

JEGYZŐKÖNYV

Webes adatkezelő környezetek

Féléves feladat

Macskamenhely

Készítette: **Vay Dominika**

Neptunkód: **VM3DIR**

Dátum: **2025. november**

Miskolc, 2025

Tartalomjegyzék

1. Feladat	3
1.1 Az adatbázis ER modell tervezése	3
1.2 Az adatbázis konvertálása XDM modellre	4
1.3 Az XDM modell alapján XML dokumentum készítése	5
1.4 Az XML dokumentum alapján XMLSchema készítése	8
1.5 Validáció	24
2. Feladat	26
2.1 Adatolvasás	26
2.2 Adat-lekérdezés	27
2.3 Adatmódosítás	30

Bevezetés

A feladat leírása:

A beadandó témája/feladata volt egy olyan adatbázis kialakítása, amelyben megtalálhatjuk egy macskamenhely minden napi működését és nyilvántartását. A rendszer célja, hogy a menhelyen élő cicák, az örökbefogadók, az alkalmazottak, az állatorvosok, valamint az egészségügyi kartonok és az örökbefogadások adatai egy közös, XML-ben legyenek elérhetőek, ahol megkaphatjuk a szükséges információkat bármelyik entitásról. A dokumentumom kialakítása során törekedtem arra, hogy az entitások közötti kapcsolatok jól láthatóak/jól nyomon követhetőek legyenek. Hogy tudjuk ezeket az adatokat lekérdezni, szűrni, módosításokat végezni rajtuk.

Entitások és attribútumaik:

- **Macska:**

- macska id: a macska egyedi azonosítója,
- név: a macska neve,
- fajta: a macska fajtája,
- születési dátum: a macska születési dátuma,
- szín: a macska színe,
- nem: a macska neme, hím vagy nőstény,
- bekerülési dátum: a macska menhelyre való bekerülésének időpontja,
- alkalmazott ref: idegen kulcs, ami az alkalmazottra mutat, ez jelzi, hogy melyik alkalmazott gondozza a macskát, egy alkalmazott gondozhat több cicát, de egy macskát csak egy alkalmazott gondozhat.

- **Egészségügyi karton:**

- karton id: a karton egyedi azonosítója,
- dátum: a vizsgálat dátuma,
- kezelés típusa: milyen beavatkozás/vizsgálat/kezelés történt a cicával,
- diagnózis: megállapítás a macska állapotáról, betegségeiről,
- gyógyszerek: a vizsgálat során/későbbiekben használt gyógyszerek/oltások,
- következő vizsgálat: a következő vizsgálat dátuma,
- macska ref: a macskához tartozó egészségügyi kartont jelzi, egy macskához egy karton tartozik,

- állatorvos ref: az állatorvoshoz tartozik, aki megírta a kartont a vizsgálat alapján, egy állatorvos több kartonhoz tartozhat, de egy karton csak egy állatorvoshoz.
- ▶ **Állatorvos:**
 - állatorvos id: az állatorvos egyedi azonosítója,
 - teljes név: szét van szedve vezetéknév és keresztnévre, ez az állatorvos teljes neve,
 - szakképesítés: több szakképesítése lehet egy állatorvosnak (pl.: orvos, sebész),
 - rendelő neve: a rendelő neve, ahol az állatorvos dolgozik,
 - telefonszám: az orvos telefonszáma,
 - email: az orvos email címe.
- ▶ **Alkalmazott:**
 - alkalmazott id: az alkalmazott egyedi azonosítója,
 - teljes név: szét van szedve vezetéknév és keresztnévre, ez az alkalmazott teljes neve,
 - cím: ez is egy összetett elem, irányítószámból, városból, utcából és házszámból áll, itt lakik az alkalmazott,
 - beosztás: az alkalmazott pozíciója,
 - belépés dátuma: mikortól dolgozik ott az alkalmazott,
 - fizetés: mennyit keres az alkalmazott.
- ▶ **Örökbefogadó:**
 - örökbefogadó id: az örökbefogadó egyedi azonosítója,
 - teljes név: szét van szedve vezetéknév és keresztnévre, ez az örökbefogadó teljes neve,
 - cím: ez is egy összetett elem, irányítószámból, városból, utcából és házszámból áll, itt lakik az örökbefogadó,
 - telefonszám: az örökbefogadó telefonszáma,
 - email: az örökbefogadó email címe,
 - születési dátum: az örökbefogadó születési dátuma.
- ▶ **Örökbefogadás:**
 - ez egy kapcsolat tulajdonságai, ami a Macska entitás és az Örökbefogadó entitás között van,
 - örökbefogadás dátuma: amikor megtörtént az örökbefogadás hivatalosan,
 - örökbefogadási díj: az örökbefogadás díja, amennyibe került,

- státusz: az örökbefogadás státusza, ami lehet folyamatban, jóváhagyva, befejezett vagy visszautasítva,
- megjegyzés: megjegyzés az örökbefogadáshoz, például, hogy hogy ment,
- macska ref: hivatkozunk a macskára, aki részt vett az örökbefogadásban, és örökbefogadják,
- örökbefogadó ref: hivatkozás az örökbefogadóra, aki örökbefogad egy cicát.

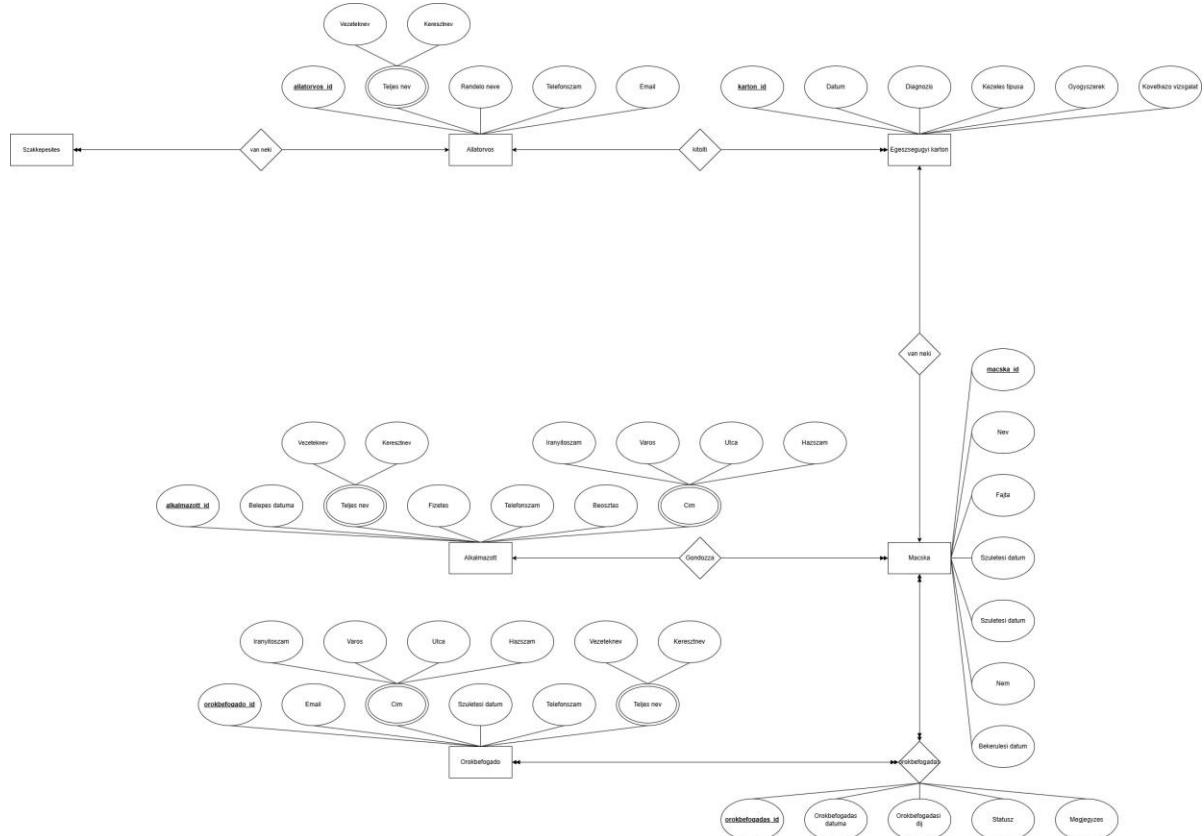
1. Feladat

1.1 Az adatbázis ER modell tervezése

Az ER modell megtervezését az Egyedek közötti kapcsolatoknak (relationship-eknek) az összevetésével kezdtem. Az alábbi kapcsolatok vannak:

- Macska -> Egészségügyi karton, egy egy kapcsolat, egy macskához egy karton tartozik (1:1).
- Macska -> Alkalmazott, egy több kapcsolat, egy alkalmazott több macskát is gondozhat, de egy macskát csak egy alkalmazott gondozhat (1:N).
- Állatorvos -> Egészségügyi karton, egy több kapcsolat, egy állatorvos több kartonhoz is tartozhat, de egy kartonhoz csak egy állatorvos (1:N).
- Macska -> Örökbefogadó, több több kapcsolat, egy macskához több örökbefogadás tartozhat, egy örökbefogadó több örökbefogadást is végezhet, tehát egy macska több örökbefogadóhoz, egy örökbefogadó több macskához is tartozhat (N:M).

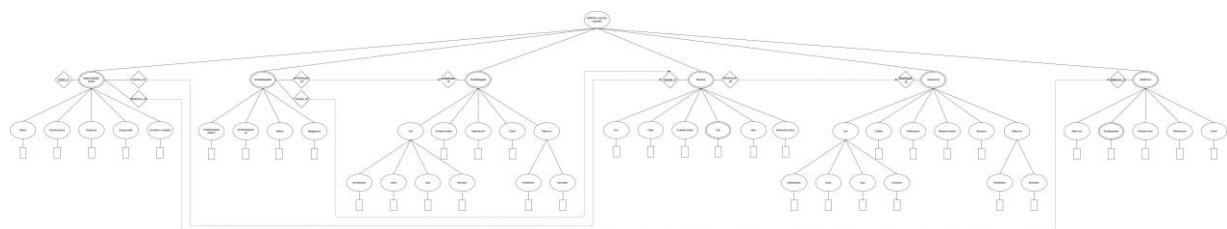
Ezek alapján a következő ER modell (1. ábra) készült az entitásokhoz tartozó egyedi kulcsokkal és tulajdonságaikkal együtt, az entitások téglalapba kerültek, a tulajdonságaik pedig ellipszisekbe, a kapcsolatokat pedig rombuszok jelzik, az 1:1, 1:N és N:M kapcsolatokat pedig a nyilak jelzik, az összetett tulajdonságok dupla kerettel rendelkeznek, és a többértékű tulajdonságok egy másik kapcsolattal lettek összekötve:



1. ábra: ER modell 1

1.2 Az adatbázis konvertálása XDM modellre

Az XDM modellt az ER modell (1. ábra) és az entitások kapcsolatai alapján hoztam létre, a kapcsolatok az ER modellben már leírásra kerültek. Az XDM modell (2. ábra) esetében az entitások ellipszisbe kerülnek, az elsődleges és az idegen kulcsok rombuszba, dupla kerettel pedig a többértékű Egyedek rendelkeznek, az elsődleges és idegen kulcsok összekötése szaggatott vonallal jelzik a kapcsolatokat:



2. ábra: XDM modell 1

1.3 Az XDM modell alapján XML dokumentum készítése

Az XDM modell (2. ábra) alapján pedig végül elkészítettem az XML dokumentumomat. A root elemmel kezdtettem, ami maga a menhely volt. Létrehoztam a gyerekelemeiből több példányt is (ez a forráskóban szerepel, ide csak a lényeges részét másoltam be). Az elemeknek az attribútumai közé tartoznak az elsődleges kulcsok és az idegen kulcsok is.

Az XML dokumentum forráskódja (részlet), kommentekkel kiegészítve:

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

<!-- A macskamenhely XML séma -->

<VM3DIR_macskamenhely xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:noNamespaceSchemaLocation="VM3DIR_XMLSchema.xsd">

    <!-- Egészségügyi kartonok adatai, id, macska_ref (idegen kulcs), allatorvos_ref (idegen
        kulcs), datum, kezeles_tipusa, diagnozis, gyogyszerek, kovetkezo_vizsgalat -->

    <!-- Egészségügyi karton 1 -->
    <egeszsegugyi_karton karton_id="k1" macska_ref="m1" allatorvos_ref="ao1">
        <datum>2023-08-16</datum>
        <kezeles_tipusa>Általános vizsgálat</kezeles_tipusa>
        <diagnozis>Egészséges, oltások beadva</diagnozis>
        <gyogyszerek>Kombinált oltás</gyogyszerek>
        <kovetkezo_vizsgalat>2024-08-16</kovetkezo_vizsgalat>
    </egeszsegugyi_karton>
    <!-- Örökbefogadások adatai, id, macska_ref (idegen kulcs), orokbefogado_ref (idegen
        kulcs), orokbefogadas_datuma, orokbefogasi_dij, statusz, megjegyzesek -->

    <!-- Örökbefogadás 1 -->
    <orokbefogadas macska_ref="m1" orokbefogado_ref="o1">
```

```
<orokbefogadas_datuma>2025-10-22</orokbefogadas_datuma>
```

```
<orokbefogasi_dij>10000</orokbefogasi_dij>
```

```
<statusz>Befejezett</statusz>
```

```
<megjegyzesek>Kiváló otthon, rendszeres kapcsolattartás</megjegyzesek>
```

```
</orokbefogadas>
```

```
<!-- Örökbefogadók adatai, id, teljes név, cím, telefonszám, email, születési dátum -->
```

```
<!-- Örökbefogadó 1 -->
```

```
<orokbefogado orokbefogado_id="o1">
```

```
<teljes_nev>
```

```
<vezeteknev>Teszt vezetéknév 1</vezeteknev>
```

```
<keresztnév>Teszt keresztnév 1</keresztnév>
```

```
</teljes_nev>
```

```
<cím>
```

```
<iranyitoszam>3529</iranyitoszam>
```

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

```
<utca>Csermőkei út</utca>
```

```
<hazszam>1</hazszam>
```

```
</cím>
```

```
<telefonszam>+36301234567</telefonszam>
```

```
<email>teszt.email.1@email.hu</email>
```

```
<szuletes_datum>2000-01-01</szuletes_datum>
```

```
</orokbefogado>
```

```
<!-- Macskák adatai, id, nev, fajta, születési dátum, szín, nem, bekerülési dátum,  
alkalmazott_ref (idegen kulcs) -->
```

```
<!-- Macska 1 -->
```

```
<macska macska_id="m1" alkalmazott_ref="a1">
```

```
<nev>Foltos</nev>
```

```
<fajta>Keverék</fajta>
```

```
<szuletes_datum>2023-01-01</szuletes_datum>  
<szin>Fekete-Fehér</szin>  
<nem>nőstény</nem>  
<bekerulesi_datum>2025-01-01</bekerulesi_datum>  
</macska>
```

```
<!-- Alkalmazottak adatai, id, teljes név, cím, telefonszam, beosztas, belepes_datuma,  
fizetes -->
```

```
<!-- Alkalmazott 1 -->  
<alkalmazott alkalmazott_id="a1">  
<teljes_nev>  
<vezeteknev>Gondozó</vezeteknev>  
<keresztnev>Neve</keresztnev>  
</teljes_nev>  
<cim>  
<iranyitoszam>3529</iranyitoszam>  
<varos>Miskolc</varos>  
<utca>Áfonyás utca</utca>  
<hazszam>1</hazszam>  
</cim>  
<telefonszam>+36301234567</telefonszam>  
<beosztas>Gondozó</beosztas>  
<belepes_datuma>2025-01-01</belepes_datuma>  
<fizetes>300000</fizetes>  
</alkalmazott>  
<!-- Állatorvosok adatai, id, teljes név, szakkepesites(ek), rendelo_neve, telefonszam,  
email -->
```

```
<!-- Állatorvos 1 -->  
<allatorvos allatorvos_id="ao1">  
<teljes_nev>
```

```

<vezeteknev>Dr. Orvos</vezeteknev>
<keresztnev>Neve</keresztnev>
</teljes_nev>
<szakkepesites>
<szakkepesites>Állatorvos diploma</szakkepesites>
<szakkepesites>Kisállat sebészeti</szakkepesites>
<szakkepesites>Orvos diploma</szakkepesites>
</szakkepesites>
<rendelo_neve>Orvos Állatklinika</rendelo_neve>
<telefonszam>+36301234000</telefonszam>
<email>orvos.neve@allatklinika.hu</email>
</allatorvos>

```

```
</VM3DIR_macskaMenhely>
```

Látható, hogy a forráskódban szerepel minden entitás (macskák, alkalmazottak, állatorvosok, örökbefogadók, örökbefogadások és az egészségügyi kartonok is), amik külön elemként jelennek meg. A kapcsolatokat pedig az idegen kulcsok jelzik (például a macska_ref, alkalmazott_ref). Az XML jól szemlélteti, hogy hogyan kapcsolódnak egymáshoz az egyedek.

1.4 Az XML dokumentum alapján XMLSchema készítése

Az XML dokumentumhoz kellett a validációt elősegítő sémát létrehozni. Ehhez először létrehoztam az egyszerű elemeket, amelyekre a felépítésben referálni tudok később, majd az ezekhez tartozó saját típusokat is. Ezekből összesen 36 darab lett, ebből 30 simpleType és 6 complexType. Van köztük 8 db regex pattern segítségével megadott típus leszűkítés (ID-k, irányítószám, email cím), illetve 2 db enumerationnal is (NemTipus, StatuszTipus). Továbbá 19 db minLength és maxLength korlátozással rendelkező típus, 2 db minInclusive/maxInclusive korlátosítású numerikus típus (FizetesTipus, OrokbefogasiDijTipus), valamint 2 db base típus leszűkítés (DatumTipus xs:date-re, IdopontTipus xs:dateTime-ra). Mindezek után létrehoztam magát a felépítést, amiben az egyszerű típusoknál referáltam a már korábban létrehozottakra, illetve adtam meg minimum és maximum előfordulást is

(minOccurs="1" maxOccurs="unbounded"), hiszen az XML-ben egy többszöri előfordulású elemek legalább 3 példány tartozik. A SzakkepesitesekTipus complexType-ban is megadtam a minOccurs="1" maxOccurs="unbounded" értékeket, mert egy állatorvosnak több szakképzettsége is lehet. minden egyes complexType után megadtam az attribútumokat is (use="required"), amikhez később létrehoztam a 6 db elsődleges kulcsot (xs:key), 5 db idegen kulcsot (xs:keyref), illetve 1 db 1:1 különleges kapcsolat referenciát (xs:unique) is a Macska és az Egészségügyi karton kapcsolathoz.

Az XMLSchema felépítése:

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

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

  <!-- Gyökérelem: A macskamenhely - Ez a macskamenhely fő eleme, itt van benne minden, a cicák, gazdák, dolgozók, állatorvosok meg az összes egészségügyi adat -->
  <xs:element name="VM3DIR_macskamenhely">
    <xs:complexType>
      <xs:sequence>
        <!-- Egészségügyi kartonok -->
        <xs:element name="egeszsegugyi_karton" type="EgeszsegugyiKartonTipus"
          minOccurs="1" maxOccurs="unbounded"/>
        <!-- Örökbefogadások (kapcsoló entitás, ID nélkül) -->
        <xs:element name="orokbefogadas" type="OrokbefogadasTipus"
          minOccurs="0" maxOccurs="unbounded"/>
        <!-- Örökbefogadók -->
        <xs:element name="orokbefogado" type="OrokbefogadoTipus" minOccurs="1"
          maxOccurs="unbounded"/>
      <!-- Macskák -->
```

```

<xs:element name="macska" type="MacskaTipus" minOccurs="1"
maxOccurs="unbounded"/>

    <!-- Alkalmazottak -->

        <xs:element name="alkalmazott" type="AlkalmazottTipus" minOccurs="1"
maxOccurs="unbounded"/>

            <!-- Állatorvosok -->

                <xs:element name="allatorvos" type="AllatorvosTipus" minOccurs="1"
maxOccurs="unbounded"/>

                </xs:sequence>

            </xs:complexType>

        <!-- Elsődleges kulcsok -->

            <!-- Elsődleges kulcs a macskának -->

            <xs:key name="macska_kulcs">
                <xs:selector xpath="macska"/>
                <xs:field xpath="@macska_id"/>
            </xs:key>

            <!-- Elsődleges kulcs az örökbefogadónak -->

            <xs:key name="orokbefogado_kulcs">
                <xs:selector xpath="orokbefogado"/>
                <xs:field xpath="@orokbefogado_id"/>
            </xs:key>

            <!-- Elsődleges kulcs az alkalmazottnak -->

            <xs:key name="alkalmazott_kulcs">
                <xs:selector xpath="alkalmazott"/>
                <xs:field xpath="@alkalmazott_id"/>
            </xs:key>

```

```
<!-- Elsődleges kulcs az állatorvosnak -->  
<xs:key name="allatorvos_kulcs">  
  <xs:selector xpath="allatorvos"/>  
  <xs:field xpath="@allatorvos_id"/>  
</xs:key>
```

```
<!-- Elsődleges kulcs az egészségügyi kartonnak -->  
<xs:key name="karton_kulcs">  
  <xs:selector xpath="egeszsegugyi_karton"/>  
  <xs:field xpath="@karton_id"/>  
</xs:key>
```

```
<!-- Idegen kulcsok -->
```

```
<!-- Idegen kulcs: FK(macska) -> PK(alkalmazott) -->  
<xs:keyref name="macska_alkalmazott_fk" refer="alkalmazott_kulcs">  
  <xs:selector xpath="macska"/>  
  <xs:field xpath="@alkalmazott_ref"/>  
</xs:keyref>
```

```
<!-- Idegen kulcs: FK(egészségügyi karton) -> PK(macska) -->  
<xs:keyref name="karton_macska_fk" refer="macska_kulcs">  
  <xs:selector xpath="egeszsegugyi_karton"/>  
  <xs:field xpath="@macska_ref"/>  
</xs:keyref>
```

```
<!-- Idegen kulcs: FK(egészségügyi karton) -> PK(állatorvos) -->  
<xs:keyref name="karton_allatorvos_fk" refer="allatorvos_kulcs">  
  <xs:selector xpath="egeszsegugyi_karton"/>  
  <xs:field xpath="@allatorvos_ref"/>  
</xs:keyref>
```

```

<!-- Idegen kulcs: FK(örökbefogadás) -> PK(macska) -->

<xs:keyref name="orokbefogadas_macska_fk" refer="macska_kulcs">
    <xs:selector xpath="orokbefogadas"/>
    <xs:field xpath="@macska_ref"/>
</xs:keyref>

<!-- Idegen kulcs: FK(örökbefogadás) -> PK(örökbefogadó) -->
<xs:keyref name="orokbefogadas_orokbefogado_fk"
refer="orokbefogado_kulcs">
    <xs:selector xpath="orokbefogadas"/>
    <xs:field xpath="@orokbefogado_ref"/>
</xs:keyref>

<!-- 1:1 Macska — Egészségügyi karton (egy macskához 1 karton jár) -->
<xs:unique name="karton_macska_unique">
    <xs:selector xpath="egeszsegugyi_karton"/>
    <xs:field xpath="@macska_ref"/>
</xs:unique>
</xs:element>

<!-- Egyszerű típusok -->

<!-- ID-k -->

<!-- Macska ID típusa: string, m betűvel kell kezdődni, majd számjegyek -->
<xs:simpleType name="MacskalDIpus">
    <xs:restriction base="xs:string">
        <xs:pattern value="m\d+/">
    </xs:restriction>
</xs:simpleType>

<!-- Örökbefogadó ID típusa: string, o betűvel kell kezdődni, majd számjegyek -->

```

```
<xs:simpleType name="OrokbefogadoldTipus">  
  <xs:restriction base="xs:string">  
    <xs:pattern value="o\d+/">  
  </xs:restriction>  
</xs:simpleType>
```

```
<!-- Alkalmazott ID típusa: string, a betűvel kell kezdődni, majd számjegyek -->  
<xs:simpleType name="AlkalmazottIdTipus">  
  <xs:restriction base="xs:string">  
    <xs:pattern value="a\d+/">  
  </xs:restriction>  
</xs:simpleType>
```

```
<!-- Állatorvos ID típusa: string, ao betűkkel kell kezdődni, majd számjegyek -->  
<xs:simpleType name="AllatorvosIdTipus">  
  <xs:restriction base="xs:string">  
    <xs:pattern value="ao\d+/">  
  </xs:restriction>  
</xs:simpleType>
```

```
<!-- Egészségügyi karton ID típusa: string, k betűvel kell kezdődni, majd  
számjegyek -->  
<xs:simpleType name="KartonIdTipus">  
  <xs:restriction base="xs:string">  
    <xs:pattern value="k\d+/">  
  </xs:restriction>  
</xs:simpleType>
```

```
<!-- Nevek -->
```

```
<!-- Név típusa: string, minimum 1 karakter, maximum 100 karakter -->  
<xs:simpleType name="NevTipus">
```

```
<xs:restriction base="xs:string">  
  <xs:minLength value="1"/>  
  <xsmaxLength value="100"/>  
</xs:restriction>  
</xs:simpleType>
```

```
<!-- Vezetéknév típusa: string, minimum 1 karakter, maximum 50 karakter -->  
<xs:simpleType name="VezeteknevTipus">  
  <xs:restriction base="xs:string">  
    <xs:minLength value="1"/>  
    <xsmaxLength value="50"/>  
</xs:restriction>  
</xs:simpleType>
```

```
<!-- Keresztnév típusa: string, minimum 1 karakter, maximum 50 karakter -->  
<xs:simpleType name="KeresztnevTipus">  
  <xs:restriction base="xs:string">  
    <xs:minLength value="1"/>  
    <xsmaxLength value="50"/>  
</xs:restriction>  
</xs:simpleType>
```

```
<!-- Cím elemei -->
```

```
<!-- Irányítószám típusa: string, pontosan 4 számjegy -->  
<xs:simpleType name="IranyitoszamTipus">  
  <xs:restriction base="xs:string">  
    <xs:pattern value="\d{4}"/>  
</xs:restriction>  
</xs:simpleType>
```

```
<!-- Város típusa: string, minimum 1 karakter, maximum 50 karakter -->
```

```
<xs:simpleType name="VarosTipus">  
  <xs:restriction base="xs:string">  
    <xs:minLength value="1"/>  
    <xsmaxLength value="50"/>  
  </xs:restriction>  
</xs:simpleType>
```

```
<!-- Utca típusa: string, minimum 1 karakter, maximum 100 karakter -->  
<xs:simpleType name="UtcaTipus">  
  <xs:restriction base="xs:string">  
    <xs:minLength value="1"/>  
    <xsmaxLength value="100"/>  
  </xs:restriction>  
</xs:simpleType>
```

```
<!-- Házszám típusa: string, minimum 1 karakter, maximum 20 karakter -->  
<xs:simpleType name="HazszamTipus">  
  <xs:restriction base="xs:string">  
    <xs:minLength value="1"/>  
    <xsmaxLength value="20"/>  
  </xs:restriction>  
</xs:simpleType>
```

```
<!-- Email címek -->
```

```
<!-- Email cím típusa: string, tartalmaznia kell @ jelet és pontot, valid email  
formátum -->  
<xs:simpleType name="EmailTipus">  
  <xs:restriction base="xs:string">  
    <xs:pattern value="[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,}" />  
  </xs:restriction>  
</xs:simpleType>
```

```
<!-- Telefonszámok -->
```

```
<!-- Telefonszám típusa: string, minimum 1 karakter, maximum 20 karakter -->
```

```
<xs:simpleType name="TelefonszamTipus">  
  <xs:restriction base="xs:string">  
    <xs:minLength value="1"/>  
    <xs:maxLength value="20"/>  
  </xs:restriction>  
</xs:simpleType>
```

```
<!-- Dátumok -->
```

```
<!-- Dátum típusa: xs:date formátum (YYYY-MM-DD) -->
```

```
<xs:simpleType name="DatumTipus">  
  <xs:restriction base="xs:date"/>  
</xs:simpleType>
```

```
<!-- Időpont típusa: xs:dateTime formátum (dátum és idő együtt) -->
```

```
<xs:simpleType name="IdopontTipus">  
  <xs:restriction base="xs:dateTime"/>  
</xs:simpleType>
```

```
<!-- Macskák -->
```

```
<!-- Macskafajta típusa: string, minimum 1 karakter, maximum 100 karakter -->
```

```
<xs:simpleType name="FajtaTipus">  
  <xs:restriction base="xs:string">  
    <xs:minLength value="1"/>  
    <xs:maxLength value="100"/>  
  </xs:restriction>  
</xs:simpleType>
```

```
<!-- Nem típusa: string, csak "hím" vagy "nőstény" értékek lehetnek -->  
<xs:simpleType name="NemTipus">  
  <xs:restriction base="xs:string">  
    <xs:enumeration value="hím"/>  
    <xs:enumeration value="nőstény"/>  
  </xs:restriction>  
</xs:simpleType>
```

```
<!-- Szín típusa: string, minimum 1 karakter, maximum 100 karakter -->  
<xs:simpleType name="Szintipus">  
  <xs:restriction base="xs:string">  
    <xs:minLength value="1"/>  
    <xs:maxLength value="100"/>  
  </xs:restriction>  
</xs:simpleType>
```

```
<!-- Alkalmazottak -->
```

```
<!-- Beosztás típusa: string, minimum 1 karakter, maximum 100 karakter  
(alkalmazott beosztása) -->  
<xs:simpleType name="BeosztasTipus">  
  <xs:restriction base="xs:string">  
    <xs:minLength value="1"/>  
    <xs:maxLength value="100"/>  
  </xs:restriction>  
</xs:simpleType>
```

```
<!-- Fizetés típusa: positiveInteger, minimum 1, maximum 10000000 (forintban) -->  
<xs:simpleType name="FizetesTipus">  
  <xs:restriction base="xs:positiveInteger">  
    <xs:minInclusive value="1"/>
```

```
<xs:maxInclusive value="10000000"/>  
</xs:restriction>  
</xs:simpleType>
```

```
<!-- Szakképzettség típusa: string, minimum 1 karakter, maximum 200 karakter  
(egy szakképzettség neve) -->  
<xs:simpleType name="SzakkepesitesTipus">  
  <xs:restriction base="xs:string">  
    <xs:minLength value="1"/>  
    <xs:maxLength value="200"/>  
  </xs:restriction>  
</xs:simpleType>
```

```
<!-- Rendelő neve típusa: string, minimum 1 karakter, maximum 100 karakter -->  
<xs:simpleType name="RendeloNeveTipus">  
  <xs:restriction base="xs:string">  
    <xs:minLength value="1"/>  
    <xs:maxLength value="100"/>  
  </xs:restriction>  
</xs:simpleType>
```

```
<!-- Egészségügyi kartonok -->
```

```
<!-- Kezelés típusa: string, minimum 1 karakter, maximum 200 karakter -->  
<xs:simpleType name="KezelesTipusaTipus">  
  <xs:restriction base="xs:string">  
    <xs:minLength value="1"/>  
    <xs:maxLength value="200"/>  
  </xs:restriction>  
</xs:simpleType>
```

```
<!-- Diagnózis típusa: string, minimum 1 karakter, maximum 500 karakter -->
```

```
<xs:simpleType name="DiagnozisTipus">  
  <xs:restriction base="xs:string">  
    <xs:minLength value="1"/>  
    <xsmaxLength value="500"/>  
  </xs:restriction>  
</xs:simpleType>
```

```
<!-- Gyógyszerek típusa: string, minimum 1 karakter, maximum 500 karakter -->  
<xs:simpleType name="GyogyszerekTipus">  
  <xs:restriction base="xs:string">  
    <xs:minLength value="1"/>  
    <xsmaxLength value="500"/>  
  </xs:restriction>  
</xs:simpleType>
```

```
<!-- Örökbefogadások -->
```

```
<!-- Örökbefogadási díj típusa: positiveInteger, minimum 1 (forintban) -->  
<xs:simpleType name="OrokbefogasiDijTipus">  
  <xs:restriction base="xs:positiveInteger">  
    <xs:minInclusive value="1"/>  
  </xs:restriction>  
</xs:simpleType>
```

```
<!-- Státusz típusa: string, csak "Folyamatban", "Jóváhagyva", "Befejezett",  
"Visszautasítva" értékek -->  
<xs:simpleType name="StatuszTipus">  
  <xs:restriction base="xs:string">  
    <xs:enumeration value="Folyamatban"/>  
    <xs:enumeration value="Jóváhagyva"/>  
    <xs:enumeration value="Befejezett"/>  
    <xs:enumeration value="Visszautasítva"/>
```

```
</xs:restriction>
```

```
</xs:simpleType>
```

```
<!-- Megjegyzések típusa: string, minimum 1 karakter, maximum 1000 karakter -->
```

```
<xs:simpleType name="MegjegyzesekTipus">  
  <xs:restriction base="xs:string">  
    <xs:minLength value="1"/>  
    <xs:maxLength value="1000"/>  
  </xs:restriction>  
</xs:simpleType>
```

```
<!-- Összetett típusok -->
```

```
<!-- Teljes név komplex típusa: vezetéknév és keresztnév külön elemként -->
```

```
<xs:complexType name="TeljesNevTipus">  
  <xs:sequence>  
    <xs:element name="vezeteknev" type="VezeteknevTipus"/>  
    <xs:element name="keresztnev" type="KeresztnevTipus"/>  
  </xs:sequence>  
</xs:complexType>
```

```
<!-- Cím komplex típusa: irányítószám, város, utca, házszám elemekből áll -->
```

```
<xs:complexType name="CimTipus">  
  <xs:sequence>  
    <xs:element name="iranyitoszam" type="IranyitoszamTipus"/>  
    <xs:element name="varos" type="VarosTipus"/>  
    <xs:element name="utca" type="UtcaTipus"/>  
    <xs:element name="hazszam" type="HazszamTipus"/>  
  </xs:sequence>  
</xs:complexType>
```

```

<!-- Szakképzettségek komplex típusa: több szakképzettség elemet tartalmazhat
(minOccurs=1, maxOccurs=unbounded) -->
<xs:complexType name="SzakkepesitesekTipus">
  <xs:sequence>
    <xs:element name="szakkepesites" type="SzakkepesitesTipus" minOccurs="1"
maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>

<!-- Entitás típusok -->

<!-- Macskák -->

<!-- Macska komplex típusa: nev, fajta, szuletes_datum, szin, nem,
bekerulesi_datum elemekből és macska_id, alkalmazott_ref attribútumokból áll -->
<xs:complexType name="MacskaTipus">
  <xs:sequence>
    <xs:element name="nev" type="NevTipus"/>
    <xs:element name="fajta" type="FajtaTipus"/>
    <xs:element name="szuletes_datum" type="DatumTipus"/>
    <xs:element name="szin" type="SzinTipus"/>
    <xs:element name="nem" type="NemTipus"/>
    <xs:element name="bekerulesi_datum" type="DatumTipus"/>
  </xs:sequence>
  <xs:attribute name="macska_id" type="MacskaldTipus" use="required"/>
    <xs:attribute name="alkalmazott_ref" type="AlkalmazottIdTipus"
use="required"/>
</xs:complexType>

<!-- Örökbefogadók -->

```

```

<!-- Örökbefogadó komplex típusa: teljes_nev, cim, telefonszam, email,
szuletes_datum elemekből és orokbefogado_id attribútumból áll -->
<xs:complexType name="OrokbefogadoTipus">
  <xs:sequence>
    <xs:element name="teljes_nev" type="TeljesNevTipus"/>
    <xs:element name="cim" type="CimTipus"/>
    <xs:element name="telefonszam" type="TelefonszamTipus"/>
    <xs:element name="email" type="EmailTipus"/>
    <xs:element name="szuletes_datum" type="DatumTipus"/>
  </xs:sequence>
  <xs:attribute name="orokbefogado_id" type="OrokbefogadoldTipus"
use="required"/>
</xs:complexType>

```

```
<!-- Alkalmazottak -->
```

```

<!-- Alkalmazott komplex típusa: teljes_nev, cim, telefonszam, beosztas,
belepes_datuma, fizetes elemekből és alkalmazott_id attribútumból áll -->
<xs:complexType name="AlkalmazottTipus">
  <xs:sequence>
    <xs:element name="teljes_nev" type="TeljesNevTipus"/>
    <xs:element name="cim" type="CimTipus"/>
    <xs:element name="telefonszam" type="TelefonszamTipus"/>
    <xs:element name="beosztas" type="BeosztasTipus"/>
    <xs:element name="belepes_datuma" type="DatumTipus"/>
    <xs:element name="fizetes" type="FizetesTipus"/>
  </xs:sequence>
  <xs:attribute name="alkalmazott_id" type="AlkalmazottIdTipus" use="required"/>
</xs:complexType>

```

```
<!-- Állatorvosok -->
```

```

<!-- Állatorvos komplex típusa: teljes_nev, szakkepesites (több is lehet),
rendelo_neve, telefonszam, email elemekből és allatorvos_id attribútumból áll -->
<xs:complexType name="AllatorvosTipus">
  <xs:sequence>
    <xs:element name="teljes_nev" type="TeljesNevTipus"/>
    <xs:element name="szakkepesites" type="SzakkepesitesekTipus"/>
    <xs:element name="rendelo_neve" type="RendeloNeveTipus"/>
    <xs:element name="telefonszam" type="TelefonszamTipus"/>
    <xs:element name="email" type="EmailTipus"/>
  </xs:sequence>
  <xs:attribute name="allatorvos_id" type="AllatorvosIdTipus" use="required"/>
</xs:complexType>

```

<!-- Egészségügyi kartonok -->

```

<!-- Egészségügyi karton komplex típusa: datum, kezeles_tipusa, diagnozis,
gyogyszerek, kovetkezo_vizsgalat elemekből és karton_id, macska_ref,
allatorvos_ref attribútumokból áll -->
<xs:complexType name="EgeszsegugyiKartonTipus">
  <xs:sequence>
    <xs:element name="datum" type="DatumTipus"/>
    <xs:element name="kezeles_tipusa" type="KezelesTipusaTipus"/>
    <xs:element name="diagnozis" type="DiagnozisTipus"/>
    <xs:element name="gyogyszerek" type="GyogyszerekTipus"/>
    <xs:element name="kovetkezo_vizsgalat" type="DatumTipus"/>
  </xs:sequence>
  <xs:attribute name="karton_id" type="KartontIdTipus" use="required"/>
  <xs:attribute name="macska_ref" type="MacskaldTipus" use="required"/>
  <xs:attribute name="allatorvos_ref" type="AllatorvosIdTipus" use="required"/>
</xs:complexType>

```

<!-- Örökbefogadások -->

```

<!-- Örökbefogadás komplex típusa (kapcsoló entitás, ID nélkül):
orokbefogadas_datuma, orokbefogasi_dij, statusz, megjegyzesek elemekből és
macska_ref, orokbefogado_ref attribútumokból áll -->

<xs:complexType name="OrokbefogadasTipus">
  <xs:sequence>
    <xs:element name="orokbefogadas_datuma" type="DatumTipus"/>
    <xs:element name="orokbefogasi_dij" type="OrokbefogasiDijTipus"/>
    <xs:element name="statusz" type="StatuszTipus"/>
    <xs:element name="megjegyzesek" type="MegjegyzesekTipus"/>
  </xs:sequence>
  <xs:attribute name="macska_ref" type="MacskaldTipus" use="required"/>
  <xs:attribute name="orokbefogado_ref" type="OrokbefogadoldTipus"
use="required"/>
</xs:complexType>

</xs:schema>

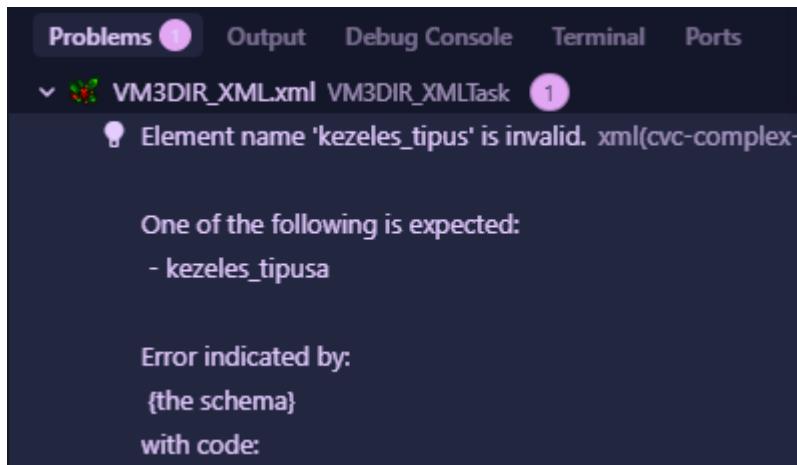
```

Úgy éreztem, hogy a teljes séma feltüntetése fogja tükrözni minden fontosat egy ilyen fájl elkészítésében. Hiszen innen is észrevehető, hogy minden kisebb és nagyobb rész a kódban befolyásolni fogja a végeredményt. A saját típusok létrehozásával teljes mértékben személyre szabható az, hogy miképp várjuk vissza az adatokat az XML fájlunkban, így egy nagyon jól strukturált és átlátható dokumentumot fogunk majd visszakapni.

1.5 Validáció

A validáció megtörténik a Visual Studio Code alapján, a „Problems” fülön keresztül üzen, ha valami hibás. Fontosnak tartottam, hogy ezt külön kiemeljem, mivel tökéletesen szemlélteti a programunk, hogy hol van hiba, és mit kell kijavítani. Számomra nagyon megkönnyíti a hibakeresést és a megoldását is.

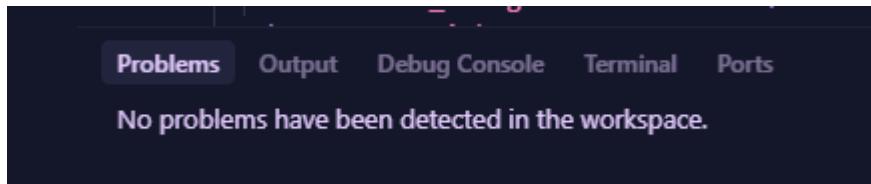
Ha hiba van az XML-ben, akkor ezt látjuk:



3. ábra: Hibaüzenet 1

A 3. ábrán is láthatjuk, hogy visszajelez számunkra a program.

Ha viszont minden rendben van, akkor ezt kell látnunk:



4. ábra: Jó megoldás

A 4. ábrán is jól látszik, hogyha sikeresen validáltuk a fájlt és nem talált benne a séma hibát, akkor ezt a szöveget fogjuk látni.

2. Feladat

A 2. feladat célja, hogy az XML dokumentumot tudjuk szerkeszteni, lekérdezéseket írni hozzá, vagy kiírni az egészet a konzolra.

2.1 Adatolvasás

A VM3DIRDOMRead.java fájl célja, hogy az XML dokumentumot feldolgozza, majd kiírja a konzolra. Először beolvassuk az XML fájlt a DocumentBuilder segítségével. Normalizáljuk ezután a dokumentumot, hogy a csomópontok összevonásra kerüljenek, így egységes lesz a feldolgozás. Az entitások feldolgozása következik, ahol minden egyedet feldolgozunk az XML-ből. Miután megtörtént a normalizálás és az entitások feldolgozása, kiírjuk őket blokk formában a konzolra.

Az XML dokumentum beolvasása:

```
File xmlFile = new File("VM3DIR_XMLTask/VM3DIR_XML.xml");
```

```
DocumentBuilderFactory factory = DocumentBuilderFactory.newInstance();
```

```
factory.setNamespaceAware(true);
```

```
DocumentBuilder dBuilder = factory.newDocumentBuilder();
```

```
Document doc = dBuilder.parse(xmlFile);
```

```
doc.getDocumentElement().normalize();
```

Az entitások feldolgozása egy ciklussal:

```
NodeList kartonList = doc.getElementsByTagName("egeszsegugyi_karton");
```

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

```
    Node nNode = kartonList.item(i);
```

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

```
        Element elem = (Element) nNode;
```

```
        System.out.println("Karton ID: " + elem.getAttribute("karton_id"));
```

```
        Node datumNode = elem.getElementsByTagName("datum").item(0);
```

```
        if (datumNode != null) {
```

```

        System.out.println("Dátum: " + datumNode.getTextContent());
    }
}
}

```

A beágyazott elemek kezelése:

```

Element teljesNevElem = (Element)
elem.getElementsByTagName("teljes_nev").item(0);
if (teljesNevElem != null) {
    Node vezeteknevNode =
teljesNevElem.getElementsByTagName("vezeteknev").item(0);
    Node keresztnévNode =
teljesNevElem.getElementsByTagName("keresztnév").item(0);
    if (vezeteknevNode != null && keresztnévNode != null) {
        System.out.println("Név: " + vezeteknevNode.getTextContent() + " " +
keresztnévNode.getTextContent());
    }
}

```

2.2 Adat-lekérdezés

A VM3DIRDOMQuery.java különböző lekérdezéseket végez az XML dokumentumon. 5 különböző lekérdezést valósítottam meg.

Először beolvassuk az XML dokumentumot, elvégezzük a lekérdezéseket, a végeredményt pedig kiíratjuk a konzolra.

Az első lekérdezés, összes macska neve és a fajtája, id-ja:

```

 NodeList macskaList = doc.getElementsByTagName("macska");
for (int i = 0; i < macskaList.getLength(); i++) {
    Element macska = (Element) macskaList.item(i);
    String macskaid = macska.getAttribute("macska_id");
    Node nevNode = macska.getElementsByTagName("nev").item(0);
    Node fajtaNode = macska.getElementsByTagName("fajta").item(0);
    if (nevNode != null && fajtaNode != null) {

```

```

        System.out.println(" - " + nevNode.getTextContent() + " (" +
fajtaNode.getTextContent() + ") [ID: " + macskald + "]");

    }

}

```

Második lekérdezés, szűrjük a befejezett státuszú örökbefogadásokra a lekérdezésünket:

```

 NodeList orokbefogadasList = doc.getElementsByTagName("orokbefogadas");
for (int i = 0; i < orokbefogadasList.getLength(); i++) {
    Element orokbefogadas = (Element) orokbefogadasList.item(i);
    Node statuszNode = orokbefogadas.getElementsByTagName("statusz").item(0);

    if (statuszNode != null && "Befejezett".equals(statuszNode.getTextContent())) {
        String macskaRef = orokbefogadas.getAttribute("macska_ref");
        String orokbefogadoRef = orokbefogadas.getAttribute("orokbefogado_ref");
        Node datumNode = orokbefogadas.getElementsByTagName("orokbefogadas_datuma").item(0);
        Node dijNode = orokbefogadas.getElementsByTagName("orokbefogasi_dij").item(0);
    }
}

```

```

System.out.println(" - Örökbefogadás (Macska: " + macskaRef + ", Örökbefogadó: " +
orokbefogadoRef + ")");
System.out.println(" Macska ref: " + macskaRef);
System.out.println(" Örökbefogadó ref: " + orokbefogadoRef);
if (datumNode != null) {
    System.out.println(" Dátum: " + datumNode.getTextContent());
}
if (dijNode != null) {
    System.out.println(" Díj: " + dijNode.getTextContent() + " Ft");
}
System.out.println();
}

```

Harmadik lekérdezés, feldolgozzuk a beágyazott elemeket, mint a teljes név és a szakképesítés:

```

Element           teljesNevElem          =      (Element)
allatorvos.getElementsByTagName("teljes_nev").item(0);

String nev = "";
if (teljesNevElem != null) {
    Node           vezeteknevNode      =
teljesNevElem.getElementsByTagName("vezeteknev").item(0);
    Node           keresztnévNode      =
teljesNevElem.getElementsByTagName("keresztnév").item(0);
    if (vezeteknevNode != null && keresztnévNode != null) {
        nev = vezeteknevNode.getTextContent() + " " + keresztnévNode.getTextContent();
    }
}

```

```

Element           szakkepesitesElem      =      (Element)
allatorvos.getElementsByTagName("szakkepesites").item(0);

if (szakkepesitesElem != null) {
    NodeList       szakkepesitesList     =
szakkepesitesElem.getElementsByTagName("szakkepesites");
    for (int j = 0; j < szakkepesitesList.getLength(); j++) {
        System.out.println(" • " + szakkepesitesList.item(j).getTextContent());
    }
}

```

Negyedik lekérdezés, egy összetett lekérdezés több egyeden, például azokat a macskákat, akikhez tartozik „betegség” diagnózis:

```

NodeList kartonList = doc.getElementsByTagName("egeszsegugyi_karton");
for (int i = 0; i < kartonList.getLength(); i++) {
    Element karton = (Element) kartonList.item(i);
    Node diagnozisNode = karton.getElementsByTagName("diagnozis").item(0);
    if (diagnozisNode != null) {
        String diagnozis = diagnozisNode.getTextContent();
        if (diagnozis.contains("Betegség")) {

```

```

String macskaRef = karton.getAttribute("macska_ref");
}

NodeList macskaList2 = doc.getElementsByTagName("macska");
for (int j = 0; j < macskaList2.getLength(); j++) {
    Element macska = (Element) macskaList2.item(j);
    if (macskaRef.equals(macska.getAttribute("macska_id"))) {
        }
    }
}
}
}

```

Ötödik lekérdezés, szűrünk azokra az örökbefogadókra, akik Miskolcon laknak:

```

Element cimElem = (Element) orokbefogado.getElementsByTagName("cim").item(0);
if (cimElem != null) {
    Node varosNode = cimElem.getElementsByTagName("varos").item(0);
    if (varosNode != null && "Miskolc".equals(varosNode.getTextContent())) {
        // Eredmények kiírása
    }
}

```

2.3 Adatmódosítás

A VM3DIRDOMModify.java lehetővé teszi, hogy módosításokat hajthassunk végre egy XML dokumentumon. Öt különböző módosítást készítettem, a módosításokat elmenti az eredeti XML fájlunkba.

Először beolvassuk az XML fájlt, végrehajtjuk rajta a módosításokat, amiket kiíratunk a konzolra és módosítjuk a fájlt is.

Első módosítás, az m1 id-jú macska nevét módosítjuk:

```

NodeList macskaList = doc.getElementsByTagName("macska");
for (int i = 0; i < macskaList.getLength(); i++) {
    Element macska = (Element) macskaList.item(i);
    if ("m1".equals(macska.getAttribute("macska_id"))) {
        Node nevNode = macska.getElementsByTagName("nev").item(0);

```

```

    if (nevNode != null) {
        String regiNev = nevNode.getTextContent();
        nevNode.setTextContent("Cirmos");
        System.out.println(" - Macska m1 neve módosítva: " + regiNev + " -> 'Cirmos'");
    }
    break;
}
}

```

Második módosítás, módosítjuk az o2 örökbefogadás státuszát:

```

 NodeList orokbefogadasList = doc.getElementsByTagName("orokbefogadas");
 for (int i = 0; i < orokbefogadasList.getLength(); i++) {
     Element orokbefogadas = (Element) orokbefogadasList.item(i);
     if ("m2".equals(orokbefogadas.getAttribute("macska_ref")) &&
         "o2".equals(orokbefogadas.getAttribute("orokbefogado_ref"))) {
         Node statuszNode = orokbefogadas.getElementsByTagName("statusz").item(0);
         if (statuszNode != null) {
             String regiStatusz = statuszNode.getTextContent();
             statuszNode.setTextContent("Jóváhagyva");
             System.out.println(" - Örökbefogadás (m2-o2) státusza módosítva: " +
                 regiStatusz + " -> 'Jóváhagyva')");
         }
     }
     break;
}

```

Harmadik módosítás, ao1 állatorvos telefonszámának módosítása:

```

 NodeList allatorvosList = doc.getElementsByTagName("allatorvos");
 for (int i = 0; i < allatorvosList.getLength(); i++) {
     Element allatorvos = (Element) allatorvosList.item(i);
     if ("ao1".equals(allatorvos.getAttribute("allatorvos_id"))){
         Node telefonszamNode =
         allatorvos.getElementsByTagName("telefonszam").item(0);
         if (telefonszamNode != null) {

```

```

        String regiTelefon = telefonszamNode.getTextContent();
        telefonszamNode.setTextContent("+36309999999");
        System.out.println(" - Állatorvos ao1 telefonszáma módosítva: " + regiTelefon + ""
-> '+36309999999");
    }
    break;
}
}

```

Negyedik módosítás, k2 egészségügyi karton diagnózisának módosítása:

```

 NodeList kartonList = doc.getElementsByTagName("egeszsegugyi_karton");
for (int i = 0; i < kartonList.getLength(); i++) {
    Element karton = (Element) kartonList.item(i);
    if ("k2".equals(karton.getAttribute("karton_id"))) {
        Node diagnozisNode = karton.getElementsByTagName("diagnozis").item(0);
        if (diagnozisNode != null) {
            String regiDiagnozis = diagnozisNode.getTextContent();
            diagnozisNode.setTextContent("Betegség, hányás - gyógyult");
            System.out.println(" - Karton k2 diagnózisa módosítva:");
            System.out.println(" Régi: " + regiDiagnozis + "'");
            System.out.println(" Új: 'Betegség, hányás - gyógyult'");
        }
        break;
    }
}

```

Ötödik módosítás, hozzáadunk egy új szakképesítést az ao2 állatorvoshoz:

```

for (int i = 0; i < allatorvosList.getLength(); i++) {
    Element allatorvos = (Element) allatorvosList.item(i);
    if ("ao2".equals(allatorvos.getAttribute("allatorvos_id"))) {
        Element szakkepesitesElem = (Element)
allatorvos.getElementsByTagName("szakkepesites").item(0);
        if (szakkepesitesElem != null) {
            // Új szakképzettség elem létrehozása

```

```
Element ujSzakkepesites = doc.createElement("szakkepesites");
ujSzakkepesites.appendChild(doc.createTextNode("Kardiológia"));
szakkepesitesElem.appendChild(ujSzakkepesites);

System.out.println(" - Új szakképzettség hozzáadva az állatorvos ao2-hez:
'Kardiológia');");
}

break;
}

}
```