JEGYZŐKÖNYV

Adatkezelés XML környezetben

Féléves feladat

Cukrászda

Készítette: **Görög Krisztina Erzsébet**

Neptunkód: **MPW46D**

Dátum: 2023. 11. 24.

Tartalom

Bevezetés

A féléves feladatomban egy cukrászdát (egészen pontosan egy cukrászdaláncot) kísérlek meg modellezni. A cukrászdaláncnak több cukrászdája van, mindegyiknek van egy igazgatója, mindegyikben vannak dolgozók. A cukrászda süteményeket szállít, amikből minimum tízet kell rendelni ahhoz, hogy kiszállítsák a rendezvényre, hiszen úgy nem éri meg és nincs értelme, ha nagyon keveset rendelnek belőle. A rendezvényeknek vannak rendezőik, akik valójában a sütemények megrendelői.

Nagyon fontos a feladat szempontjából a rendelés megvalósítása, amiben le kell adni a rendelt sütemények darabszámát, valamint tárolni szükséges a fizetendő összeget, amelyet a rendelt sütemény egységárából és a rendelt sütemények számából számítunk ki. Már említettük, hogy minimum tíz sütemény rendelése szükséges.

Röviden összefoglalva: egy cukrászdalánc megrendelésre szállít süteményeket rendezvényekre. Egy cukrászdában dolgozók (főcukrász, cukrász, cukrászgyakornok) dolgoznak, süteményeket árulnak, ezen süteményeket megrendelők rendelik meg egy rendezvényre, amelyre a cukrászda szállítja minden esetben ingyenesen a megrendelt süteményeket. Mindegyik cukrászdában el tudják készíteni mindegyik süteményt. Minden cukrászdának van egy igazgatója.

1. feladat

1a) Az adatbázis ER modell tervezése

Az alábbi egyedeink lesznek: cukrászda, sütemény, igazgató, dolgozó, megrendelő, rendezvény.

Tárolnunk kell a cukrászda esetében a nevét, címét, valamint a CID azonosító lesz a kulcsa (PK). A cím összetett tulajdonság, a városból, irányítószámból, utcából és házszámból áll.

A sütemények esetében tároljuk a nevüket, alapanyagaikat, egységárukat. A kulcstulajdonság a SID lesz. Az allergének egy többértékű tulajdonság. Egy süteményben több allergén is lehet.

Tároljuk az igazgatók nevét, lakcímét, beosztását, fizetését. A kulcstulajdonság az IID lesz. Az elérhetőség többértékű tulajdonság.

Tároljuk a dolgozók nevét, beosztását, fizetését. A kulcstulajdonság a DID lesz.

A megrendelő esetében tároljuk a nevét, lakcímét, elérhetőségét. A kulcs a MID lesz. A lakcím összetett tulajdonság, a városból, irányítószámból, utcából és házszámból áll. Az elérhetőség többértékű tulajdonság.

A rendezvénynek tároljuk a helyszínét, kezdetét (azaz az időpontot amire a cukrászdának szállítania kell). A kulcstulajdonság a RID azonosító lesz. A helyszín összetett tulajdonság, a városból, irányítószámból, utcából és házszámból áll.

A cukrászdát és dolgozót 1:n kapcsolat (dolgozik) köti össze, egy dolgozó egy cukrászdában dolgozik, de egy cukrászdának több dolgozója is lehet.

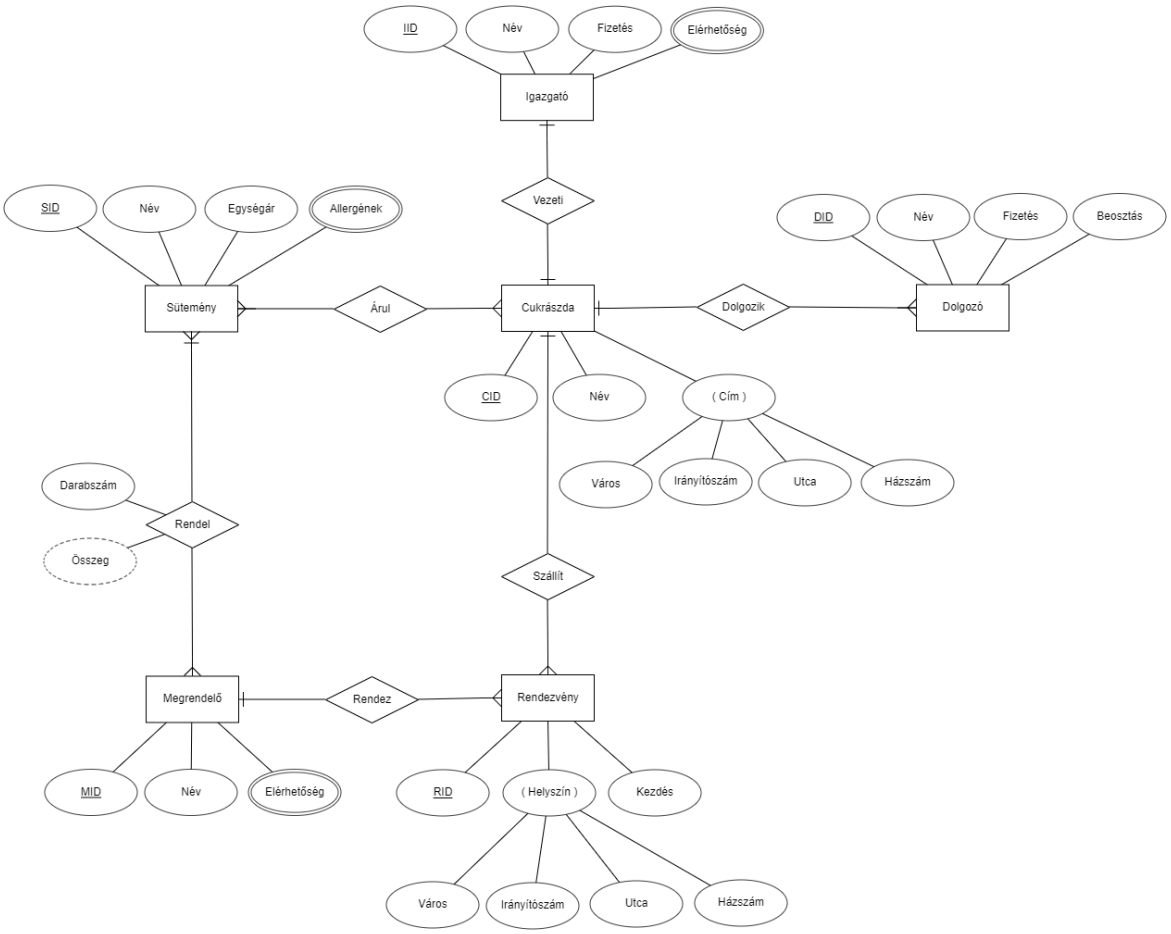
Az igazgatót és a cukrászdát 1:1 kapcsolat köti össze.

A cukrászdát és a süteményt n:m (árul) kapcsolat köti össze.

A süteményt és a megrendelőt egy n:m (rendel) kapcsolat köti össze. A kapcsolatnak két tulajdonsága van: összeg és darabszám. Az összeg egy származtatott tulajdonság, a darabszám szorozva a sütemény egységárával képletből kapjuk meg értékét.

A megrendelő és a rendezvény között 1:n (rendez) kapcsolat van. Feltételezzük, hogy egy rendezvényhez csak egy megrendelő tartozik, azonban egy megrendelő több rendezvényhez is tartozhat.

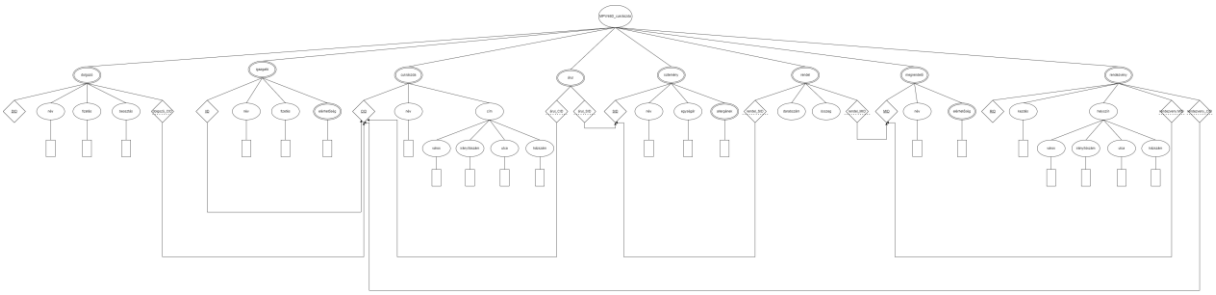
A rendezvény és a cukrászda között 1:n (szállít) kapcsolat van. A cukrászda több rendezvényre is szállíthat süteményt, azonban egy rendezvényre csak egy cukrászda szállít.



1b) Az adatbázis konvertálása XDM modellre

Konvertáljuk az ER modellt XDM modellre.

Először is szükségünk van egy gyökérelemre, amely a mi esetünkben az MPW46D_cukrászda lesz. Ebből ágaznak el az egyedeink, valamint a n:n kapcsolat megvalósítására létrejött egyedeink. Egy szinttel lejjebb találhatók az egyedek tulajdonságai, kulcsai, idegen kulcsai. Az összetett tulajdonságok egy szinttel lejjebb terjednek, ott találhatók azon tulajdonságok, amikből összeállnak. Téglalappal jelöljük a szöveget tartalmazást, ezt egy szinttel lejjebb csatlakoztatjuk az elemekhez. Az idegenkulcsok nyíllal mutatnak a kulcsra, amelyre vonatkoznak.



A kép teljes méretben megtekinthető XDMMPW46D.drawio.png és MDMMPW46D.drawio néven a forrásfájlok között.

1c) Az XDM modell alapján XML dokumentum készítése

Az XDM modell alapján elkészítjük az XML dokumentumot. A gyökérelemtől indulunk, és egyre beljebb haladunk az elemek megadásával. Egyesével megadjuk a példányokat, mindegyikből minimum hármat. A példányok tulajdonságaiknak is értéket adunk.

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

<MPW46D_cukrászda xmlns:xs="http://www.w3.org/2001/XMLSchema-instance"
xs:noNamespaceSchemaLocation="XMLSchemaMPW46D.xsd">

    <!-- dolgozó példányok -->
    <dolgozó DID="d1" dolgozó_CID="c1">
        <név>Nagy Emese</név>
        <fizetés>300000</fizetés>
        <beosztás>cukrász</beosztás>
    </dolgozó>

    <dolgozó DID="d2" dolgozó_CID="c1">
        <név>Kiss Tamás</név>
        <fizetés>700000</fizetés>
        <beosztás>főcukrász</beosztás>
    </dolgozó>

    <dolgozó DID="d3" dolgozó_CID="c1">
        <név>Kovács Evelin</név>
        <fizetés>250000</fizetés>
        <beosztás>cukrászgyakornok</beosztás>
    </dolgozó>

    <dolgozó DID="d4" dolgozó_CID="c2">
        <név>Varga Levente</név>
        <fizetés>500000</fizetés>
        <beosztás>cukrász</beosztás>
    </dolgozó>

    <dolgozó DID="d5" dolgozó_CID="c2">
        <név>Sárosi Ferencné</név>
        <fizetés>800000</fizetés>
```

```
        <beosztás>főcukrász</beosztás>
    </dolgozó>

    <dolgozó DID="d6" dolgozó_CID="c2">
        <név>Nagy Andor</név>
        <fizetés>600000</fizetés>
        <beosztás>cukrász</beosztás>
    </dolgozó>

    <dolgozó DID="d7" dolgozó_CID="c3">
        <név>Nagy Andrea</név>
        <fizetés>900000</fizetés>
        <beosztás>főcukrász</beosztás>
    </dolgozó>

    <dolgozó DID="d8" dolgozó_CID="c3">
        <név>Tóth Tamás</név>
        <fizetés>600000</fizetés>
        <beosztás>cukrász</beosztás>
    </dolgozó>

    <!-- igazgató példányok -->
    <igazgató IID="c1">
        <név>Farkas Benedek</név>
        <fizetés>1000000</fizetés>
        <elérhetőség>farkas.benedek@gmail.com</elérhetőség>
        <elérhetőség>+36201234567</elérhetőség>
    </igazgató>

    <igazgató IID="c2">
        <név>Szőke László</név>
        <fizetés>2000000</fizetés>
        <elérhetőség>+36702345645</elérhetőség>
    </igazgató>

    <igazgató IID="c3">
        <név>Takács Ádám</név>
        <fizetés>1500000</fizetés>
        <elérhetőség>+36705656789</elérhetőség>
        <elérhetőség>adam.takacs@gmail.com</elérhetőség>
    </igazgató>

    <!-- cukrászda példányok -->
    <cukrászda CID="c1">
        <név>Kristály Cukrászda</név>
        <cím>
            <város>Budapest</város>
            <irányítószám>1082</irányítószám>
            <utca>József krt.</utca>
```

```
            <házszám>53</házszám>
        </cím>
    </cukrászda>

    <cukrászda CID="c2">
        <név>Daubner Cukrászda</név>
        <cím>
            <város>Budapest</város>
            <irányítószám>1025</irányítószám>
            <utca>Szépvölgyi út</utca>
            <házszám>50</házszám>
        </cím>
    </cukrászda>

    <cukrászda CID="c3">
        <név>Kismandula Cukrászda</név>
        <cím>
            <város>Debrecen</város>
            <irányítószám>4024</irányítószám>
            <utca>Liszt Ferenc utca</utca>
            <házszám>10</házszám>
        </cím>
    </cukrászda>

    <!-- árul példányok -->
    <árul árul_CID="c1" árul_SID="s1"></árul>
    <árul árul_CID="c1" árul_SID="s2"></árul>
    <árul árul_CID="c1" árul_SID="s3"></árul>
    <árul árul_CID="c1" árul_SID="s4"></árul>
    <árul árul_CID="c2" árul_SID="s1"></árul>
    <árul árul_CID="c2" árul_SID="s2"></árul>
    <árul árul_CID="c2" árul_SID="s3"></árul>
    <árul árul_CID="c2" árul_SID="s4"></árul>
    <árul árul_CID="c3" árul_SID="s1"></árul>
    <árul árul_CID="c3" árul_SID="s2"></árul>
    <árul árul_CID="c3" árul_SID="s3"></árul>
    <árul árul_CID="c3" árul_SID="s4"></árul>

    <!-- sütemény példányok -->
    <sütemény SID="s1">
        <név>Mindenmentes süti</név>
        <egységár>3000</egységár>
    </sütemény>

    <sütemény SID="s2">
        <név>kókuszos kocka</név>
        <egységár>600</egységár>
        <allergének>tojás</allergének>
        <allergének>tej</allergének>
```

```
        <allergének>liszt</allergének>
    </sütemény>

    <sütemény SID="s3">
        <név>csokis szelet</név>
        <egységár>700</egységár>
        <allergének>liszt</allergének>
        <allergének>tojás</allergének>
    </sütemény>

    <sütemény SID="s4">
        <név>Madeira sütemény</név>
        <egységár>900</egységár>
        <allergének>liszt</allergének>
        <allergének>tej</allergének>
        <allergének>tojás</allergének>
        <allergének>sajt</allergének>
    </sütemény>

    <!-- rendel példányok -->
    <rendel rendel_SID="s1" rendel_MID="m1">
        <darabszám>30</darabszám>
    </rendel>

    <rendel rendel_SID="s4" rendel_MID="m2">
        <darabszám>20</darabszám>
    </rendel>

    <rendel rendel_SID="s3" rendel_MID="m3">
        <darabszám>50</darabszám>
    </rendel>

    <!-- megrendelő példányok -->
    <megrendelő MID="m1">
        <név>Kiss Lászlóné</név>
        <elérhetőség>+36203453434</elérhetőség>
    </megrendelő>

    <megrendelő MID="m2">
        <név>Horváth Csaba</név>
        <elérhetőség>+36305675656</elérhetőség>
    </megrendelő>

    <megrendelő MID="m3">
        <név>Remete Ákos</név>
        <elérhetőség>+36204557887</elérhetőség>
    </megrendelő>

    <!-- rendezvény példányok -->
```

```xml
    <rendezvény RID="r1" rendezvény_MID="m1" rendezvény_CID="c1">
        <kezdés>2023-12-12 16:00</kezdés>
        <helyszín>
            <város>Budapest</város>
            <irányítószám>1181</irányítószám>
            <utca>Városház utca</utca>
            <házszám>1</házszám>
        </helyszín>
    </rendezvény>

    <rendezvény RID="r2" rendezvény_MID="m2" rendezvény_CID="c1">
        <kezdés>2023-12-21 18:00</kezdés>
        <helyszín>
            <város>Budapest</város>
            <irányítószám>1119</irányítószám>
            <utca>Fehérvári út</utca>
            <házszám>47</házszám>
        </helyszín>
    </rendezvény>

    <rendezvény RID="r3" rendezvény_MID="m3" rendezvény_CID="c2">
        <kezdés>2023-12-29 19:00</kezdés>
        <helyszín>
            <város>Budapest</város>
            <irányítószám>1133</irányítószám>
            <utca>Kárpát utca</utca>
            <házszám>23</házszám>
        </helyszín>
    </rendezvény>

</MPW46D_cukrászda>
```

1d) Az XML dokumentum alapján XMLSchema készítése

Elkészítjük a szükséges xsd dokumentumot az xml-hez. Először kigyűjtjük az egyszerűtípusokat, valamint megvalósítjuk a megszorításokat. Ezután a gyökérelemtől haladva meghatározzuk a komplex típusokat, majd a saját típusokat definiáljuk. A kapcsolatok megvalósításához meghatározzuk a kulcsokat, idegenkulcsokat.

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

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

<!-- Egyszerű típusok kigyűjtése, saját típusok meghatározása, megszorítás -->

    <xs:element name="név" type="xs:string"/>
    <xs:element name="fizetés" type="xs:int"/>
    <xs:element name="beosztás" type="xs:string"/>
    <xs:element name="elérhetőség" type="xs:string"/>
```

```
    <xs:element name="város" type="xs:string"/>
    <xs:element name="irányítószám" type="xs:string"/>
    <xs:element name="utca" type="xs:string"/>
    <xs:element name="házszám" type="xs:string"/>
    <xs:element name="egységár" type="xs:int"/>
    <xs:element name="allergének" type="xs:string"/>
    <xs:element name="kezdés" type="xs:string"/>

    <xs:element name="darabszám">
        <xs:simpleType>
            <xs:restriction base="xs:int">
            <xs:minInclusive value="10"/>
            </xs:restriction>
        </xs:simpleType>
    </xs:element>

<!-- komplex típusok-->

<xs:element name="MPW46D_cukrászda">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="dolgozó" type="dolgozóTipus" minOccurs="0"
maxOccurs="unbounded"/>
            <xs:element name="igazgató" type="igazgatóTipus" minOccurs="0"
maxOccurs="unbounded"/>
            <xs:element name="cukrászda" type="cukrászdaTipus" minOccurs="0"
maxOccurs="unbounded"/>
            <xs:element name="árul" type="árulTipus" minOccurs="0"
maxOccurs="unbounded"/>
            <xs:element name="sütemény" type="süteményTipus" minOccurs="0"
maxOccurs="unbounded"/>
            <xs:element name="rendel" type="rendelTipus" minOccurs="0"
maxOccurs="unbounded"/>
            <xs:element name="megrendelő" type="megrendelőTipus" minOccurs="0"
maxOccurs="unbounded"/>
            <xs:element name="rendezvény" type="rendezvényTipus" minOccurs="0"
maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>

    <!-- Kulcsok megvalósítása -->
    <xs:key name="dolgozoKey">
        <xs:selector xpath=".//dolgozó"/>
        <xs:field xpath="@DID"/>
    </xs:key>
    <xs:key name="igazgatoKey">
        <xs:selector xpath=".//igazgató"/>
        <xs:field xpath="@IID"/>
    </xs:key>
```

```
    <xs:key name="cukraszdaKey">
        <xs:selector xpath=".//cukrászda"/>
        <xs:field xpath="@CID"/>
    </xs:key>
    <xs:key name="sutemenyKey">
        <xs:selector xpath=".//sütemény"/>
        <xs:field xpath="@SID"/>
    </xs:key>
    <xs:key name="megrendeloKey">
        <xs:selector xpath=".//megrendelő"/>
        <xs:field xpath="@MID"/>
    </xs:key>
    <xs:key name="rendezvenyKey">
        <xs:selector xpath=".//rendezvény"/>
        <xs:field xpath="@RID"/>
    </xs:key>

    <!-- Idegenkulcsok megvalósítása -->
    <xs:keyref name="dolgozoKeyRef" refer="cukraszdaKey">
        <xs:selector xpath=".//dolgozó"/>
        <xs:field xpath="@dolgozó_CID"/>
    </xs:keyref>
    <xs:keyref name="arulCukraszdaKeyRef" refer="cukraszdaKey">
        <xs:selector xpath=".//árul"/>
        <xs:field xpath="@árul_CID"/>
    </xs:keyref>
    <xs:keyref name="arulSutemenyKeyRef" refer="sutemenyKey">
        <xs:selector xpath=".//árul"/>
        <xs:field xpath="@árul_SID"/>
    </xs:keyref>
    <xs:keyref name="rendelSutemenyKeyRef" refer="sutemenyKey">
        <xs:selector xpath=".//rendel"/>
        <xs:field xpath="@rendel_SID"/>
    </xs:keyref>
    <xs:keyref name="rendelMegrendeloKeyRef" refer="megrendeloKey">
        <xs:selector xpath=".//rendel"/>
        <xs:field xpath="@rendel_MID"/>
    </xs:keyref>
    <xs:keyref name="rendezvenyMegrendeloKeyRef" refer="megrendeloKey">
        <xs:selector xpath=".//rendezvény"/>
        <xs:field xpath="@rendezvény_MID"/>
    </xs:keyref>

    <xs:keyref name="rendezvenyCukraszdaKeyRef" refer="cukraszdaKey">
        <xs:selector xpath=".//rendezvény"/>
        <xs:field xpath="@rendezvény_CID"/>
    </xs:keyref>

    <!-- 1:1 kapcsolat-->
```

```
    <xs:unique name="igazgatoKeyUnique">
        <xs:selector xpath=".//igazgató"/>
        <xs:field xpath="@CID"/>
    </xs:unique>
</xs:element>

<!-- saját típusok-->

<xs:complexType name="dolgozóTipus">
    <xs:sequence>
        <xs:element ref="név"/>
        <xs:element ref="fizetés"/>
        <xs:element ref="beosztás"/>
    </xs:sequence>
    <xs:attribute ref="DID"/>
    <xs:attribute ref="dolgozó_CID"/>
</xs:complexType>

<xs:complexType name="igazgatóTipus">
    <xs:sequence>
        <xs:element ref="név"/>
        <xs:element ref="fizetés"/>
        <xs:element ref="elérhetőség" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute ref="IID"/>
</xs:complexType>

<xs:complexType name="cukrászdaTipus">
    <xs:sequence>
        <xs:element ref="név"/>
        <xs:element name="cím" type="címTipus"/>
    </xs:sequence>
    <xs:attribute ref="CID"/>
</xs:complexType>

<xs:complexType name="árulTipus">
    <xs:attribute ref="árul_CID"/>
    <xs:attribute ref="árul_SID"/>
</xs:complexType>

<xs:complexType name="süteményTipus">
    <xs:sequence>
        <xs:element ref="név"/>
        <xs:element ref="egységár"/>
        <xs:element ref="allergének" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute ref="SID"/>
</xs:complexType>
```

```
<xs:complexType name="rendelTipus">
    <xs:sequence>
        <xs:element ref="darabszám"/>
        <xs:element name="összeg" type="xs:int" minOccurs="0" maxOccurs="1"/>
    </xs:sequence>
    <xs:attribute ref="rendel_SID"/>
    <xs:attribute ref="rendel_MID"/>
</xs:complexType>

<xs:complexType name="megrendelőTipus">
    <xs:sequence>
        <xs:element ref="név"/>
        <xs:element ref="elérhetőség" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute ref="MID"/>
</xs:complexType>

<xs:complexType name="rendezvényTipus">
    <xs:sequence>
        <xs:element ref="kezdés"/>
        <xs:element name="helyszín" type="címTipus"/>
    </xs:sequence>
    <xs:attribute ref="RID"/>
    <xs:attribute ref="rendezvény_MID"/>
    <xs:attribute ref="rendezvény_CID"/>
</xs:complexType>

<xs:complexType name="címTipus">
    <xs:sequence>
        <xs:element ref="város"/>
        <xs:element ref="irányítószám"/>
        <xs:element ref="utca"/>
        <xs:element ref="házszám"/>
    </xs:sequence>
</xs:complexType>

<!-- kulcsok-->

    <xs:attribute name="DID" type="xs:ID"/>
    <xs:attribute name="CID" type="xs:ID"/>
    <xs:attribute name="SID" type="xs:ID"/>
    <xs:attribute name="MID" type="xs:ID"/>
    <xs:attribute name="RID" type="xs:ID"/>

    <xs:attribute name="dolgozó_CID" type="xs:IDREF"/>
    <xs:attribute name="árul_CID" type="xs:IDREF"/>
    <xs:attribute name="árul_SID" type="xs:IDREF"/>
    <xs:attribute name="rendel_SID" type="xs:IDREF"/>
    <xs:attribute name="rendel_MID" type="xs:IDREF"/>
```

```
    <xs:attribute name="rendezvény_MID" type="xs:IDREF"/>
    <xs:attribute name="rendezvény_CID" type="xs:IDREF"/>

    <!--1:1 kapcsolat megvalósítása-->
    <xs:attribute name="IID" type="xs:IDREF"/>
</xs:schema>
```

2. feladat

Az XML dokumentumhoz egy DOM programot készítünk. A projekt neve: DOMParseMPW46D. A package legyen hu.domparse.mpw46d. Az osztályok nevei: DomReadMPW46D, DomModifyMPW46D, DomQueryMPW46D, DomWriteMPW46D.

2a) adatolvasás

Ezt a feladatot a DomReadMPW46D.java fájlban valósítjuk meg. Feldolgozzuk a teljes dokumentumot majd kiírjuk a konzolra fastruktúrában és elmentjük egy fájlba, jelen esetben a DomReadMPW46D_output.xml fájlba írjuk. A kód működését kommentekkel magyarázzuk.

```
package hu.domparse.mpw46d;

import java.io.File;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.OutputStream;
import java.io.OutputStreamWriter;
import java.io.Writer;

import javax.xml.parsers.DocumentBuilderFactory;
import javax.xml.parsers.DocumentBuilder;
import org.w3c.dom.Document;
import org.w3c.dom.NodeList;
import org.w3c.dom.Node;
import org.w3c.dom.Element;
import org.w3c.dom.NamedNodeMap;

public class DomReadMPW46D {

    public static void main(String[] args) {
        try {
            File inputFile = new File("XMLTaskMPW46D/XMLMPW46D.xml");
            DocumentBuilderFactory dbFactory =
DocumentBuilderFactory.newInstance();
            DocumentBuilder dBuilder = dbFactory.newDocumentBuilder();
            Document doc = dBuilder.parse(inputFile);
            System.out.println("Gyökérelem :" +
doc.getDocumentElement().getNodeName());

            System.out.println("\n----------------\n");
```

```
            // XML kiírása a konzolra
            printElement(doc.getDocumentElement(), "");

            // XML mentése a DomReadMPW46D_output.xml fájlba
            writeXmlToFile(doc,
"XMLTaskMPW46D/DomParseMPW46D/DomReadMPW46D_output.xml");

            System.out.println("\n----------------\nXML kiírása az output
fájlba teljesítve.\n");

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

    private static void printElement(Element element, String indent) {
        // jelenlegi elem
        System.out.print(indent + "<" + element.getTagName());

        // az elem attribútumai
        NamedNodeMap attributes = element.getAttributes();
        for (int i = 0; i < attributes.getLength(); i++) {
            Node attribute = attributes.item(i);
            System.out.print(" " + attribute.getNodeName() + "=\"" +
attribute.getNodeValue() + "\"");
        }

        // az elem gyermekelemei
        NodeList children = element.getChildNodes();
        if (children.getLength() == 0) {
            // ha nincsenek gyermekelemei, akkor záró taggel fejezzük be
            System.out.println("/>");
        } else {
            // ha vannak gyermekelemei, akkor normális záró taggel folytatjuk
            System.out.println(">");

            for (int i = 0; i < children.getLength(); i++) {
                Node child = children.item(i);
                // ha a gyermekelem is tartalmaz elemet, akkor arra is
meghívjuk a printElementet, ha nem, akkor kiírjuk a szöveget
                if (child.getNodeType() == Node.ELEMENT_NODE) {
                    printElement((Element) child, indent + "  ");
                } else if (child.getNodeType() == Node.TEXT_NODE &&
child.getNodeValue().trim().length() > 0) {
                    System.out.println(indent + "  " +
child.getNodeValue().trim());
                }
            }
```

```
            // a jelenlegi elem záró tagje
            System.out.println(indent + "</" + element.getTagName() + ">");
        }
    }

    private static void writeXmlToFile(Document document, String
outputFilePath) throws IOException {
        try (
                // outputstream nyitása a fájlnak
                OutputStream outputStream = new
FileOutputStream(outputFilePath);

                // UTF-8 writer
                Writer writer = new OutputStreamWriter(outputStream, "UTF-8"))
{
            // transformer készítése a DOM streammé formálásához
            javax.xml.transform.TransformerFactory transformerFactory =
javax.xml.transform.TransformerFactory
                    .newInstance();
            javax.xml.transform.Transformer transformer =
transformerFactory.newTransformer();

            // az indentálás, a szóközök beállítása, az xml deklarációjának
            // elvetése/meghagyása
            //
transformer.setOutputProperty(javax.xml.transform.OutputKeys.INDENT, "no");
            //
transformer.setOutputProperty("{http://xml.apache.org/xslt}indent-amount",
            // "1");
            //
transformer.setOutputProperty(javax.xml.transform.OutputKeys.OMIT_XML_DECLARAT
ION,
            // "yes");

            // a forrás (tehát a DOM) és a cél (az output) deklarálása
            javax.xml.transform.dom.DOMSource source = new
javax.xml.transform.dom.DOMSource(document);
            javax.xml.transform.stream.StreamResult result = new
javax.xml.transform.stream.StreamResult(writer);

            // transzformálás
            transformer.transform(source, result);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }

}
```

2b) adatmódosítás

Ezt a feladatot a DomModifyMPW46D.java fájlban valósírjuk meg. A feladat öt darab módosítás készítése és kiírása a konzolra.

Végezzük el az alábbi módosításokat:

1. Változtassuk meg az első dolgozó nevét!
2. Változtassuk meg a rendezvény helyszínének a házszámát 1000-re ott, ahol nem üres!
3. Adjunk hozzá egy új allergént a csokis szelethez!
4. Emeljük meg az igazgatók fizetését 50000-rel!
5. Számítsuk ki az összeg származtatott tulajdonságot és adjuk hozzá a dokumentumhoz!

A kód működésének a magyarázata megtalálható a kódban. A módosítások elvégzése után kiírjuk a teljes dokumentumot strukturált formában, amelyből látszódik, hogy a módosításokat helyesen végrehajtottuk.

```
package hu.domparse.mpw46d;

import java.io.File;

import javax.xml.parsers.DocumentBuilderFactory;
import javax.xml.parsers.DocumentBuilder;
import org.w3c.dom.Document;
import org.w3c.dom.NodeList;
import org.w3c.dom.Node;
import org.w3c.dom.Element;
import org.w3c.dom.NamedNodeMap;


public class DomModifyMPW46D {

    public static void main(String[] args) {
        try {
            File inputFile = new File("XMLTaskMPW46D/XMLMPW46D.xml");
            DocumentBuilderFactory dbFactory =
DocumentBuilderFactory.newInstance();
            DocumentBuilder dBuilder = dbFactory.newDocumentBuilder();
            Document doc = dBuilder.parse(inputFile);
            System.out.println("Gyökérelem :" +
doc.getDocumentElement().getNodeName());

            System.out.println("\n-----------------\n");

            // az első dolgozó nevének megváltoztatása
            NodeList dolgozoList = doc.getElementsByTagName("dolgozó");
            for (int temp = 0; temp < dolgozoList.getLength(); temp++) {
                Node nNode = dolgozoList.item(temp);

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

```
                Element dolgozoElement = (Element) nNode;

                NodeList nevList = dolgozoElement.getElementsByTagName("név");

                if (dolgozoElement.getAttribute("DID").equals("d1")) {
                    nevList.item(0).setTextContent("Kiss Márta");
                }
                }
            }

            // ahol a rendezvény házszáma nem üres, megváltoztatjuk 1000-re
            NodeList rendezvenyList = doc.getElementsByTagName("rendezvény");
            for (int temp = 0; temp < rendezvenyList.getLength(); temp++) {
                Node nNode = rendezvenyList.item(temp);

                if (nNode.getNodeType() == Node.ELEMENT_NODE) {
                Element rendezvenyElement = (Element) nNode;

                NodeList hazszamList =
rendezvenyElement.getElementsByTagName("házszám");

                if (hazszamList.getLength() != 0) {
                    hazszamList.item(0).setTextContent("1000");
                }
                }
            }

            // új allergén adása a csokis szelethez
            NodeList sutemenyList = doc.getElementsByTagName("sütemény");
            for (int i = 0; i < sutemenyList.getLength(); i++) {
                Node sutemenyNode = sutemenyList.item(i);
                if (sutemenyNode.getNodeType() == Node.ELEMENT_NODE) {
                    Element sutemenyElement = (Element) sutemenyNode;
                    String currentSid = sutemenyElement.getAttribute("SID");
                    if (currentSid.equals("s3")) {
                        Element newOsszegElement =
doc.createElement("allergének");
                        newOsszegElement.appendChild(doc.createTextNode("mogyo
ró"));

                        sutemenyElement.appendChild(newOsszegElement);
                    }
                }
            }

            // az igatgatók fizetésének megemelése 50000-el
            NodeList igazgatoList = doc.getElementsByTagName("igazgató");
            for (int i = 0; i < igazgatoList.getLength(); i++) {
                Node node = igazgatoList.item(i);
```

```
                if (node.getNodeType() == Node.ELEMENT_NODE) {
                    Element element = (Element) node;
                    Element fizetesElement = (Element)
element.getElementsByTagName("fizetés").item(0);

                    int currentSalary =
Integer.parseInt(fizetesElement.getTextContent());
                    int newSalary = currentSalary + 50000;
                    fizetesElement.setTextContent(Integer.toString(newSalary))
;
                }
            }

            // az összeg származtatott tulajdonság kiszámítása (rendelés
darabszám * sütemény egységár) és módosítása
            calcOsszeg(doc);

            System.out.println("\nMódosítások elvégezve!------------\n");

            // a módosított XML kiírása a konzolra
            printElement(doc.getDocumentElement(), "");

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

    private static void printElement(Element element, String indent) {
        // jelenlegi elem
        System.out.print(indent + "<" + element.getTagName());

        // az elem attribútumai
        NamedNodeMap attributes = element.getAttributes();
        for (int i = 0; i < attributes.getLength(); i++) {
            Node attribute = attributes.item(i);
            System.out.print(" " + attribute.getNodeName() + "=\"" +
attribute.getNodeValue() + "\"");
        }

        // az elem gyermekelemei
        NodeList children = element.getChildNodes();
        if (children.getLength() == 0) {
            // ha nincsenek gyermekelemei, akkor záró taggel fejezzük be
            System.out.println("/>");
        } else {
            // ha vannak gyermekelemei, akkor normális záró taggel folytatjuk
            System.out.println(">");

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

```
                Node child = children.item(i);
                // ha a gyermekelem is tartalmaz elemet, akkor arra is
meghívjuk a printElementet, ha nem, akkor kiírjuk a szöveget
                if (child.getNodeType() == Node.ELEMENT_NODE) {
                    printElement((Element) child, indent + "  ");
                } else if (child.getNodeType() == Node.TEXT_NODE &&
child.getNodeValue().trim().length() > 0) {
                    System.out.println(indent + "  " +
child.getNodeValue().trim());
                }
            }

            // a jelenlegi elem záró tagje
            System.out.println(indent + "</" + element.getTagName() + ">");
        }
    }

    // az összeg kiszámítása
    private static void calcOsszeg(Document document) {
        NodeList rendelList = document.getElementsByTagName("rendel");

        for (int i = 0; i < rendelList.getLength(); i++) {
            Node node = rendelList.item(i);
            if (node.getNodeType() == Node.ELEMENT_NODE) {
                Element element = (Element) node;

                Element darabszamElement = (Element)
element.getElementsByTagName("darabszám").item(0);
                String sidValue = element.getAttribute("rendel_SID");

                // a megfelelő sütemény elem egységárának lekérdezése
                Element sutemenyElement = getSutemenyElement(document,
sidValue);
                Element egysegarElement = (Element)
sutemenyElement.getElementsByTagName("egységár").item(0);

                // az összeg kiszámítása
                int darabszam =
Integer.parseInt(darabszamElement.getTextContent());
                int egysegAr =
Integer.parseInt(egysegarElement.getTextContent());
                int osszeg = darabszam * egysegAr;

                Element newOsszegElement = document.createElement("összeg");
                newOsszegElement.appendChild(document.createTextNode(Integer.t
oString(osszeg)));

                element.appendChild(newOsszegElement);
            }
```

```
        }
    }


    // a sütemény lekérése SID alapján
    private static Element getSutemenyElement(Document document, String sid) {
        NodeList sutemenyList = document.getElementsByTagName("sütemény");

        for (int i = 0; i < sutemenyList.getLength(); i++) {
            Node node = sutemenyList.item(i);
            if (node.getNodeType() == Node.ELEMENT_NODE) {
                Element element = (Element) node;
                if (element.getAttribute("SID").equals(sid)) {
                    return element;
                }
            }
        }

        return null; // nullot dob, ha nem találta a süteményt
    }
}
```

2c) adatlekérdezés

A következő feladatot a DomQueryMPW46D.java fájlban hajtjuk végre. Hajtsunk végre öt lekérdezést, majd írjuk ki őket a konzolra!

A kérdezzük le az alábbiakat:

1. Írjuk ki a dolgozók neveit!
2. Írjuk ki a budapesti cukrászdák neveit!
3. Írjuk ki a rendelt sütemények neveit!
4. Írjuk ki az igazgatók adatait!
5. Írjuk ki annak az igazgatónak a nevét, aki azt a cukrászdát igazgatja, amelyik az r1-es rendezvényre készít süteményeket!

A kód működésének magyarázata megtalálható a kommentekben.

```
package hu.domparse.mpw46d;

import java.io.File;

import javax.xml.parsers.DocumentBuilderFactory;
import javax.xml.parsers.DocumentBuilder;
import org.w3c.dom.Document;
import org.w3c.dom.NodeList;
import org.w3c.dom.Node;
import org.w3c.dom.Element;

public class DomQueryMPW46D {
```

```
    public static void main(String[] args) {
        try {
            File inputFile = new File("XMLTaskMPW46D/XMLMPW46D.xml");
            DocumentBuilderFactory dbFactory =
DocumentBuilderFactory.newInstance();
            DocumentBuilder dBuilder = dbFactory.newDocumentBuilder();
            Document doc = dBuilder.parse(inputFile);
            System.out.println("Gyökérelem :" +
doc.getDocumentElement().getNodeName());

            System.out.println("\n----------------\n");

            //dolgozók neveinek kiírása
            System.out.println("\nDolgozók: ");
            NodeList dolgozoList = doc.getElementsByTagName("dolgozó");

            for (int i = 0; i < dolgozoList.getLength(); i++) {
                Node node = dolgozoList.item(i);
                if (node.getNodeType() == Node.ELEMENT_NODE) {
                    Element dolgozoElement = (Element) node;

                    // név elem lekérése
                    Element nevElement = (Element)
dolgozoElement.getElementsByTagName("név").item(0);

                    // név kiírása
                    String nevValue = nevElement.getTextContent();
                    System.out.println("Dolgozó neve: " + nevValue);
                }
            }

            // budapesti cukrászdák neveinek kiírása
            System.out.println("\nBudapesti cukrászdák: ");
            NodeList cukraszdaList = doc.getElementsByTagName("cukrászda");

            for (int i = 0; i < cukraszdaList.getLength(); i++) {
                Node node = cukraszdaList.item(i);
                if (node.getNodeType() == Node.ELEMENT_NODE) {
                    Element cukraszdaElement = (Element) node;

                    // a város lekérése
                    Element cimElement = (Element)
cukraszdaElement.getElementsByTagName("cím").item(0);
                    String varosValue =
cimElement.getElementsByTagName("város").item(0).getTextContent();

                    // ha Budapest, akkor kiíratjuk
                    if ("Budapest".equals(varosValue)) {
```

```
                        String nevValue =
cukraszdaElement.getElementsByTagName("név").item(0).getTextContent();
                        System.out.println("Cukrászda neve: " + nevValue);
                    }
                }
            }

            // azoknak a süteményeknek a kiírása, amelyeket rendeltek
            System.out.println("\nRendelt sütemények: ");
            NodeList rendelList = doc.getElementsByTagName("rendel");

            for (int i = 0; i < rendelList.getLength(); i++) {
                Node node = rendelList.item(i);
                if (node.getNodeType() == Node.ELEMENT_NODE) {
                    Element rendelElement = (Element) node;

                    // a rendel elem SID értéke
                    String sidValue =
rendelElement.getAttribute("rendel_SID");

                    // a megfelelő értékű sütemény elem
                    Element sutemenyElement = findElementByAttribute(doc,
"sütemény", "SID", sidValue);

                    // a sütemény nevének lekérdezése és kiíratása
                    Element nevElement = (Element)
sutemenyElement.getElementsByTagName("név").item(0);
                    String sutemenyNev = nevElement.getTextContent();

                    System.out.println("Rendelt sütemény neve: " +
sutemenyNev);
                }
            }

            // az igazgatók adatainak kiírása
            System.out.println("\nIgazgatók és adataik:");
            NodeList igazgatoList = doc.getElementsByTagName("igazgató");

            for (int i = 0; i < igazgatoList.getLength(); i++) {
                Node node = igazgatoList.item(i);
                if (node.getNodeType() == Node.ELEMENT_NODE) {
                    Element igazgatoElement = (Element) node;

                    // tulajdonság kiírása
                    String iidValue = igazgatoElement.getAttribute("IID");
                    System.out.println("IID: " + iidValue);

                    // a gyermekelemek kiírása
```

```
                    Element nevElement = (Element)
igazgatoElement.getElementsByTagName("név").item(0);
                    String nevValue = nevElement.getTextContent();
                    System.out.println("név: " + nevValue);

                    Element fizetesElement = (Element)
igazgatoElement.getElementsByTagName("fizetés").item(0);
                    String fizetesValue = fizetesElement.getTextContent();
                    System.out.println("fizetés: " + fizetesValue);

                    NodeList elerhetosegList =
igazgatoElement.getElementsByTagName("elérhetőség");
                    for (int j = 0; j < elerhetosegList.getLength(); j++) {
                        Element elerhetosegElement = (Element)
elerhetosegList.item(j);
                        String elerhetosegValue =
elerhetosegElement.getTextContent();
                        System.out.println("elérhetőség: " +
elerhetosegValue);
                    }

                    System.out.println();
                }
            }

            // annak az igazgatónak a neve, aki azt a cukrászdát igazgatja,
amelyik az r1 rendezvényre készít süteményeket
            System.out.println("\nIgazgató az r1 rendezvényre süteményt
készítő cukrászdánál: ");

            // r1-es rendezvény megkeresése
            Element rendezvenyElement = findElementByAttribute(doc,
"rendezvény", "RID", "r1");

            // a r1 cukrászda_CID-je
            String cukraszdaCID =
rendezvenyElement.getAttribute("rendezvény_CID");

            // az ennek megfelelő cukrászda
            Element cukraszdaElement = findElementByAttribute(doc,
"cukrászda", "CID", cukraszdaCID);

            // az igazgatóhoz szükséges CID
            String igazgatoCID = cukraszdaElement.getAttribute("CID");

            // a megfelelő IID-jű igazgató nevének kiírása
            Element igazgatoElement = findElementByAttribute(doc, "igazgató",
"IID", igazgatoCID);
```

```
            Element nevElement = (Element)
igazgatoElement.getElementsByTagName("név").item(0);
            System.out.println("Név: " + nevElement.getTextContent());

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

    // az attribútumnak megfelelő elem megkeresése
    private static Element findElementByAttribute(Document document, String
elementName, String attributeName, String attributeValue) {
        NodeList nodeList = document.getElementsByTagName(elementName);

        for (int i = 0; i < nodeList.getLength(); i++) {
            Node node = nodeList.item(i);
            if (node.getNodeType() == Node.ELEMENT_NODE) {
                Element element = (Element) node;

                if
(attributeValue.equals(element.getAttribute(attributeName))) {
                    return element;
                }
            }
        }

        return null;
    }
}
```

2d) adatírás

A feladatunk ebben az esetben egy olyan program készítése, amely az XML dokumentum tartalmát kiírja fa struktúra formában a konzolra és egy XMLMPW46D1.xml nevű fájlba. A program neve legyen DomWriteMPW46D.java. A program az egyes node-okhoz külön definiált metódusok segítségével építi fel az xml dokumentumot. A kód működésének a magyarázata megtalálható a kommentekben.

```
package hu.domparse.mpw46d;

import org.w3c.dom.Document;
import org.w3c.dom.Element;
import org.w3c.dom.Text;

import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.DocumentBuilderFactory;
import javax.xml.transform.Transformer;
import javax.xml.transform.TransformerFactory;
import javax.xml.transform.dom.DOMSource;
import javax.xml.transform.stream.StreamResult;
```

```
import java.io.FileOutputStream;

public class DomWriteMPW46D {

    private Document doc;
    private Element root;

    public DomWriteMPW46D() throws Exception {
        // konstruktor a dokumentum építőhöz
        DocumentBuilderFactory docFactory =
DocumentBuilderFactory.newInstance();
        DocumentBuilder docBuilder = docFactory.newDocumentBuilder();
        doc = docBuilder.newDocument();

        // létrehozzuk a gyökérelemet
        root = doc.createElement("MPW46D_cukrászda");
        root.setAttribute("xmlns:xs", "http://www.w3.org/2001/XMLSchema-
instance");
        root.setAttribute("xs:noNamespaceSchemaLocation",
"XMLSchemaMPW46D.xsd");
        doc.appendChild(root);
    }

    // metódus a dolgozó hozzáadásához
    public void addDolgozo(String did, String cid, String nev, int fizetes,
String beosztas) {
        // elem létrehozása
        Element dolgozo = doc.createElement("dolgozó");
        // attribútumok hozzáadása
        dolgozo.setAttribute("DID", did);
        dolgozo.setAttribute("dolgozó_CID", cid);

        // a node elemei hozzádása
        createElement("név", nev, dolgozo);
        createElement("fizetés", String.valueOf(fizetes), dolgozo);
        createElement("beosztás", beosztas, dolgozo);

        // hozzáadjuk a dokumentumhoz
        root.appendChild(dolgozo);
    }

    // metódus az igazgató hozzáadásához
    public void addIgazgato(String iid, String nev, int fizetes, String...
elerhetoseg) {
        // elem létrehozása
        Element igazgato = doc.createElement("igazgató");
        // attribútum hozzáadása
        igazgato.setAttribute("IID", iid);
```

```
        // a node elemei
        createElement("név", nev, igazgato);
        createElement("fizetés", String.valueOf(fizetes), igazgato);

        // az elérhetőségek hozzáadása iteratívan
        for (String elerhet : elerhetoseg) {
            createElement("elérhetőség", elerhet, igazgato);
        }

        // hozzáadás a dokumentumhoz
        root.appendChild(igazgato);
    }

    // metódus a cukrászda hozzáadásához
    public void addCukraszda(String cid, String nev, String varos, int
iranyitoszam, String utca, int hazszam) {
        // cukrászda létrehozása
        Element cukraszda = doc.createElement("cukrászda");
        // attribútum hozzáadása
        cukraszda.setAttribute("CID", cid);

        // elem hozzáadása
        createElement("név", nev, cukraszda);

        // a cím létrehozása
        Element cim = doc.createElement("cím");
        createElement("város", varos, cim);
        createElement("irányítószám", String.valueOf(iranyitoszam), cim);
        createElement("utca", utca, cim);
        createElement("házszám", String.valueOf(hazszam), cim);

        // a cím hozzáadása a cukrászdához
        cukraszda.appendChild(cim);
        // a cukrászda hozzáadása a dokumentumhoz
        root.appendChild(cukraszda);
    }

    // az árul hozzáadása
    public void addArul(String cid, String sid) {
        // árul létrehozása
        Element arul = doc.createElement("árul");
        // attrubútumok létrehozása
        arul.setAttribute("árul_CID", cid);
        arul.setAttribute("árul_SID", sid);

        // arul hozzáadása a dokumentumhoz
        root.appendChild(arul);
    }
```

```
    // sütemény hozzáadása
    public void addSutemeny(String sid, String nev, int egysegar, String...
allergenek) {
        // sütemény létrehozása
        Element sutemeny = doc.createElement("sütemény");
        // ID hozzáadása
        sutemeny.setAttribute("SID", sid);

        // elemek hozzáadása
        createElement("név", nev, sutemeny);
        createElement("egységár", String.valueOf(egysegar), sutemeny);

        // az allergéneket iteratívan adjuk hozzá
        for (String allergen : allergenek) {
            createElement("allergének", allergen, sutemeny);
        }

        // hozzáadás a dokumentumhoz
        root.appendChild(sutemeny);
    }

    // rendel hozzáadása
    public void addRendel(String sid, String mid, int darabszam) {
        // rendel létrehozása
        Element rendel = doc.createElement("rendel");
        // attribútumok hozzáadása
        rendel.setAttribute("rendel_SID", sid);
        rendel.setAttribute("rendel_MID", mid);

        // darabszám hozzáadása
        createElement("darabszám", String.valueOf(darabszam), rendel);

        // rendel hozzáadása a dokumentumhoz
        root.appendChild(rendel);
    }

    // megrendelő hozzáadása
    public void addMegrendelo(String mid, String nev, String... elerhetoseg) {
        // létrehozás és attribútum hozzáadása
        Element megrendelo = doc.createElement("megrendelő");
        megrendelo.setAttribute("MID", mid);

        createElement("név", nev, megrendelo);
        // az elérhetőségek hozzáadása iteratívan
        for (String elerhet : elerhetoseg) {
            createElement("elérhetőség", elerhet, megrendelo);
        }

        // hozzáadás a dokumentumhoz
```

```
        root.appendChild(megrendelo);
    }

    // rendezvény hozzáadása
    public void addRendezveny(String rid, String mid, String cid, String
kezdes, String varos, int iranyitoszam, String utca, int hazszam) {
        // rendezvény létrehozása
        Element rendezveny = doc.createElement("rendezvény");
        // attribútumok hozzáadása
        rendezveny.setAttribute("RID", rid);
        rendezveny.setAttribute("rendezvény_MID", mid);
        rendezveny.setAttribute("rendezvény_CID", cid);

        // kezdés elem létrehozása
        createElement("kezdes", kezdes, rendezveny);

        // helyszín összetett tulajdonság létrehozása
        Element helyszin = doc.createElement("helyszín");
        createElement("város", varos, helyszin);
        createElement("irányítószám", String.valueOf(iranyitoszam), helyszin);
        createElement("utca", utca, helyszin);
        createElement("házszám", String.valueOf(hazszam), helyszin);

        // helyszín hozzáadása a rendezvényhez, rendezvény hozzáadása a
dokumenumhoz
        rendezveny.appendChild(helyszin);
        root.appendChild(rendezveny);
    }

    // a dokumentum kiírása fájlba
    public void docToFile(String fileName) throws Exception {
        // transzformer létrehozása
        TransformerFactory transformerFactory =
TransformerFactory.newInstance();
        Transformer transformer = transformerFactory.newTransformer();
        // indentálás beállítása
        transformer.setOutputProperty(javax.xml.transform.OutputKeys.INDENT,
"yes");

        // forrás és cél beállítása
        DOMSource source = new DOMSource(doc);
        StreamResult result = new StreamResult(new
FileOutputStream(fileName));

        // transzformálás
        transformer.transform(source, result);
    }

    // elem létrehozása
```

```
    private void createElement(String elementName, String value, Element
parent) {
        Element element = doc.createElement(elementName);
        Text textNode = doc.createTextNode(value);
        element.appendChild(textNode);
        parent.appendChild(element);
    }

    // xml kiíratása a konzolra
    public void printXml() throws Exception {
        TransformerFactory transformerFactory =
TransformerFactory.newInstance();
        Transformer transformer = transformerFactory.newTransformer();
        transformer.setOutputProperty(javax.xml.transform.OutputKeys.INDENT,
"yes");

        DOMSource source = new DOMSource(doc);

        // DOMSource áttranszformálása StreamResulttá
        java.io.StringWriter stringWriter = new java.io.StringWriter();
        StreamResult result = new StreamResult(stringWriter);
        transformer.transform(source, result);

        // kiíratás
        System.out.println(stringWriter.toString());
    }

    public static void main(String[] args) {
        try {
            DomWriteMPW46D builder = new DomWriteMPW46D();

            // dolgozók hozzáadása
            builder.addDolgozo("d1", "c1", "Nagy Emese", 300000, "cukrász");
            builder.addDolgozo("d2", "c1", "Kiss Tamás", 700000, "főcukrász");
            builder.addDolgozo("d3", "c1", "Kovács Evelin", 250000,
"cukrászgyakornok");
            builder.addDolgozo("d4", "c2", "Varga Levente", 500000,
"cukrász");
            builder.addDolgozo("d5", "c2", "Sárosi Ferencné", 800000,
"főcukrász");
            builder.addDolgozo("d6", "c2", "Nagy Andor", 600000, "cukrász");
            builder.addDolgozo("d7", "c3", "Nagy Andrea", 900000,
"főcukrász");
            builder.addDolgozo("d8", "c3", "Tóth Tamás", 600000, "cukrász");

            // igazgatók hozzáadása
            builder.addIgazgato("c1", "Farkas Benedek", 1000000,
"farkas.benedek@gmail.com", "+36201234567");
```

```
            builder.addIgazgato("c2", "Szőke László", 2000000, "+36702345645",
"");
            builder.addIgazgato("c3", "Takács Ádám", 1500000, "+36705656789",
"adam.takacs@gmail.com");

            // cukrászdák hozzáadása
            builder.addCukraszda("c1", "Kristály Cukrászda", "Budapest", 1082,
"József krt.", 53);
            builder.addCukraszda("c2", "Daubner Cukrászda", "Budapest", 1025,
"Szépvölgyi út", 50);
            builder.addCukraszda("c3", "Kismandula Cukrászda", "Debrecen",
4024, "Liszt Ferenc utca", 10);

            // árusítások hozzáadása
            builder.addArul("c1", "s1");
            builder.addArul("c1", "s2");
            builder.addArul("c1", "s3");
            builder.addArul("c1", "s4");
            builder.addArul("c2", "s1");
            builder.addArul("c2", "s2");
            builder.addArul("c2", "s3");
            builder.addArul("c2", "s4");
            builder.addArul("c3", "s1");
            builder.addArul("c3", "s2");
            builder.addArul("c3", "s3");
            builder.addArul("c3", "s4");

            // sütemények hozzáadása
            builder.addSutemeny("s1", "Mindenmentes süti", 3000);
            builder.addSutemeny("s2", "kókuszos kocka", 600, "tojás", "tej",
"liszt");
            builder.addSutemeny("s3", "csokis szelet", 700, "liszt", "tojás");
            builder.addSutemeny("s4", "Madeira sütemény", 900, "liszt", "tej",
"tojás", "sajt");

            // rendelések
            builder.addRendel("s1", "m1", 30);
            builder.addRendel("s4", "m2", 20);
            builder.addRendel("s3", "m3", 50);

            // megrendelők hozzáadása
            builder.addMegrendelo("m1", "Kiss Lászlóné", "+36203453434");
            builder.addMegrendelo("m2", "Horváth Csaba", "+36305675656");
            builder.addMegrendelo("m3", "Remete Ákos", "+36204557887");

            // rendezvények hozzáadása
            builder.addRendezveny("r1", "m1", "c1", "2023-12-12 16:00",
"Budapest", 1181, "Városház utca", 1);
```

```
            builder.addRendezveny("r2", "m2", "c1", "2023-12-21 18:00",
"Budapest", 1119, "Fehérvári út", 47);
            builder.addRendezveny("r3", "m3", "c2", "2023-12-29 19:00",
"Budapest", 1133, "Kárpát utca", 23);

            System.out.println("A dokumentum felépítése megtörtént. ");

            builder.docToFile("XMLTaskMPW46D/DomParseMPW46D/XMLMPW46D1.xml");
            System.out.println("A kiírás megtörtént az XMLMPW46D1.xml
dokumentumba. ");
            System.out.println("\n----------------------\n");

            builder.printXml();
            System.out.println("A kiírás megtörtént. ");
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```