JEGYZŐKÖNYV

FÉLÉVES FELADAT

ADATKEZELÉS XML-BEN

Készítette: Pető Ádám

Neptunkód: S0KK3C

Az én beadandóm egy bizonyos élelmiszer átalakulását és útját szeretné leírni, ahogyan eljut a vevő-/megrendelőhöz egy éttermen keresztül.

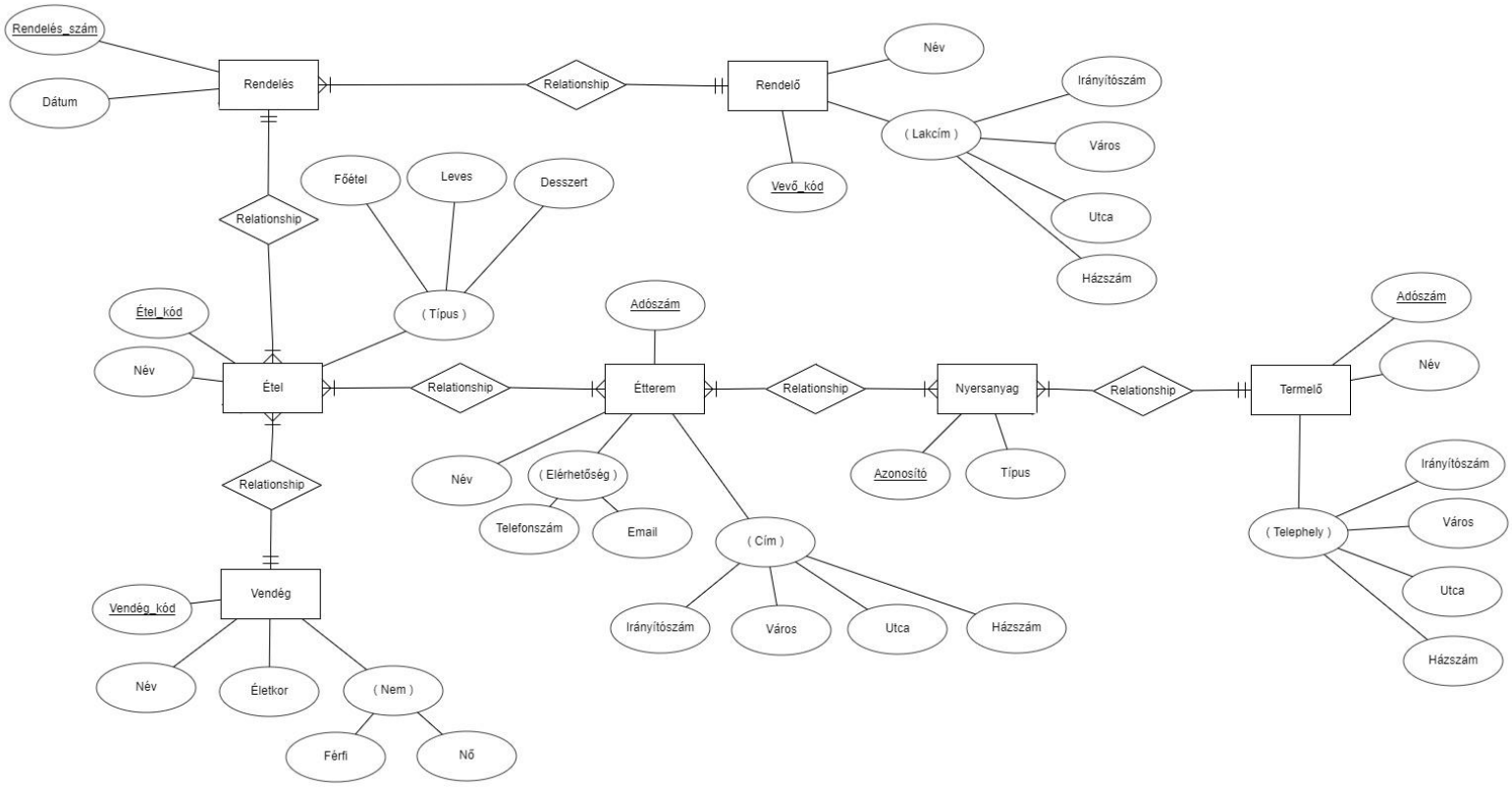
Melyben kiindulunk egy termelőtől aki termelhet sokféle nyersanyagot, ezt el kell juttatnia egy étterembe, ahol készítenek belőle ételeket, italokat.

Az ételt és italt meg lehet rendelni, amelyet egy rendelő azaz vevő ad le, akár személyesen akár online. Majd így végül elérünk a vevőhöz.

ER modell:

Egyedek: 7 db

Termelő, Nyersanyag, Étterem, Étel, Rendelés, Rendelő, Vendég



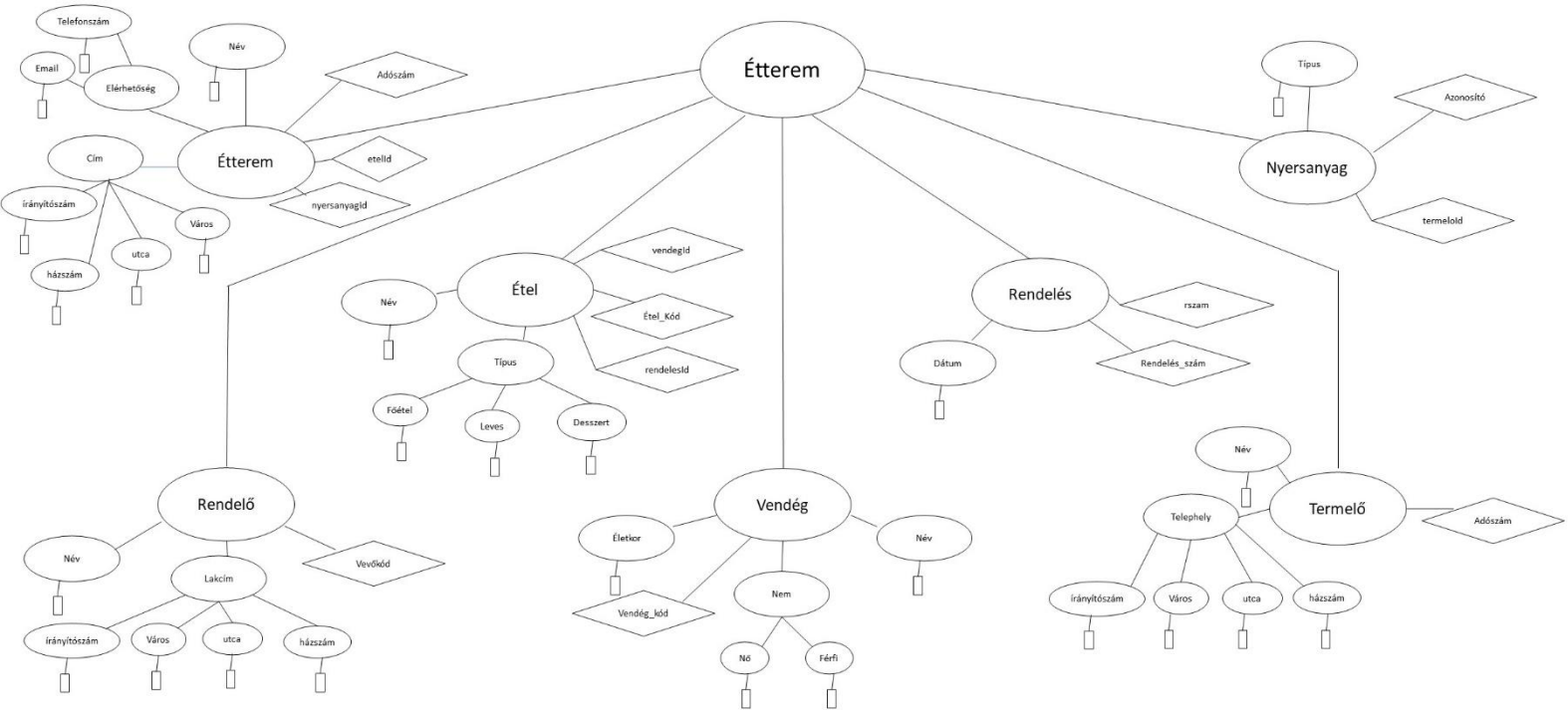
XDM modell:

Létrejönnek idegen kulcsok: -Étteremhez: etelId, nyersanyagId

-Nyersanyaghoz: termeloId

-Rendeléshez: rszam

-Ételhez: vendegId, rendelesId



XML Dokumentum:

```
<?xml version="1.0" encoding="utf-8" standalone="no"?><etterem_service
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="XMLSchemas0kk3c.xsd">

	<etterem adoszam="00001" etelId="101" nyersanyagId="001">
		<nev>Csülök Bisztró</nev>
		<elerhetoseg>
			<telefonszam>+30-123-4567</telefonszam>
			<email>csulokbusz@gmail.com</email>
		</elerhetoseg>
		<cim>
			<iranyitoszam>3000</iranyitoszam>
			<varos>Miskolc</varos>
			<utca>Futó utca</utca>
			<hazszam>25</hazszam>
		</cim>
	</etterem>
	<etterem adoszam="00002" etelId="102" nyersanyagId="002">
		<nev>Végállomás Bisztró</nev>
		<elerhetoseg>
			<telefonszam>+30-987-6543</telefonszam>
			<email>vegallomas@gmail.com</email>
		</elerhetoseg>
```

```
        <cim>
                <iranyitoszam>3501</iranyitoszam>
                <varos>Miskolc</varos>
                <utca>Széchenyi István utca</utca>
                <hazszam>107</hazszam>
        </cim>
</etterem>
<etterem adoszam="00003" etelId="103" nyersanyagId="003">
        <nev>Pető Hambizó</nev>
        <elerhetoseg>
                <telefonszam>30-456-7981</telefonszam>
                <email>petohambi@gmail.com</email>
        </elerhetoseg>
        <cim>
                <iranyitoszam>3525</iranyitoszam>
                <varos>Budapest</varos>
                <utca>Zsedényi József utca</utca>
                <hazszam>1</hazszam>
        </cim>
</etterem>
<nyersanyag azonosito="001" termeloId="0001">
        <tipus>kacsahús</tipus>
</nyersanyag>
<nyersanyag azonosito="002" termeloId="0002">
        <tipus>borjúhús</tipus>
</nyersanyag>
<nyersanyag azonosito="003" termeloId="0001">
        <tipus>Krumpli</tipus>
</nyersanyag>
<nyersanyag azonosito="004" termeloId="0002">
        <tipus>Répa</tipus>
</nyersanyag>
<nyersanyag azonosito="005" termeloId="0001">
        <tipus>Liszt</tipus>
</nyersanyag>
<termelo tid="0001">
        <nev>Hús Nagyker</nev>
        <telephely>
                <iranyitoszam>3525</iranyitoszam>
                <varos>Budapest</varos>
                <utca>Zsedényi József utca</utca>
                <hazszam>95</hazszam>
        </telephely>
</termelo>
<termelo tid="0002">
        <nev>Zöldség, Gyümölcs Nagyker</nev>
        <telephely>
                <iranyitoszam>3520</iranyitoszam>
                <varos>Győr</varos>
                <utca>Berzsenyi Dániel utca</utca>
                <hazszam>8</hazszam>
        </telephely>
</termelo>
<etel etelkod="101" rendelesId="000000001" vendegId="01">
        <nev>Sushi</nev>
</etel>
<etel etelkod="102" rendelesId="000000002" vendegId="02">
        <nev>Csokis palacsinta</nev>
</etel>
```

```
	<etel etelkod="103" rendelesId="000000001" vendegId="03">
		<nev>Cigány pecsenye</nev>
	</etel>
	<etel etelkod="104" rendelesId="000000003" vendegId="02">
		<nev>Paprikás krumpli</nev>
	</etel>
	<etel etelkod="105" rendelesId="000000002" vendegId="01">
		<nev>Pho leves</nev>
	</etel>
	<vendeg vendegkod="01">
		<nev>Pető Ádám</nev>
		<eletkor>20</eletkor>
		<nem>Férfi</nem>
	</vendeg>
	<vendeg vendegkod="02">
		<nev>Iványi Fruzsina</nev>
		<eletkor>21</eletkor>
		<nem>Nő</nem>
	</vendeg>
	<vendeg vendegkod="03">
		<nev>Szalai Márton</nev>
		<eletkor>21</eletkor>
		<nem>Férfi</nem>
	</vendeg>
	<rendeles rendeloId="001" rszam="000000001">
		<datum>2021.11.15</datum>
	</rendeles>
	<rendeles rendeloId="002" rszam="000000002">
		<datum>2021.11.16</datum>
	</rendeles>
	<rendeles rendeloId="002" rszam="000000003">
		<datum>2021.11.17</datum>
	</rendeles>
	<rendelo vevokod="001">
		<nev>Pető Ádám</nev>
		<lakcim>
			<iranyitoszam>3525</iranyitoszam>
			<varos>Szeged</varos>
			<utca>Berzsenyi Dániel utca</utca>
			<hazszam>9</hazszam>
		</lakcim>
	</rendelo>
	<rendelo vevokod="002">
		<nev>Sebák Petra</nev>
		<lakcim>
			<iranyitoszam>3522</iranyitoszam>
			<varos>Gesztely</varos>
			<utca>Aba utca</utca>
			<hazszam>15</hazszam>
		</lakcim>
	</rendelo>
</etterem_service>
```

XML Schema file:

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

    <xs:element name="etterem_service">
        <xs:complexType>
            <xs:sequence>
```

```
        <xs:choice maxOccurs="unbounded">
            <xs:element name="etterem" type="etterem_type"/>
            <xs:element name="nyersanyag" type="nyersanyag_type"/>
            <xs:element name="termelo" type="termelo_type"/>
            <xs:element name="etel" type="etel_type"/>
            <xs:element name="vendeg" type="vendeg_type"/>
            <xs:element name="rendeles" type="rendeles_type"/>
            <xs:element name="rendelo" type="rendelo_type"/>
        </xs:choice>
    </xs:sequence>
</xs:complexType>

<xs:key name="etteremid_PK">
    <xs:selector xpath="etterem"/>
    <xs:field xpath="@adoszam"/>
</xs:key>
<xs:key name="nyeranyagId_PK">
    <xs:selector xpath="nyersanyag"/>
    <xs:field xpath="@azonosito"/>
</xs:key>
<xs:key name="termeloId_PK">
    <xs:selector xpath="termelo"/>
    <xs:field xpath="@tid"/>
</xs:key>
<xs:key name="etelId_PK">
    <xs:selector xpath="etel"/>
    <xs:field xpath="@etelkod"/>
</xs:key>
<xs:key name="vendegId_PK">
    <xs:selector xpath="vendeg"/>
    <xs:field xpath="@vendegkod"/>
</xs:key>
<xs:key name="rendelesId_PK">
    <xs:selector xpath="rendeles"/>
    <xs:field xpath="@rszam"/>
</xs:key>
<xs:key name="rendeloId_PK">
    <xs:selector xpath="rendelo"/>
    <xs:field xpath="@vevokod"/>
</xs:key>
<xs:keyref name="etelId_FK" refer="etelId_PK">
    <xs:selector xpath="etterem"/>
    <xs:field xpath="@etelId"/>
</xs:keyref>
<xs:keyref name="nyeranyagId_FK" refer="nyeranyagId_PK">
    <xs:selector xpath="etterem"/>
    <xs:field xpath="@nyersanyagId"/>
</xs:keyref>
<xs:keyref name="termeloId_FK" refer="termeloId_PK">
    <xs:selector xpath="nyersanyag"/>
    <xs:field xpath="@termeloId"/>
</xs:keyref>
<xs:keyref name="vendegId_FK" refer="vendegId_PK">
    <xs:selector xpath="etel"/>
    <xs:field xpath="@vendegId"/>
</xs:keyref>
<xs:keyref name="rendelesId_FK" refer="rendelesId_PK">
    <xs:selector xpath="etel"/>
    <xs:field xpath="@rendelesId"/>
```

```
    </xs:keyref>
    <xs:keyref name="rendeloId_FK" refer="rendeloId_PK">
        <xs:selector xpath="rendeles"/>
        <xs:field xpath="@rendeloId"/>
    </xs:keyref>
</xs:element>

 <xs:complexType name="etterem_type">
    <xs:sequence>
        <xs:element name="nev" type="xs:string"/>
        <xs:element name="elerhetoseg" type="elerhetoseg_type"/>
        <xs:element name="cim" type="cim_type"/>
    </xs:sequence>
    <xs:attribute name="adoszam" type="xs:unsignedByte" use="required"/>
    <xs:attribute name="etelId" type="xs:unsignedByte" use="required"/>
    <xs:attribute name="nyersanyagId" type="xs:unsignedByte" use="required"/>
</xs:complexType>
<xs:complexType name="nyersanyag_type">
    <xs:sequence>
        <xs:element name="tipus" type="xs:string"/>
    </xs:sequence>
    <xs:attribute name="azonosito" type="xs:unsignedByte" use="required"/>
    <xs:attribute name="termeloId" type="xs:unsignedByte" use="required"/>
</xs:complexType>
<xs:complexType name="termelo_type">
    <xs:sequence>
        <xs:element name="nev" type="xs:string"/>
        <xs:element name="telephely" type="cim_type"/>
    </xs:sequence>
    <xs:attribute name="tid" type="xs:unsignedByte" use="required"/>
</xs:complexType>
<xs:complexType name="etel_type">
    <xs:sequence>
        <xs:element name="nev" type="xs:string"/>
    </xs:sequence>
    <xs:attribute name="etelkod" type="xs:unsignedByte" use="required"/>
    <xs:attribute name="vendegId" type="xs:unsignedByte" use="required"/>
    <xs:attribute name="rendelesId" type="xs:unsignedByte" use="required"/>
</xs:complexType>
<xs:complexType name="vendeg_type">
    <xs:sequence>
        <xs:element name="nev" type="xs:string"/>
        <xs:element name="eletkor" type="xs:integer"/>
        <xs:element name="nem" type="xs:string"/>
    </xs:sequence>
    <xs:attribute name="vendegkod" type="xs:unsignedByte" use="required"/>
</xs:complexType>
<xs:complexType name="rendeles_type">
    <xs:sequence>
        <xs:element name="datum" type="datum_type"/>
    </xs:sequence>
    <xs:attribute name="rszam" type="xs:unsignedByte" use="required"/>
    <xs:attribute name="rendeloId" type="xs:unsignedByte" use="required"/>
</xs:complexType>
<xs:complexType name="rendelo_type">
    <xs:sequence>
        <xs:element name="nev" type="xs:string"/>
        <xs:element name="lakcim" type="cim_type"/>
    </xs:sequence>
```

```
        <xs:attribute name="vevokod" type="xs:unsignedByte" use="required"/>
    </xs:complexType>
    <xs:complexType name="elerhetoseg_type">
        <xs:sequence>
            <xs:element name="telefonszam" type="xs:string"/>
            <xs:element name="email" type="xs:string"/>
        </xs:sequence>
    </xs:complexType>
    <xs:complexType name="cim_type">
        <xs:sequence>
            <xs:element name="iranyitoszam" type="xs:string"/>
            <xs:element name="varos" type="xs:string"/>
            <xs:element name="utca" type="xs:string"/>
            <xs:element name="hazszam" type="xs:integer"/>
        </xs:sequence>
    </xs:complexType>

      <xs:simpleType name="datum_type">
        <xs:restriction base="xs:string">
            <xs:pattern value="\d{4}\.(0[1-9]|1[012])\.(0[1-9]|[12][0-9]|3[01])"/>
<!-- YYYY.MM.DD -->
        </xs:restriction>
    </xs:simpleType>
      <xs:simpleType name="telefonszam_type">
        <xs:restriction base="xs:string">
            <xs:pattern value="\d{2}-\d{3}-\d{4}"/>
            <xs:pattern value="\d{2}-\d{3}-\d{3}"/>
        </xs:restriction>
    </xs:simpleType>

</xs:schema>
```

DOM Read class:

```
package hu.domparse.s0kk3c;

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

import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.DocumentBuilderFactory;
import javax.xml.parsers.ParserConfigurationException;

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

public class DomReads0kk3c {

      public static void main(String[] args) {
            try {
                  File xmlFile = new File("XMLs0kk3c.xml"); // xml fájl, amelyből olvasunk
                  DocumentBuilderFactory factory = DocumentBuilderFactory.newInstance(); // XML dokumentumból DOM objektum lehetővé tétele
                  DocumentBuilder dBuilder = factory.newDocumentBuilder(); // XML fájl, Document lekéréséhez
```

```
                Document doc = dBuilder.parse(xmlFile); // dokument lekérése
                System.out.println(doc);
                doc.getDocumentElement().normalize();
                System.out.println("Étterem adatok lekérése");
                System.out.println(xmlFile);
                System.out.println(doc);
                Read(doc); // fő metódus
            } catch (ParserConfigurationException pce) {
                pce.printStackTrace();
            } catch (IOException ioe) {
                ioe.printStackTrace();
            } catch (SAXException sae) {
                sae.printStackTrace();
            }
        }

        public static void Read(Document doc) {
            NodeList nList = doc.getElementsByTagName("etterem"); // Fellepes
taggal rendelkezõ elemek lekérése

                            // listába
            for (int i = 0; i < nList.getLength(); i++) { // listán végigmegyünk
                Node nNode = nList.item(i); // lekérjük a lista aktuális
elemét, majd elementé konvertáljuk
                Element element = (Element) nNode;
                // Lekérjük az attribútumokat, majd azok segítségével meghívjuk
a definiált metódusokat
                if (nNode.getNodeType() == Node.ELEMENT_NODE) {
                    String nev =
element.getElementsByTagName("nev").item(0).getTextContent(); // darabszám
lekérdezése
                    String elerhetoseg =
element.getElementsByTagName("elerhetoseg").item(0).getTextContent();
                    String cim =
element.getElementsByTagName("cim").item(0).getTextContent();
                    String etelId = element.getAttribute("etelId");
                    String nyersanyagId =
element.getAttribute("nyersanyagId");
                    System.out.println("\n----------------------------------
-" + (i + 1) + ". Étterem-----------------------------------");
                    System.out.println("\n\tAdószám:\t" + nev);
                    System.out.println("\n\tElérhetőség:\t" + elerhetoseg);
                    System.out.println("\n\tCím:\t" + cim);
                    ReadEtelById(doc, etelId);
                    ReadNyersanyagById(doc, nyersanyagId);

                }
            }
        }

        public static void ReadEtelById(Document doc, String id) {
            NodeList nList = doc.getElementsByTagName("etel");
            for (int i = 0; i < nList.getLength(); i++) {
                Node nNode = nList.item(i);
                Element element = (Element) nNode;
                if (nNode.getNodeType() == Node.ELEMENT_NODE) {
                    if (element.getAttribute("etelkod").equals(id)) {
                        String nev =
element.getElementsByTagName("nev").item(0).getTextContent();
```

```
                                System.out.println("Etel adatok: \n\tNev:\t" +
nev);
                                String vendegId =
element.getAttribute("vendegId");
                                String rendelesId =
element.getAttribute("rendelesId");
                                ReadVendegById(doc, vendegId);
                                ReadRendelesById(doc, rendelesId);
                        }
                }
        }
    }
    public static void ReadVendegById(Document doc, String id) {
        NodeList nList = doc.getElementsByTagName("vendeg");
        for (int i = 0; i < nList.getLength(); i++) {
            Node nNode = nList.item(i);
            Element element = (Element) nNode;
            if (nNode.getNodeType() == Node.ELEMENT_NODE) {
                if (element.getAttribute("vendegkod").equals(id)) {
                    String nev =
element.getElementsByTagName("nev").item(0).getTextContent();
                    String eletkor =
element.getElementsByTagName("eletkor").item(0).getTextContent();
                    String nem =
element.getElementsByTagName("nem").item(0).getTextContent();
                    System.out.println("Vendeg adatok: \n\tNev:\t" +
nev + "\n\tEletkor:\t" + eletkor + "\n\tNem:\t" + nem);
                }
            }
        }
    }
    public static void ReadRendelesById(Document doc, String id) {
        NodeList nList = doc.getElementsByTagName("rendeles");
        for (int i = 0; i < nList.getLength(); i++) {
            Node nNode = nList.item(i);
            Element element = (Element) nNode;
            if (nNode.getNodeType() == Node.ELEMENT_NODE) {
                if (element.getAttribute("rszam").equals(id)) {
                    String datum =
element.getElementsByTagName("datum").item(0).getTextContent();
                    System.out.println("Rendeles adatok: \n\tDatum:\t"
+ datum);
                    String rendeloId =
element.getAttribute("rendeloId");
                    ReadRendeloById(doc, rendeloId);
                }
            }
        }
    }
    public static void ReadRendeloById(Document doc, String id) {
        NodeList nList = doc.getElementsByTagName("rendelo");
        for (int i = 0; i < nList.getLength(); i++) {
            Node nNode = nList.item(i);
            Element element = (Element) nNode;
            if (nNode.getNodeType() == Node.ELEMENT_NODE) {
                if (element.getAttribute("vevokod").equals(id)) {
                    String nev =
element.getElementsByTagName("nev").item(0).getTextContent();
```

```
                                String lakcim =
element.getElementsByTagName("lakcim").item(0).getTextContent();
                                System.out.println("Rendelo adatok: \n\tNev:\t" +
nev + "\n\tLakcim:\t" + lakcim);
                        }
                    }
                }
        }
        public static void ReadNyersanyagById(Document doc, String id) {
                NodeList nList = doc.getElementsByTagName("nyersanyag");
                for (int i = 0; i < nList.getLength(); i++) {
                        Node nNode = nList.item(i);
                        Element element = (Element) nNode;
                        if (nNode.getNodeType() == Node.ELEMENT_NODE) {
                                if (element.getAttribute("azonosito").equals(id)) {
                                String tipus =
element.getElementsByTagName("tipus").item(0).getTextContent();
                                System.out.println("Nyersanyag adatok:
\n\tTipus:\t" + tipus);
                                String termeloId =
element.getAttribute("termeloId");
                                ReadTermeloById(doc, termeloId);
                        }
                    }
                }
        }
        public static void ReadTermeloById(Document doc, String id) {
                NodeList nList = doc.getElementsByTagName("termelo");
                for (int i = 0; i < nList.getLength(); i++) {
                        Node nNode = nList.item(i);
                        Element element = (Element) nNode;
                        if (nNode.getNodeType() == Node.ELEMENT_NODE) {
                                if (element.getAttribute("tid").equals(id)) {
                                String nev =
element.getElementsByTagName("nev").item(0).getTextContent();
                                String telephely =
element.getElementsByTagName("telephely").item(0).getTextContent();
                                System.out.println("Termelo adatok: \n\tNev:\t" +
nev + "\n\tTelephely:\t" + telephely);
                        }
                    }
                }
        }

}
```

DOM Modify class:

```
package hu.domparse.s0kk3c;

import org.w3c.dom.Document;
import org.w3c.dom.Element;
import org.w3c.dom.Node;
import org.w3c.dom.NodeList;
import org.xml.sax.SAXException;
import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.DocumentBuilderFactory;
import javax.xml.parsers.ParserConfigurationException;
import javax.xml.transform.Transformer;
import javax.xml.transform.TransformerException;
```

```java
import javax.xml.transform.TransformerFactory;
import javax.xml.transform.dom.DOMSource;
import javax.xml.transform.stream.StreamResult;
import java.io.File;
import java.io.IOException;
import java.util.Scanner;

public class DomModifys0kk3c {

	public static void main String[] args)
			throws ParserConfigurationException, IOException, SAXException,
TransformerException {

		File xmlFile = new File("XMLs0kk3c.xml"); // xml fájl bekérése
		DocumentBuilderFactory factory =
DocumentBuilderFactory.newInstance(); // olvasás lehetõvé tétele
		DocumentBuilder dBuilder = factory.newDocumentBuilder();
		Document doc = dBuilder.parse(xmlFile);
		doc.getDocumentElement().normalize();
		System.out.println("XML Módosítása");
		System.out.println("Adja meg mit szeretne módosítani: ");
		System.out.println("1 Nyersanyag módosítása\n2 Etel módosítása\n3
Termelo módosítása\n4 Vendeg módosítása");
		Modify(doc);

	}
	public static void Modify(Document doc) throws TransformerException {
		int nyersanyagSzama =
doc.getElementsByTagName("nyersanyag").getLength(); // nyersanyag számának
lekérdezése
		int etelSzama = doc.getElementsByTagName("etel").getLength(); // etel
számának lekérdezése
		int termeloSzama = doc.getElementsByTagName("termelo").getLength();
// termelo számának lekérdezése
		int vendegSzama = doc.getElementsByTagName("vendeg").getLength(); //
vendeg számának lekérdezése


		Scanner scan = new Scanner(System.in);
		System.out.println("Adja meg a sorszamot: ");
		int readCategory = scan.nextInt();
		switch (readCategory) {
		case 1:
			ModifyNyersanyag(doc, nyersanyagSzama);
			break;
		case 2:
			ModifyEtel(doc, etelSzama);
			break;
		case 3:
			ModifyTermelo(doc, termeloSzama);
			break;
		case 4:
			ModifyVendeg(doc, vendegSzama);
			break;
		}
		scan.close();
	}
```

```
	private static void ModifyNyersanyag(Document doc, int nyersanyagSzama)
throws TransformerException {
		// Kiiratjuk a jelenlegi nyersanyagokat, majd lekérdezzük melyiket
kívánja módosítani
		System.out.println("Melyik nyersanyag adatait szeretné módosítani?");
		for (int i = 1; i < nyersanyagSzama + 1; i++) {
			System.out.println(i + ". nyersanyag");
			DomReads0kk3c.ReadNyersanyagById(doc, String.valueOf(i));
			System.out.println("----------------------------------------
");
		}
		String id = ReadId();
		System.out.println(id);
		// Bekérjük az új adatokat
		Scanner scanner = new Scanner(System.in);
		System.out.print("Tipus: ");
		String tipus = scanner.nextLine();
		scanner.close();
		// lekérdezzük az Elementeket, majd setTextContent-el módosítjuk.
		NodeList nodeList = doc.getElementsByTagName("nyersanyag");
		for (int i = 0; i < nodeList.getLength(); i++) {
			Node nNode = nodeList.item(i);
			if (nNode.getNodeType() == Node.ELEMENT_NODE) {
				Element element = (Element) nNode;
				String sid = element.getAttribute("azonosito");
				if (sid.equals(id)) {
					Node node1 =
element.getElementsByTagName("tipus").item(0);
					node1.setTextContent(tipus);
					System.out.println("Sikeres módosítás");
				}
			}
		}
		ModifyXML(doc); // Létrehozzuk az XML-t
	}
	/*private static void ModifyCsoport(Document doc, int csoportszam) throws
TransformerException {
		System.out.println("Melyik csoportot kívánja módosítani?");
		for (int i = 1; i < csoportszam + 1; i++) {
			System.out.println(i + ". csoport");
			DomReadFB8YPQ.ReadCsoportById(doc, String.valueOf(i));
			System.out.println("----------------------------------------
");
		}
		String id = ReadId();
		// Bekérjük az új adatokat
		Scanner sc = new Scanner(System.in);
		System.out.print("Korosztály: ");
		String korosztaly = sc.nextLine();
		System.out.print("Létszám: ");
		String letszam = sc.nextLine();
		System.out.print("Csoportnév: ");
		String csoportnev = sc.nextLine();
		sc.close();
		// lekérdezzük az Elementeket, majd setTextContent-el módosítjuk.
		NodeList nodeList = doc.getElementsByTagName("Csoport");
		for (int i = 0; i < nodeList.getLength(); i++) {
			Node nNode = nodeList.item(i);
			if (nNode.getNodeType() == Node.ELEMENT_NODE) {
```

```
                        Element element = (Element) nNode;
                        String sid = element.getAttribute("id");
                        if (sid.equals(id)) {
                              Node node1 =
element.getElementsByTagName("Korosztaly").item(0);
                              node1.setTextContent(korosztaly);
                              Node node2 =
element.getElementsByTagName("Letszam").item(0);
                              node2.setTextContent(letszam);
                              Node node3 =
element.getElementsByTagName("Csoportnev").item(0);
                              node3.setTextContent(csoportnev);
                              System.out.println("Sikeres módosítás");
                        }
                  }
            }
            ModifyXML(doc); // Létrehozzuk az XML-t
      }*/
      private static void ModifyEtel(Document doc, int etelSzama) throws
TransformerException {
            // Kiiratjuk a jelenlegi eteleket, majd lekérdezzük melyiket kívánja
módosítani
            System.out.println("Melyik etel adatait szeretné módosítani?");
            for (int i = 1; i < etelSzama + 1; i++) {
                  System.out.println(i + ". etel");
                  DomReads0kk3c.ReadEtelById(doc, String.valueOf(i));
                  System.out.println("----------------------------------------
");
            }
            String id = ReadId();
            // Bekérjük az új adatokat
            Scanner sc = new Scanner(System.in);
            System.out.print("Nev: ");
            String nev = sc.nextLine();
            System.out.print("Tipus: ");
            String tipus = sc.nextLine();
            sc.close();
            // lekérdezzük az Elementeket, majd setTextContent-el módosítjuk.
            NodeList nodeList = doc.getElementsByTagName("etel");
            for (int i = 0; i < nodeList.getLength(); i++) {
                  Node nNode = nodeList.item(i);
                  if (nNode.getNodeType() == Node.ELEMENT_NODE) {
                        Element element = (Element) nNode;
                        String sid = element.getAttribute("id");
                        if (sid.equals(id)) {
                              Node node1 =
element.getElementsByTagName("nev").item(0);
                              node1.setTextContent(nev);
                              Node node2 =
element.getElementsByTagName("tipus").item(0);
                              node2.setTextContent(tipus);
                              System.out.println("Sikeres módosítás");
                        }
                  }
            }
            ModifyXML(doc); // Létrehozzuk az XML-t
      }
      private static void ModifyTermelo(Document doc, int termeloSzama) throws
TransformerException {
```

```
            // Kiiratjuk a jelenlegi termeloket, majd lekérdezzük melyiket
kívánja módosítani
            System.out.println("Melyik termelo adatait szeretné módosítani?");
            for (int i = 1; i < termeloSzama + 1; i++) {
                  System.out.println(i + ". termelo");
                  DomReads0kk3c.ReadTermeloById(doc, String.valueOf(i));
                  System.out.println("----------------------------------------
");
            }
            String id = ReadId();
            // Bekérjük az új adatokat
            Scanner sc = new Scanner(System.in);
            System.out.print("Nev: ");
            String nev = sc.nextLine();
            System.out.print("Telephely: ");
            String telephely = sc.nextLine();
            sc.close();
            // lekérdezzük az Elementeket, majd setTextContent-el módosítjuk.
            NodeList nodeList = doc.getElementsByTagName("termelo");
            for (int i = 0; i < nodeList.getLength(); i++) {
                  Node nNode = nodeList.item(i);
                  if (nNode.getNodeType() == Node.ELEMENT_NODE) {
                        Element element = (Element) nNode;
                        String sid = element.getAttribute("id");
                        if (sid.equals(id)) {
                              Node node1 =
element.getElementsByTagName("nev").item(0);
                              node1.setTextContent(nev);
                              Node node2 =
element.getElementsByTagName("telephely").item(0);
                              node2.setTextContent(telephely);
                              System.out.println("Sikeres módosítás");
                        }
                  }
            }
            ModifyXML(doc); // Létrehozzuk az XML-t
      }
      private static void ModifyVendeg(Document doc, int vendegSzama) throws
TransformerException {
            // Kiiratjuk a jelenlegi vendegeket, majd lekérdezzük melyiket
kívánja módosítani
            System.out.println("Melyik vendeg adatait szeretné módosítani?");
            for (int i = 1; i < vendegSzama + 1; i++) {
                  System.out.println(i + ". vendeg");
                  DomReads0kk3c.ReadVendegById(doc, String.valueOf(i));
                  System.out.println("----------------------------------------
");
            }
            String id = ReadId();
            // Bekérjük az új adatokat
            Scanner sc = new Scanner(System.in);
            System.out.print("Nev: ");
            String nev = sc.nextLine();
            System.out.print("Eletkor: ");
            String eletkor = sc.nextLine();
            System.out.print("Nem: ");
            String nem = sc.nextLine();
            sc.close();
            // lekérdezzük az Elementeket, majd setTextContent-el módosítjuk.
```

```
            NodeList nodeList = doc.getElementsByTagName("vendeg");
            for (int i = 0; i < nodeList.getLength(); i++) {
                  Node nNode = nodeList.item(i);
                  if (nNode.getNodeType() == Node.ELEMENT_NODE) {
                        Element element = (Element) nNode;
                        String sid = element.getAttribute("id");
                        if (sid.equals(id)) {
                              Node node1 =
element.getElementsByTagName("nev").item(0);
                              node1.setTextContent(nev);
                              Node node2 =
element.getElementsByTagName("eletkor").item(0);
                              node2.setTextContent(eletkor);
                              Node node3 =
element.getElementsByTagName("nem").item(0);
                              node3.setTextContent(nem);
                              System.out.println("Sikeres módosítás");
                        }
                  }
            }
            ModifyXML(doc); // Létrehozzuk az XML-t
      }

      public static String ReadId() {
            Scanner sc = new Scanner(System.in);
            System.out.print("\nid:");
            String id = sc.nextLine();
            //sc.close();
            return id;
      }
      public static void ModifyXML(Document doc) throws TransformerException {
            TransformerFactory transformerFactory =
TransformerFactory.newInstance();
            Transformer transformer = transformerFactory.newTransformer();
            DOMSource source = new DOMSource(doc);
            StreamResult result = new StreamResult(new File("XMLs0kk3c.xml"));
            transformer.transform(source, result);
      }

}
```

DOM Query class:

```
package hu.domparse.s0kk3c;

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

import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.DocumentBuilderFactory;
import javax.xml.parsers.ParserConfigurationException;
import javax.xml.transform.TransformerException;

import org.w3c.dom.Document;
import org.w3c.dom.Element;
import org.w3c.dom.Node;
import org.w3c.dom.NodeList;
import org.xml.sax.SAXException;
```

```java
public class DomQuerys0kk3c {

	public static void main String[] args)
			throws ParserConfigurationException, IOException, SAXException,
TransformerException {

		File xmlFile = new File("XMLs0kk3c.xml"); // xml fájl bekérése
		DocumentBuilderFactory factory =
DocumentBuilderFactory.newInstance(); // olvasás lehetővé tétele
		DocumentBuilder dBuilder = factory.newDocumentBuilder();
		Document doc = dBuilder.parse(xmlFile);
		doc.getDocumentElement().normalize();

		System.out.println("Root element: " +
doc.getDocumentElement().getNodeName());
		System.out.println("-----------------------------");
		LoadVendegQuery(doc);

	}

	public static void LoadVendegQuery(Document doc) throws TransformerException
{
		NodeList nodeList = doc.getElementsByTagName("vendeg"); // vendeg
elemek listázása
		String vendeg;
		Element element = null;
		Node nNode = null;
		for (int i = 0; i < nodeList.getLength(); i++) {
			nNode = nodeList.item(i);
			element = (Element) nNode;
			String nev =
element.getElementsByTagName("nev").item(0).getTextContent();
			System.out.println((i + 1) + ") " + nev);

		}
		// Tánciskola választása
		System.out.println("Írja be annak a vendég nevét, amelyiknek az
ételeit szeretné látni: ");
		Scanner sc = new Scanner(System.in);
		vendeg = sc.nextLine();
		for (int i = 0; i < nodeList.getLength(); i++) {
			nNode = nodeList.item(i);
			element = (Element) nNode;
			if (nNode.getNodeType() == Node.ELEMENT_NODE) {

				if (vendeg.equals("Pető Ádám")) {
					LoadEtelQuery(doc, "01");
					break;
				}
				if (vendeg.equals("Ivanyi Fruzsina")) {
					LoadEtelQuery(doc, "02");
					break;
				}
			}
		}
		sc.close();

	}
```

```
	public static void LoadEtelQuery(Document doc, String id) throws
TransformerException {
		NodeList nodeList = doc.getElementsByTagName("etel");
		int etel = 0;

		for (int i = 0; i < nodeList.getLength(); i++) {
			Node nNode = nodeList.item(i);
			Element element = (Element) nNode;
			String vId = element.getAttribute("vendegId");
			if (nNode.getNodeType() == Node.ELEMENT_NODE) {
				if (id.equals(vId)) {
					etel += 1;
					System.out.println(etel + ". etel adatai:");
					String etelkod = element.getAttribute("etelkod");
					DomReads0kk3c.ReadEtelById(doc, etelkod);

				}
			}
		}

	}
}
```