JEGYZŐKÖNYV

Adatkezelés XML környezetben

Féléves feladat

Pizzázó

Készítette: **Drig Dávid**
Neptunkód: **EZ3YRC**
Dátum: 2023.12.02

Tartalomjegyzék:

1a) A feladat témája

A beadandó témája egy olyan adatbázis, amely több pizzázót kezel. Rákereshetünk benne a pizzázóban dolgozó futárokra, vagy beszállítókra, a vevő adatait is lekérdezhetjük.

<u>Az ER modell egyedei és tulajdonságai</u>:

- **A Bankkártya egyed tulajdonságai**
  - Kártyaszám: A Bankkártya egyed elsődleges kulcsa.
  - Bank: A bank neve, amelyhez a bankkártya tartozik.
  - Lejárati dátum: A kártya lejárati dátuma.
  - Típus: A bankkártya típusa.

- **A Vevő egyed tulajdonságai**
  - VevőID: A Vevő egyed elsődleges kulcsa.
  - Név: A vevő neve.
  - Telefonszám: A vevő telefonszáma.
  - Cím: Összetett tulajdonság. A vevő címe.

- **A Pizza egyed tulajdonságai**
  - PizzaID: A Pizza egyed elsődleges kulcsa.
  - Teljes ár: A rendelt pizza/pizzák teljes ára. Származtatott tulajdonság.
  - Pizza neve: A pizza neve.
  - Méret: Többértékű tulajdonság. A pizza méretét tárolja.
  - Feltét: Többértékű tulajdonság. A pizzán lévő feltéteket tárolja.

- **A Futár egyed tulajdonságai**
  - FutárID: A Futár egyed elsődleges kulcsa.
  - Telefonszám: A futár telefonszáma.
  - Név: A futár neve.

- **A Beszállító egyed tulajdonságai**
  - BeszállítóID: A Beszállító egyed elsődleges kulcsa.
  - Elérhetőség: A beszállító elérhetősége.
  - Név: A beszállító cég neve.
  - Cím: Összetett tulajdonság. A beszállító cég címe.

- **A Pizzázó egyed tulajdonságai**
  - PizzázóID: A Pizzázó egyed elsődleges kulcsa.
  - Név: A pizzázó neve.
  - Elérhetőség: Összetett tulajdonság. A pizzázó elérhetőségei.

Egyedek közötti kapcsolat:

- **Pizzázó és Futár:**

  A Pizzázó és a Futár egyedek között egy a többhöz kapcsolat van, mivel egy pizzázó alkalmazhat több futárt, de egy futár csak egy pizzázónál dolgozik.

- **Pizzázó és Beszállító:**

  A Pizzázó és a Beszállító egyedek között több a többhöz kapcsolat van, mivel egy pizzázó rendelhet több beszállítótól, valamint egy beszállító beszállíthat több pizzázónak is. A kapcsolat paraméterei: a Hozzávalók, amely a beszállító által beszállított hozzávalókat jelenti, valamint a Dátum, azaz a beszállítás dátuma.

- **Pizzázó és Pizza:**

  A Pizzázó és a Pizza egyedek között egy a többhöz kapcsolat van, mivel egy pizzázónak lehet több pizzája, de egy pizza csak egy pizzázóhoz tartozhat.

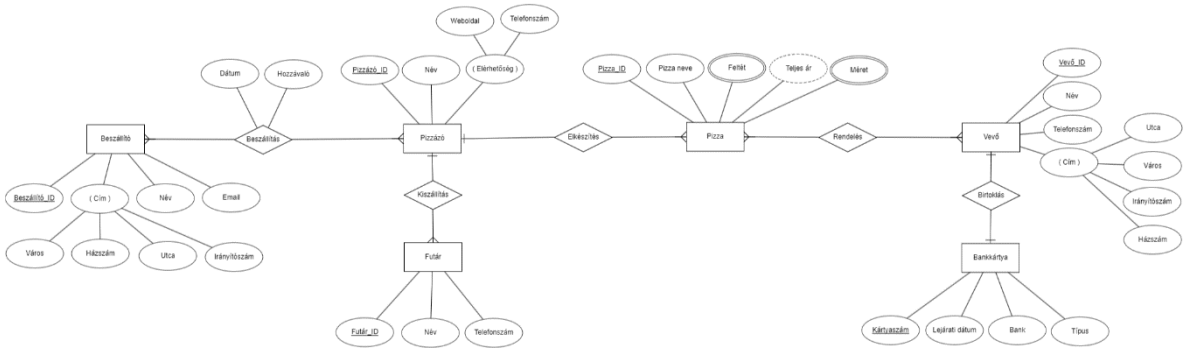
- **Pizza és Vevő:**

  A Pizza és a Vevő egyedek között több a többhöz kapcsolat van, mivel egy vevő rendelhet többfajta pizzát, és a pizzából rendelhet több különböző vevő is.

- **Vevő és Bankkártya:**

  A Vevő és a Bankkártya egyedek között egy-egy kapcsolat van, mivel egy vevőnek csak egy bankkártyája lehet, és egy bankkártyának nem lehet több tulajdonosa.

A feladat ER modellje:



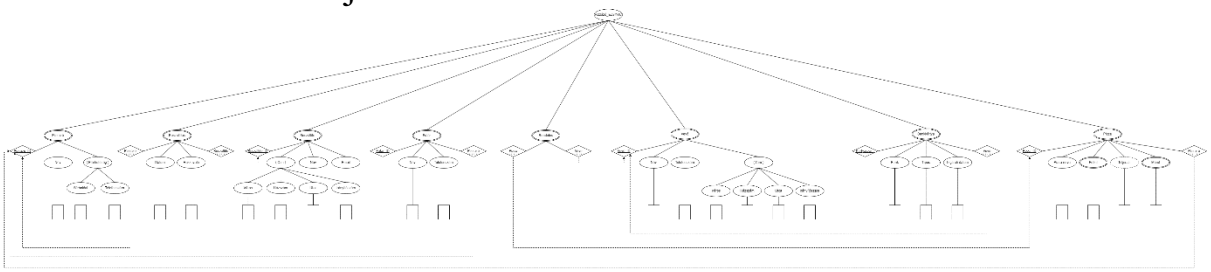
Az egyedek közötti kapcsolat

- Beszállítás kapcsolat: Több pizzázóhoz több beszállítás tartozik N:N
- Kiszállítás kapcsolat: Egy pizzázóhoz több futár tartozik: 1:N
- Elkészítés kapcsolat: Egy pizzázóhoz több pizza tartozik: 1:N
- Rendelés kapcsolat: Több pizzához több vevő tartozik: N:N
- Birtoklás kapcsolat: Egy vevőhöz egy bankkártya tartozik: 1:1

1b) Az ER-modell konvertálása XDM modellre

XDM modellnél háromféle jelölést alkalmazhatunk. Ezek az ellipszis, a rombusz, illetve a téglalap. Az ellipszis jelöli az elemeket minden egyedből elem lesz, ezen felül a tulajdonságokból is. A rombusz jelöli az attribútumokat, amelyek a kulcs tulajdonságokból keletkeznek. A téglalap jelöli a szöveget, amely majd az XML dokumentumban fog megjelenni. Azoknak az elemeknek, amelyek többször is előfordulhatnak, a jelölése dupla ellipszissel történik. Az idegenkulcsok és a kulcsok közötti kapcsolatot szaggatott vonalas nyíllal jelöljük.

A feladat XDM modellje:



1c) XML dokumentum készítése

Az XDM modell alapján az XML dokumentumot úgy készítettem el, hogy először is a root elementtel kezdtem, ami az Pizzazo_EZ3YRC volt.

A gyermek elemeiből 3-3 példányt hoztam létre, ezeknek az elemeknek az attribútumai közé tartoznak a kulcsok, illetve idegenkulcsok is, mindezek után ezeknek az elemeknek létrehoztam a többi gyermek elementet is.

XML dokumentum forráskódja

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Pizzazo_EZ3YRC xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:noNamespaceSchemaLocation="XMLSchemaEZ3YRC.xsd">


    <!--Pizzázók-->

    <Pizzazo pizzazo_id="1">
        <nev>Don Pepe</nev>
        <elerhetoseg>
            <weboldal>https://www.donpepe.hu/hu</weboldal>
            <telefonszam>309082091</telefonszam>
        </elerhetoseg>
    </Pizzazo>

    <Pizzazo pizzazo_id="2">
        <nev>Pizza Tábor</nev>
        <elerhetoseg>
            <weboldal>https://www.pizzatabor.hu/</weboldal>
            <telefonszam>204716655</telefonszam>
        </elerhetoseg>
    </Pizzazo>

    <Pizzazo pizzazo_id="3">
        <nev>Fortuna Pizzéria</nev>
        <elerhetoseg>
            <weboldal>https://pizzafortuna.hu/</weboldal>
```

```
            <telefonszam>302415505</telefonszam>
        </elerhetoseg>
    </Pizzazo>


    <!--Beszállítás-->

    <Beszallitas beszallito="1" pizzazo="1">
        <datum>2023-11-04</datum>
        <hozzavalo>hagyma</hozzavalo>
    </Beszallitas>

    <Beszallitas beszallito="2" pizzazo="2">
        <datum>2023-11-19</datum>
        <hozzavalo>cukor</hozzavalo>
    </Beszallitas>

    <Beszallitas beszallito="3" pizzazo="3">
        <datum>2023-11-25</datum>
        <hozzavalo>olaj</hozzavalo>
    </Beszallitas>



    <!--Beszállítók-->


    <Beszallito  beszallito_id="1">
        <nev>Kovács Lajos</nev>
        <email>mintalajos@gmail.com</email>
        <cim>
            <varos>Miskolc</varos>
            <hazszam>5</hazszam>
            <utca>Klapka György</utca>
            <iranyitoszam>3524</iranyitoszam>
        </cim>
    </Beszallito>

    <Beszallito  beszallito_id="2">
        <nev>Nehéz István</nev>
        <email>mintaistvan@gmail.com</email>
        <cim>
            <varos>Miskolc</varos>
            <hazszam>5</hazszam>
            <utca>Petőfi Sándor</utca>
            <iranyitoszam>3527</iranyitoszam>
        </cim>
    </Beszallito>
```

```xml
<Beszallito  beszallito_id="3">
    <nev>Kiss István</nev>
    <email>mintakiss@gmail.com</email>
    <cim>
        <varos>Miskolc</varos>
        <hazszam>4</hazszam>
        <utca>József Attila</utca>
        <iranyitoszam>3531</iranyitoszam>
    </cim>
</Beszallito>


<!--Futárok-->


<Futar  futar_id="1" pizzazo="1">
    <nev>Dudás Lajos</nev>
    <telefonszam>309562179</telefonszam>
</Futar>

<Futar  futar_id="2" pizzazo="2">
    <nev>Lajos Imre</nev>
    <telefonszam>309652179</telefonszam>
</Futar>

<Futar  futar_id="3" pizzazo="3">
    <nev>Nagy Márkó</nev>
    <telefonszam>209852179</telefonszam>
</Futar>


<!--Vevők-->


<Vevo vevo_id="1">
    <nev>Nehéz Gábor</nev>
    <telefonszam>306153384</telefonszam>
    <cim>
        <varos>Miskolc</varos>
        <hazszam>6</hazszam>
        <utca>Hajós Alfréd</utca>
        <iranyitoszam>3524</iranyitoszam>
    </cim>
</Vevo>


<Vevo vevo_id="2">
    <nev>Kiss Lajos</nev>
```

```
        <telefonszam>206453384</telefonszam>
        <cim>
            <varos>Miskolc</varos>
            <hazszam>4</hazszam>
            <utca>Vörösmarty Mihály</utca>
            <iranyitoszam>3532</iranyitoszam>
        </cim>
    </Vevo>

    <Vevo vevo_id="3">
        <nev>Elek János</nev>
        <telefonszam>206158484</telefonszam>
        <cim>
            <varos>Miskolc</varos>
            <hazszam>6</hazszam>
            <utca>Árok utca</utca>
            <iranyitoszam>3531</iranyitoszam>
        </cim>
    </Vevo>


    <!--Rendelés-->

    <Rendeles  pizza="1" vevo="1"></Rendeles>

    <Rendeles  pizza="2" vevo="2"></Rendeles>

    <Rendeles  pizza="3" vevo="3"></Rendeles>

    <!--Pizza-->

    <Pizza  pizza_id="1" pizzazo="1">
        <pizzaneve>Sonkás Pizza</pizzaneve>
        <feltet>paradicsom</feltet>
        <feltet>mozzarella</feltet>
        <feltet>sonka</feltet>
        <teljes_ar>2000</teljes_ar>
        <meret>17 cm</meret>
        <meret>25 cm</meret>
        <meret>30 cm</meret>
    </Pizza>

    <Pizza  pizza_id="2" pizzazo="2">
        <pizzaneve>Pizza Mexikói</pizzaneve>
        <feltet>paradicsom</feltet>
        <feltet>chili</feltet>
        <feltet>sajt</feltet>
        <teljes_ar>2550</teljes_ar>
```

```
        <meret>25 cm</meret>
        <meret>32 cm</meret>
        <meret>50 cm</meret>
    </Pizza>

    <Pizza  pizza_id="3" pizzazo="3">
        <pizzaneve>Aladdin Pizza</pizzaneve>
        <feltet>aszalt paradicsom</feltet>
        <feltet>ananász</feltet>
        <feltet>csirkemell</feltet>
        <teljes_ar>2750</teljes_ar>
        <meret>17 cm</meret>
        <meret>25 cm</meret>
        <meret>30 cm</meret>
    </Pizza>

    <!--Bankkártyák-->

    <Bankkartya  kartyaszam="1" vevo="1">
        <bank>OTP</bank>
        <tipus>bankkartya</tipus>
        <lejaratidatum>2024-05</lejaratidatum>
    </Bankkartya>

    <Bankkartya  kartyaszam="2" vevo="2">
        <bank>ERSTE</bank>
        <tipus>hitelkartya</tipus>
        <lejaratidatum>2025-06</lejaratidatum>
    </Bankkartya>

    <Bankkartya  kartyaszam="3" vevo="3">
        <bank>UNICREDIT</bank>
        <tipus>bankkartya</tipus>
        <lejaratidatum>2026-05</lejaratidatum>
    </Bankkartya>



</Pizzazo_EZ3YRC>
```

1d) XMLSchema készítése

Az XML Schemám meghatározza az adatokat, mint például a pizzázó nevét, hozzávalókat, pizza nevét stb. A telefonszamTipus azt adja meg, hogy az első karakter 1-9 közötti szám kell hogy legyen, utána a következő 8 karakter 0-9 közötti számok lehetnek. A bankTipus csak a felsorolt bankok közül enged választani (pl. OTP, KH).A ttipus pedig csak hitelkártya illetve bankkártya közül enged választani. Továbbá komplex típusokat is definiál, mint a pizzazoTipus, beszallitasTipus, beszallitoTipus, futarTipus, vevoTipus, rendelesTipus, pizzaTipus, bankkartyaTipus melyek különféle attribútumokat és elemeket tartalmaznak. Az adatbázis integritásának megőrzése érdekében elsődleges (PK) és idegen kulcsok (FK) meghatározására kerül sor, valamint egyediség biztosítása (pl. minden vevőnek egyedülálló bankkártyája lehet) az vevo_bankkartya_egyegy elem esetében. Az XML séma így biztosítja, hogy az adatok szerkezete és kapcsolatai érvényesek és következetesek legyenek.
Az XMLSchema forráskódja:

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema attributeFormDefault="unqualified" elementFormDefault="qualified"
    xmlns:xs="http://www.w3.org/2001/XMLSchema">
    <!-- Egyszeű tipusok kigyűjtése, saját tipusok meghatározása, megszorítás-
->
    <xs:element name="nev" type="xs:string"/>
    <xs:element name="datum" type="xs:date"/>
    <xs:element name="hozzavalo" type="xs:string"/>
    <xs:element name="telefonszam" type="telefonszamTipus"/>
    <xs:element name="email" type="xs:string"/>
    <xs:element name="pizzaneve" type="pizzanevTipus"/>
    <xs:element name="feltet" type="feltetTipus"/>
    <xs:element name="teljes_ar" type="xs:int"/>
    <xs:element name="meret" type="xs:string"/>
    <xs:element name="bank" type="bankTipus"/>
    <xs:element name="tipus" type="ttipus"/>
    <xs:element name="lejaratidatum" type="xs:gYearMonth"/>


    <xs:simpleType name="telefonszamTipus">
        <xs:restriction base="xs:string">
            <xs:pattern value="[1-9]{1}[0-9]{8}"/>
        </xs:restriction>
    </xs:simpleType>

    <xs:simpleType name="pizzanevTipus">
        <xs:restriction base="xs:string">
            <xs:pattern value="[a-zA-Z0-9áÁéÉíÍóÓöÖőŐúÚüÜűŰ, ]+"/>
        </xs:restriction>
    </xs:simpleType>
```

```
    <xs:simpleType name="feltetTipus">
        <xs:restriction base="xs:string">
            <xs:pattern value="[a-zA-Z0-9áÁéÉíÍóÓöÖőŐúÚüÜűŰ, ]+"/>
        </xs:restriction>
    </xs:simpleType>

    <xs:simpleType name="bankTipus">
        <xs:restriction base="xs:string">
            <xs:enumeration value="OTP"/>
            <xs:enumeration value="KH"/>
            <xs:enumeration value="MBH"/>
            <xs:enumeration value="ERSTE"/>
            <xs:enumeration value="UNICREDIT"/>
        </xs:restriction>
    </xs:simpleType>

    <xs:simpleType name="ttipus">
        <xs:restriction base="xs:string">
            <xs:enumeration value="hitelkartya"/>
            <xs:enumeration value="bankkartya"/>
        </xs:restriction>
    </xs:simpleType>

    <!--Komplex tipusokhoz saját tipus meghatározása, sorrendiség, számosság
etc. -->

    <xs:complexType name="pizzazoTipus">
        <xs:sequence>
            <xs:element ref="nev"/>
            <xs:element name="elerhetoseg">
                <xs:complexType>
                    <xs:sequence>
                        <xs:element name="weboldal" type="xs:string"/>
                        <xs:element name="telefonszam"
type="telefonszamTipus"/>
                    </xs:sequence>
                </xs:complexType>
            </xs:element>
        </xs:sequence>
        <xs:attribute name="pizzazo_id" type="xs:integer" use="required"/>
    </xs:complexType>

    <xs:complexType name="beszallitasTipus">
        <xs:sequence>
            <xs:element ref="datum"/>
            <xs:element ref="hozzavalo"/>
        </xs:sequence>
        <xs:attribute name="pizzazo" type="xs:integer" use="required"/>
```

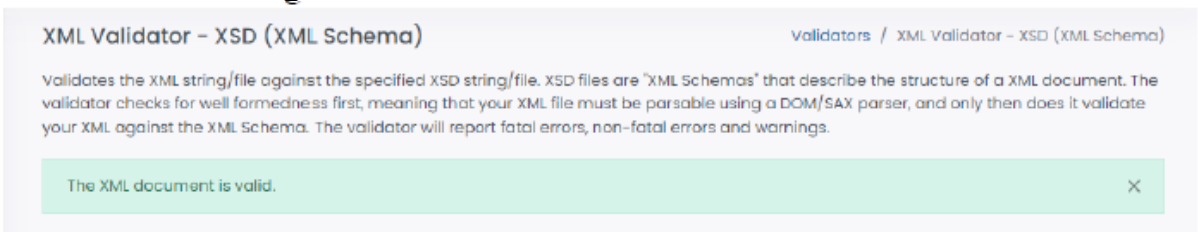
```
    <xs:attribute name="beszallito" type="xs:integer" use="required"/>
</xs:complexType>

<xs:complexType name="beszallitoTipus">
    <xs:sequence>
        <xs:element ref="nev"/>
        <xs:element ref="email"/>
        <xs:element name="cim">
            <xs:complexType>
                <xs:sequence>
                    <xs:element name="varos" type="xs:string"/>
                    <xs:element name="hazszam" type="xs:int"/>
                    <xs:element name="utca" type="xs:string"/>
                    <xs:element name="iranyitoszam" type="xs:int"/>
                </xs:sequence>
            </xs:complexType>
        </xs:element>
    </xs:sequence>
    <xs:attribute name="beszallito_id" type="xs:integer" use="required"/>
</xs:complexType>

<xs:complexType name="futarTipus">
    <xs:sequence>
        <xs:element ref="nev"/>
        <xs:element ref="telefonszam"/>
    </xs:sequence>
    <xs:attribute name="futar_id" type="xs:integer" use="required"/>
    <xs:attribute name="pizzazo" type="xs:integer" use="required"/>
</xs:complexType>

<xs:complexType name="vevoTipus">
    <xs:sequence>
        <xs:element ref="nev"/>
        <xs:element ref="telefonszam"/>
        <xs:element name="cim">
            <xs:complexType>
                <xs:sequence>
                    <xs:element name="varos" type="xs:string"/>
                    <xs:element name="hazszam" type="xs:int"/>
                    <xs:element name="utca" type="xs:string"/>
                    <xs:element name="iranyitoszam" type="xs:int"/>
                </xs:sequence>
            </xs:complexType>
        </xs:element>
    </xs:sequence>
    <xs:attribute name="vevo_id" type="xs:integer" use="required"/>
</xs:complexType>
```

```
    <xs:complexType name="rendelesTipus">
        <xs:attribute name="pizza" type="xs:integer" use="required"/>
        <xs:attribute name="vevo" type="xs:integer" use="required"/>
    </xs:complexType>

    <xs:complexType name="pizzaTipus">
        <xs:sequence>
            <xs:element ref="pizzaneve"/>
            <xs:element ref="feltet" minOccurs="1" maxOccurs="unbounded"/>
            <xs:element name="teljes_ar" minOccurs="1" maxOccurs="unbounded"/>
            <xs:element ref="meret" minOccurs="1" maxOccurs="3"/>
        </xs:sequence>
        <xs:attribute name="pizza_id" type="xs:integer" use="required"/>
        <xs:attribute name="pizzazo" type="xs:integer" use="required"/>
    </xs:complexType>

    <xs:complexType name="bankkartyaTipus">
        <xs:sequence>
            <xs:element ref="bank"/>
            <xs:element ref="tipus"/>
            <xs:element name="lejaratidatum"/>
        </xs:sequence>
        <xs:attribute name="kartyaszam" type="xs:integer" use="required"/>
        <xs:attribute name="vevo" type="xs:integer" use="required"/>
    </xs:complexType>




    <!-- Gyökérelemtől az elemek felhasználása -->

    <xs:element name="Pizzazo_EZ3YRC">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="Pizzazo" type="pizzazoTipus" minOccurs="0"
maxOccurs="100"/>
                <xs:element name="Beszallitas" type="beszallitasTipus"
minOccurs="0"
                    maxOccurs="unbounded"/>
                <xs:element name="Beszallito" type="beszallitoTipus"
minOccurs="0"
                    maxOccurs="unbounded"/>
                <xs:element name="Futar" type="futarTipus" minOccurs="0"
maxOccurs="unbounded"/>
                <xs:element name="Vevo" type="vevoTipus" minOccurs="0"
maxOccurs="unbounded"/>
                <xs:element name="Rendeles" type="rendelesTipus" minOccurs="0"
maxOccurs="unbounded"/>
```

```
                <xs:element name="Pizza" type="pizzaTipus" minOccurs="0"
maxOccurs="100"/>
                <xs:element name="Bankkartya" type="bankkartyaTipus"
minOccurs="0"
                    maxOccurs="unbounded"/>
            </xs:sequence>
        </xs:complexType>

        <!-- Elsődleges kulcsok -->

        <xs:key name="pizzazo_kulcs">
            <xs:selector xpath="Pizzazo"/>
            <xs:field xpath="@pizzazo_id"/>
        </xs:key>

        <xs:key name="beszallito_kulcs">
            <xs:selector xpath="Beszallito"/>
            <xs:field xpath="@beszallito_id"/>
        </xs:key>

        <xs:key name="futar_kulcs">
            <xs:selector xpath="Futar"/>
            <xs:field xpath="@futar_id"/>
        </xs:key>

        <xs:key name="vevo_kulcs">
            <xs:selector xpath="Vevo"/>
            <xs:field xpath="@vevo_id"/>
        </xs:key>

        <xs:key name="pizza_kulcs">
            <xs:selector xpath="Pizza"/>
            <xs:field xpath="@pizza_id"/>
        </xs:key>

        <xs:key name="bankkartya_kulcs">
            <xs:selector xpath="Bankkartya"/>
            <xs:field xpath="@kartyaszam"/>
        </xs:key>

        <!-- Idegen kulcsok -->

        <xs:keyref name="pizzazo_beszallitas_kulcs" refer="pizzazo_kulcs">
            <xs:selector xpath="Beszallitas"/>
            <xs:field xpath="@pizzazo"/>
        </xs:keyref>
```

```
        <xs:keyref name="beszallito_beszallitas_kulcs"
refer="beszallito_kulcs">
            <xs:selector xpath="Beszallitas"/>
            <xs:field xpath="@beszallito"/>
        </xs:keyref>

        <xs:keyref name="pizzazo_futar_kulcs" refer="pizzazo_kulcs">
            <xs:selector xpath="Futar"/>
            <xs:field xpath="@pizzazo"/>
        </xs:keyref>

        <xs:keyref name="pizza_rendeles_kulcs" refer="pizza_kulcs">
            <xs:selector xpath="Rendeles"/>
            <xs:field xpath="@pizza"/>
        </xs:keyref>

        <xs:keyref name="vevo_rendeles_kulcs" refer="vevo_kulcs">
            <xs:selector xpath="Rendeles"/>
            <xs:field xpath="@vevo"/>
        </xs:keyref>

        <xs:keyref name="pizzazo_pizza_kulcs" refer="pizzazo_kulcs">
            <xs:selector xpath="Pizza"/>
            <xs:field xpath="@pizzazo"/>
        </xs:keyref>

        <xs:keyref name="vevo_bankkartya_kulcs" refer="vevo_kulcs">
            <xs:selector xpath="Bankkartya"/>
            <xs:field xpath="@vevo"/>
        </xs:keyref>

        <!-- Az 1:1 kapcsolat megvalósítás -->
        <xs:unique name="vevo_bankkartya_egyegy">
            <xs:selector xpath="Bankkartya"/>
            <xs:field xpath="@vevo"/>
        </xs:unique>
    </xs:element>
</xs:schema>
```

Validáció sikeressége:



2a) DOM file beolvasás

A kód egy egyszerű alkalmazást tartalmaz, amely egy XML fájlt dolgoz fel a DOM (Document Object Model) parser segítségével. Az alkalmazás a Pizzazo_EZ3YRC XML adatstruktúráját kezeli, amely pizzériával kapcsolatos információkat, rendeléseket, vevőket, beszállítókat és egyebeket tartalmaz. A kód különböző metódusokat tartalmaz, amelyek minden egyes XML elem típusát kezelik, és kiírják vagy feldolgozzák azok adatait. A kód célja a strukturált adatok könnyű és érthető módon történő beolvasása és megjelenítése a konzolon.

```
package hu.domparse.ez3yrc;

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

import java.io.*;

public class DOMReadEZ3YRC {

    public static void main(String[] args) {
        try {
            // XML dokumentum feldolgozásához szükséges objektumok
létrehozása
            DocumentBuilderFactory factory =
DocumentBuilderFactory.newInstance();
            DocumentBuilder builder = factory.newDocumentBuilder();
            Document document = builder.parse(new File("XMLEZ3YRC.xml"));

            // XML struktúra kiíratása
            document.getDocumentElement().normalize();
            System.out.println("<?xml version=\"1.0\" encoding=\"UTF-
8\"?>\n");
            System.out.println("<Pizzazo_EZ3YRC
xmlns:xsi=\"http://www.w3.org/2001/XMLSchema-instance\"
xsi:noNamespaceSchemaLocation=\"XMLSchemaEZ3YRC.xsd\">\n");

            readPizzazos(document);
            readBeszallitasok(document);
            readBeszallitok(document);
            readFutarok(document);
            readVevok(document);
            readRendelesek(document);
            readPizzak(document);
            readBankkartyak(document);

            System.out.println("\n</Pizzazo_EZ3YRC>");
        } catch (ParserConfigurationException | IOException | SAXException
e) {
```

```
            System.out.println("Valami baj van: " + e);
        }
    }

    // Pizzázók adatainak kiolvasása és kiíratása
    private static void readPizzazos(Document document) {
        NodeList pizzazoList = document.getElementsByTagName("Pizzazo");
        for (int temp = 0; temp < pizzazoList.getLength(); temp++) {
            Node node = pizzazoList.item(temp);
            if (node.getNodeType() == Node.ELEMENT_NODE) {
                Element eElement = (Element) node;
                String pizzazoId = eElement.getAttribute("pizzazo_id");
                String pizzazoName =
eElement.getElementsByTagName("nev").item(0).getTextContent();
                String pizzazoWebsite =
eElement.getElementsByTagName("weboldal").item(0).getTextContent();
                String pizzazoPhoneNumber =
eElement.getElementsByTagName("telefonszam").item(0).getTextContent();

                System.out.println("    <Pizzazo pizzazo_id=\"" + pizzazoId
+ "\">");
                printElement("nev", pizzazoName);
                System.out.println("        <elerhetoseg>");
                printElement("weboldal", pizzazoWebsite);
                printElement("telefonszam", pizzazoPhoneNumber);
                System.out.println("        </elerhetoseg>");
                System.out.println("    </Pizzazo>");
            }
        }
    }

    // Beszállítások adatainak kiolvasása és kiíratása
    private static void readBeszallitasok(Document document) {
        NodeList beszallitasList =
document.getElementsByTagName("Beszallitas");
        for (int temp = 0; temp < beszallitasList.getLength(); temp++) {
            Node node = beszallitasList.item(temp);
            if (node.getNodeType() == Node.ELEMENT_NODE) {
                Element eElement = (Element) node;
                String beszallitoId = eElement.getAttribute("beszallito");
                String pizzazoId = eElement.getAttribute("pizzazo");
                String beszallitasDate =
eElement.getElementsByTagName("datum").item(0).getTextContent();
                String hozzavalo =
eElement.getElementsByTagName("hozzavalo").item(0).getTextContent();

                System.out.println("    <Beszallitas beszallito=\"" +
beszallitoId + "\" pizzazo=\"" + pizzazoId + "\">");
                printElement("datum", beszallitasDate);
                printElement("hozzavalo", hozzavalo);
                System.out.println("    </Beszallitas>");
            }
        }
    }

    // Beszállítók adatainak kiolvasása és kiíratása
    private static void readBeszallitok(Document document) {
        NodeList beszallitoList =
document.getElementsByTagName("Beszallito");
        for (int temp = 0; temp < beszallitoList.getLength(); temp++) {
```

```
            Node node = beszallitoList.item(temp);
            if (node.getNodeType() == Node.ELEMENT_NODE) {
                Element eElement = (Element) node;
                String beszallitoId =
eElement.getAttribute("beszallito_id");
                String beszallitoName =
eElement.getElementsByTagName("nev").item(0).getTextContent();
                String beszallitoEmail =
eElement.getElementsByTagName("email").item(0).getTextContent();
                String varos =
eElement.getElementsByTagName("varos").item(0).getTextContent();
                String hazszam =
eElement.getElementsByTagName("hazszam").item(0).getTextContent();
                String utca =
eElement.getElementsByTagName("utca").item(0).getTextContent();
                String iranyitoszam =
eElement.getElementsByTagName("iranyitoszam").item(0).getTextContent();

                System.out.println("    <Beszallito beszallito_id=\"" +
beszallitoId + "\">");
                printElement("nev", beszallitoName);
                printElement("email", beszallitoEmail);
                System.out.println("        <cim>");
                printElement("varos", varos);
                printElement("hazszam", hazszam);
                printElement("utca", utca);
                printElement("iranyitoszam", iranyitoszam);
                System.out.println("        </cim>");
                System.out.println("    </Beszallito>");
            }
        }
    }

    // Futárok adatainak kiolvasása és kiíratása
    private static void readFutarok(Document document) {
        NodeList futarList = document.getElementsByTagName("Futar");
        for (int temp = 0; temp < futarList.getLength(); temp++) {
            Node node = futarList.item(temp);
            if (node.getNodeType() == Node.ELEMENT_NODE) {
                Element eElement = (Element) node;
                String futarId = eElement.getAttribute("futar_id");
                String pizzazoId = eElement.getAttribute("pizzazo");
                String futarName =
eElement.getElementsByTagName("nev").item(0).getTextContent();
                String futarPhoneNumber =
eElement.getElementsByTagName("telefonszam").item(0).getTextContent();

                System.out.println("    <Futar futar_id=\"" + futarId + "\"
pizzazo=\"" + pizzazoId + "\">");
                printElement("nev", futarName);
                printElement("telefonszam", futarPhoneNumber);
                System.out.println("    </Futar>");
            }
        }
    }

    // Vevők adatainak kiolvasása és kiíratása
    private static void readVevok(Document document) {
        NodeList vevoList = document.getElementsByTagName("Vevo");
        for (int temp = 0; temp < vevoList.getLength(); temp++) {
```

```
            Node node = vevoList.item(temp);
            if (node.getNodeType() == Node.ELEMENT_NODE) {
                Element eElement = (Element) node;
                String vevoId = eElement.getAttribute("vevo_id");
                String vevoName =
eElement.getElementsByTagName("nev").item(0).getTextContent();
                String vevoPhoneNumber =
eElement.getElementsByTagName("telefonszam").item(0).getTextContent();
                String varos =
eElement.getElementsByTagName("varos").item(0).getTextContent();
                String hazszam =
eElement.getElementsByTagName("hazszam").item(0).getTextContent();
                String utca =
eElement.getElementsByTagName("utca").item(0).getTextContent();
                String iranyitoszam =
eElement.getElementsByTagName("iranyitoszam").item(0).getTextContent();

                System.out.println("    <Vevo vevo_id=\"" + vevoId +
"\">");
                printElement("nev", vevoName);
                printElement("telefonszam", vevoPhoneNumber);
                System.out.println("        <cim>");
                printElement("varos", varos);
                printElement("hazszam", hazszam);
                printElement("utca", utca);
                printElement("iranyitoszam", iranyitoszam);
                System.out.println("        </cim>");
                System.out.println("    </Vevo>");
            }
        }
    }

    // Rendelések adatainak kiolvasása és kiíratása
    private static void readRendelesek(Document document) {
        NodeList rendelesList = document.getElementsByTagName("Rendeles");
        for (int temp = 0; temp < rendelesList.getLength(); temp++) {
            Node node = rendelesList.item(temp);
            if (node.getNodeType() == Node.ELEMENT_NODE) {
                Element eElement = (Element) node;
                String pizzaId = eElement.getAttribute("pizza");
                String vevoId = eElement.getAttribute("vevo");

                System.out.println("    <Rendeles pizza=\"" + pizzaId + "\"
vevo=\"" + vevoId + "\"></Rendeles>");
            }
        }
    }

    // Pizzák adatainak kiolvasása és kiíratása
    private static void readPizzak(Document document) {
        NodeList pizzaList = document.getElementsByTagName("Pizza");
        for (int temp = 0; temp < pizzaList.getLength(); temp++) {
            Node node = pizzaList.item(temp);
            if (node.getNodeType() == Node.ELEMENT_NODE) {
                Element eElement = (Element) node;
                String pizzaId = eElement.getAttribute("pizza_id");
                String pizzazoId = eElement.getAttribute("pizzazo");
                String pizzaName =
eElement.getElementsByTagName("pizzaneve").item(0).getTextContent();
                NodeList feltetList =
```

```
eElement.getElementsByTagName("feltet");
                NodeList meretList =
eElement.getElementsByTagName("meret");
                String teljesAr =
eElement.getElementsByTagName("teljes_ar").item(0).getTextContent();

                System.out.println("    <Pizza pizza_id=\"" + pizzaId + "\"
pizzazo=\"" + pizzazoId + "\">");
                printElement("pizzaneve", pizzaName);
                for (int i = 0; i < feltetList.getLength(); i++) {
                    printElement("feltet",
feltetList.item(i).getTextContent());
                }
                for (int i = 0; i < meretList.getLength(); i++) {
                    printElement("meret",
meretList.item(i).getTextContent());
                }
                printElement("teljes_ar", teljesAr);
                System.out.println("    </Pizza>");
            }
        }
    }

    // Bankkártyák adatainak kiolvasása és kiíratása
    private static void readBankkartyak(Document document) {
        NodeList bankkartyaList =
document.getElementsByTagName("Bankkartya");
        for (int temp = 0; temp < bankkartyaList.getLength(); temp++) {
            Node node = bankkartyaList.item(temp);
            if (node.getNodeType() == Node.ELEMENT_NODE) {
                Element eElement = (Element) node;
                String kartyaszam = eElement.getAttribute("kartyaszam");
                String vevoId = eElement.getAttribute("vevo");
                String bank =
eElement.getElementsByTagName("bank").item(0).getTextContent();
                String tipus =
eElement.getElementsByTagName("tipus").item(0).getTextContent();
                String lejaratiDatum =
eElement.getElementsByTagName("lejaratidatum").item(0).getTextContent();

                System.out.println("    <Bankkartya kartyaszam=\"" +
kartyaszam + "\" vevo=\"" + vevoId + "\">");
                printElement("bank", bank);
                printElement("tipus", tipus);
                printElement("lejaratidatum", lejaratiDatum);
                System.out.println("    </Bankkartya>");
            }
        }
    }

    // Segédfüggvény az XML elemek kiíratásához
    private static void printElement(String elementName, String content) {
        System.out.println("        <" + elementName + ">" + content + "</"
+ elementName + ">");
    }
    // Dokumentum írásához használt függvény
    public static void writeDocument(Document document, StreamResult
result) throws TransformerException {
        TransformerFactory transformerFactory =
TransformerFactory.newInstance();
```

```
        Transformer transformer = transformerFactory.newTransformer();
        transformer.setOutputProperty(OutputKeys.INDENT, "yes");
        DOMSource source = new DOMSource(document);
        transformer.transform(source, result);
    }
}
```

2b) DOM adatmódosítás

A kód először beolvassa az "XMLEZ3YRC.xml" nevű XML fájlt, majd végrehajt néhány módosítást a dokumentumon, és végül kiírja az eredményt a konzolra.

1. A végrehajtott módosítások a következők:
2. A "Pizzazo" elem nevének módosítása "Fortuna Pizzéria"-ra.
3. A "Pizza" elemek méreteinek átállítása 25, 32 és 50 értékekre.
4. A második "Beszallitas" elem hozzávalójának módosítása "olaj"-ra.
5. A harmadik "Vevo" elem telefonszámának módosítása "408883091"-re.
6. A harmadik "Bankkartya" elem bankjának "OTP"-re, típusának "hitelkartya"-ra és lejárati dátumának "2028-12"-re állítása.

Végül a program kiírja az eredményt az XML dokumentumra végrehajtott összes módosítással együtt, formázva és behúzásokkal az olvashatóság javítása érdekében.

```
package hu.domparse.ez3yrc;

import javax.xml.parsers.*;
import javax.xml.transform.OutputKeys;
import javax.xml.transform.Transformer;
import javax.xml.transform.TransformerFactory;
import javax.xml.transform.dom.DOMSource;
import javax.xml.transform.stream.StreamResult;
import org.w3c.dom.*;
import java.io.File;

public class DOMModifyEZ3YRC {
    // Main metódus
    public static void main(String argv[]) {
        try {
            File inputFile = new File("XMLEZ3YRC.xml");

            DocumentBuilderFactory docFactory =
DocumentBuilderFactory.newInstance();
            DocumentBuilder docBuilder = docFactory.newDocumentBuilder();
            Document doc = docBuilder.parse(inputFile);

            // Módosítások a main függvényben
            // Módosítás 1: Pizzazo nevének módosítása
            Element firstPizzazo = (Element)
doc.getElementsByTagName("Pizzazo").item(0);

firstPizzazo.getElementsByTagName("nev").item(0).setTextContent("Fortuna
Pizzéria");

            // Módosítás 2: Pizza méreteinek átállítása
            NodeList pizzaList = doc.getElementsByTagName("Pizza");
```

```
            for (int i = 0; i < pizzaList.getLength(); i++) {
                Element pizzaElement = (Element) pizzaList.item(i);
                NodeList meretList =
pizzaElement.getElementsByTagName("meret");
                meretList.item(0).setTextContent("25");
                meretList.item(1).setTextContent("32");
                meretList.item(2).setTextContent("50");
            }

            // Módosítás 3: Beszállítás hozzávaló módosítása
            Element beszallitasElement = (Element)
doc.getElementsByTagName("Beszallitas").item(1);

beszallitasElement.getElementsByTagName("hozzavalo").item(0).setTextContent
("olaj");

            // Módosítás 4: Vevő telefonszámának módosítása
            Element thirdVevo = (Element)
doc.getElementsByTagName("Vevo").item(2);

thirdVevo.getElementsByTagName("telefonszam").item(0).setTextContent("40888
3091");

            // Módosítás 5: Bankkártya adatainak módosítása
            Element thirdBankkartya = (Element)
doc.getElementsByTagName("Bankkartya").item(2);

thirdBankkartya.getElementsByTagName("bank").item(0).setTextContent("OTP");

thirdBankkartya.getElementsByTagName("tipus").item(0).setTextContent("hitel
kartya");

thirdBankkartya.getElementsByTagName("lejaratidatum").item(0).setTextConten
t("2028-12");

            // Kimenet kiírása
            TransformerFactory transformerFactory =
TransformerFactory.newInstance();
            Transformer transformer = transformerFactory.newTransformer();
            transformer.setOutputProperty(OutputKeys.INDENT, "yes");
            DOMSource source = new DOMSource(doc);
            StreamResult consoleResult = new StreamResult(System.out);
            transformer.transform(source, consoleResult);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

2c) DOM adat lekérdezés

A különböző lekérdezéseket külön metódusok implementálják, és ezeket hívja meg a main metódus.

Az egyes lekérdezések a következők:

1. ExpiringAfter2024: Kilistázza azokat a bankkártyákat, amelyeknek a lejárati dátuma 2024 után van.
2. AffordablePizzas: Kilistázza azokat a pizzákat, amelyek teljes ára 2600-nál olcsóbb.
3. MiskolciBeszallitok: Miskolci beszállítók kilistázása.
4. FortunaPizzaIngredientsAndSupplier: Kilistázza a Fortuna Pizzéria által rendelt hozzávalókat és a beszállítókat.
5. DonPepePizzasAndPrices: Kilistázza a Don Pepe pizzériában rendelt pizzákat és azok teljes árát.

Minden lekérdezésnél a megfelelő XML elemeket keresi meg a DOM segítségével, majd a kívánt adatokat kiírja a konzolra.

```
package hu.domparse.ez3yrc;

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 DOMQueryEZ3YRC {

    public static void main(String[] argv) throws SAXException,
IOException, ParserConfigurationException {
        File xmlFile = new File("XMLEZ3YRC.xml");

        DocumentBuilderFactory factory =
DocumentBuilderFactory.newInstance();
        DocumentBuilder dBuilder = factory.newDocumentBuilder();

        Document doc = dBuilder.parse(xmlFile);
        doc.getDocumentElement().normalize();

        StringBuilder outputBuilder = new StringBuilder();
```

```
        // 2024 utáni lejáratú bankkártyák kiírása
        NodeList bankCardList = doc.getElementsByTagName("Bankkartya");
        outputBuilder.append("\n<ExpiringAfter2024>\n");
        for (int i = 0; i < bankCardList.getLength(); i++) {
            Node node = bankCardList.item(i);
            if (node.getNodeType() == Node.ELEMENT_NODE) {
                Element element = (Element) node;
                String lejaratidatum =
element.getElementsByTagName("lejaratidatum").item(0).getTextContent();
                if (lejaratidatum.compareTo("2024-12") > 0) {
                    String kartyaszam = element.getAttribute("kartyaszam");
                    String bank =
element.getElementsByTagName("bank").item(0).getTextContent();
                    String tipus =
element.getElementsByTagName("tipus").item(0).getTextContent();
                    outputBuilder.append(String.format("  <Bankkartya
kartyaszam=\"%s\">\n", kartyaszam));
                    outputBuilder.append(String.format("
<Bank>%s</Bank>\n", bank));
                    outputBuilder.append(String.format("
<Tipus>%s</Tipus>\n", tipus));
                    outputBuilder.append(String.format("
<LejaratIdatum>%s</LejaratIdatum>\n", lejaratidatum));
                    outputBuilder.append("  </Bankkartya>\n");
                }
            }
        }
        outputBuilder.append("</ExpiringAfter2024>\n");

        //Kilistázza azokat a pizzákat, amelyek teljes ára 2600-nál
olcsóbb.
        NodeList pizzaList = doc.getElementsByTagName("Pizza");
        outputBuilder.append("\n<AffordablePizzas>\n");
        for (int i = 0; i < pizzaList.getLength(); i++) {
            Node node = pizzaList.item(i);
            if (node.getNodeType() == Node.ELEMENT_NODE) {
                Element element = (Element) node;
                String teljesAr =
element.getElementsByTagName("teljes_ar").item(0).getTextContent();
                int teljesArInt = Integer.parseInt(teljesAr);
                if (teljesArInt <= 2600) {
                    String pizzaId = element.getAttribute("pizza_id");
                    String pizzaneve =
element.getElementsByTagName("pizzaneve").item(0).getTextContent();
                    outputBuilder.append(String.format("  <Pizza
pizza_id=\"%s\">\n", pizzaId));
                    outputBuilder.append(String.format("
<Nev>%s</Nev>\n", pizzaneve));
                    outputBuilder.append(String.format("
<TeljesAr>%s</TeljesAr>\n", teljesAr));
                    outputBuilder.append("  </Pizza>\n");
                }
            }
        }
        outputBuilder.append("</AffordablePizzas>\n");

        // Miskolci beszállítók kilistázása
        NodeList beszallitoList = doc.getElementsByTagName("Beszallito");
        outputBuilder.append("\n<MiskolciBeszallitok>\n");
        for (int i = 0; i < beszallitoList.getLength(); i++) {
```

```
            Node node = beszallitoList.item(i);
            if (node.getNodeType() == Node.ELEMENT_NODE) {
                Element element = (Element) node;
                NodeList varosList = element.getElementsByTagName("varos");
                if (varosList.getLength() > 0) {
                    String varos = varosList.item(0).getTextContent();
                    if ("Miskolc".equals(varos)) {
                        String beszallitoId =
element.getAttribute("beszallito_id");
                        String nev =
element.getElementsByTagName("nev").item(0).getTextContent();
                        outputBuilder.append(String.format("  <Beszallito
beszallito_id=\"%s\">\n", beszallitoId));
                        outputBuilder.append(String.format("
<Nev>%s</Nev>\n", nev));
                        outputBuilder.append(String.format("
<Varos>%s</Varos>\n", varos));
                        outputBuilder.append("  </Beszallito>\n");
                    }
                }
            }
        }
        outputBuilder.append("</MiskolciBeszallitok>\n");


        // Fortuna pizzéria hozzávalóinak és beszállítójának kilistázása
        NodeList pizzazoList = doc.getElementsByTagName("Pizzazo");
        outputBuilder.append("\n<FortunaPizzaIngredientsAndSupplier>\n");
        String fortunaPizzazoId = "";
        for (int i = 0; i < pizzazoList.getLength(); i++) {
            Node node = pizzazoList.item(i);
            if (node.getNodeType() == Node.ELEMENT_NODE) {
                Element element = (Element) node;
                String nev =
element.getElementsByTagName("nev").item(0).getTextContent();
                if (nev.equals("Fortuna Pizzéria")) {
                    fortunaPizzazoId = element.getAttribute("pizzazo_id");
                    break;
                }
            }
        }

        NodeList beszallitasList = doc.getElementsByTagName("Beszallitas");
        for (int i = 0; i < beszallitasList.getLength(); i++) {
            Node node = beszallitasList.item(i);
            if (node.getNodeType() == Node.ELEMENT_NODE) {
                Element element = (Element) node;
                String pizzazo = element.getAttribute("pizzazo");
                if (pizzazo.equals(fortunaPizzazoId)) {
                    String datum =
element.getElementsByTagName("datum").item(0).getTextContent();
                    String hozzavalo =
element.getElementsByTagName("hozzavalo").item(0).getTextContent();
                    String beszallito = element.getAttribute("beszallito");
                    outputBuilder.append(String.format("  <Beszallitas
datum=\"%s\" hozzavalo=\"%s\" beszallito=\"%s\">\n", datum, hozzavalo,
beszallito));
                    outputBuilder.append("  </Beszallitas>\n");
                }
            }
```

```
        }
        outputBuilder.append("</FortunaPizzaIngredientsAndSupplier>\n");

        // Don Pepe pizzéria pizzáinak és árainak kilistázása
        outputBuilder.append("\n<DonPepePizzasAndPrices>\n");
        String donPepePizzazoId = "";
        for (int i = 0; i < pizzazoList.getLength(); i++) {
            Node node = pizzazoList.item(i);
            if (node.getNodeType() == Node.ELEMENT_NODE) {
                Element element = (Element) node;
                String nev =
element.getElementsByTagName("nev").item(0).getTextContent();
                if (nev.equals("Don Pepe")) {
                    donPepePizzazoId = element.getAttribute("pizzazo_id");
                    break;
                }
            }
        }


        for (int i = 0; i < pizzaList.getLength(); i++) {
            Node node = pizzaList.item(i);
            if (node.getNodeType() == Node.ELEMENT_NODE) {
                Element element = (Element) node;
                String pizzazo = element.getAttribute("pizzazo");
                if (pizzazo.equals(donPepePizzazoId)) {
                    String pizzaId = element.getAttribute("pizza_id");
                    String pizzaneve =
element.getElementsByTagName("pizzaneve").item(0).getTextContent();
                    String teljesAr =
element.getElementsByTagName("teljes_ar").item(0).getTextContent();
                    outputBuilder.append(String.format("  <Pizza
pizza_id=\"%s\">\n", pizzaId));
                    outputBuilder.append(String.format("
<Nev>%s</Nev>\n", pizzaneve));
                    outputBuilder.append(String.format("
<TeljesAr>%s</TeljesAr>\n", teljesAr));
                    outputBuilder.append("  </Pizza>\n");
                }
            }
        }
        outputBuilder.append("</DonPepePizzasAndPrices>\n");

        System.out.println(outputBuilder);
    }
}
}
```

2d) DOM adatírás

Az osztályban a main metódusban XML elemeket hoz létre és ír ki egy XML fájlba. Az elkészített XML fájl a Pizza rendelési rendszer adatait reprezentálja.

A kód fő részei: Az XML dokumentum létrehozása és a gyökér elem inicializálása. Pizzazo, Beszallitas, Beszallito, Futar, Vevo, Rendeles, Pizza és Bankkartya elemek hozzáadása a gyökér elemhez. Az XML fájl kimenetének előkészítése és a fájlba írás.

Végül, az XML fájlba való írás a TransformerFactory és a Transformer objektumok segítségével történik. Az elkészített XML fájl neve "XMLEZ3YRC1.xml" és a kimeneti fájlba írás után kiírja a konzolra a sikeres üzenetet vagy a hibaüzenetet.

```
package hu.domparse.ez3yrc;

import org.w3c.dom.*;
import javax.xml.parsers.*;
import javax.xml.transform.*;
import javax.xml.transform.dom.DOMSource;
import javax.xml.transform.stream.StreamResult;
import java.io.*;

public class DOMWriteEZ3YRC {

    public static void main(String[] args) {
        try {
            DocumentBuilderFactory dbFactory =
DocumentBuilderFactory.newInstance();
            DocumentBuilder dBuilder = dbFactory.newDocumentBuilder();
            Document doc = dBuilder.newDocument();

            // Gyökér elem létrehozása
            Element rootElement = doc.createElement("Pizzazo_EZ3YRC");
            doc.appendChild(rootElement);

            // Namespace attribútumok hozzáadása
            rootElement.setAttribute("xmlns:xsi",
"http://www.w3.org/2001/XMLSchema-instance");
            rootElement.setAttribute("xsi:noNamespaceSchemaLocation",
"XMLSchemaEZ3YRC.xsd");

            // Pizzazo elemek hozzáadása
            addPizzazo(doc, rootElement, "1", "Don Pepe",
"https://www.donpepe.hu/hu", "309082091");
            addPizzazo(doc, rootElement, "2", "Pizza Tábor",
"https://www.pizzatabor.hu/", "204716655");
            addPizzazo(doc, rootElement, "3", "Fortuna Pizzéria",
"https://pizzafortuna.hu/", "302415505");

            // Beszallitas elemek hozzáadása
            addBeszallitas(doc, rootElement, "1", "1", "2023-11-04",
"hagyma");
            addBeszallitas(doc, rootElement, "2", "2", "2023-11-19",
```

```
"cukor");
            addBeszallitas(doc, rootElement, "3", "3", "2023-11-25",
"olaj");

            // Beszallito elemek hozzáadása
            addBeszallito(doc, rootElement, "1", "Kovács Lajos",
"mintalajos@gmail.com", "Miskolc", "5", "Klapka György", "3524");
            addBeszallito(doc, rootElement, "2", "Nehéz István",
"mintaistvan@gmail.com", "Miskolc", "5", "Petőfi Sándor", "3527");
            addBeszallito(doc, rootElement, "3", "Kiss István",
"mintakiss@gmail.com", "Miskolc", "4", "József Attila", "3531");

            // Futar elemek hozzáadása
            addFutar(doc, rootElement, "1", "1", "Dudás Lajos",
"309562179");
            addFutar(doc, rootElement, "2", "2", "Lajos Imre",
"309652179");
            addFutar(doc, rootElement, "3", "3", "Nagy Márkó",
"209852179");

            // Vevo elemek hozzáadása
            addVevo(doc, rootElement, "1", "Nehéz Gábor", "306153384",
"Miskolc", "6", "Hajós Alfréd", "3524");
            addVevo(doc, rootElement, "2", "Kiss Lajos", "206453384",
"Miskolc", "4", "Vörösmarty Mihály", "3532");
            addVevo(doc, rootElement, "3", "Elek János", "206158484",
"Miskolc", "6", "Árok utca", "3531");

            // Rendeles elemek hozzáadása
            addRendeles(doc, rootElement, "1", "1", "1");
            addRendeles(doc, rootElement, "2", "2", "2");
            addRendeles(doc, rootElement, "3", "3", "3");

            // Pizza elemek hozzáadása
            addPizza(doc, rootElement, "1", "1", "Sonkás Pizza",
"paradicsom,mozzarella,sonka", "2000", "17 cm,25 cm,30 cm");
            addPizza(doc, rootElement, "2", "2", "Pizza Mexikói",
"paradicsom,chili,sajt", "2550", "25 cm,32 cm,50 cm");
            addPizza(doc, rootElement, "3", "3", "Aladdin Pizza", "aszalt
paradicsom,ananász,csirkemell", "2750", "17 cm,25 cm,30 cm");

            // Bankkartya elemek hozzáadása
            addBankkartya(doc, rootElement, "1", "1", "OTP", "bankkartya",
"2024-05");
            addBankkartya(doc, rootElement, "2", "2", "ERSTE",
"hitelkartya", "2025-06");
            addBankkartya(doc, rootElement, "3", "3", "UNICREDIT",
"bankkartya", "2026-05");

            // XML fájl írása
            TransformerFactory transformerFactory =
TransformerFactory.newInstance();
            Transformer transformer = transformerFactory.newTransformer();
            transformer.setOutputProperty(OutputKeys.INDENT, "yes");
            DOMSource source = new DOMSource(doc);

            StreamResult fileResult = new StreamResult(new
File("XMLEZ3YRC1.xml"));
            transformer.transform(source, fileResult);
```

```
            System.out.println("The content has been written to the output
file successfully.");
        } catch (ParserConfigurationException | TransformerException e) {
            System.out.println("Valami baj van: " + e);
        }
    }

    // Segédfüggvények az elemek hozzáadásához
    private static void addPizzazo(Document doc, Element root, String
pizzazoId, String nev, String weboldal, String telefonszam) {
        Element pizzazo = doc.createElement("Pizzazo");
        pizzazo.setAttribute("pizzazo_id", pizzazoId);
        root.appendChild(pizzazo);

        Element nevElement = doc.createElement("nev");
        nevElement.appendChild(doc.createTextNode(nev));
        pizzazo.appendChild(nevElement);

        Element elerhetoseg = doc.createElement("elerhetoseg");
        pizzazo.appendChild(elerhetoseg);

        Element weboldalElement = doc.createElement("weboldal");
        weboldalElement.appendChild(doc.createTextNode(weboldal));
        elerhetoseg.appendChild(weboldalElement);

        Element telefonszamElement = doc.createElement("telefonszam");
        telefonszamElement.appendChild(doc.createTextNode(telefonszam));
        elerhetoseg.appendChild(telefonszamElement);
    }

    private static void addBeszallitas(Document doc, Element root, String
beszallito, String pizzazo, String datum, String hozzavalo) {
        Element beszallitas = doc.createElement("Beszallitas");
        beszallitas.setAttribute("beszallito", beszallito);
        beszallitas.setAttribute("pizzazo", pizzazo);
        root.appendChild(beszallitas);

        Element datumElement = doc.createElement("datum");
        datumElement.appendChild(doc.createTextNode(datum));
        beszallitas.appendChild(datumElement);

        Element hozzavaloElement = doc.createElement("hozzavalo");
        hozzavaloElement.appendChild(doc.createTextNode(hozzavalo));
        beszallitas.appendChild(hozzavaloElement);
    }

    private static void addBeszallito(Document doc, Element root, String
beszallitoId, String nev, String email, String varos, String hazszam,
String utca, String iranyitoszam) {
        Element beszallitoElem = doc.createElement("Beszallito");
        beszallitoElem.setAttribute("beszallito_id", beszallitoId);
        root.appendChild(beszallitoElem);

        Element nevElement = doc.createElement("nev");
        nevElement.appendChild(doc.createTextNode(nev));
        beszallitoElem.appendChild(nevElement);

        Element emailElement = doc.createElement("email");
        emailElement.appendChild(doc.createTextNode(email));
        beszallitoElem.appendChild(emailElement);
```

```
        Element cimElem = doc.createElement("cim");
        beszallitoElem.appendChild(cimElem);

        Element varosElement = doc.createElement("varos");
        varosElement.appendChild(doc.createTextNode(varos));
        cimElem.appendChild(varosElement);

        Element hazszamElement = doc.createElement("hazszam");
        hazszamElement.appendChild(doc.createTextNode(hazszam));
        cimElem.appendChild(hazszamElement);

        Element utcaElement = doc.createElement("utca");
        utcaElement.appendChild(doc.createTextNode(utca));
        cimElem.appendChild(utcaElement);

        Element iranyitoszamElement = doc.createElement("iranyitoszam");
        iranyitoszamElement.appendChild(doc.createTextNode(iranyitoszam));
        cimElem.appendChild(iranyitoszamElement);
    }

    private static void addFutar(Document doc, Element root, String
futarId, String pizzazo, String nev, String telefonszam) {
        Element futarElem = doc.createElement("Futar");
        futarElem.setAttribute("futar_id", futarId);
        futarElem.setAttribute("pizzazo", pizzazo);
        root.appendChild(futarElem);

        Element nevElement = doc.createElement("nev");
        nevElement.appendChild(doc.createTextNode(nev));
        futarElem.appendChild(nevElement);

        Element telefonszamElement = doc.createElement("telefonszam");
        telefonszamElement.appendChild(doc.createTextNode(telefonszam));
        futarElem.appendChild(telefonszamElement);
    }

    private static void addVevo(Document doc, Element root, String vevoId,
String nev, String telefonszam, String varos, String hazszam, String utca,
String iranyitoszam) {
        Element vevoElem = doc.createElement("Vevo");
        vevoElem.setAttribute("vevo_id", vevoId);
        root.appendChild(vevoElem);

        Element nevElement = doc.createElement("nev");
        nevElement.appendChild(doc.createTextNode(nev));
        vevoElem.appendChild(nevElement);

        Element telefonszamElement = doc.createElement("telefonszam");
        telefonszamElement.appendChild(doc.createTextNode(telefonszam));
        vevoElem.appendChild(telefonszamElement);

        Element cimElem = doc.createElement("cim");
        vevoElem.appendChild(cimElem);

        Element varosElement = doc.createElement("varos");
        varosElement.appendChild(doc.createTextNode(varos));
        cimElem.appendChild(varosElement);

        Element hazszamElement = doc.createElement("hazszam");
```

```
        hazszamElement.appendChild(doc.createTextNode(hazszam));
        cimElem.appendChild(hazszamElement);

        Element utcaElement = doc.createElement("utca");
        utcaElement.appendChild(doc.createTextNode(utca));
        cimElem.appendChild(utcaElement);

        Element iranyitoszamElement = doc.createElement("iranyitoszam");
        iranyitoszamElement.appendChild(doc.createTextNode(iranyitoszam));
        cimElem.appendChild(iranyitoszamElement);
    }

    private static void addRendeles(Document doc, Element root, String
pizza, String vevo, String rendelesId) {
        Element rendelesElem = doc.createElement("Rendeles");
        rendelesElem.setAttribute("pizza", pizza);
        rendelesElem.setAttribute("vevo", vevo);
        root.appendChild(rendelesElem);
    }

    private static void addPizza(Document doc, Element root, String
pizzaId, String pizzazo, String pizzaneve, String feltetek, String
teljesAr, String meretek) {
        Element pizzaElem = doc.createElement("Pizza");
        pizzaElem.setAttribute("pizza_id", pizzaId);
        pizzaElem.setAttribute("pizzazo", pizzazo);
        root.appendChild(pizzaElem);

        Element pizzaneveElement = doc.createElement("pizzaneve");
        pizzaneveElement.appendChild(doc.createTextNode(pizzaneve));
        pizzaElem.appendChild(pizzaneveElement);

        String[] feltetArray = feltetek.split(",");
        for (String feltet : feltetArray) {
            Element feltetElement = doc.createElement("feltet");
            feltetElement.appendChild(doc.createTextNode(feltet.trim()));
            pizzaElem.appendChild(feltetElement);
        }

        Element teljesArElement = doc.createElement("teljes_ar");
        teljesArElement.appendChild(doc.createTextNode(teljesAr));
        pizzaElem.appendChild(teljesArElement);

        String[] meretArray = meretek.split(",");
        for (String meret : meretArray) {
            Element meretElement = doc.createElement("meret");
            meretElement.appendChild(doc.createTextNode(meret.trim()));
            pizzaElem.appendChild(meretElement);
        }
    }

    private static void addBankkartya(Document doc, Element root, String
kartyaszam, String vevo, String bank, String tipus, String lejaratidatum) {
        Element bankkartyaElem = doc.createElement("Bankkartya");
        bankkartyaElem.setAttribute("kartyaszam", kartyaszam);
        bankkartyaElem.setAttribute("vevo", vevo);
        root.appendChild(bankkartyaElem);

        Element bankElem = doc.createElement("bank");
        bankElem.appendChild(doc.createTextNode(bank));
```

```
        bankkartyaElem.appendChild(bankElem);

        Element tipusElem = doc.createElement("tipus");
        tipusElem.appendChild(doc.createTextNode(tipus));
        bankkartyaElem.appendChild(tipusElem);

        Element lejaratElem = doc.createElement("lejaratidatum");
        lejaratElem.appendChild(doc.createTextNode(lejaratidatum));
        bankkartyaElem.appendChild(lejaratElem);
    }


    public static void addPizzaElem(Document doc, Element root, String
pizzaId, String pizzazo, String pizzaneve, String feltetek, String
teljesAr, String meretek) {
        Element pizzaElem = doc.createElement("Pizza");
        pizzaElem.setAttribute("pizza_id", pizzaId);
        pizzaElem.setAttribute("pizzazo", pizzazo);
        root.appendChild(pizzaElem);

        Element pizzaneveElem = doc.createElement("pizzaneve");
        pizzaneveElem.appendChild(doc.createTextNode(pizzaneve));
        pizzaElem.appendChild(pizzaneveElem);

        String[] feltetArray = feltetek.split(",");
        for (String feltet : feltetArray) {
            Element feltetElem = doc.createElement("feltet");
            feltetElem.appendChild(doc.createTextNode(feltet.trim()));
            pizzaElem.appendChild(feltetElem);
        }

        Element teljesArElem = doc.createElement("teljes_ar");
        teljesArElem.appendChild(doc.createTextNode(teljesAr));
        pizzaElem.appendChild(teljesArElem);

        String[] meretArray = meretek.split(",");
        for (String meret : meretArray) {
            Element meretElem = doc.createElement("meret");
            meretElem.appendChild(doc.createTextNode(meret.trim()));
            pizzaElem.appendChild(meretElem);
        }
    }
}
```