JEGYZŐKÖNYV

Adatkezelés XML környezetben
Féléves feladat

Filmadatbázis

Tóth Dávid Olivér
YIF399
2022

Tartalomjegyzék

Témakör leírása 3
Első feladat 4
Az adatbázis ER modell 4
Az adatbázis konvertálása XDM modellre 5
Az XDM modell alapján XML dokumentum készítése 6
Az XML dokumentum alapján XMLSchema készítése 11
Második feladat 16
Adatolvasás 16
Adatmódosítás 19
Adatlekérdezés 23

Témakör leírása

A választott beadandó feladatom filmek nyilvántartásához való xml dokumentum elkészítése. Ezzel az XML dokumentummal az emberek többet tudhatnak meg a filmek és az azzal kapcsolatos személyek és stúdiókról. Nem csupán a filmek, de az ahhoz tartozó stúdiók és rendezők illetve persze a benne játszó színészek is belekerültek a dokumentumba. A filmekről a címük és a megjelenés pontos dátuma mellett eltárolásra kerül még a terjedelmek és a bevételük. A stúdiókról eltároljuk a nevüket, az alapításuk évét, a vezérigazgatójának a nevét és az országot és várost ahol a központja található ennek az egésznek. Tárolásra kerülnek adatok a film forgatókönyvéről is, ezek a forgatókönyv oldalszáma, az hogy elektronikus vagy nyomtatott formában adták-e ki, és nyomtatott formátum esetén hány példány készült belőle. A színészekről csak néhány adatott tarolunk, amelyek a nevük, az életkoruk és legutoljára a lakhelyük országra és városra pontosan. A rendezőkről hasonlóan a színészekhez nem tárolunk túl sok információt csak, mint a színészeknél a nevüket, a korukat, valamelyest a tartózkodásuknak a helyét. Ezeken kívül még rendelkeznek ID-val is, amely a pontos azonosításért fellel, hiszen több filmből is készült újrafeldolgozás, amelyek ugyan azt a címet viselik, mint az előző. Illetve annak sincs kizárva a lehetősége, hogy két vagy több ugyanolyan nevű színész vagy rendező legyen. Illetve nem egy filmbe nem egy szereplő van, így azokat az ID segítségével könnyen rendszerezhető és még a karakter nevét is el tudjuk menteni melléjük.

Első feladat

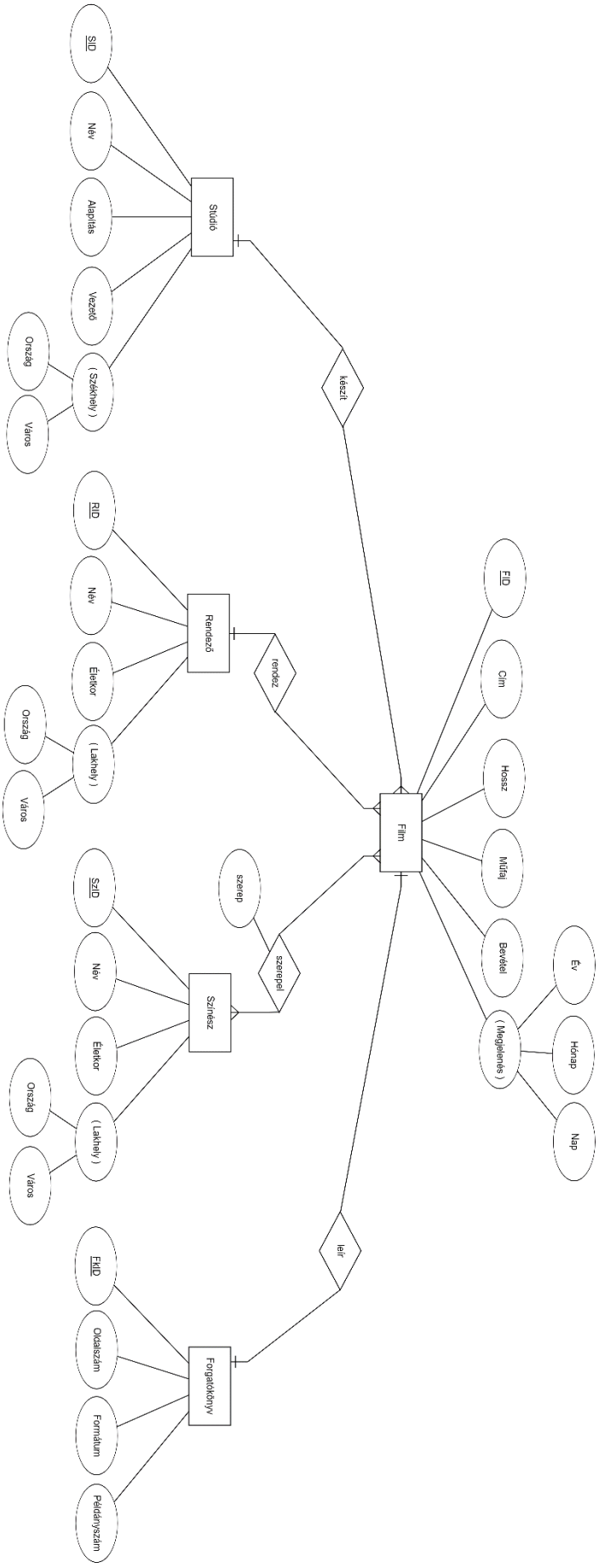
Az adatbázis ER modell

Az ER kapcsolati modellünkben, minden nagyobb egységet entitásként tüntetünk fel, ezek a következők:

- **Filmek**, ez a fő egység
- **Forgatókönyvek**
- **Stúdiók**
- **Rendezők**
- **Színészek**

ezeknek az egyedeknek pedig attribútumaik vannak, mint pl. egy film címe, ezek az attribútumok többnyire egyszerűek, de van néhány összetett attribútuma az egyedeknek

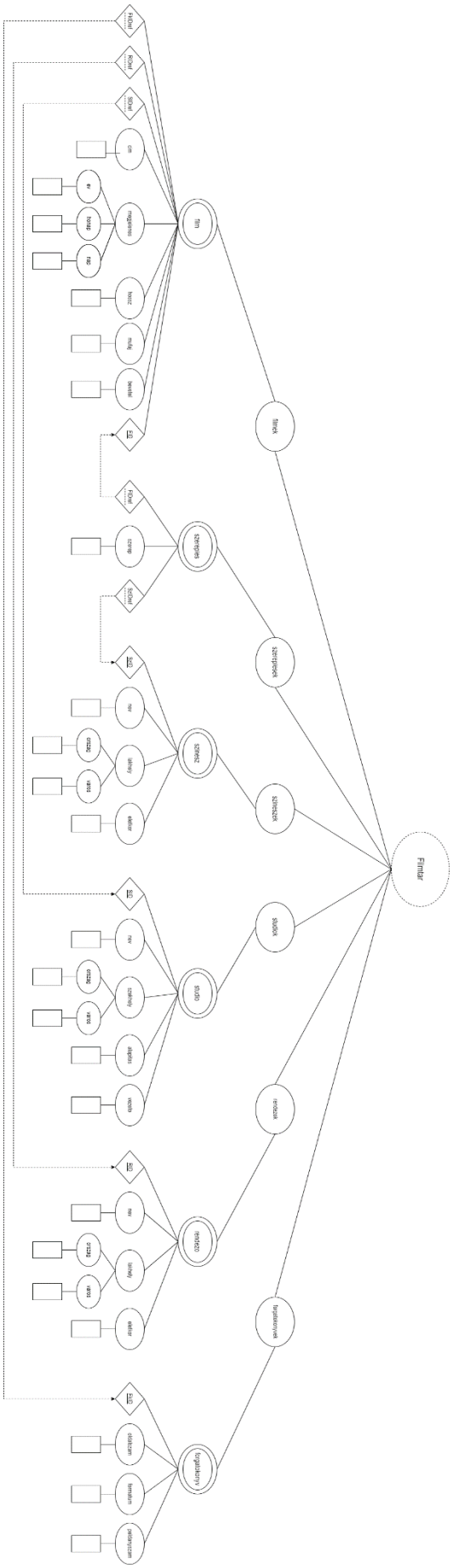
Nagyobb méret érdekében a kép elforgatva



Az adatbázis konvertálása XDM modellre

Az ER modellben szereplő egyedeket ellipszissel,
a tulajdonságokat ellipszissel és téglalappal,
a kulcsokat rombusszal ábrázoljuk.
A többszörös előfordulást dupla vonallal, vagyis
koncentrikus ellipszisekkel jelezzük
Az adatbázis root eleme a *filmtar* nevet kapja.

Nagyobb méret érdekében a kép elforgatva



Az XDM modell alapján XML dokumentum készítése

Az XDM modell alapján késztettem el az XML dokumentumot, kezdve a root elementtel. Figyeltem arra, hogy a gyerekelemekből legalább 3 legyen, mert úgy szemléletes.

```xml
<?xml version="1.0" encoding="UTF-8" standalone="no"?>

<filmtar xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="./XMLSchemaYIF399.xsd">
    <rendezok>
        <rendezo RID="r01">
            <nev>David Zucker</nev>
            <eletkor>74</eletkor>
            <lakhely>
                <orszag>USA</orszag>
                <varos>Los Angeles</varos>
            </lakhely>
        </rendezo>
        <rendezo RID="r02">
            <nev>Peter Segal</nev>
            <eletkor>58</eletkor>
            <lakhely>
                <orszag>USA</orszag>
                <varos>Chicago</varos>
            </lakhely>
        </rendezo>
        <rendezo RID="r03">
            <nev>Keenen Ivory Wayans</nev>
            <eletkor>63</eletkor>
            <lakhely>
                <orszag>USA</orszag>
                <varos>New York</varos>
            </lakhely>
        </rendezo>
    </rendezok>
    <szineszek>
        <szinesz SzID="sz01">
            <nev>Leslie Nielsen</nev>
            <eletkor>84</eletkor>
            <lakhely>
                <orszag>USA</orszag>
                <varos>Fort Lauderdale</varos>
            </lakhely>
        </szinesz>
        <szinesz SzID="sz02">
            <nev>Priscilla Presley</nev>
```

```
            <eletkor>76</eletkor>
            <lakhely>
                <orszag>USA</orszag>
                <varos>Los Angeles</varos>
            </lakhely>
        </szinesz>
        <szinesz SzID="sz03">
            <nev>Marlon Wayans</nev>
            <eletkor>49</eletkor>
            <lakhely>
                <orszag>USA</orszag>
                <varos>Los Angeles</varos>
            </lakhely>
        </szinesz>
    </szineszek>
    <studiok>
        <studio SID="s01">
            <nev>Paramount Pictures</nev>
            <alapitas>1912</alapitas>
            <vezeto>Adolf Zukor</vezeto>
            <szekhely>
                <orszag>USA</orszag>
                <varos>Los Angeles</varos>
            </szekhely>
        </studio>
        <studio SID="s02">
            <nev>Miramax</nev>
            <alapitas>1979</alapitas>
            <vezeto>Bob Weinstein</vezeto>
            <szekhely>
                <orszag>USA</orszag>
                <varos>Santa Monica</varos>
            </szekhely>
        </studio>
    </studiok>
    <forgatokonyvek>
        <forgatokonyv FkID="fk01">
            <oldalszam>251</oldalszam>
            <formatum>nyomtatott</formatum>
            <peldanyszam>10000</peldanyszam>
        </forgatokonyv>
        <forgatokonyv FkID="fk02">
            <oldalszam>223</oldalszam>
            <formatum>nyomtatott</formatum>
            <peldanyszam>12000</peldanyszam>
        </forgatokonyv>
        <forgatokonyv FkID="fk03">
            <oldalszam>217</oldalszam>
            <formatum>nyomtatott</formatum>
```

```
            <peldanyszam>5000</peldanyszam>
        </forgatokonyv>
        <forgatokonyv FkID="fk04">
            <oldalszam>146</oldalszam>
            <formatum>digitális</formatum>
            <peldanyszam>1</peldanyszam>
        </forgatokonyv>
    </forgatokonyvek>
    <filmek>
        <film FID="f01">
            <cim>Csupasz pisztoly</cim>
            <bevetel>78756177</bevetel>
            <hossz>85</hossz>
            <mufaj>akció/vígjáték</mufaj>
            <megjelenes>
                <ev>1988</ev>
                <honap>12</honap>
                <nap>2</nap>
            </megjelenes>
            <RID refID="r01"/>
            <SID refID="s01"/>
            <FkID refID="fk01"/>
        </film>
        <film FID="f02">
            <cim>Csupasz pisztoly 2 és 1/2</cim>
            <bevetel>192000000</bevetel>
            <hossz>81</hossz>
            <mufaj>akció/vígjáték</mufaj>
            <megjelenes>
                <ev>1991</ev>
                <honap>6</honap>
                <nap>28</nap>
            </megjelenes>
            <RID refID="r01"/>
            <SID refID="s01"/>
            <FkID refID="fk02"/>
        </film>
        <film FID="f03">
            <cim>Csupasz pisztoly 33 1/3 – Az utolsó merénylet</cim>
            <bevetel>51132598</bevetel>
            <hossz>81</hossz>
            <mufaj>akció/vígjáték</mufaj>
            <megjelenes>
                <ev>1994</ev>
                <honap>3</honap>
                <nap>18</nap>
            </megjelenes>
            <RID refID="r02"/>
            <SID refID="s01"/>
```

```
            <FkID refID="fk03"/>
        </film>
        <film FID="f04">
            <cim>Horrorra akadva 2</cim>
            <bevetel>71308997</bevetel>
            <hossz>83</hossz>
            <mufaj>horror/vígjáték</mufaj>
            <megjelenes>
                <ev>2001</ev>
                <honap>7</honap>
                <nap>4</nap>
            </megjelenes>
            <RID refID="r03"/>
            <SID refID="s02"/>
            <FkID refID="fk04"/>
        </film>
    </filmek>
    <szereplesek>
        <szereples>
            <FID refID="f01"/>
            <SzID refID="sz01"/>
            <szerep>Frank Drebin hadnagy</szerep>
        </szereples>
        <szereples>
            <FID refID="f01"/>
            <SzID refID="sz02"/>
            <szerep>Jane Spencer</szerep>
        </szereples>
        <szereples>
            <FID refID="f02"/>
            <SzID refID="sz01"/>
            <szerep>Frank Drebin hadnagy</szerep>
        </szereples>
        <szereples>
            <FID refID="f02"/>
            <SzID refID="sz02"/>
            <szerep>Jane Spencer</szerep>
        </szereples>
        <szereples>
            <FID refID="f03"/>
            <SzID refID="sz01"/>
            <szerep>Frank Drebin hadnagy</szerep>
        </szereples>
        <szereples>
            <FID refID="f03"/>
            <SzID refID="sz02"/>
            <szerep>Jane Spencer</szerep>
        </szereples>
        <szereples>
```

```
            <FID refID="f04"/>
            <SzID refID="sz03"/>
            <szerep>Kurta Meeks</szerep>
        </szereples>
    </szereplesek>
</filmtar>
```

Az XML dokumentum alapján XMLSchema készítése

Az XML dokumentum validálásához saját sémát készítettem, melyben saját típusokat használtam, amelyett az xs:alaptípusokból hoztam létre és helyenként restriction-ökkel is kiegészítettem. A séma szerkezetét azalábbiak szerint alakítottam: saját típusok, xml felépítésére vonatkozó megkötés a root element-től kezdve, majd elsődleges kulcsok és végül idegen kulcsok, vagyis kulcs hivatkozások.

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

    <!-- típusok-->

    <xs:complexType name="rendezotype">
        <xs:sequence>
            <xs:element name="rendezo" maxOccurs="unbounded">
                <xs:complexType>
                    <xs:sequence>
                        <xs:element name="nev" type="xs:string"/>
                        <xs:element name="eletkor" type="xs:integer"/>
                        <xs:element name="lakhely" type="lakszekhelytype"/>
                    </xs:sequence>
                    <xs:attribute name="RID" type="xs:ID" use="required"/>
                </xs:complexType>
            </xs:element>
        </xs:sequence>
    </xs:complexType>

    <xs:complexType name="megjelenestype">
        <xs:sequence>
            <xs:element name="ev" type="xs:integer"/>
            <xs:element name="honap" type="xs:integer"/>
            <xs:element name="nap" type="xs:integer"/>
        </xs:sequence>
    </xs:complexType>

    <xs:complexType name="lakszekhelytype">
        <xs:sequence>
            <xs:element name="orszag" type="xs:string"/>
            <xs:element name="varos" type="xs:string"/>
        </xs:sequence>
    </xs:complexType>

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

```
        <xs:sequence>
            <xs:element name="szinesz" maxOccurs="unbounded">
                <xs:complexType>
                    <xs:sequence>
                        <xs:element name="nev" type="xs:string"/>
                        <xs:element name="eletkor" type="xs:integer"/>
                        <xs:element name="lakhely" type="lakszekhelytype"/>
                    </xs:sequence>
                    <xs:attribute name="SzID" type="xs:ID" use="required"/>
                </xs:complexType>
            </xs:element>
        </xs:sequence>
    </xs:complexType>

    <xs:complexType name="studiotype">
        <xs:sequence>
            <xs:element name="studio" maxOccurs="unbounded">
                <xs:complexType>
                    <xs:sequence>
                        <xs:element name="nev" type="xs:string"/>
                        <xs:element name="alapitas" type="xs:integer"/>
                        <xs:element name="vezeto" type="xs:string"/>
                        <xs:element name="szekhely" type="lakszekhelytype"/>
                    </xs:sequence>
                    <xs:attribute name="SID" type="xs:ID" use="required"/>
                </xs:complexType>
            </xs:element>
        </xs:sequence>
    </xs:complexType>

    <xs:complexType name="forgatokonyvtype">
        <xs:sequence>
            <xs:element name="forgatokonyv" maxOccurs="unbounded">
                <xs:complexType>
                    <xs:sequence>
                        <xs:element name="oldalszam"
type="xs:positiveInteger"/>
                        <xs:element name="formatum">
                            <xs:simpleType>
                                <xs:restriction base="xs:string">
                                    <xs:enumeration value="nyomtatott" />
                                    <xs:enumeration value="digitális" />
                                </xs:restriction>
                            </xs:simpleType>
                        </xs:element>
                        <xs:element name="peldanyszam"
type="xs:positiveInteger"/>
                    </xs:sequence>
                    <xs:attribute name="FkID" type="xs:ID" use="required"/>
```

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

<xs:complexType name="filmtype">
    <xs:sequence>
        <xs:element name="film" maxOccurs="unbounded">
            <xs:complexType>
                <xs:sequence>
                    <xs:element name="cim" type="xs:string"/>
                    <xs:element name="bevetel" type="xs:long"/>
                    <xs:element name="hossz" type="xs:integer"/>
                    <xs:element name="mufaj" type="xs:string"/>
                    <xs:element name="megjelenes" type="megjelenestype"/>
                    <xs:element name="RID">
                        <xs:complexType>
                            <xs:attribute name="refID" type="xs:IDREF"/>
                        </xs:complexType>
                    </xs:element>
                    <xs:element name="SID">
                        <xs:complexType>
                            <xs:attribute name="refID" type="xs:IDREF"/>
                        </xs:complexType>
                    </xs:element>
                    <xs:element name="FkID">
                        <xs:complexType>
                            <xs:attribute name="refID" type="xs:IDREF"/>
                        </xs:complexType>
                    </xs:element>
                </xs:sequence>
                <xs:attribute name="FID" type="xs:ID" use="required"/>
            </xs:complexType>
        </xs:element>
    </xs:sequence>
</xs:complexType>

<xs:complexType name="szereplestype">
    <xs:sequence>
        <xs:element name="szereples" maxOccurs="unbounded">
            <xs:complexType>
                <xs:sequence>
                    <xs:element name="FID">
                        <xs:complexType>
                            <xs:attribute name="refID" type="xs:IDREF"/>
                        </xs:complexType>
                    </xs:element>
                    <xs:element name="SzID">
                        <xs:complexType>
```

```
                            <xs:attribute name="refID" type="xs:IDREF"/>
                        </xs:complexType>
                    </xs:element>
                    <xs:element name="szerep" type="xs:string"/>
                </xs:sequence>
            </xs:complexType>
        </xs:element>
    </xs:sequence>
</xs:complexType>


<!-- root element-->

<xs:element name="filmtar">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="rendezok" type="rendezotype" maxOccurs="1"/>
            <xs:element name="szineszek" type="szinesztype"
maxOccurs="1"/>
            <xs:element name="studiok" type="studiotype" maxOccurs="1"/>
            <xs:element name="forgatokonyvek" type="forgatokonyvtype"
maxOccurs="1"/>
            <xs:element name="filmek" type="filmtype" maxOccurs="1"/>
            <xs:element name="szereplesek" type="szereplestype"
maxOccurs="1"/>
        </xs:sequence>
    </xs:complexType>

    <!-- kulcsok-->
    <xs:key name="RID">
        <xs:selector xpath="rendezo"/>
        <xs:field xpath="@RID"/>
    </xs:key>
    <xs:key name="SzID">
        <xs:selector xpath="szinesz"/>
        <xs:field xpath="@SzID"/>
    </xs:key>
    <xs:key name="SID">
        <xs:selector xpath="studio"/>
        <xs:field xpath="@SID"/>
    </xs:key>
    <xs:unique name="FkID">
        <xs:selector xpath="forgatokonyv"/>
        <xs:field xpath="@FkID"/>
    </xs:unique>
    <xs:key name="FID">
        <xs:selector xpath="film"/>
        <xs:field xpath="@FID"/>
    </xs:key>
```

```
        <xs:keyref name="refRID" refer="RID">
            <xs:selector xpath="film/RID"/>
            <xs:field xpath="@refID"/>
        </xs:keyref>
        <xs:keyref name="refSID" refer="SID">
            <xs:selector xpath="film/SID"/>
            <xs:field xpath="@refID"/>
        </xs:keyref>
        <xs:keyref name="refFkID" refer="FkID">
            <xs:selector xpath="film/FkID"/>
            <xs:field xpath="@refID"/>
        </xs:keyref>
        <xs:keyref name="refFID" refer="FID">
            <xs:selector xpath="szereples/FID"/>
            <xs:field xpath="@refID"/>
        </xs:keyref>
        <xs:keyref name="refSzID" refer="SzID">
            <xs:selector xpath="szereples/SzID"/>
            <xs:field xpath="@refID"/>
        </xs:keyref>
    </xs:element>

</xs:schema>
```

Második feladat

Adatolvasás

A cél, hogy egy adott XMl dokumentot beolvassunk, itt név szerint a feladat előző részében létrehozott XMLYF399.xml dokumentumot fogjuk beolvasni, a megkapott adatokat ezután console-ra kiírja a program.

```
package hu.domparse.yif399;

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

public class DomReadYIF399 {

    public static void main(String[] args) {
        // A DOM l�trehoz�sa az XML dokumentumb�l
        DocumentBuilderFactory documentBuilderFactory =
DocumentBuilderFactory.newInstance();
        try {
            DocumentBuilder documentBuilder =
documentBuilderFactory.newDocumentBuilder();
            Document document = documentBuilder.parse(new
File("XMLYIF399.xml"));
            document.getDocumentElement().normalize();//normaliz�l�s
            Node root = document.getDocumentElement();//root elem
kiv�laszt�sa
            listAll(root);
        } catch (ParserConfigurationException | SAXException | IOException e)
{
            e.printStackTrace();
        }
    }

    public static void listAll(Node root) {
        //root elem gyerek elemeinek kiv�laszt�sa
        NodeList nodeList = root.getChildNodes();
        //kiirat�shoz String-ek
```

```
        String rendezo = "Rendezok\n\n";
        String szinesz = "Szineszek\n\n";
        String studio = "Studiok\n\n";
        String film = "Filmek\n\n";
        String szereples = "Szereplesek\n\n";
        for (int i = 0; i < nodeList.getLength(); i++) {
            Node node = nodeList.item(i);//root gyerek elemei egyess?vel
            if(node.getNodeType() == Node.ELEMENT_NODE) {
                //root gyerek elemeinek a gyerekelemeinek a kiv?laszt?sa
                NodeList subNodeList = nodeList.item(i).getChildNodes();
                for (int j = 0; j < subNodeList.getLength(); j++) {
                    //root gyerek elemeinek a gyerekelemei egyess?vel
                    Node subNode = subNodeList.item(j);
                    if(subNode.getNodeType() == Node.ELEMENT_NODE) {
                        //Stringekbe csoportos?t?s
                        rendezo += oneNode(subNode,"rendezo");
                        szinesz += oneNode(subNode,"szinesz");
                        studio += oneNode(subNode,"studio");
                        film += oneNode(subNode,"film");
                        szereples += oneNode(subNode,"szereples");
                    }
                }
            }
        }
        //kiirat?sok
        System.out.println(rendezo);
        System.out.println(szinesz);
        System.out.println(studio);
        System.out.println(film);
        System.out.println(szereples);
    }

    public static String oneNode(Node subNode, String tag) {
        String out = "";
        if(subNode.getNodeName().equals(tag)) {//a tag String-gel megegyez?
nev? node-ot keres
            //ha van attribute-ja, akkor ki?ratja a legels?t a String-be
            if(subNode.getAttributes().getLength()>0) {
                out +=tag + " ID : " +
subNode.getAttributes().item(0).getTextContent()+"\n";
            }

            NodeList subSubNodeList = subNode.getChildNodes();
            for (int k = 0; k < subSubNodeList.getLength(); k++) {
                Node subSubNode = subSubNodeList.item(k);//node gyerek elemei
egyess?vel
                if(subSubNode.getNodeType() == Node.ELEMENT_NODE) {
                    //az elem nev?t?l f?gg?en m?shogy rakja bele a Sring-
be
```

```
                    switch(subSubNode.getNodeName()) {
                        //a refID-kat tartalmaz? elemek olvas?sa
                        case "RID":
                        case "SzID":
                        case "SID":
                        case "FID":
                            //az els? attribute-okat rakja bele a String-be
                            out += subSubNode.getNodeName() + " : "
                                    +
subSubNode.getAttributes().item(0).getTextContent()+"\n";
                            break;
                        case "megjelenes":
                        case "lakhely":
                        case "szekhely":
                            NodeList sssNode = subSubNode.getChildNodes();
                            // gyerek elemein for ciklussal v?gigmegy ?s ?gy
rakja bele a String-be
                            for (int l = 0; l < sssNode.getLength(); l++) {
                                if (sssNode.item(l).getNodeType() ==
Node.ELEMENT_NODE) {
                                    out += subSubNode.getNodeName() + "-" +
sssNode.item(l).getNodeName()
                                            + " : " +
sssNode.item(l).getTextContent()+"\n";
                                }
                            }
                            break;
                        default:
                            //norm?l esetben az elem nev?t ?s kontextus?t
rakja bele a String-be
                            out += subSubNode.getNodeName() + " : " +
subSubNode.getTextContent() + "\n";
                    }
                }
            }
            out+="\n";
        }
        return out;
    }

}
```

Adatmódosítás

Az adatmódosítás során cél, hogy az eredeti XML dokumentumból olvassunk be, de egy másikban tároljuk el a módosításokat, így nem vész el az eredeti fájlunk és a módosítások is érvényt nyernek. Menürendszerrel dolgoztam, ahol különböző attribute-okat lehet megváltoztatni, a megfelelő elemet az elsődleges kulcs alapján lehet kiválasztani. A kimeneti fájl neve: *XMLYIF399.out.xml*

```
package hu.domparse.yif399;

import java.io.File;
import java.io.IOException;
import java.io.UnsupportedEncodingException;
import java.util.Scanner;
import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.DocumentBuilderFactory;
import javax.xml.parsers.ParserConfigurationException;
import javax.xml.transform.Transformer;
import javax.xml.transform.TransformerException;
import javax.xml.transform.TransformerFactory;
import javax.xml.transform.dom.DOMSource;
import javax.xml.transform.stream.StreamResult;
import org.w3c.dom.Document;
import org.w3c.dom.Node;
import org.w3c.dom.NodeList;
import org.xml.sax.SAXException;

public class DomModifyYIF399 {

    public static void main(String[] args) {

        // A DOM l�trehoz�sa az XML dokumentumb�l
        DocumentBuilderFactory documentBuilderFactory =
DocumentBuilderFactory.newInstance();
        try {
            DocumentBuilder documentBuilder =
documentBuilderFactory.newDocumentBuilder();
            Document document = documentBuilder.parse(new
File("XMLYIF399.xml"));
            document.getDocumentElement().normalize();//normaliz�l�s
            Scanner scanner = new Scanner(System.in);
            boolean exit=true;
            while(exit) {//menu rendszer
                System.out.println("1 : Adja meg egy film bevetelet\n2 : Adja
meg egy szin�sz �letkor�t\n"
                        + "3 : Adja meg egy studio �j vezet�j�nek a
nev�t");
```

```
                switch(scanner.nextInt()) {
                    case 1 : filmIncome(document, scanner);break;
                    case 2 : szineszAge(document, scanner);break;
                    case 3 : studioBoss(document, scanner);break;
                    default : exit=false;
                }
            }
            scanner.close();
            writeToXml(document);
        } catch (ParserConfigurationException | SAXException | IOException |
TransformerException e) {
            e.printStackTrace();
        }
    }

    public static void filmIncome(Document document, Scanner scanner) {
        NodeList nodeList = document.getElementsByTagName("film");//film
elemek kiv�laszt�sa
        System.out.println("Adja meg a film ID-j�t :");
        String input = scanner.next();//beolvas�s
        for (int i = 0; i < nodeList.getLength(); i++) {
            Node node = nodeList.item(i);
            if (node.getNodeType() == Node.ELEMENT_NODE) {
                //a megfelel� ID megkeres�se
                if
(node.getAttributes().getNamedItem("FID").getTextContent().equals(input)) {
                    NodeList subNodeList = node.getChildNodes();
                    for (int j = 0; j < subNodeList.getLength(); j++) {
                        Node subNode = subNodeList.item(j);
                        if (subNode.getNodeName().equals("bevetel"))
{//bevetel elem keres�se
                            System.out.println("Adja meg a film bev�tel�t
:");
                            //�j bevetel bek�r�se �s be�r�sa az xml-be
                            subNode.setTextContent(scanner.next());
                        }
                    }
                }
            }
        }
    }

    public static void szineszAge(Document document, Scanner scanner) {
        NodeList nodeList = document.getElementsByTagName("szinesz");//szinesz
elemek kiv�laszt�sa
        System.out.println("Adja meg a szinesz ID-j�t :");
        String input = scanner.next();//beolvas�s
        for (int i = 0; i < nodeList.getLength(); i++) {
            Node node = nodeList.item(i);
```

```
            if (node.getNodeType() == Node.ELEMENT_NODE) {
                //a megfelel? ID megkeres?se
                if
(node.getAttributes().getNamedItem("SzID").getTextContent().equals(input)) {
                    NodeList subNodeList = node.getChildNodes();
                    for (int j = 0; j < subNodeList.getLength(); j++) {
                        Node subNode = subNodeList.item(j);
                        if (subNode.getNodeName().equals("eletkor"))
{//eletkor elem keres?se
                            //?j kor bek?r?se ?s be?r?sa az xml-be
                            System.out.println("Adja meg a szinesz ?j
?letkor?t :");
                            subNode.setTextContent(scanner.next());
                        }
                    }
                }
            }
        }
    }

    public static void studioBoss(Document document, Scanner scanner) {
        NodeList nodeList = document.getElementsByTagName("studio");//studio
elemek kiv?laszt?sa
        System.out.println("Adja meg a studio ID-j?t :");
        String input = scanner.next();//beolvas?s
        for (int i = 0; i < nodeList.getLength(); i++) {
            Node node = nodeList.item(i);
            if (node.getNodeType() == Node.ELEMENT_NODE) {
                //a megfelel? ID megkeres?se
                if
(node.getAttributes().getNamedItem("SID").getTextContent().equals(input)) {
                    NodeList subNodeList = node.getChildNodes();
                    for (int j = 0; j < subNodeList.getLength(); j++) {
                        Node subNode = subNodeList.item(j);
                        if (subNode.getNodeName().equals("vezeto")) {//vezeto
elem kiv?laszt?sa
                            System.out.println("Adja meg a vezeto ?j nev?t
:");
                            //?j n?v bek?r?se ?s kiir?sa az xml-be
                            String name = scanner.next();
                            name +=scanner.nextLine();
                            subNode.setTextContent(name);
                        }
                    }
                }
            }
        }
    }
    //f?jlba ?s konzolra kiir?s
```

```
    public static void writeToXml(Document document) throws
TransformerException, UnsupportedEncodingException {
        TransformerFactory transformerFactory =
TransformerFactory.newInstance();
        Transformer transf = transformerFactory.newTransformer();
        DOMSource source = new DOMSource(document);
        //kimeneti csatorna konzolra
        StreamResult console = new StreamResult(System.out);
        //kimeneti csatorna f?jlba
        StreamResult file = new StreamResult(new File("XMLYIF399.out.xml"));
        // A m?dos?t?sok ki?r?sa a k?perny?re
        transf.transform(source, console);
        // A m?dos?t?sok ki?r?sa ?j f?jlba
        transf.transform(source, file);
    }

}
```

Adatlekérdezés

Az adatlekérdezés során rendezett keretek között nyerünk ki információt az adatbázisunkból, viszont, hogy az eredmény letisztult legyen, a forráskódban a lekérdezések ki vannak kommentelve. Itt szintén egy menürendszerben választhatunk, hogy a felkínált adatokból melyikre szeretnénk szűrést végrehajtani. A lekérdezett eredményket console-on adja vissza a program.

```
package hu.domparse.yif399;

import java.io.File;
import java.io.IOException;
import java.util.Calendar;
import java.util.Scanner;

import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.DocumentBuilderFactory;
import javax.xml.parsers.ParserConfigurationException;
import org.w3c.dom.*;
import org.xml.sax.SAXException;

public class DomQueryYIF399 {

    public static void main(String[] args) {

        // A DOM l�trehoz�sa az XML dokumentumb�l
        DocumentBuilderFactory documentBuilderFactory =
DocumentBuilderFactory.newInstance();
        try {
            DocumentBuilder documentBuilder =
documentBuilderFactory.newDocumentBuilder();
            Document document = documentBuilder.parse(new
File("XMLYIF399.xml"));
            document.getDocumentElement().normalize();//normaliz�l�s
            Scanner scanner = new Scanner(System.in);
            boolean exit=true;
            while(exit) { //menu rendszer
                System.out.println("1 : Film(ek) amik t�bb bev�telt hoztak
mint...\n"
                        + "2 : Szin�sz(ek) akik fiatalabbak...\n3 : Studi�(k)
amik fiatalabbak mint....");
                switch(scanner.nextInt()) {
                    case 1: filmIncome(document, scanner);break;
                    case 2: szineszAge(document, scanner);break;
                    case 3: studioFoundation(document, scanner);break;
                    default: exit=false;
                }
```

```
            }
            scanner.close();
        } catch (ParserConfigurationException | SAXException | IOException e)
{
            e.printStackTrace();
        }
    }

    public static void filmIncome(Document document,Scanner scanner) {
        NodeList nodeList = document.getElementsByTagName("film");//film
elemek kiv?laszt?sa
        System.out.println("Adja meg mennyin?l :");
        Long input = scanner.nextLong();//beolvas?s
        System.out.println("\n");
        for (int i = 0; i < nodeList.getLength(); i++) {
            Node node = nodeList.item(i);//film gyerek elemei egyess?vel
            if(node.getNodeType() == Node.ELEMENT_NODE) {
                //root gyerek elemeinek a gyerek elemeinek a kiv?laszt?sa
                NodeList subNodeList = node.getChildNodes();
                for (int j = 0; j < subNodeList.getLength(); j++) {
                Node subNode = subNodeList.item(j);
                if(subNode.getNodeType() == Node.ELEMENT_NODE) {
                    if (subNode.getNodeName().equals("bevetel") )//bevetele
elem keres?se
                        //a krit?riumnak megfelel? elem kiv?laszt?sa
                        if (Long.parseLong(subNode.getTextContent()) > input)
                            listSub(node, "Film");
                    }
                }
            }
        }
    }

    public static void szineszAge(Document document,Scanner scanner) {
        NodeList nodeList = document.getElementsByTagName("szinesz");//szinesz
elemek kiv?laszt?sa
        System.out.println("Adja meg mennyin?l :");
        int input = scanner.nextInt();//beolvas?s
        System.out.println("\n");
        for (int i = 0; i < nodeList.getLength(); i++) {
            Node node = nodeList.item(i);//szinesz gyerek elemei egyess?vel
            if(node.getNodeType() == Node.ELEMENT_NODE) {
                //root gyerek elemeinek a gyerek elemeinek a kiv?laszt?sa
                NodeList subNodeList = node.getChildNodes();
                for (int j = 0; j < subNodeList.getLength(); j++) {
                Node subNode = subNodeList.item(j);
                if(subNode.getNodeType() == Node.ELEMENT_NODE) {
                    if (subNode.getNodeName().equals("eletkor") )//eletkor
elem megkeres?se
```

```
                        //a krit�riumnak megfelel� elem kiv�laszt�sa
                        if (Integer.parseInt(subNode.getTextContent()) <
input)
                            listSub(node, "Szinesz");
                    }
                }
            }
        }
    }

    public static void studioFoundation(Document document,Scanner scanner) {
        NodeList nodeList = document.getElementsByTagName("studio");//studio
elemek kiv�laszt�sa
        System.out.println("Adja meg mennyin�l :");
        Long input = scanner.nextLong();//beolvas�s
        System.out.println("\n");
        int year = Calendar.getInstance().get(Calendar.YEAR);
        for (int i = 0; i < nodeList.getLength(); i++) {
            Node node = nodeList.item(i);//studio gyerek elemei egyess�vel
            if(node.getNodeType() == Node.ELEMENT_NODE) {
                //root gyerek elemeinek a gyerek elemeinek a kiv�laszt�sa
                NodeList subNodeList = node.getChildNodes();
                for (int j = 0; j < subNodeList.getLength(); j++) {
                Node subNode = subNodeList.item(j);
                if(subNode.getNodeType() == Node.ELEMENT_NODE) {
                    if (subNode.getNodeName().equals("alapitas") )//alapitas
elem megkeres�se
                        //a krit�riumnak megfelel� elem kiv�laszt�sa
                        if (year - Integer.parseInt(subNode.getTextContent())
< input)
                            listSub(node, "Studio");
                    }
                }
            }
        }
    }

    private static void listSub(Node node, String tag) {
        String out="";
        out +=tag + " ID : " +
node.getAttributes().item(0).getTextContent()+"\n";//ID Stringbe ir�sa
        Element element = (Element) node;
        switch(tag) {//k�l�n kezel�se a kiir�soknak
            case "Film"://film cim �s bevetel String-be ir�sa
                out +=
element.getElementsByTagName("cim").item(0).getNodeName()
                    + " : "
+element.getElementsByTagName("cim").item(0).getTextContent()+"\n";
```

```
                out +=
element.getElementsByTagName("bevetel").item(0).getNodeName()
                    + " : "
+element.getElementsByTagName("bevetel").item(0).getTextContent()+"\n";
                break;
            case "Szinesz"://szinesz n�v �s eletkor String-be ir�sa
                out +=
element.getElementsByTagName("nev").item(0).getNodeName()
                    + " : "
+element.getElementsByTagName("nev").item(0).getTextContent()+"\n";
                out +=
element.getElementsByTagName("eletkor").item(0).getNodeName()
                    + " : "
+element.getElementsByTagName("eletkor").item(0).getTextContent()+"\n";
                break;
            case "Studio"://studio nev �s alapitas String-be ir�sa
                out +=
element.getElementsByTagName("nev").item(0).getNodeName()
                    + " : "
+element.getElementsByTagName("nev").item(0).getTextContent()+"\n";
                out +=
element.getElementsByTagName("alapitas").item(0).getNodeName()
                    + " : "
+element.getElementsByTagName("alapitas").item(0).getTextContent()+"\n";
                break;
        }

        System.out.println(out);//kiir�s
    }

}
```