs:int"
/>
                <xs:attribute name="vendeg_id" type="xs:int" />
            </xs:complexType>
        </xs:element>
<xs:element name="vendeg" minOccurs="1" maxOccurs="unbounded">
    <xs:complexType>
        <xs:sequence>
            <xs:element ref="telefonszam" />
            <xs:element ref="email" />
            <xs:element ref="fizetett" minOccurs="0"
maxOccurs="unbounded"/>
            <xs:element ref="nev" />
        </xs:sequence>
        <xs:attribute name="vendeg_id" type="xs:int" />
        <xs:attribute name="kedvezmeny_id" type="xs:int" />
    </xs:complexType>
</xs:element>

```

```

        <xs:element name="kert_szolgaltatas" minOccurs="1"
maxOccurs="unbounded">
        <xs:complexType>
        <xs:sequence>
        <xs:element ref="datum" />
        </xs:sequence>
        <xs:attribute name="ksz_id" type="xs:int" />
        <xs:attribute name="vevo_id" type="xs:int" />
        <xs:attribute name="szolgaltatas_id" type="xs:int"
/>
        </xs:complexType>
    </xs:element>
    <xs:element name="kerheto_szolgaltatas" minOccurs="1"
maxOccurs="unbounded">
    <xs:complexType>
    <xs:attribute name="szalon_id" type="xs:int" />
    <xs:attribute name="szolgaltatas_id" type="xs:int"
/>
    </xs:complexType>
</xs:element>
</xs:sequence>
</xs:complexType>

<!--Kulcsok-->

<xs:key name="karbantarto_ceg_kulcs">
    <xs:selector xpath="karbantarto_ceg" />
    <xs:field xpath="@ceg_id" />
</xs:key>

<xs:key name="szalon_kulcs">
    <xs:selector xpath="szalon" />
    <xs:field xpath="@szalon_id" />
</xs:key>

<xs:key name="kedvezmeny_kulcs">
    <xs:selector xpath="kedvezmeny" />
    <xs:field xpath="@kedvezmeny_id" />
</xs:key>

<xs:key name="szolgaltatas_kulcs">
    <xs:selector xpath="szolgaltatas" />
    <xs:field xpath="@szolgaltatas_id" />
</xs:key>

<xs:key name="vendeg_kulcs">

```



```

        <xs:selector xpath="vendeg" />
        <xs:field xpath="@vendeg_id" />
    </xs:key>

    <xs:key name="kert_szolgaltatas_kulcs">
        <xs:selector xpath="kert_szolgaltatas" />
        <xs:field xpath="@ksz_id" />
    </xs:key>

    <!--Idegen kulcsok-->

        <xs:keyref
name="karbantarto_ceg_idegen_kulcs">
            refer="karbantarto_ceg_kulcs"
            <xs:selector xpath="szalon" />
            <xs:field xpath="@karbantartja" />
        </xs:keyref>

    <xs:keyref refer="szalon_kulcs" name="szalon_idegen_kulcs">
        <xs:selector xpath="alkalmazott" />
        <xs:field xpath="@munkaviszony_szalon" />
    </xs:keyref>

    <xs:keyref refer="kedvezmeny_kulcs" name="kedvezmeny_idegen_kulcs">
        <xs:selector xpath="vendeg" />
        <xs:field xpath="@kedvezmeny_id" />
    </xs:keyref>

        <xs:keyref
name="vendeg_kertszolgaltatas_idegen_kulcs">
            refer="vendeg_kulcs"
            <xs:selector xpath="kert_szolgaltatas" />
            <xs:field xpath="@vevo_id" />
        </xs:keyref>

        <xs:keyref
name="szolgaltatas_kertszolgaltatas_idegen_kulcs">
            refer="szolgaltatas_kulcs"
            <xs:selector xpath="kert_szolgaltatas" />
            <xs:field xpath="@szolgaltatas_id" />
        </xs:keyref>

        <xs:keyref
name="szalon_kerhetoszolgaltatas_idegen_kulcs">
            refer="szalon_kulcs"
            <xs:selector xpath="kerheto_szolgaltatas" />
            <xs:field xpath="@szalon_id" />
        </xs:keyref>

```

```

                                <xs:keyref          refer="szolgaltatas_kulcs"
name="szolgaltatas_kerhetoszolgaltatas_idegen_kulcs">
    <xs:selector xpath="kerheto_szolgaltatas" />
    <xs:field xpath="@szolgaltatas_id" />
</xs:keyref>

<!--1:1-->

<xs:unique name="unique_szalon">
    <xs:selector xpath="szalon" />
    <xs:field xpath="@karbantartja" />
</xs:unique>

<xs:unique name="unique_vendeg">
    <xs:selector xpath="vendeg" />
    <xs:field xpath="@kedvezmeny_id" />
</xs:unique>

</xs:element>

</xs:schema>

```

5. DOM adatolvasás

Először is a java kódban beállítom, hogy melyik az a file, amiből kellene olvasni, ezt a file-t, pedig a projekt mappájában helyeztem el. Létrehozok egy document elemet, amely segítségével le fogom tudni kérdezni a root elementet, illetve a több parent, illetve gyerekelemet is. NodeListet kell létrehozni, hogy a többszöri előfordulású elementeket el lehessen tárolni, ezeket a document `getElementsByTagName(„String”)` segítségével lehet lekérdezni. Miután ez megtörtént for ciklussal kell végigiterálni az adott parentelemeenten, hogy az attribútumokat és a gyerekelemeket el tudjuk menteni egy-egy stringbe, majd ki tudjuk írni őket. Ahol olyan elem van, aminek több értéke lehet, ott megvizsgálom, hogy az elemek száma több e, mint akkor, hogy ha nem lenne az elemnek több értéke. Ha igen, akkor ciklussal iterálok, és kiíratom a több értékeket.

Kód:

```

package DOMParseVFX06G;

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 DOMReadVFX06G {

    public static void main(String argv[]) throws SAXException, IOException,
ParserConfigurationException {

        File xmlFile = new File("XML_VFX06G.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());

        NodeList kcList = doc.getElementsByTagName("karbantarto_ceg");

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

            Node nNode = kcList.item(i);
            System.out.println("\nCurrent element: " + nNode.getNodeName());

            if (nNode.getNodeType() == Node.ELEMENT_NODE) {

                Element elem = (Element) nNode;
                String id = elem.getAttribute("ceg_id");

                Node node1 = elem.getElementsByTagName("ceg").item(0);
                String ceg = node1.getTextContent();

                System.out.println("A cég id-je: " + id);
                System.out.println("Cég: " + ceg);

                if (kcList.item(i).getChildNodes().getLength() > 2) {
                    int db = 0;

                    Node node2 =
elem.getElementsByTagName("kapcsolattarto").item(0);
                    while (node2 != null) {

```

```

                                                                    node2      =
elem.getElementsByTagName("kapcsolattarto").item(db);
    if (node2 != null) {

                                                                    Node      n      =
elem.getElementsByTagName("telefonszam").item(db);
        String t = n.getTextContent();
        System.out.println("Kapcsolattarto telefonszama:
" + t);

                                                                    Node      n2     =
elem.getElementsByTagName("nev").item(db);
        String nev = n2.getTextContent();
        System.out.println("Kapcsolattarto neve: " +
nev);

        }
        db++;
    }

    if (kcList.item(i).getChildNodes().getLength() > 3) {
        int db = 0;

                                                                    Node      node2    =
elem.getElementsByTagName("telephely").item(0);
        while (node2 != null) {

                                                                    node2      =
elem.getElementsByTagName("telephely").item(db);
            if (node2 != null) {

                                                                    Node      n      =
elem.getElementsByTagName("varos").item(db);
                String v = n.getTextContent();
                System.out.println("Telephely városa: " + v);

                                                                    Node      n2     =
elem.getElementsByTagName("iranyitoszam").item(db);
                String isz = n2.getTextContent();
                System.out.println("Telephely irányítószáma: " +
isz);

                                                                    Node      n3     =
elem.getElementsByTagName("utca").item(db);
                String u = n3.getTextContent();
                System.out.println("Telephely utcája: " + u);

```

```

Node n4 =
elem.getElementsByTagName("hazszam").item(db);
    String hsz = n4.getTextContent();
    System.out.println("Telephely házszáma: " + hsz);

    }
    db++;
}

}

}

NodeList szList = doc.getElementsByTagName("szalon");

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

    Node fNode = szList.item(i);
    System.out.println("\nCurrent element: " + fNode.getNodeName());

    if (fNode.getNodeType() == Node.ELEMENT_NODE) {

        Element elem = (Element) fNode;
        String id = elem.getAttribute("szalon_id");
        String karbantart = elem.getAttribute("karbantartja");

        Node node1 = elem.getElementsByTagName("fonok").item(0);
        String fonok = node1.getTextContent();

        Node node2 = elem.getElementsByTagName("neve").item(0);
        String nev = node2.getTextContent();

        System.out.println("A szalon id-je: " + id);
        System.out.println("A szalon karbantartojanak id-je: " +
karbantart);

        System.out.println("A fonok neve: " + fonok);
        System.out.println("A szalon neve: " + nev);

        if (szList.item(i).getChildNodes().getLength() > 4) {
            int db = 0;
            Node node3 = elem.getElementsByTagName("cim").item(0);
            while (node3 != null) {
                node3 = elem.getElementsByTagName("cim").item(db);
                if (node3 != null) {

```

```

Node n =
elem.getElementsByTagName("iranyitoszam").item(db);
    String isz = n.getTextContent();
    System.out.println("Telephely irányítószáma: " +
isz);

Node n2 =
elem.getElementsByTagName("varos").item(db);
    String v = n2.getTextContent();
    System.out.println("Telephely városa: " + v);

Node n3 =
elem.getElementsByTagName("utca_hazszam").item(db);
    String u = n3.getTextContent();
    System.out.println("Telephely utcája és házszáma:
" + u);
    }
    db++;
}

if (szList.item(i).getChildNodes().getLength() > 5) {
    int db = 0;

Node node3 =
elem.getElementsByTagName("elerhetoseg").item(0);
    while (node3 != null) {

node3 =
elem.getElementsByTagName("elerhetoseg").item(db);
        if (node3 != null) {

Node n =
elem.getElementsByTagName("telefonszam").item(db);
            String tsz = n.getTextContent();
            System.out.println("Szalon telefonszama: " +
tsz);

Node n2 =
elem.getElementsByTagName("email").item(db);
            String email = n2.getTextContent();
            System.out.println("Szalon emailje: " + email);

Node n3 =
elem.getElementsByTagName("facebook").item(db);
            String face = n3.getTextContent();
            System.out.println("Szalon facebook oldala: " +
face);

```

```

        }
        db++;
    }

}

}

}

NodeList tList = doc.getElementsByTagName("kedvezmeny");

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

    Node tNode = tList.item(i);
    System.out.println("\nCurrent element: " + tNode.getNodeName());

    if (tNode.getNodeType() == Node.ELEMENT_NODE) {

        Element elem = (Element) tNode;
        String aid = elem.getAttribute("kedvezmeny_id");

        Node node2 = elem.getElementsByTagName("elsoIdopont").item(0);
        String nev = node2.getTextContent();

        Node node3 =
elem.getElementsByTagName("szolgaltatasok").item(0);
        String fizetes = node3.getTextContent();

        System.out.println("A kedvezmeny kulcsa: " + aid);
        System.out.println("Az elsoIdopont: " + nev);
        System.out.println("Szolgaltatas: " + fizetes);

    }

}

NodeList csList = doc.getElementsByTagName("szolgaltatas");

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

    Node vNode = csList.item(i);
    System.out.println("\nCurrent element: " + vNode.getNodeName());

    if (vNode.getNodeType() == Node.ELEMENT_NODE) {

```

```

        Element elem = (Element) vNode;
        String sz_id = elem.getAttribute("szolgaltatas_id");

        Node node1 = elem.getElementsByTagName("tipus").item(0);
        String tipus = node1.getTextContent();

        Node node2 = elem.getElementsByTagName("ar").item(0);
        String ar = node2.getTextContent();

        Node node3 = elem.getElementsByTagName("idotartam").item(0);
        String idotart = node3.getTextContent();

        System.out.println("A szolgaltatas azonosítója: " + sz_id);
        System.out.println("A szolgáltatás típusa: " + tipus);
        System.out.println("Ára: " + ar);
        System.out.println("Időtartama: " + idotart);

    }
}

NodeList vList = doc.getElementsByTagName("vendeg");

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

    Node gyNode = vList.item(i);
    System.out.println("\nCurrent element: " + gyNode.getNodeName());

    if (gyNode.getNodeType() == Node.ELEMENT_NODE) {

        Element elem = (Element) gyNode;
        String id = elem.getAttribute("vendeg_id");
        String kid = elem.getAttribute("kedvezmeny_id");

        Node node1 = elem.getElementsByTagName("telefonszam").item(0);
        String tf = node1.getTextContent();

        Node node2 = elem.getElementsByTagName("email").item(0);
        String email = node2.getTextContent();

        Node node3 = null;
        String fizetett = null;
        if (elem.getElementsByTagName("fizetett").item(0) != null) {
            node3 = elem.getElementsByTagName("fizetett").item(0);
            fizetett = node3.getTextContent();
        }
    }
}

```



```

        Node node4 = elem.getElementsByTagName("nev").item(0);
        String nev = node4.getTextContent();

        System.out.println("Vendég egyedi azonosítója: " + id);
        System.out.println("Szolgáltatás kedvezmény egyedi
azonosítója: " + kid);
        System.out.println("Vendég telefonszáma: " + tf);
        System.out.println("Vendég emailje: " + email);
        if (node3 != null)
            System.out.println("Vendég által fizetett összeg: " +
fizetett);

        System.out.println("A vendég neve: " + nev);

    }
}

NodeList thList = doc.getElementsByTagName("kert_szolgaltatas");

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

    Node lNode = thList.item(i);
    System.out.println("\nCurrent element: " + lNode.getNodeName());

    if (lNode.getNodeType() == Node.ELEMENT_NODE) {

        Element elem = (Element) lNode;
        String ksz_id = elem.getAttribute("ksz_id");
        String vid = elem.getAttribute("vevo_id");
        String szid = elem.getAttribute("szolgaltatas_id");

        Node node1 = elem.getElementsByTagName("datum").item(0);
        String date = node1.getTextContent();

        System.out.println("A kért szolgáltatás kapcsolat egyedi
azonosítója: " + ksz_id);
        System.out.println("A vevő azonosítója: " + vid);
        System.out.println("A szolgáltatás azonosítója: " + szid);
        System.out.println("A dátum: " + date);

    }
}

NodeList lzList = doc.getElementsByTagName("kerheto_szolgaltatas");

```

```

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

            Node tnNode = lzList.item(i);
            System.out.println("\nCurrent element: " + tnNode.getNodeName());

            if (tnNode.getNodeType() == Node.ELEMENT_NODE) {

                Element elem = (Element) tnNode;
                String szid = elem.getAttribute("szalon_id");
                String szolid = elem.getAttribute("szolgaltatas_id");

                System.out.println("A szalon azonosítója: " + szid);
                System.out.println("A szolgáltatás azonosítója: " + szolid);

            }

        }

    }

}

```

6. DOM adatmódosítás

Kiválasztottam, hogy az első szalon attribútumát szeretném módosítani, illetve egy gyerekelemét a nevet. Mivel ez az attribútum több helyen is megjelent, ezért ezeket ott is meg kellett változtatnom. A módosított fájlt mentettem, illetve kiírtam a konzolra.

```

package DOMParseVFX06G;

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

import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.DocumentBuilderFactory;
import javax.xml.parsers.ParserConfigurationException;
import javax.xml.transform.Transformer;
import javax.xml.transform.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.NamedNodeMap;
import org.w3c.dom.Node;
import org.w3c.dom.NodeList;
import org.xml.sax.SAXException;

public class DOMModifyVFX06G {

```

```

    public static void main(String argv[]) throws SAXException, IOException,
ParserConfigurationException {

    try {

        File inputFile = new File("XML_VFX06G2.xml");

        DocumentBuilderFactory documentBuilderFactory =
DocumentBuilderFactory.newInstance();
        DocumentBuilder documentBuilder =
documentBuilderFactory.newDocumentBuilder();

        Document doc = documentBuilder.parse(inputFile);

        // szalon attribútumának módosítása
        Node csoport = doc.getElementsByTagName("szalon").item(0);

        NamedNodeMap attr = csoport.getAttributes();
        Node nodeAttr = attr.getNamedItem("szalon_id");
        nodeAttr.setTextContent("5");

        // szalonnév módosítása
        NodeList list = csoport.getChildNodes();

        for (int i = 0; i < list.getLength(); i++) {
            Node node = list.item(i);

            if (node.getNodeType() == Node.ELEMENT_NODE) {
                Element eElement = (Element) node;

                if ("neve".equals(eElement.getNodeName())) {
                    if ("Szalon 1".equals(eElement.getTextContent())) {
                        eElement.setTextContent("Szalon 5");
                    }
                }
            }
        }

        // kerheto_szolgaltatas módosítás
        NodeList kszList =
doc.getElementsByTagName("kerheto_szolgaltatas");

        for (int i = 0; i < kszList.getLength(); i++) {
            Node node = kszList.item(i);

            if (node.getNodeType() == Node.ELEMENT_NODE) {

                Element eElement = (Element) node;

```

```

        String tagValue = eElement.getAttribute("szalon_id");

        if ("1".equals(tagValue)) {
            eElement.setAttribute("szalon_id", "5");
        }

    }

}

// Tartalom írása
TransformerFactory transformerFactory =
TransformerFactory.newInstance();
Transformer transformer = transformerFactory.newTransformer();

DOMSource source = new DOMSource(doc);

System.out.println("!Módosított fájl!");

StreamResult consoleResult = new StreamResult(System.out);
StreamResult file = new StreamResult(inputFile);

transformer.transform(source, consoleResult);
transformer.transform(source, file);

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

}
}

```

7. DOM adatlekérdezés

A lekérdezéseket az XPath segítségével hajtottam végre, ebből 8-at készítettem el, majd ezeket teszteltem, illetve a megfelelő kritériumok mellett kiíratam a konzolra.

- a Szalonok_VFX06G root elem szolgáltatás gyerekelemei
- szalonok, amik rendelkeznek attribútummal
- szolgáltatások, amik 60 percesek
- szolgáltatások típusa, amik több, mint 12000Ft-ba kerülnek
- 2-es azonosítójú szalon
- a harmadik szalon kiválasztása
- vendégek telefonszámának és fizetett mennyiségének kiírása
- első két vendég elem

Kód:

```
package DOMParseVFX06G;
```

```

import java.io.IOException;

import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.DocumentBuilderFactory;
import javax.xml.parsers.ParserConfigurationException;
import javax.xml.xpath.XPath;
import javax.xml.xpath.XPathConstants;
import javax.xml.xpath.XPathExpressionException;
import javax.xml.xpath.XPathFactory;

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 DOMQueryVFX06G {

    public static void main(String[] args) {

        try {

            // DocumentBuilder
            DocumentBuilderFactory documentBuilderFactory =
DocumentBuilderFactory.newInstance();
            DocumentBuilder documentBuilder =
documentBuilderFactory.newDocumentBuilder();

            Document document = documentBuilder.parse("XML_VFX06G.xml");

            document.getDocumentElement().normalize();

            // XPath
            XPath xPath = XPathFactory.newInstance().newXPath();

            // a Szalonok_VFX06G root elem szolgáltatás gyerekelemei
            String expression = "Szalonok_VFX06G / szolgáltatás";

            // szalonok, amik rendelkeznek attributummal
            // String expression = "//szalon[@*]";

            // szolgáltatások, amik 60 percesek
            // String expression = "//szolgáltatás[idotartam='60']";

            // szolgáltatások típusa, amik több, mint 12000Ft-ba kerülnek
            // String expression = "//szolgáltatás[ar>12000]/típus";

            // 2-es azonosítóju szalon
            // String expression = "//szalon[@szalon_id='2']";

```

```

// a harmadik szalon kiválasztása
// String expression = "Salonok_VFX06G/salon[3]";

// vendégek telefonszámának és fizetett mennyiségének kiírása
// String expression = "//telefonszam | //fizetett";

// vendég első két eleme
// String expression = "//vendeg[position()<3]";

// készítünk egy listát, majd az XPath kifejezést le kell fordítani
// és ki kell értékelni
NodeList nodeList = (NodeList)
XPath.compile(expression).evaluate(document, XPathConstants.NODESET);

for (int i = 0; i < nodeList.getLength(); i++) {
    Node node = nodeList.item(i);
    System.out.println("\nAktualis elem: " + node.getNodeName());

    if (node.getNodeType() == Node.ELEMENT_NODE &&
node.getNodeName().equals("szolgaltatas")) {

        Element elem = (Element) node;

        String id = elem.getAttribute("szolgaltatas_id");
        String munk_id =
elem.getAttribute("munka_alkalmazott");

        Node node1 =
elem.getElementsByTagName("tipus").item(0);
        String tipus = node1.getTextContent();

        Node node2 = elem.getElementsByTagName("ar").item(0);
        String ar = node2.getTextContent();

        Node node3 =
elem.getElementsByTagName("idotartam").item(0);
        String idotart = node3.getTextContent();

        System.out.println("Szolgáltatás id-je: " + id);
        System.out.println("Alkalmazott id-je: " + munk_id);
        System.out.println("Típus: " + tipus);
        System.out.println("Ár: " + ar);
        System.out.println("Időtartam: " + idotart);

    }

    // szolgáltatás típusa

```

```

        if (node.getNodeType() == Node.ELEMENT_NODE &&
node.getNodeName().equals("tipus")) {

            Element element = (Element) node;

            System.out.println("Szolgáltatás típusa: " +
element.getTextContent());

        }

        // vendég telefonszáma
        if (node.getNodeType() == Node.ELEMENT_NODE &&
node.getNodeName().equals("telefonszam")) {

            Element element = (Element) node;

            System.out.println("Telefonszám: " +
element.getTextContent());

        }

        // vendég fizetett összege
        if (node.getNodeType() == Node.ELEMENT_NODE &&
node.getNodeName().equals("fizetett")) {

            Element element = (Element) node;

            System.out.println("Fizetett: " +
element.getTextContent());

        }

        //szalon kiiratasa
        if (node.getNodeType() == Node.ELEMENT_NODE &&
node.getNodeName().equals("szalon")) {

            Element element = (Element) node;

            System.out.println("ID: " +
element.getAttribute("szalon_id"));
            System.out.println("Karbantartó cég id: " +
element.getAttribute("karbantartja"));

            System.out.println(
                "Főnök neve: " +
element.getElementsByTagName("fonok").item(0).getTextContent());

            System.out.println(

```



```

        System.out.println("Szalon emailje: " +
email);

Node n3 =
element.getElementsByTagName("facebook").item(db);
        String face = n3.getTextContent();
        System.out.println("Szalon facebook oldala:
" + face);
    }
    db++;
}

}

//vendég kiiratasa
        if (node.getNodeType() == Node.ELEMENT_NODE &&
node.getNodeName().equals("vendeg")) {

            Element element = (Element) node;

            System.out.println("ID: " +
element.getAttribute("vendeg_id"));

            System.out.println("Telefonszám: " +
element.getElementsByTagName("telefonszam").item(0).getTextContent());

            System.out.println(
                "Email: " +
element.getElementsByTagName("email").item(0).getTextContent());

            System.out.println(
                "Fizetett: " +
element.getElementsByTagName("fizetett").item(0).getTextContent());

            System.out.println(
                "Név: " +
element.getElementsByTagName("nev").item(0).getTextContent());

        }

    }

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

```

```

        } catch (IOException e) {
            e.printStackTrace();
        } catch (XPathExpressionException e) {
            e.printStackTrace();
        }
    }
}
}

```

8. DOM adatírás

Minden egyes gyerekelemnek, amelyek a root-hoz tartoznak, létrehoztam egy-egy metódust aminek segítségével a kód futtatásakor ezeket hozzáadom a root-hoz. A metódusok paraméterei az attribútumok, illetve a megfelelő tulajdonságok, amiket ezen belül hozzáadok az elemhez. Mindegyik root gyerekelemből legalább hármat létrehozok, amely megjelenik az xml dokumentumban is kiíráskor.

Kód:

```

package DOMParseVFX06G;

import java.io.File;

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

public class DOMWriteVFX06G {

    public static void main(String argv[]) throws Exception {

        DocumentBuilderFactory factory =
DocumentBuilderFactory.newInstance();
        DocumentBuilder dBuilder = factory.newDocumentBuilder();

        Document doc = dBuilder.newDocument();
    }
}

```

```

Element root = doc.createElementNS("VFX06G", "Szalonok_VFX06G");
doc.appendChild(root);

// karbantarto_ceg

    root.appendChild(createCeg(doc, "1", "Ceg 1", "06705684589", "Kiss
István", "Budapest", "1024", "Kossuth utca", "10"));
    root.appendChild(createCeg(doc, "2", "Ceg 2", "06705684111", "Kovács
Anna", "Budapest", "1132", "Rákospatak utca", "5"));
    root.appendChild(createCeg(doc, "3", "Ceg 3", "06705684233", "Szabó
István", "Debrecen", "3521", "Sport utca", "8"));

                                Element          element          =          (Element)
doc.getElementsByTagName("karbantarto_ceg").item(0);
    Comment comment = doc.createComment("Karbantartó cégek");
    element.getParentNode().insertBefore(comment, element);

// szalon

    root.appendChild(createSzalon(doc, "1", "1", "Nagy Éliás", "Szalon
1", "1024", "Budapest", "Kossuth utca 10", "06705214787",
"nagyelias@gmail.com", "nagyelias32"));
    root.appendChild(createSzalon(doc, "2", "2", "Szalon 2", "Szegedi
Dóra", "1132", "Budapest", "Rákospatak utca 5", "06701112547",
"szegedidora@gmail.com", "doraszo2"));
    root.appendChild(createSzalon(doc, "3", "3", "Szalon 3", "Kovács
Adrienn", "3521", "Debrecen", "Sport utca 8", "06705214733",
"kovacsadrienn@gmail.com", "szepsegvarazsszalon"));

    element = (Element) doc.getElementsByTagName("szalon").item(0);
    comment = doc.createComment("Szalonok");
    element.getParentNode().insertBefore(comment, element);

// kedvezmeny

    root.appendChild(createKedvezmeny(doc, "1", "2023.05.12",
"Manikűr"));
    root.appendChild(createKedvezmeny(doc, "2", "2023.05.12",
"Manikűr"));
    root.appendChild(createKedvezmeny(doc, "3", "2023.05.12",
"Manikűr"));

    element = (Element) doc.getElementsByTagName("kedvezmeny").item(0);
    comment = doc.createComment("Kedvezmények");
    element.getParentNode().insertBefore(comment, element);

```

```

        // szolgaltatas

        root.appendChild(createSzolgaltatas(doc, "1", "Smink", "15000",
"60"));
        root.appendChild(createSzolgaltatas(doc, "2", "Fodrászat", "25000",
"90"));
        root.appendChild(createSzolgaltatas(doc, "3", "Manikűr", "8000",
"45"));

        element = (Element) doc.getElementsByTagName("szolgaltatas").item(0);
        comment = doc.createComment("Szolgáltatások");
        element.parentNode().insertBefore(comment, element);

        // vendeg

        root.appendChild(createVendeg(doc, "1", "1", "06301234567",
"kovacsнора@gmail.com", "25000", "Kovács Nóra"));
        root.appendChild(createVendeg(doc, "2", "2", "06705555555",
"szabo.andras@gmail.com", "", "Szabó András"));
        root.appendChild(createVendeg(doc, "3", "3", "06708887777",
"molnar.rita@gmail.com", "12000", "Molnár Rita"));

        element = (Element) doc.getElementsByTagName("vendeg").item(0);
        comment = doc.createComment("Vendégek");
        element.parentNode().insertBefore(comment, element);

        // kert_szolgaltatas

        root.appendChild(createKertSzolgaltatas(doc, "3", "1", "2", "2023-
05-15"));
        root.appendChild(createKertSzolgaltatas(doc, "1", "3", "1", "2023-
05-15"));
        root.appendChild(createKertSzolgaltatas(doc, "2", "2", "3", "2023-
05-15"));

        element = (Element)
doc.getElementsByTagName("kert_szolgaltatas").item(0);
        comment = doc.createComment("Kért szolgáltatás több-több kapcsolat");
        element.parentNode().insertBefore(comment, element);

        // kerheto_szolgaltatas

        root.appendChild(createKerhetoSzolgaltatas(doc, "2", "1"));
        root.appendChild(createKerhetoSzolgaltatas(doc, "1", "3"));
        root.appendChild(createKerhetoSzolgaltatas(doc, "3", "2"));

```

```

                element = (Element)
doc.getElementsByTagName("kerheto_szolgaltatas").item(0);
        comment = doc.createComment("Kérhető szolgáltatások több-több
kapcsolat");
        element.getParentNode().insertBefore(comment, element);

// Transform

        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");

// File létrehozás

DOMSource source = new DOMSource(doc);
File myFile = new File("VFX06G_write.xml");

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

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

}

private static Node createCeg(Document doc, String ceg_id, String ceg,
String telefonszam, String nev,
        String varos, String iranyitoszam, String utca, String hazzsam)
{

        Element kc = doc.createElement("karbantarto_ceg");

        kc.setAttribute("ceg_id", ceg_id);
        kc.appendChild(createElement(doc, "ceg", ceg));

        Element kt = doc.createElement("kapcsolattarto");
        kt.appendChild(createElement(doc, "telefonszam", telefonszam));
        kt.appendChild(createElement(doc, "nev", nev));

        kc.appendChild(kt);

```

```

        Element cim = doc.createElement("telephely");
        cim.appendChild(createElement(doc, "varos", varos));
        cim.appendChild(createElement(doc, "iranyitoszam", iranyitoszam));
        cim.appendChild(createElement(doc, "utca", utca));
        cim.appendChild(createElement(doc, "hazszam", hazszam));

        kc.appendChild(cim);

        return kc;
    }

    private static Node createSzalon(Document doc, String szalon_id, String
    karbantartja, String fonok,
        String neve, String isz, String varos, String utcahazszam, String
    telefonszam, String email, String facebook) {

        Element sz = doc.createElement("szalon");

        sz.setAttribute("szalon_id", szalon_id);
        sz.setAttribute("karbantartja", karbantartja);
        sz.appendChild(createElement(doc, "fonok", fonok));
        sz.appendChild(createElement(doc, "neve", neve));

        Element cim = doc.createElement("cim");
        cim.appendChild(createElement(doc, "iranyitoszam", isz));
        cim.appendChild(createElement(doc, "varos", varos));
        cim.appendChild(createElement(doc, "utca_hazszam", utcahazszam));

        sz.appendChild(cim);

        Element elerh = doc.createElement("elerhetoseg");
        elerh.appendChild(createElement(doc, "telefonszam", telefonszam));
        elerh.appendChild(createElement(doc, "email", email));
        elerh.appendChild(createElement(doc, "facebook", facebook));

        sz.appendChild(elerh);

        return sz;
    }

    private static Node createKedvezmeny(Document doc, String kedvezmeny_id,
    String elsoIdopont, String szolgáltatás) {

```

```

        Element kedv = doc.createElement("kedvezmeny");

        kedv.setAttribute("kedvezmeny_id", kedvezmeny_id);

        kedv.appendChild(createElement(doc, "elsoIdopont", elsoIdopont));
        kedv.appendChild(createElement(doc, "szolgaltatasok", szolgaltatas));

        return kedv;
    }

    private static Node createSzolgaltatas(Document doc, String
szolgaltatas_id, String tipus,
        String ar, String idotartam) {

        Element csop = doc.createElement("szolgaltatas");

        csop.setAttribute("szolgaltatas_id", szolgaltatas_id);
        csop.appendChild(createElement(doc, "tipus", tipus));
        csop.appendChild(createElement(doc, "ar", ar));
        csop.appendChild(createElement(doc, "idotartam", idotartam));

        return csop;
    }

    private static Node createVendeg(Document doc, String vendeg_id, String
kedvezmeny_id, String telefonszam, String email, String fizetett,
        String nev) {

        Element v = doc.createElement("vendeg");

        v.setAttribute("vendeg_id", vendeg_id);
        v.setAttribute("kedvezmeny_id", kedvezmeny_id);
        v.appendChild(createElement(doc, "telefonszam", telefonszam));
        v.appendChild(createElement(doc, "email", email));

        if(!(fizetett.equals(""))){
            v.appendChild(createElement(doc, "fizetett", fizetett));
        }

        v.appendChild(createElement(doc, "nev", nev));

        return v;
    }

```

```

        private static Node createKertSzolgaltatas(Document doc, String ksz_id,
String vevo_id, String szolgaltatas_id,
        String datum) {

            Element th = doc.createElement("kert_szolgaltatas");

            th.setAttribute("ksz_id", ksz_id);
            th.setAttribute("vevo_id", vevo_id);
            th.setAttribute("szolgaltatas_id", szolgaltatas_id);
            th.appendChild(createElement(doc, "datum", datum));

            return th;

        }

        private static Node createKerhetoSzolgaltatas(Document doc, String
szalon_id, String szolgaltatas_id) {

            Element lz = doc.createElement("kerheto_szolgaltatas");

            lz.setAttribute("szalon_id", szalon_id);
            lz.setAttribute("szolgaltatas_id", szolgaltatas_id);

            return lz;

        }

        private static Node createElement(Document doc, String name, String value)
{

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

            return node;

        }

    }

```