JEGYZŐKÖNYV

Adatkezelés XML környezetben

Féléves feladat

Cukrászda

Készítette: Tamás Kinga

Neptun kód: F75CP6

Dátum: 2023.10.30

Tartalomjegyzék

CUKRÁSZDA

TÉMA:

Az adatbázis témája, hogy a Cukrászda működését, rendelésfelvételét megkönnyítse és hatékonyabbá tegye az adatok, vásárlások, rendelések, szállítások kezelését.

EGYEDEK:

Minden megrendelő köteles kitöltetni a megadott információkat (*név, irányítószám, település, utca/házszám*, és *telefonszámból* többet is megadhat)

Szállításnál ki kell választania a módot a megrendelőnek(*személyes átvétel, házhoz szállítás*) és ennek függvényében történik a *szállítási költség* felszámolása, valamint még *megjegyzés*.

Termékek közül választhat majd, hogy milyen *fajtát* szeretne venni (torta, szelet sütemények, sós sütemények) és ezekhez kötelezően tartozik *leírás, ár.*

Cukrászoknál a *név* megadása, *minősítése* kötelező és a hozzá tartozó elérhetőség, mint *e-mail cím*.

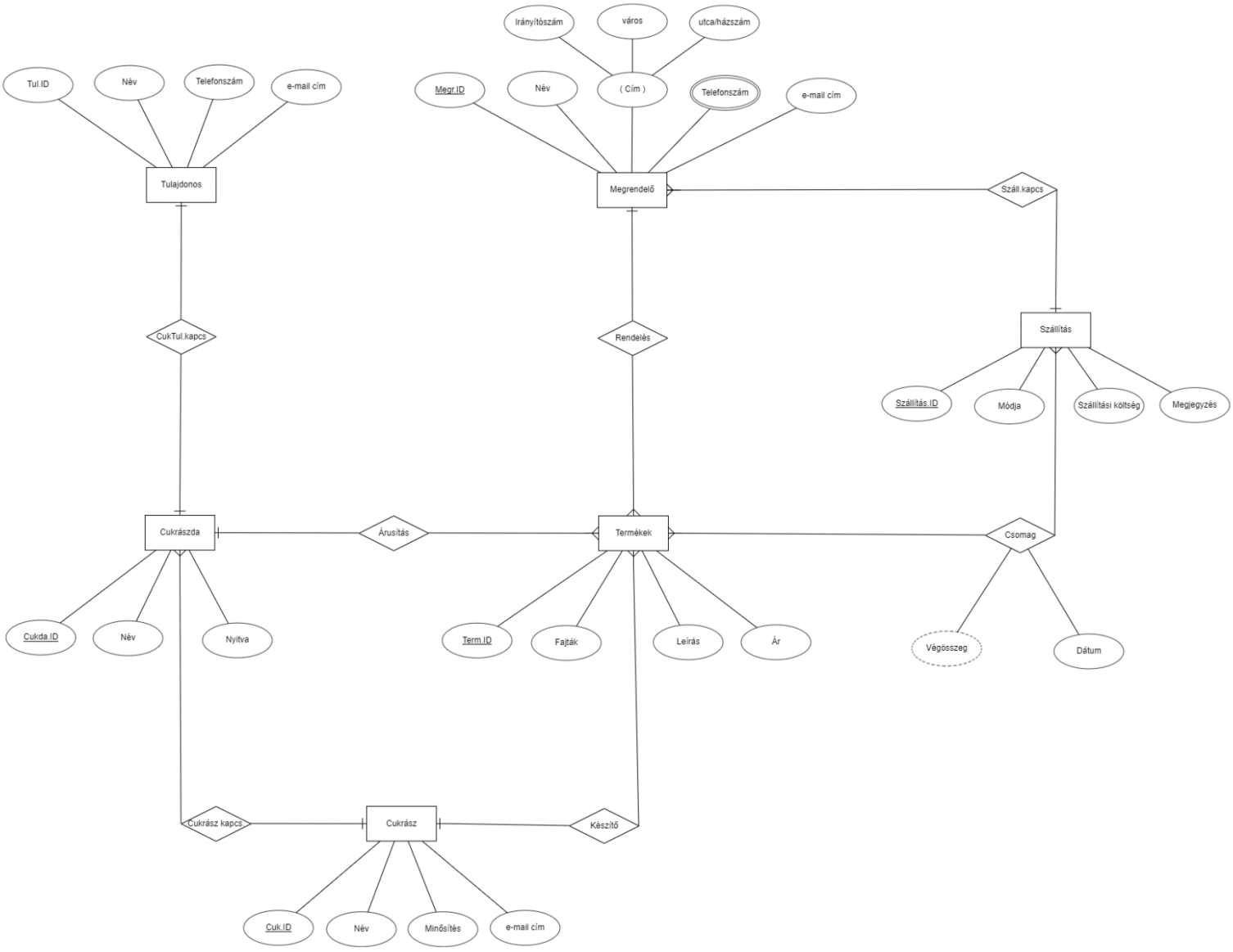
Tulajdonosnál *név, telefonszám, e-mail cím* megadása kötelező.

Cukrászda *neve* és *nyitva (tartása)* kötelező.

- **Megrendelő**: Megr.ID[elsődleges kulcs], név, cím(irányítószám, város, utca/házszám), telefonszám[többértékű]
- **Szállítás**: Szállít.ID[elsődleges kulcs], módja, szállítási költség, megjegyzés
- **Termékek**: Term.ID[elsődleges kulcs],fajták, leírás, ár, mennyiség
- **Cukrász:** Cuk.ID[elsődleges kulcs], név, e-mail cím, minősítés
- **Tulajdonos:** Tul.ID[elsődleges kulcs], név, telefonszám, e-mail cím
- **Cukrászda:** Cukda.ID[elsődleges kulcs], név, nyitva

1. Feladat

1a) Az adatbázis ER modell tervezése



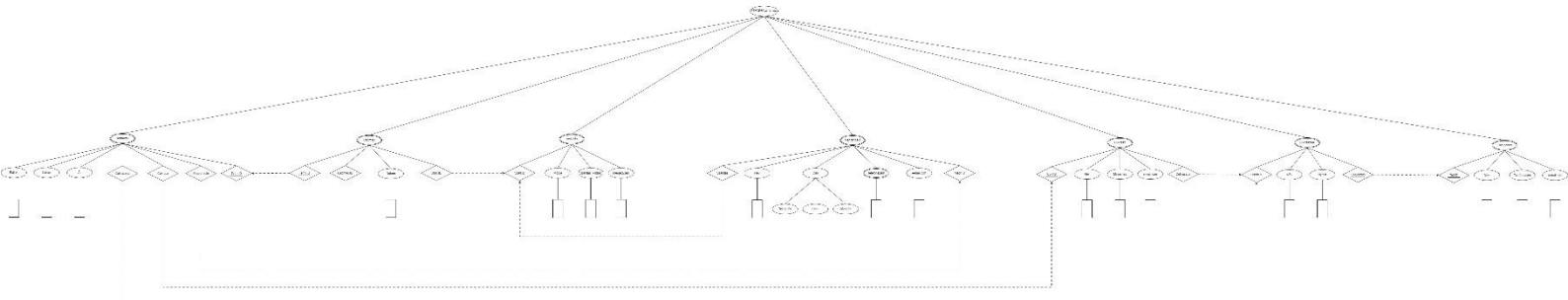
KAPCSOLATOK:

- Megrendelő↔Szállítás *(1:N)*
- Termékek↔Megrendelő *(1:N)*
- Cukrász↔Termékek *(1:N)*
- Termékek↔Szállítás *(N:M)*
- Tulajdonos↔Cukrászda *(1:1)*
- Cukrászda↔Termékek *(1:N)*
- Cukrászda↔Cukrász *(1:N)*

1b) Az adatbázis konvertálása XDM modellre

Az XDM modellt, nagyon egyszerűen tudjuk elkészíteni.

- Ellipszissel jelöljük a az elemeket és minden egyedből majd egy elem lesz és a tulajdonságaikból is.
- A rombuszok jelölik azokat az attribútumokat, amelyek kulcstulajdonságokból keletkeztek.
- Téglalapok jelölik majd a konkrét szöveget, amelyet az XML dokumentumba fogunk beleírni.
- Többszörösen előforduló elemeket duplavonalú ellipszissel jelöljük.
- A szaggatott nyíllal jelöljük az idegen kulcs és az elsődleges kulcs közötti kapcsolatokat.



1c) Az XDM modell alapján XML dokumentum készítése

Az XDM modell alapján felépítettem az XML dokumentumot, és a root element-nek az *F75CP6_Cukraszda* -t írtam.

Minden gyermek elemből létrehoztam 3 példányt. Az attribútumai a kulcsok és idegenkulcsok ezeknek az elemeknek létrehoztam a többi gyermek elemet is.

Az XML forráskód:

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


<F75CP6_Cukraszda xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="XMLSchemaF75CP6.xsd">

<!--- termekek -->

<termekek Term_ID="1" Cukda_ID="11" Cuk_ID="01" Megr_ID="101">
<fajtak>Epres csokitorta</fajtak>
<leiras>csokis piskóta, étcsokis-tejszínes krémmel, epervelővel</leiras>
<ar>12000</ar>
</termekek>

<termekek Term_ID="2" Cukda_ID="12" Cuk_ID="02" Megr_ID="102">
<fajtak>Ishler</fajtak>
<leiras>linzer tészta erdeigyümölcsös dzsemmel</leiras>
<ar>240</ar>
</termekek>

<termekek Term_ID="3" Cukda_ID="13" Cuk_ID="03" Megr_ID="103">
<fajtak>Macaron</fajtak>
<leiras>őrölt mandulából készült korongok sós karamellás krémmel töltve</leiras>
<ar>300</ar>
</termekek>


<!--- Csomag -->

<csomag Term_ID="1" Szall_ID="901">
<datum>2023-07-07</datum>
</csomag>

<csomag  Term_ID="2" Szall_ID="902">
<datum>2023-08-02</datum>
</csomag>

<csomag  Term_ID="3" Szall_ID="903">
<datum>2023-08-11</datum>
</csomag>









<!--- SZÁLLÍTÁS -->
```

```
<szallitas Szall_ID="901">
<modja>Házhoz szállítás</modja>
<szallitasi_koltseg>3250</szallitasi_koltseg>
<megjegyzes>A csomag várhatóan 12:55 és 14:55 között érkezik meg.</megjegyzes>
</szallitas>

<szallitas Szall_ID="902">
<modja>Személyes átvétel</modja>
<szallitasi_koltseg>0</szallitasi_koltseg>
<megjegyzes>Nyitvatartási időben bármikor átveheti termékét.</megjegyzes>
</szallitas>


<szallitas Szall_ID="903">
<modja>Házhoz szállítás</modja>
<szallitasi_koltseg>2700</szallitasi_koltseg>
<megjegyzes>A csomag várhatóan 11:20 és 13:20 között érkezik meg.</megjegyzes>
</szallitas>


<!--- MEGRENDELŐ -->

<megrendelo Megr_ID="101" Szall_ID="901">
<nev>Kis Béla</nev>
<cim>
<telepules>Edelény</telepules>
<utca>Uitz Béla utca</utca>
<hazszam>28</hazszam>
</cim>
<telefonszam>36403127492</telefonszam>
<telefonszam>36805551232</telefonszam>
<e-mail_cim>beluka33@citromail.hu</e-mail_cim>
</megrendelo>


<megrendelo Megr_ID="102" Szall_ID="902">
<nev>Afonyi Dóra</nev>
<cim>
<telepules>Kakucs</telepules>
<utca>Arany János utca</utca>
<hazszam>4</hazszam>
</cim>
<telefonszam>36803978912</telefonszam>
<telefonszam>36408869769</telefonszam>
<e-mail_cim>dora12@gmail.com</e-mail_cim>
</megrendelo>
```

```
<megrendelo Megr_ID="103" Szall_ID="903">
<nev>Stoll Barbara</nev>
<cim>
<telepules>Veresegyház</telepules>
<utca>Petőfi utca</utca>
<hazszam>48</hazszam>
</cim>
<telefonszam>36807958169</telefonszam>
<e-mail_cim>rebarbara18@gmail.com</e-mail_cim>
</megrendelo>



<!--- CUKRÁSZ -->

<cukrasz Cuk_ID="01" Cukda_ID="11">
<nev>Réfi Réka</nev>
<minosites>2</minosites>
<e-mail_cim>rekuci15@gmail.com</e-mail_cim>
</cukrasz>

<cukrasz Cuk_ID="02" Cukda_ID="12">
<nev>Túró Rudolf</nev>
<minosites>3</minosites>
<e-mail_cim>turorudi11@gmail.com</e-mail_cim>
</cukrasz>

<cukrasz Cuk_ID="03" Cukda_ID="13">
<nev>Kerepesi Rebeka</nev>
<minosites>5</minosites>
<e-mail_cim>rebekabeka36@gmail.com</e-mail_cim>
</cukrasz>



<!--- CUKRÁSZDA -->

<cukraszda Cukda_ID="11" Tul_ID="001">
<nev>Hajnali Álom</nev>
<nyitva>8-16</nyitva>
</cukraszda>


<cukraszda Cukda_ID="12" Tul_ID="002">
<nev>Éjféli Kívánság</nev>
<nyitva>8-20</nyitva>
</cukraszda>
```

```
<cukraszda Cukda_ID="13" Tul_ID="003">
<nev>Napsütéses Virágoskert</nev>
<nyitva>10-22</nyitva>
</cukraszda>


<!--- TULAJDONOS -->

<tulajdonos Tul_ID="001">
<nev>Tompa János</nev>
<telefonszam>36803974892</telefonszam>
<e-mail_cim>tmpjni56@gmail.com</e-mail_cim>
</tulajdonos>


<tulajdonos Tul_ID="002">
<nev>Mészáros Mária</nev>
<telefonszam>36904258798</telefonszam>
<e-mail_cim>marikameszaros55@freemail.hu</e-mail_cim>
</tulajdonos>


<tulajdonos Tul_ID="003">
<nev>Várkonyi Erika</nev>
<telefonszam>36503459678</telefonszam>
<e-mail_cim>varkonyierika16@gmail.com</e-mail_cim>
</tulajdonos>



</F75CP6_Cukraszda>
```

1d) Az XML dokumentum alapján XMLSchema készítése

A F75CP6_Cukraszda gyökérelem tartalmazza az összes adatot, és alárendelt elemeiben tárolja a Cukrászdával kapcsolatos különböző információkat. Az XML séma hatféle összetett típust definiál: *termékekTípus, csomagTípus, szállításTípus, megrendelőTípus, cukrászTípus, cukrászdaTípus, tulajdonosTípus*

A séma emellett definiál egy egyszerű típust is, mint például a minősítést (*minősítésTípus*), amely egy 1 és 5 közötti egész számot engedélyez. Emellett definiál kulcsokat és idegen kulcsokat is a kapcsolatok meghatározására az entitások között.

Az egyedi kulcsok és az idegen kulcsok segítenek a kapcsolatok fenntartásában, és biztosítják az adatok integritását. A 1:1 kapcsolat is jelen van a tulajdonos és a cukrászda közötti kapcsolatban.

Az XMLSchema forráskód:

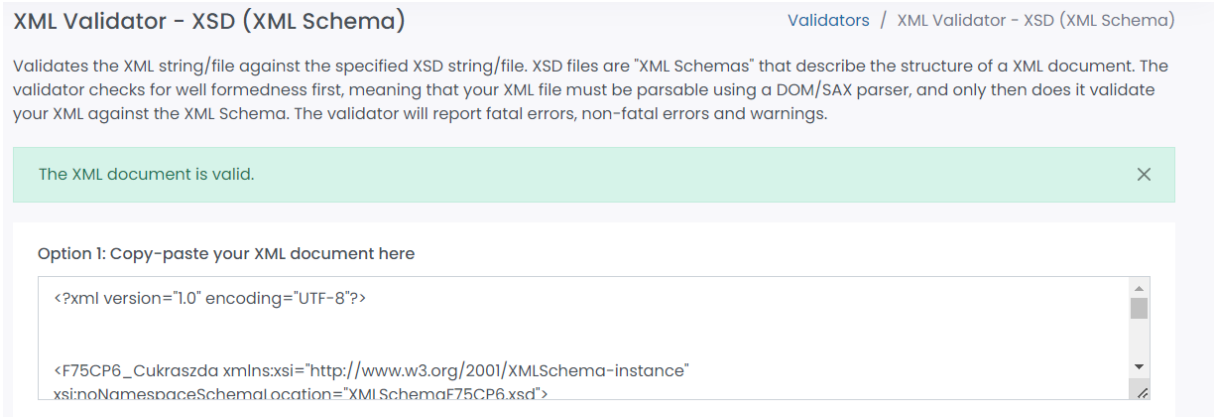
```
<?xml version="1.0" encoding="UTF-8"?>

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

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


    <xs:element name="fajtak" type="xs:string"/>
    <xs:element name="leiras" type="xs:string"/>
    <xs:element name="ar" type="xs:integer"/>
    <xs:element name="datum" type="xs:date"/>
    <xs:element name="modja" type="xs:string"/>
    <xs:element name="szallitasi_koltseg" type="xs:integer"/>
    <xs:element name="megjegyzes" type="xs:string"/>
    <xs:element name="nev" type="xs:string"/>
    <xs:element name="telepules" type="xs:string"/>
    <xs:element name="utca" type="xs:string"/>
    <xs:element name="hazszam" type="xs:integer"/>
    <xs:element name="e-mail_cim" type="xs:string"/>
    <xs:element name="telefonszam" type="xs:integer"/>
    <xs:element name="minosites" type="minősítésTípus"/>
    <xs:element name="nyitva" type="xs:string"/>
```

```
    <xs:simpleType name="minősítésTípus">
        <xs:restriction base="xs:integer">
            <xs:minInclusive value="1" />
            <xs:maxInclusive value="5" />
        </xs:restriction>
    </xs:simpleType>



    <!--Összetett típusok meghatározása-->

    <xs:complexType name="termékekTípus" >
        <xs:sequence>
            <xs:element name="fajtak" type="xs:string"/>
            <xs:element name="leiras" type="xs:string"/>
            <xs:element ref="ar" maxOccurs="unbounded"/>
        </xs:sequence>
        <xs:attribute name="Term_ID" type="xs:integer"/>
        <xs:attribute name="Cukda_ID" type="xs:integer"/>
        <xs:attribute name="Cuk_ID" type="xs:integer"/>
        <xs:attribute name="Megr_ID" type="xs:integer"/>
    </xs:complexType>

    <xs:complexType name="csomagTípus">
        <xs:sequence>
            <xs:element ref="datum" />
        </xs:sequence>
        <xs:attribute name="Term_ID" type="xs:integer"/>
        <xs:attribute name="Szall_ID" type="xs:integer"/>
    </xs:complexType>


    <xs:complexType name="szállításTípus">
        <xs:sequence>
            <xs:element ref="modja" />
            <xs:element ref="szallitasi_koltseg" minOccurs="0" />
            <xs:element name="megjegyzes" type="xs:string"/>
        </xs:sequence>
        <xs:attribute name="Szall_ID" type="xs:integer"/>
    </xs:complexType>









<xs:complexType name="megrendelőTípus">
        <xs:sequence>
```

```
            <xs:element ref="nev"/>
            <xs:element name="cim">
                <xs:complexType>
                    <xs:sequence>
                        <xs:element name="telepules" type="xs:string"/>
                        <xs:element name="utca" type="xs:string"/>
                        <xs:element name="hazszam" type="xs:integer"/>
                    </xs:sequence>
                </xs:complexType>
            </xs:element>
            <xs:element ref="telefonszam" maxOccurs="unbounded"/>
            <xs:element ref="e-mail_cim"/>
        </xs:sequence>
        <xs:attribute name="Megr_ID" type="xs:integer"/>
        <xs:attribute name="Szall_ID" type="xs:integer"/>
    </xs:complexType>


    <xs:complexType name="cukrászTípus">
        <xs:sequence>
            <xs:element ref="nev"/>
            <xs:element ref="minosites"/>
            <xs:element ref="e-mail_cim"/>
        </xs:sequence>
        <xs:attribute name="Cuk_ID" type="xs:integer"/>
        <xs:attribute name="Cukda_ID" type="xs:integer"/>
    </xs:complexType>


    <xs:complexType name="cukrászdaTípus">
        <xs:sequence>
            <xs:element ref="nev"/>
            <xs:element ref="nyitva"/>
        </xs:sequence>
        <xs:attribute name="Cukda_ID" type="xs:integer"/>
        <xs:attribute name="Tul_ID" type="xs:integer"/>
    </xs:complexType>


    <xs:complexType name="tulajdonosTípus">
        <xs:sequence>
            <xs:element ref="nev"/>
            <xs:element ref="telefonszam"/>
            <xs:element ref="e-mail_cim"/>
        </xs:sequence>
        <xs:attribute name="Tul_ID" type="xs:integer"/>
    </xs:complexType>


<!--Gyökérelem meghatározása-->
```

```
<xs:element name="F75CP6_Cukraszda">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="termekek" type="termékekTípus" maxOccurs="unbounded"/>
            <xs:element name="csomag" type="csomagTípus" maxOccurs="unbounded"/>
            <xs:element name="szallitas" type="szállításTípus" maxOccurs="unbounded"/>
            <xs:element name="megrendelo" type="megrendelőTípus" maxOccurs="unbounded"/>
            <xs:element name="cukrasz" type="cukrászTípus" maxOccurs="unbounded"/>
            <xs:element name="cukraszda" type="cukrászdaTípus" maxOccurs="unbounded"/>
            <xs:element name="tulajdonos" type="tulajdonosTípus" maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>



    <!--Elsődleges kulcsok-->

    <xs:key name="termekekPK">
        <xs:selector xpath="termekek"/>
        <xs:field xpath="@Term_ID"/>
    </xs:key>


    <xs:key name="szállításPK">
        <xs:selector xpath="szallitas"/>
        <xs:field xpath="@Szall_ID"/>
    </xs:key>
    <xs:key name="megrendelőPK">
        <xs:selector xpath="megrendelo"/>
        <xs:field xpath="@Megr_ID"/>
    </xs:key>


    <xs:key name="cukrászPK">
        <xs:selector xpath="cukrasz"/>
        <xs:field xpath="@Cuk_ID"/>
    </xs:key>


    <xs:key name="cukrászdaPK">
        <xs:selector xpath="cukraszda"/>
        <xs:field xpath="@Cukda_ID"/>
    </xs:key>


    <xs:key name="tulajdonosPK">
        <xs:selector xpath="tulajdonos"/>
        <xs:field xpath="@Tul_ID"/>
    </xs:key>
    <!-- Idegen kulcsok -->
    <xs:keyref name="cukrászdaTermékekFK" refer="cukrászdaPK">
```

```
        <xs:selector xpath="termekek"/>
        <xs:field xpath="@Cukda_ID"/>
    </xs:keyref>

    <xs:keyref name="cukrászTermékekFK" refer="cukrászPK">
        <xs:selector xpath="termekek"/>
        <xs:field xpath="@Cuk_ID"/>
    </xs:keyref>


    <xs:keyref name="megrendelőTermékekFK" refer="megrendelőPK">
        <xs:selector xpath="termekek"/>
        <xs:field xpath="@Megr_ID"/>
    </xs:keyref>


    <xs:keyref name="termékekCsomagFK" refer="termekekPK">
        <xs:selector xpath="csomag"/>
        <xs:field xpath="@Term_ID"/>
    </xs:keyref>

    <xs:keyref name="szállításCsomagFK" refer="szállításPK">
        <xs:selector xpath="csomag"/>
        <xs:field xpath="@Szall_ID"/>
    </xs:keyref>


    <xs:keyref name="szállításMegrendelőFK" refer="szállításPK">
        <xs:selector xpath="megrendelo"/>
        <xs:field xpath="@Szall_ID"/>
    </xs:keyref>


    <xs:keyref name="cukrászdaCukrászFK" refer="cukrászdaPK">
        <xs:selector xpath="cukrasz"/>
        <xs:field xpath="@Cukda_ID"/>
    </xs:keyref>


    <!-- 1:1 kapcsolat -->

    <xs:unique name="tulajdonosCukrászda" >
        <xs:selector xpath="cukraszda"/>
        <xs:field xpath="@Tul_ID"/>
    </xs:unique>

  </xs:element>
</xs:schema>
```



2. Feladat

2a) DOM adatolvasás

***ReadXMLDocument metódus*:** Ez a metódus felelős az XML fájl beolvasásáért. A filePath paraméterként megadott elérési útvonalon található fájlt nyitja meg.

DocumentBuilderFactory és DocumentBuilder: Ezek az osztályok a DOM parser alapvető elemei. A DocumentBuilderFactory létrehoz egy olyan környezetet, amelyben DocumentBuilder objektumokat hozhatunk létre. A DocumentBuilder feladata az XML fájl beolvasása és egy Document objektum létrehozása, amely a fájl DOM reprezentációját tartalmazza.

Normalizálás: A doc.getDocumentElement().normalize() hívás eltávolítja a felesleges whitespace karaktereket az XML dokumentumból.

printDocument metódus: A metódus az XML dokumentum tartalmának kiírását oldja meg. A dokumentum gyökérelemét (root element) és annak attribútumait írja ki először, majd a többi elemet.

DOMReadF75CP6 kódja:

```java
package hu.domparse.f75cp6;

import org.w3c.dom.*;
import org.xml.sax.SAXException;

import javax.xml.parsers.*;
import java.io.*;
import java.util.StringJoiner;

public class DOMReadF75CP6 {

    public static void ReadXMLDocument(String filePath) {
        try {

            // Fájl beolvasása
            filePath = ("DOMParseF75CP6\\XMLF75CP6.xml");
            File XMLFile = new File(filePath);

            // A DocumentumBuilderFactoryból megkapjuk a DocumentBuildert
            // A DocumentumBuilder tartalmazza az API-t a DOM dokumentumok példányok
            // XML-dokumentumból való beszerzéséhez
   DocumentBuilderFactory dbFactory =  DocumentBuilderFactory.newInstance();

            // DocumentBuilder a példányok létrehozására szolgál
            DocumentBuilder dBuilder = dbFactory.newDocumentBuilder();

            // Ez a dokumentum építésére szolgál
            Document doc = dBuilder.parse(XMLFile);

            // A dokumentum normalizálását segíti a helyes eredmény elérése érdekében
            doc.getDocumentElement().normalize();
            printDocument(doc);

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

    private static void printDocument(Document doc) {
        try {

            // Mentés fájlba
            File outputFile = new File("DOMParseF75CP6\\XMLF75CP6_2.xml");
            PrintWriter writer = new PrintWriter(new FileWriter(outputFile, false));
```

```
// Kiírjuk az XML főgyökér elemét a konzolra és fájlba
Element rootElement = doc.getDocumentElement();
String rootName = rootElement.getTagName();
StringJoiner rootAttributes = new StringJoiner(" ");
NamedNodeMap rootAttributeMap = rootElement.getAttributes();

for (int i = 0; i < rootAttributeMap.getLength(); i++) {
    Node attribute = rootAttributeMap.item(i);
    rootAttributes.add(attribute.getNodeName() + "=\"" + attribute.getNodeValue() + "\"");
}

System.out.print("<?xml version=\"1.0\" encoding=\"UTF-8\"?>\n");
writer.print("<?xml version=\"1.0\" encoding=\"UTF-8\"?>\n");

System.out.print("<" + rootName + " " + rootAttributes.toString() + ">\n");
writer.print("<" + rootName + " " + rootAttributes.toString() + ">\n");

// A gyökér elem alatti elemek lekérése
NodeList termekekList = doc.getElementsByTagName("termekek");
NodeList csomagList = doc.getElementsByTagName("csomag");
NodeList szallitasList = doc.getElementsByTagName("szallitas");
NodeList megrendeloList = doc.getElementsByTagName("megrendelo");
NodeList cukraszList = doc.getElementsByTagName("cukrasz");
NodeList cukraszdaList = doc.getElementsByTagName("cukraszda");
NodeList tulajdonosList = doc.getElementsByTagName("tulajdonos");

// Kiírjuk az XML-t a konzolra megtartva az eredeti formázást
printNodeList(termekekList, writer);
System.out.println("");
writer.println("");
printNodeList(csomagList, writer);
System.out.println("");
writer.println("");
printNodeList(szallitasList, writer);
System.out.println("");
writer.println("");
printNodeList(megrendeloList, writer);
System.out.println("");
writer.println("");
printNodeList(cukraszList, writer);
System.out.println("");
writer.println("");
printNodeList(cukraszdaList, writer);
System.out.println("");
writer.println("");
printNodeList(tulajdonosList, writer);
System.out.println("");

// Zárjuk le az XML gyökér elemét
```

```
        System.out.println("</" + rootName + ">");
        writer.append("</" + rootName + ">");

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

// NodeList kiírása
private static void printNodeList(NodeList, PrintWriter writer) {
    for (int i = 0; i < nodeList.getLength(); i++) {
        Node = nodeList.item(i);
        printNode(node, 1, writer);
        System.out.println("");
        writer.println("");
    }
}

// Node kiírása
private static void printNode(Node node, int indent, PrintWriter writer)

{
    // Ha az elem típusa ELEMENT_NODE, akkor kiírjuk az elem nevét és attribútumait
    if (node.getNodeType() == Node.ELEMENT_NODE) {
        Element = (Element) node;
        String nodeName = element.getTagName();
        StringJoiner attributes = new StringJoiner(" ");
        NamedNodeMap attributeMap = element.getAttributes();

        // Kiírjuk az elem nevét és attribútumait
        for (int i = 0; i < attributeMap.getLength(); i++) {
            Node attribute = attributeMap.item(i);
            attributes.add(attribute.getNodeName() + "=\"" + attribute.getNodeValue() + "\"");
        }

        // Kiírjuk az elem nevét és attribútumait (tartalmát)
        System.out.print(getIndentString(indent));
        System.out.print("<" + nodeName + " " + attributes.toString() + ">");

        writer.print(getIndentString(indent));
        writer.print("<" + nodeName + " " + attributes.toString() + ">");

        NodeList children = element.getChildNodes();
        if (children.getLength() == 1 && children.item(0).getNodeType() == Node.TEXT_NODE) {
            System.out.print(children.item(0).getNodeValue());
            writer.print(children.item(0).getNodeValue());
        } else {
            System.out.println();
            writer.println();
```

```
                for (int i = 0; i < children.getLength(); i++) {
                    printNode(children.item(i), indent + 1, writer);
                }
                System.out.print(getIndentString(indent));
                writer.print(getIndentString(indent));
            }
            System.out.println("</" + nodeName + ">");
            writer.println("</" + nodeName + ">");
        }

    }

    // Behúzások hozzáadása
    private static String getIndentString(int indent) {
        StringBuilder sb = new StringBuilder();
        for (int i = 0; i < indent; i++) {
            // A szóközök száma, minden iteráció után, amit hozzáfűz a Stringbuilderhez
            // amivel indentálunk
            sb.append("  ");
        }
        return sb.toString();
    }
}
```

2b) DOM adatlekérdezés

A DOMQueryF75CP6 osztály létrehozásával adat lekérdezéseket végeztem.

A 6 lekérdezés:

1. Összes termék
2. Kiírja azoknak a Szállításoknak az ID-ját, ahol a szállítási költség nagyobb, mint 3000.
3. Kiírja a megrendelőknek az adatait, ahol házhoz szállítással kérték a terméket.
4. Cukrászok nevei minősítéssel együtt
5. Cukrászdák nevei tulajdonosok neveivel együtt
6. Csomagok szállításának adatai

DOMQueryF75CP6 kódja:

```
package hu.domparse.f75cp6;

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

import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.DocumentBuilderFactory;
import java.io.File;

public class DOMQueryF75CP6 {

    public static void QueryPrescribedDetails(String filePath)

    {
        Document doc = null;

        try {

            // Fájl beolvasása
            filePath = ("DOMParseF75CP6\\XMLF75CP6.xml");
            File inputFile = new File(filePath);

            // Ez létrehoz egy singleton objektumot, amely lehetővé teszi a dokumentumok
            DocumentBuilderFactory dbFactory = DocumentBuilderFactory.newInstance();

            // Ez a dokumentumépítő példányok létrehozására szolgál
            DocumentBuilder dBuilder = dbFactory.newDocumentBuilder();

            // A dokumentum beolvasása
            doc = dBuilder.parse(inputFile);

            // A dokumentum normalizálása
            doc.getDocumentElement().normalize();
        } catch (Exception e) {
            e.printStackTrace();
        }

        // 1. lekérdezés
        System.out.println();
        System.out.println("\n1. Lekérdezés:");
        System.out.println();
        // Kiírja, hogy "Összes a termék:"
        System.out.println("Összes termék:");

        // Lekéri az összes "termékek" elemet az XML-ből
```

```
        NodeList termekekList = doc.getElementsByTagName("termekek");

        // Végigmegy az összes termék elemen
        for (int i = 0; i < termekekList.getLength(); i++)

        {
            Node node = termekekList.item(i);

            // Lellenőrzi, hogy az elemem az tényleg elem típusú-e tehát nem szöveg vagy
            if (node.getNodeType() == Node.ELEMENT_NODE) {
                Element termekek = (Element) node;

                // Kiírja a termékek ID-ját és fajtáit
                System.out.println("Termekék ID: " + termekek.getAttribute("Term_ID"));
                System.out.println("Fajták: " +
termekek.getElementsByTagName("fajtak").item(0).getTextContent());
            }
        }

        // 2. lekérdezés
        System.out.println();
        System.out.println("\n2. Lekérdezés:");
        System.out.println();
        System.out.println("Kiírja azoknak a Szállításoknak az ID-ját, ahol a szállítási költség
nagyobb, mint 3000");

        // Lekéri az összes "szállítás" elemet az XML-ből
        NodeList szallitasList = doc.getElementsByTagName("szallitas");

        // Végigmegy az összes szállítás elemen
        for (int i = 0; i < szallitasList.getLength(); i++) {

            Element szallitasElement = (Element) szallitasList.item(i);
            int szallitasi_koltseg = Integer
                    .parseInt(szallitasElement.getElementsByTagName("szallitasi_koltseg").item(0).
getTextContent());

            // Ha a szállítási költség több, mint 3000, akkor kiírja a szállítás ID-ját
            if (szallitasi_koltseg > 3000) {
                String Szall_ID = szallitasElement.getAttribute("Szall_ID");
                System.out.println("Szall_ID:" + Szall_ID);
            }
        }

        // 3. lekérdezés
        System.out.println();
        NodeList szallitasRendelesList = doc.getElementsByTagName("szallitas");
        System.out.println("\n3. Lekérdezés:");
        System.out.println();
```

```
        System.out.println("Kiírja a megrendelőknek az adatait, ahol házhoz szállítással kérték a
terméket.");

        for (int i = 0; i < szallitasRendelesList.getLength(); i++) {
            Element szallitasElement = (Element) szallitasRendelesList.item(i);
            String szallitasModja =
szallitasElement.getElementsByTagName("modja").item(0).getTextContent();
            String szallitasID = szallitasElement.getAttribute("Szall_ID");

            // Csak a "Házhoz szállítás" módú szállításokra vagyunk kíváncsiak
            if ("Házhoz szállítás".equals(szallitasModja)) {
                // Megrendelő elemek lekérdezése a megfelelő Szall_ID alapján
                NodeList megrendeloList = doc.getElementsByTagName("megrendelo");
                for (int j = 0; j < megrendeloList.getLength(); j++) {
                    Element megrendeloElement = (Element) megrendeloList.item(j);
                    String megrendeloSzállID = megrendeloElement.getAttribute("Szall_ID");

                    // Ha a Szall_ID megegyezik és a Szállítási mód "Házhoz szállítás", akkor
kiírja
                    // a megrendelő adatait
                    if (szallitasID.equals(megrendeloSzállID)) {
                        System.out.println("Megrendelő adatai Szall_ID=" + szallitasID + ":");
                        System.out.println(
                                "Név: " +
megrendeloElement.getElementsByTagName("nev").item(0).getTextContent());

                        // Cím elem lekérdezése
                        Element cimElement = (Element)
megrendeloElement.getElementsByTagName("cim").item(0);
                        if (cimElement != null) {
                            System.out.println("Cím: " +
                    cimElement.getElementsByTagName("telepules").item(0).getTextContent() + ", " +
                    cimElement.getElementsByTagName("utca").item(0).getTextContent() + " " +
                    cimElement.getElementsByTagName("hazszam").item(0).getTextContent());
                        }

       // Telefonszámok lekérdezése
                    NodeList telefonList megrendeloElement.getElementsByTagName("telefonszam");
                        for (int k = 0; k < telefonList.getLength(); k++) {
                            System.out.println("Telefonszám: " +
telefonList.item(k).getTextContent());
                        }

                        System.out.println("E-mail cím: " +
                                megrendeloElement.getElementsByTagName("e-
mail_cim").item(0).getTextContent());

                    }
                }
            }
```

```
        }


        // 4. lekérdezés
        System.out.println();
        NodeList cukraszList = doc.getElementsByTagName("cukrasz");
        System.out.println("\n4. Lekérdezés:");
        System.out.println("\nCukrászok nevei minősítéssel együtt:");

        for (int i = 0; i < cukraszList.getLength(); i++) {
            Element cukraszElement = (Element) cukraszList.item(i);
            String cukraszNev =
cukraszElement.getElementsByTagName("nev").item(0).getTextContent();
            String minosites =
cukraszElement.getElementsByTagName("minosites").item(0).getTextContent();

            System.out.println("Cukrász: " + cukraszNev + ", Minősítés: " + minosites);
        }

        // 5. lekérdezés
        System.out.println();
        NodeList cukraszdaList = doc.getElementsByTagName("cukraszda");
        NodeList tulajdonosList = doc.getElementsByTagName("tulajdonos");

        System.out.println("\n5. Lekérdezés:");
        System.out.println("\nCukrászdák nevei tulajdonosok neveivel együtt:");

        for (int i = 0; i < cukraszdaList.getLength(); i++) {
            Element cukraszdaElement = (Element) cukraszdaList.item(i);
            String cukraszdaNev =
cukraszdaElement.getElementsByTagName("nev").item(0).getTextContent();
            String tulajdonosID = cukraszdaElement.getAttribute("Tul_ID");

            // Tulajdonos nevének lekérdezése a megfelelő Tul_ID alapján
            for (int j = 0; j < tulajdonosList.getLength(); j++) {
                Element tulajdonosElement = (Element) tulajdonosList.item(j);
                String tulajdonosIDinXML = tulajdonosElement.getAttribute("Tul_ID");

                // Ha a Tul_ID megegyezik, akkor kiírja a cukrászda nevét és tulajdonos nevét
                if (tulajdonosID.equals(tulajdonosIDinXML)) {
                    String tulajdonosNev =
tulajdonosElement.getElementsByTagName("nev").item(0).getTextContent();
                    System.out.println("Cukrászda: " + cukraszdaNev + ", Tulajdonos: " +
tulajdonosNev);
                    break; // Kilép a belső ciklusból, mert megtaláltuk a megfelelő tulajdonost
                }
            }
        }
```

```
        // 6. lekérdezés
        System.out.println();

        NodeList csomagList = doc.getElementsByTagName("csomag");
        System.out.println("\n6. Lekérdezés:");
        System.out.println("\nCsomagok szállításának adatai:");

        for (int i = 0; i < csomagList.getLength(); i++) {
            Element csomagElement = (Element) csomagList.item(i);
            String termID = csomagElement.getAttribute("Term_ID");
            String szallID = csomagElement.getAttribute("Szall_ID");

            // Dátum kiírása
            String datum = csomagElement.getElementsByTagName("datum").item(0).getTextContent();
            System.out.println("Csomag Szall_ID=" + szallID + ", Term_ID=" + termID + ":");
            System.out.println("datum: " + datum);

            // Szállítás adatainak kiírása
            NodeList szallitasCsomagList = doc.getElementsByTagName("szallitas");
            for (int j = 0; j < szallitasCsomagList.getLength(); j++) {
                Element szallitasElement = (Element) szallitasCsomagList.item(j);
                String szallIDinXML = szallitasElement.getAttribute("Szall_ID");

                // Ha a Szall_ID megegyezik, akkor kiírja a szállítás adatait
                if (szallID.equals(szallIDinXML)) {
                    String szallitasModja =
szallitasElement.getElementsByTagName("modja").item(0).getTextContent();
                    System.out.println("Szállítási mód: " + szallitasModja);

                    // Termék fajtájának kiírása
                    NodeList termekList = doc.getElementsByTagName("termekek");
                    for (int k = 0; k < termekList.getLength(); k++) {
                        Element termekElement = (Element) termekList.item(k);
                        String termIDinXML = termekElement.getAttribute("Term_ID");

                        // Ha a Term_ID megegyezik, akkor kiírja a termék fajtáját
                        if (termID.equals(termIDinXML)) {
                            String termekFajta =
termekElement.getElementsByTagName("fajtak").item(0).getTextContent();
                         System.out.println("Termék fajta: " + termekFajta);
                         break; // Kilép a belső ciklusból, mert megtaláltuk a megfelelő terméket
                        }
                    }
                    break; // Kilép a középső ciklusból, mert megtaláltuk a megfelelő szállítást
                }
            }
            System.out.println();
        }
    }
}
```

Output:

```
1. Lekérdezés:

Összes termék:
Termekék ID: 1
Fajták: Epres csokitorta
Termekék ID: 2
Fajták: Ishler
Termekék ID: 3
Fajták: Macaron


2. Lekérdezés:

Kiírja azoknak a Szállításoknak az ID-ját, ahol a szállítási költség nagyobb, mint 3000
Szall_ID:901


3. Lekérdezés:

Kiírja a megrendelőknek az adatait, ahol házhoz szállítással kérték a terméket.
Megrendelő adatai Szall_ID=901:
Név: Kis Béla
Cím: Edelény, Uitz Béla utca 28
Telefonszám: 36403127492
Telefonszám: 36805551232
E-mail cím: beluka33@citromail.hu
Megrendelő adatai Szall_ID=903:
Név: Stoll Barbara
Cím: Veresegyház, Petőfi utca 48
Telefonszám: 36807958169
E-mail cím: rebarbara18@gmail.com


4. Lekérdezés:

Cukrászok nevei minősítéssel együtt:
Cukrász: Réfi Réka, Minősítés: 2
Cukrász: Túró Rudolf, Minősítés: 3
Cukrász: Kerepesi Rebeka, Minősítés: 5

5. Lekérdezés:

Cukrászdák nevei tulajdonosok neveivel együtt:
Cukrászda: Hajnali Álom, Tulajdonos: Tompa János
Cukrászda: Éjféli Kívánság, Tulajdonos: Mészáros Mária
Cukrászda: Napsütéses Virágoskert, Tulajdonos: Várkonyi Erika


6. Lekérdezés:

Csomagok szállításának adatai:
Csomag Szall_ID=901, Term_ID=1:
datum: 2023-07-07
Szállítási mód: Házhoz szállítás
Termék fajta: Epres csokitorta

Csomag Szall_ID=902, Term_ID=2:
datum: 2023-08-02
Szállítási mód: Személyes átvétel
Termék fajta: Ishler

Csomag Szall_ID=903, Term_ID=3:
datum: 2023-08-11
Szállítási mód: Házhoz szállítás
Termék fajta: Macaron
```

2c) DOM adatmódosítás

Az adat módosításhoz a következő segéd metódusokat használtam:

- ***getElementsByTagName***: ezzel kértem le a különböző elemeket egy listába amelyeket módosítottam.

- ***setTextContext***: ezzel módosítottam az elementhez tartozó szöveget.

- ***setAttribute:*** ezzel módosítottam az attribútumokat.

Ennek létrehoztam egy osztályt DomQueryF75CP6 néven, és egy metódust ModifyElement. Ez a metódus végig iterál az elements-eken és az összes egyedre lefuttattam a ModifyPrescribedElements metódust, amely módosítja a tulajdonságokat.

DOMModifyF75CP6 kódja:

```
package hu.domparse.f75cp6;

import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.DocumentBuilderFactory;
import javax.xml.transform.OutputKeys;
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 java.io.File;
import java.io.StringWriter;

import org.w3c.dom.*;

public class DOMModifyF75CP6 {

    public static void ModifyElement(String filePath) {
        // Fájl beolvasása

        try {
            filePath = ("DOMParseF75CP6\\XMLF75CP6.xml");
            File inputFile = new File(filePath);
            // Ez létrehoz egy singleton objektumot, amely lehetővé teszi a dokumentumok
            // építését
            DocumentBuilderFactory dbFactory = DocumentBuilderFactory.newInstance();
            // Ez a dokumentumépítő példányok létrehozására szolgál
            DocumentBuilder dBuilder = dbFactory.newDocumentBuilder();
            // Ez a dokumentum építésére szolgál
            Document doc = dBuilder.parse(inputFile);
```

```
        // A dokumentum normalizálását segíti a helyes eredmény elérése érdekében
        doc.getDocumentElement().normalize();
        ModifyPrescribedElements(doc);
    } catch (Exception e) {
        e.printStackTrace();
    }
}

private static void ModifyPrescribedElements(Document doc) throws TransformerException {
    // Root Element lekérése
    NodeList nList = doc.getElementsByTagName("F75CP6_Cukraszda");
    Element element = (Element) nList.item(0);

    // Megváltoztatom az Termékekben az első elem nevét
    NodeList termekekList = element.getElementsByTagName("termekek");
    Element termekek = (Element) termekekList.item(0);
    termekek.getElementsByTagName("fajtak").item(0).setTextContent("Gyümölcsös csokitorta");

    // Például az első Szállítás költségét változtatja meg
    NodeList szallitasList = element.getElementsByTagName("szallitas");
    Element szallitas = (Element) szallitasList.item(0);
    szallitas.getElementsByTagName("szallitasi_koltseg").item(0).setTextContent("6350");

    // Az első megrendelo Települését változtatja meg
    NodeList megrendeloList = element.getElementsByTagName("megrendelo");
    Element megrendelo = (Element) megrendeloList.item(1);
    megrendelo.getElementsByTagName("telepules").item(0).setTextContent("Miskolc");

    // Az első cukrasz minősítését változtatja meg
    NodeList cukraszList = element.getElementsByTagName("cukrasz");
    Element cukrasz = (Element) cukraszList.item(1);
    cukrasz.getElementsByTagName("minosites").item(0).setTextContent("4");

    // Az első Cukrászda nevét változtatja meg
    NodeList cukraszdaList = element.getElementsByTagName("cukraszda");
    Element cukraszda = (Element) cukraszdaList.item(0);
    cukraszda.getElementsByTagName("nev").item(0).setTextContent("Hajnali Harmat Cukrászda");

    // Az első tulajdonos nevét változtatja meg
    NodeList tulajdonosList = element.getElementsByTagName("tulajdonos");
    Element tulajdonos = (Element) tulajdonosList.item(0);
    tulajdonos.getElementsByTagName("nev").item(0).setTextContent("Tommy Hilfiger");

    // Az első csomag dátumát változtatja meg
    NodeList csomagList = element.getElementsByTagName("csomag");
    Element csomag = (Element) csomagList.item(0);
    csomag.getElementsByTagName("datum").item(0).setTextContent("2022-01-01");

    printDocument(doc);
}
```

```
    private static void printDocument(Document doc) throws TransformerException {
        TransformerFactory tf = TransformerFactory.newInstance();
        Transformer transformer = tf.newTransformer();
        transformer.setOutputProperty(OutputKeys.OMIT_XML_DECLARATION, "no");
        transformer.setOutputProperty(OutputKeys.INDENT, "yes");
        StringWriter writer = new StringWriter();
        transformer.transform(new DOMSource(doc), new StreamResult(writer));
        String output = writer.getBuffer().toString();
        System.out.println(output);
    }
}
```

2d) DOM adatírás

Létrehoztam egy osztályt, amely leírja majd az elemeket a konzolba az XML forráskód mintájára. A metódus a dokumentum létrehozásáért és annak különböző elemekkel való feltöltéséért felelős. A DocumentBuilderFactory és DocumentBuilder segítségével hozza létre az üres XML dokumentumot, amelybe különböző elemeket ad hozzá.

Root Element: Az *F75CP6_Cukraszda* nevű gyökérelemet hoz létre, amely az XML dokumentum alapját képezi. Ehhez attribútumokat is rendel, a névtér és a séma helye.

A kód tartalmaz különböző segédmetódusokat (***addTermekek, addSzallitas, addCsomag, addMegrendelo*** stb.), amelyekkel ezeket az entitásokat hozzáadja majd a dokumentumhoz. Ezek az entitások különböző attribútumokkal, gyerek elemekkel rendelkeznek.

createElement segédmetódus: Ez a metódus egy új XML elemet hoz létre a megadott névvel és értékkel. Segít abban, hogy az elemek létrehozásának folyamatát egyszerűsítse

Dokumentum kiíratása: A Transformer osztály létrehozásával a kód kiírja az XML dokumentumot egy fájlba, valamint a konzolra is. Figyel arra, hogy jól olvasható legyen.

printDocument, printNodeList, és printNode metódusok: Ezek a metódusok felelősek a dokumentum és annak egyes elemeinek kiírásáért. A printNode metódus rekurzívan járja be az XML elemeket, és gondoskodik a megfelelő behúzásokról.

DOMWriteF75CP6 kódja:

```java
package hu.domparse.f75cp6;

import org.w3c.dom.Document;
import org.w3c.dom.Element;
import org.w3c.dom.NamedNodeMap;
import org.w3c.dom.Node;
import org.w3c.dom.NodeList;
import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.DocumentBuilderFactory;
import javax.xml.transform.OutputKeys;
import javax.xml.transform.Transformer;
import javax.xml.transform.TransformerFactory;
import java.io.File;
import java.io.FileWriter;
import java.io.PrintWriter;
import java.util.Arrays;
import java.util.List;
import java.util.StringJoiner;

public class DOMWriteF75CP6 {

    public static void WriteElementsToFileAndConsole() {

        try {
            // Előkészítjük a dokumentumot
            DocumentBuilderFactory factory = DocumentBuilderFactory.newInstance();

            // Ez a dokumentumépítő példányok létrehozására szolgál
            DocumentBuilder builder = factory.newDocumentBuilder();

            // Ez a dokumentum építésére szolgál
            Document doc = builder.newDocument();

            // Root Element létrehozása
            Element rootElement = doc.createElement("F75CP6_Cukraszda");
            rootElement.setAttribute("xmlns:xsi", "http://www.w3.org/2001/XMLSchema-
instance");
            rootElement.setAttribute("xsi:noNamespaceSchemaLocation", "XMLSchemaF75CP6.xsd");
            doc.appendChild(rootElement);

            // Termékek létrehozása
            addTermekek(doc, rootElement, "1", "11", "01", "101", "Epres csokitorta",
                    "csokis piskóta, étcsokis-tejszínes krémmel, epervelővel", "12000");
            addTermekek(doc, rootElement, "2", "12", "02", "102", "Ishler", "linzer tészta
erdeigyümölcsös dzsemmel",
                    "240");
            addTermekek(doc, rootElement, "3", "13", "03", "103", "Macaron",
                    "őrölt mandulából készült korongok sós karamellás krémmel töltve",
"300");
```

```
            // Szállítás létrehozása
            addSzallitas(doc, rootElement, "901", "Házhoz szállítás", "3250",
                    "A csomag várhatóan 12:55 és 14:55 között érkezik meg.");
            addSzallitas(doc, rootElement, "902", "Személyes átvétel", "0",
                    "Nyitvatartási időben bármikor átveheti termékét.");
            addSzallitas(doc, rootElement, "903", "Házhoz szállítás", "2700",
                    "A csomag várhatóan 11:20 és 13:20 között érkezik meg.");

            // Megrendelő létrehozása
            addMegrendelo(doc, rootElement, "101", "901", "Kis Béla", "Edelény", "Uitz Béla
utca", "28",
                    Arrays.asList("36403127492", "36805551232"), "beluka33@citromail.hu");

            addMegrendelo(doc, rootElement, "102", "902", "Afonyi Dóra", "Kakucs", "Arany
János utca", "4",
                    Arrays.asList("36803978912", "36408869769"), "dora12@gmail.com");

            addMegrendelo(doc, rootElement, "103", "903", "Stoll Barbara", "Veresegyház",
"Petőfi utca", "48",
                    Arrays.asList("36807958169"), "rebarbara18@gmail.com");

            // Cukrász létrehozása
            addCukrasz(doc, rootElement, "01", "11", "Réfi Réka", "2", "rekuci15@gmail.com");
            addCukrasz(doc, rootElement, "02", "12", "Túró Rudolf", "3",
"turorudi11@gmail.com");
            addCukrasz(doc, rootElement, "03", "13", "Kerepesi Rebeka", "5",
"rebekabeka36@gmail.com");

            // Cukrászda létrehozása

            addCukraszda(doc, rootElement, "11", "001", "Hajnali Álom", "8-16");
            addCukraszda(doc, rootElement, "12", "002", "Éjféli Kívánság", "8-20");
            addCukraszda(doc, rootElement, "13", "003", "Napsütéses Virágoskert", "10-22");

            // Tuladonos létrehozása
            addTulajdonos(doc, rootElement, "001", "Tompa János", "36803974892",
"tmpjni56@gmail.com");
            addTulajdonos(doc, rootElement, "002", "Mészáros Mária", "36904258798",
"marikameszaros55@freemail.hu");
            addTulajdonos(doc, rootElement, "003", "Várkonyi Erika", "36503459678",
"varkonyierika16@gmail.com");

            // Csomag létrehozása
            addCsomag(doc, rootElement, "1", "901", "2023-07-07");
            addCsomag(doc, rootElement, "2", "902", "2023-08-02");
            addCsomag(doc, rootElement, "3", "903", "2023-08-11");

            // Dokumentum mentése
            TransformerFactory transformerFactory = TransformerFactory.newInstance();
            Transformer transformer = transformerFactory.newTransformer();
```

```
            transformer.setOutputProperty(OutputKeys.ENCODING, "UTF-8");
            transformer.setOutputProperty(OutputKeys.INDENT, "yes");
            transformer.setOutputProperty("{https://xml.apache.org/xslt}indent-amount", "2");

            printDocument(doc);
        } catch (

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

    private static void addTermekek(Document doc, Element rootElement, String Term_ID, String
Cukda_ID, String Cuk_ID,
            String Megr_ID, String fajtak, String leiras, String ar)

    {
        Element termekek = doc.createElement("termekek");
        termekek.setAttribute("Term_ID", Term_ID);
        termekek.setAttribute("Cukda_ID", Cukda_ID);
        termekek.setAttribute("Cuk_ID", Cuk_ID);
        termekek.setAttribute("Megr_ID", Megr_ID);
        rootElement.appendChild(termekek);

        Element fajtakElem = doc.createElement("fajtak");
        fajtakElem.appendChild(doc.createTextNode(fajtak));
        termekek.appendChild(fajtakElem);

        Element leirasElem = doc.createElement("leiras");
        leirasElem.appendChild(doc.createTextNode(leiras));
        termekek.appendChild(leirasElem);

        Element arElem = doc.createElement("ar");
        arElem.appendChild(doc.createTextNode(ar));
        termekek.appendChild(arElem);
    }

    private static void addCsomag(Document doc, Element rootElement, String Term_ID, String
Szall_ID, String datum)

    {
        Element csomag = doc.createElement("csomag");
        csomag.setAttribute("Term_ID", Term_ID);
        csomag.setAttribute("Szall_ID", Szall_ID);
        rootElement.appendChild(csomag);

        Element datumElem = doc.createElement("datum");
        datumElem.appendChild(doc.createTextNode(datum));
        csomag.appendChild(datumElem);
    }
```

```
    private static void addSzallitas(Document doc, Element rootElement, String Szall_ID,
String modja,
            String szallitasi_koltseg, String megjegyzes) {
        Element szallitas = doc.createElement("szallitas");
        szallitas.setAttribute("Szall_ID", Szall_ID);
        rootElement.appendChild(szallitas);

        Element modjaElem = doc.createElement("modja");
        modjaElem.appendChild(doc.createTextNode(modja));
        szallitas.appendChild(modjaElem);

        Element szallitasiKoltsegElem = doc.createElement("szallitasi_koltseg");
        szallitasiKoltsegElem.appendChild(doc.createTextNode(szallitasi_koltseg));
        szallitas.appendChild(szallitasiKoltsegElem);

        Element megjegyzesElem = doc.createElement("megjegyzes");
        megjegyzesElem.appendChild(doc.createTextNode(megjegyzes));
        szallitas.appendChild(megjegyzesElem);
    }

    private static void addMegrendelo(Document doc, Element rootElement, String Megr_ID,
String Szall_ID, String nev,
            String telepules, String utca, String hazszam, List<String> telefonszam, String
email)

    {
        Element megrendelo = doc.createElement("megrendelo");
        megrendelo.setAttribute("Megr_ID", Megr_ID);
        megrendelo.setAttribute("Szall_ID", Szall_ID);
        rootElement.appendChild(megrendelo);

        Element nevElem = doc.createElement("nev");
        nevElem.appendChild(doc.createTextNode(nev));
        megrendelo.appendChild(nevElem);

        Element cim = doc.createElement("cim");
        Element telepulesElem = doc.createElement("telepules");
        telepulesElem.appendChild(doc.createTextNode(telepules));
        cim.appendChild(telepulesElem);

        Element utcaElem = doc.createElement("utca");
        utcaElem.appendChild(doc.createTextNode(utca));
        cim.appendChild(utcaElem);

        Element hazszamElem = doc.createElement("hazszam");
        hazszamElem.appendChild(doc.createTextNode(hazszam));
        cim.appendChild(hazszamElem);

        megrendelo.appendChild(cim);
```

```
        for (String szam : telefonszam) {
            Element telefonszamElem = doc.createElement("telefonszam");
            telefonszamElem.appendChild(doc.createTextNode(szam));
            megrendelo.appendChild(telefonszamElem);
        }

        Element emailElem = doc.createElement("e-mail_cim");
        emailElem.appendChild(doc.createTextNode(email));
        megrendelo.appendChild(emailElem);
    }

    private static void addCukrasz(Document doc, Element rootElement, String Cuk_ID, String
Cukda_ID, String nev,
            String minosites, String email) {

        Element cukrasz = doc.createElement("cukrasz");
        cukrasz.setAttribute("Cuk_ID", Cuk_ID);
        cukrasz.setAttribute("Cukda_ID", Cukda_ID);
        rootElement.appendChild(cukrasz);

        Element nevElem = doc.createElement("nev");
        nevElem.appendChild(doc.createTextNode(nev));
        cukrasz.appendChild(nevElem);

        Element minositestElem = doc.createElement("minosites");
        minositestElem.appendChild(doc.createTextNode(minosites));
        cukrasz.appendChild(minositestElem);

        Element emailElem = doc.createElement("e-mail_cim");
        emailElem.appendChild(doc.createTextNode(email));
        cukrasz.appendChild(emailElem);

    }

    private static void addCukraszda(Document doc, Element rootElement, String Cukda_ID,
String Tul_ID, String nev,
            String nyitva) {

        Element cukraszda = doc.createElement("cukraszda");
        cukraszda.setAttribute("Cukda_ID", Cukda_ID);
        cukraszda.setAttribute("Tul_ID", Tul_ID);
        rootElement.appendChild(cukraszda);

        Element nevElem = doc.createElement("nev");
        nevElem.appendChild(doc.createTextNode(nev));
        cukraszda.appendChild(nevElem);

        Element nyitvaElem = doc.createElement("nyitva");
        nyitvaElem.appendChild(doc.createTextNode(nyitva));
```

```
        cukraszda.appendChild(nyitvaElem);

    }

    private static void addTulajdonos(Document doc, Element rootElement, String Tul_ID,
String nev, String telefonszam,
            String email) {

        Element tulajdonos = doc.createElement("tulajdonos");
        tulajdonos.setAttribute("Tul_ID", Tul_ID);
        rootElement.appendChild(tulajdonos);

        Element nevElem = doc.createElement("nev");
        nevElem.appendChild(doc.createTextNode(nev));
        tulajdonos.appendChild(nevElem);

        Element telefonszamElem = doc.createElement("telefonszam");
        telefonszamElem.appendChild(doc.createTextNode(telefonszam));
        tulajdonos.appendChild(telefonszamElem);

        Element emailElem = doc.createElement("e-mail_cim");
        emailElem.appendChild(doc.createTextNode(email));
        tulajdonos.appendChild(emailElem);
    }

    private static void printDocument(Document doc) {

        try {
            // Fájlba írás
            File outputFile = new File("DOMParseF75CP6\\XMLF75CP6_3.xml");
            // Írás a konzolra
            PrintWriter writer = new PrintWriter(new FileWriter(outputFile, false));
            // Kiírja az XML főgyökér elemét a konzolra és fájlba
            Element rootElement = doc.getDocumentElement();
            String rootName = rootElement.getTagName();
            // A gyökér elem attribútumainak kiírása
            StringJoiner rootAttributes = new StringJoiner(" ");
            // A gyökér elem attribútumainak lekérése
            NamedNodeMap rootAttributeMap = rootElement.getAttributes();

            for (int i = 0; i < rootAttributeMap.getLength(); i++) {
                Node attribute = rootAttributeMap.item(i);
                rootAttributes.add(attribute.getNodeName() + "=\"" + attribute.getNodeValue()
+ "\"");
            }

            System.out.print("<?xml version=\"1.0\" encoding=\"UTF-8\"?>\n");
            writer.print("<?xml version=\"1.0\" encoding=\"UTF-8\"?>\n");

            System.out.print("<" + rootName + " " + rootAttributes.toString() + ">\n");
```

```
            writer.print("<" + rootName + " " + rootAttributes.toString() + ">\n");

            // A gyökér elem alatti elemek lekérése
            NodeList termekekList = doc.getElementsByTagName("termekek");
            NodeList csomagList = doc.getElementsByTagName("csomag");
            NodeList szallitasList = doc.getElementsByTagName("szallitas");
            NodeList megrendeloList = doc.getElementsByTagName("megrendelo");
            NodeList cukraszList = doc.getElementsByTagName("cukrasz");
            NodeList cukraszdaList = doc.getElementsByTagName("cukraszda");
            NodeList tulajdonosList = doc.getElementsByTagName("tulajdonos");

            // Kiírjuk az XML-t a konzolra megtartva az eredeti formázást
            printNodeList(termekekList, writer);
            System.out.println("");
            writer.println("");
            printNodeList(csomagList, writer);
            System.out.println("");
            writer.println("");
            printNodeList(szallitasList, writer);
            System.out.println("");
            writer.println("");
            printNodeList(megrendeloList, writer);
            System.out.println("");
            writer.println("");
            printNodeList(cukraszList, writer);
            System.out.println("");
            writer.println("");
            printNodeList(cukraszdaList, writer);
            System.out.println("");
            writer.println("");
            printNodeList(tulajdonosList, writer);
            System.out.println("");

            // XML gyökér lezárása
            System.out.println("</" + rootName + ">");
            writer.append("</" + rootName + ">");

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

    // NodeList kiírása
    private static void printNodeList(NodeList nodeList, PrintWriter writer) {
        for (int i = 0; i < nodeList.getLength(); i++) {
            Node node = nodeList.item(i);
            printNode(node, 1, writer);
            System.out.println("");
            writer.println("");
```

```
        }
    }

    // Node kiírása
    private static void printNode(Node node, int indent, PrintWriter writer)

    {
        // Ha az elem típusa ELEMENT_NODE, akkor kiírjuk az elem nevét és attribútumait
        if (node.getNodeType() == Node.ELEMENT_NODE) {
            Element element = (Element) node;
            String nodeName = element.getTagName();
            StringJoiner attributes = new StringJoiner(" ");
            NamedNodeMap attributeMap = element.getAttributes();

            // Kiírjuk az elem nevét és attribútumait
            for (int i = 0; i < attributeMap.getLength(); i++) {
                Node attribute = attributeMap.item(i);
                attributes.add(attribute.getNodeName() + "=\"" + attribute.getNodeValue() +
"\"");
            }

            // Kiírjuk az elem nevét és attribútumait (tartalmát)
            System.out.print(getIndentString(indent));
            System.out.print("<" + nodeName + " " + attributes.toString() + ">");

            writer.print(getIndentString(indent));
            writer.print("<" + nodeName + " " + attributes.toString() + ">");

            NodeList children = element.getChildNodes();
            if (children.getLength() == 1 && children.item(0).getNodeType() ==
Node.TEXT_NODE) {
                System.out.print(children.item(0).getNodeValue());
                writer.print(children.item(0).getNodeValue());
            }

            else {
                System.out.println();
                writer.println();
                for (int i = 0; i < children.getLength(); i++) {
                    printNode(children.item(i), indent + 1, writer);
                }
                System.out.print(getIndentString(indent));
                writer.print(getIndentString(indent));
            }
            System.out.println("</" + nodeName + ">");
            writer.println("</" + nodeName + ">");
        }
    }
```

```java
    // Behúzások hozzáadása
    private static String getIndentString(int indent) {
        StringBuilder sb = new StringBuilder();
        for (int i = 0; i < indent; i++) {
            sb.append("  ");
        }
        return sb.toString();
    }
}
```

Output:

```
</F75CP6_Cukraszda>
<?xml version="1.0" encoding="UTF-8"?>
<F75CP6_Cukraszda xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="XMLSchemaF75CP6.xsd">
  <termekek Cuk_ID="01" Cukda_ID="11" Megr_ID="101" Term_ID="1">
    <fajtak >Epres csokitorta</fajtak>
    <leiras >csokis piskóta, étcsokis-tejszínes krémmel, epervelővel</leiras>
    <ar >12000</ar>
  </termekek>

  <termekek Cuk_ID="02" Cukda_ID="12" Megr_ID="102" Term_ID="2">
    <fajtak >Ishler</fajtak>
    <leiras >linzer tészta erdeigyümölcsös dzsemmel</leiras>
    <ar >240</ar>
  </termekek>

  <termekek Cuk_ID="03" Cukda_ID="13" Megr_ID="103" Term_ID="3">
    <fajtak >Macaron</fajtak>
    <leiras >őrölt mandulából készült korongok sós karamellás krémmel töltve</leiras>
    <ar >300</ar>
  </termekek>


  <csomag Szall_ID="901" Term_ID="1">
    <datum >2023-07-07</datum>
  </csomag>

  <csomag Szall_ID="902" Term_ID="2">
    <datum >2023-08-02</datum>
  </csomag>

  <csomag Szall_ID="903" Term_ID="3">
    <datum >2023-08-11</datum>
  </csomag>


  <szallitas Szall_ID="901">
    <modja >Házhoz szállítás</modja>
    <szallitasi_koltseg >3250</szallitasi_koltseg>
    <megjegyzes >A csomag várhatóan 12:55 és 14:55 között érkezik meg.</megjegyzes>
  </szallitas>

  <szallitas Szall_ID="902">
    <modja >Személyes átvétel</modja>
    <szallitasi_koltseg >0</szallitasi_koltseg>
    <megjegyzes >Nyitvatartási időben bármikor átveheti termékét.</megjegyzes>
  </szallitas>

  <szallitas Szall_ID="903">
    <modja >Házhoz szállítás</modja>
    <szallitasi_koltseg >2700</szallitasi_koltseg>
    <megjegyzes >A csomag várhatóan 11:20 és 13:20 között érkezik meg.</megjegyzes>
  </szallitas>


  <megrendelo Megr_ID="101" Szall_ID="901">
    <nev >Kis Béla</nev>
    <cim >
      <telepules >Edelény</telepules>
      <utca >Uitz Béla utca</utca>
      <hazszam >28</hazszam>
    </cim>
    <telefonszam >36403127492</telefonszam>
    <telefonszam >36805551232</telefonszam>
    <e-mail_cim >beluka33@citromail.hu</e-mail_cim>
  </megrendelo>

  <megrendelo Megr_ID="102" Szall_ID="902">
    <nev >Afonyi Dóra</nev>
    <cim >
      <telepules >Kakucs</telepules>
      <utca >Arany János utca</utca>
      <hazszam >4</hazszam>
    </cim>
    <telefonszam >36803978912</telefonszam>
    <telefonszam >36408869769</telefonszam>
    <e-mail_cim >dora12@gmail.com</e-mail_cim>
  </megrendelo>
```

```
 <megrendelo Megr_ID="103" Szall_ID="903">
   <nev >Stoll Barbara</nev>
   <cim >
     <telepules >Veresegyház</telepules>
     <utca >Petőfi utca</utca>
     <hazszam >48</hazszam>
   </cim>
   <telefonszam >36807958169</telefonszam>
   <e-mail_cim >rebarbara18@gmail.com</e-mail_cim>
 </megrendelo>


 <cukrasz Cuk_ID="01" Cukda_ID="11">
   <nev >Réfi Réka</nev>
   <minosites >2</minosites>
   <e-mail_cim >rekuci15@gmail.com</e-mail_cim>
 </cukrasz>

 <cukrasz Cuk_ID="02" Cukda_ID="12">
   <nev >Túró Rudolf</nev>
   <minosites >3</minosites>
   <e-mail_cim >turorudi11@gmail.com</e-mail_cim>
 </cukrasz>

 <cukrasz Cuk_ID="03" Cukda_ID="13">
   <nev >Kerepesi Rebeka</nev>
   <minosites >5</minosites>
   <e-mail_cim >rebekabeka36@gmail.com</e-mail_cim>
 </cukrasz>


 <cukraszda Cukda_ID="11" Tul_ID="001">
   <nev >Hajnali Álom</nev>
   <nyitva >8-16</nyitva>
 </cukraszda>

 <cukraszda Cukda_ID="12" Tul_ID="002">
   <nev >Éjféli Kívánság</nev>
   <nyitva >8-20</nyitva>
 </cukraszda>

 <cukraszda Cukda_ID="13" Tul_ID="003">
   <nev >Napsütéses Virágoskert</nev>
   <nyitva >10-22</nyitva>
 </cukraszda>
 <tulajdonos Tul_ID="001">
   <nev >Tompa János</nev>
   <telefonszam >36803974892</telefonszam>
   <e-mail_cim >tmpjni56@gmail.com</e-mail_cim>
 </tulajdonos>

 <tulajdonos Tul_ID="002">
   <nev >Mészáros Mária</nev>
   <telefonszam >36904258798</telefonszam>
   <e-mail_cim >marikameszaros55@freemail.hu</e-mail_cim>
 </tulajdonos>

 <tulajdonos Tul_ID="003">
   <nev >Várkonyi Erika</nev>
   <telefonszam >36503459678</telefonszam>
   <e-mail_cim >varkonyierika16@gmail.com</e-mail_cim>
 </tulajdonos>


</F75CP6_Cukraszda>
PS D:\XML\XMLTaskF75CP6>
```