# JEGYZŐKÖNYV

Adatkezelés XML környezetben

Féléves feladat

Bolt

Készítette: **Regecz Márk**

Neptunkód: CNGDZ3

Dátum: 2024.12.09

**A feladat leírása:**

A rendszer lehetővé teszi a bolt teljes adattárolási és -kezelési folyamatainak digitális nyomon követését és kezelését. A vásárlók adatainak tárolása magában foglalja az alapvető személyes adatokat, például nevet, elérhetőségeket, illetve a vásárlási előzményeket. A dolgozók nyilvántartása pedig segíti a munkavállalók adatainak rendszerezését és kezelését, beleértve a beosztásukat és szerepkörüket.

Ezen kívül a termékekhez tartozó típus egyed részletes információkat nyújt a termékek kategorizálásáról, ami lehetővé teszi az árukészlet könnyebb kezelhetőségét. A rendszer képes a rendelési folyamatok részletes dokumentálására, a kapcsolódó dolgozók, vásárlók és termékek adataival összekötve, ezáltal hatékonyabbá téve az adminisztrációt. Az átlátható adatkapcsolatok révén a rendszer kiválóan alkalmas a bolt üzleti folyamatai optimalizálására és az ügyfélélmény növelésére.

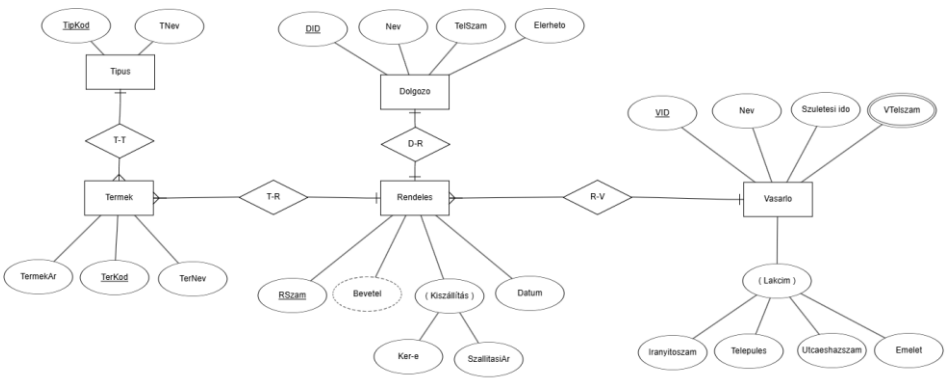
A rendszer emellett lehetőséget biztosít az adatok elemzésére és riportok készítésére is, amelyek segíthetnek az üzleti döntéshozatalban. Például meg lehet vizsgálni a legnépszerűbb termékeket, a legaktívabb vásárlókat, vagy éppen a dolgozók teljesítményét a rendelések alapján. Az adattárolás struktúrája biztosítja, hogy minden információ konzisztensen és könnyen elérhetően legyen kezelve, minimalizálva az adatduplikációt és az esetleges hibákat.

# 1. Bolt

## 1.1 Az adatbázis ER modell tervezése

- Az E-R diagram tisztán mutatja, hogy az adatbázis jól strukturált, a redundancia minimalizálására törekedtem az entitások és kapcsolatok megfelelő felbontásával.
- Az 1:N és N:N kapcsolatok megfelelő kezelése biztosítja, hogy a rendszer rugalmasan kezelje a bolt bővülő adattárait.
- Az attribútumok részletezése (pl. lakcím bontása) megkönnyíti az adatok pontos nyilvántartását.

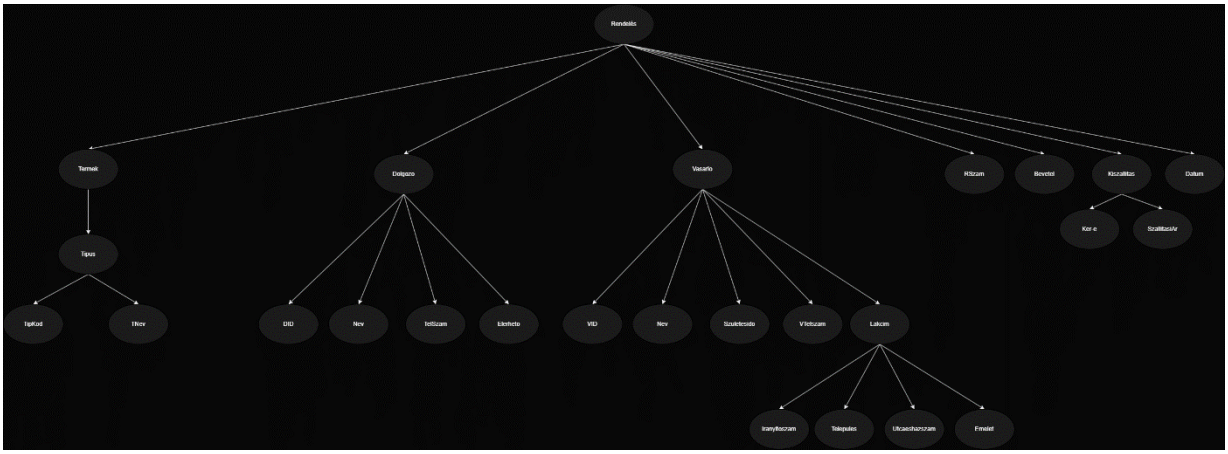
Ez a tervezés lehetőséget biztosít a bolt működésének hatékony menedzselésére, az információk gyors lekérdezésére és az üzleti folyamatok követésére.




1.ábra: ER modell

## 1.2 Az adatbázis konvertálása XDM modellre

Az XDM (XML Data Model) egy olyan struktúra, amely az XML dokumentumokat egy fa-alapú modellként ábrázolja. Az XDM modell segítségével az XML dokumentum hierarchikus felépítését és elemei közötti kapcsolatokat pontosan leírhatjuk.




## 1.3 Az XDM modell alapján XML dokumentum készítése

A projekt célja egy bolt adatainak rendszerezett és átlátható tárolása egy XML-alapú megoldás segítségével. Az XML dokumentum felépítése megfelel a korábban készített E-R diagramnak, amely az üzlet működéséhez kapcsolódó entitásokat és azok közötti kapcsolatokat írja le. Az adatstruktúrában helyet kaptak a termékek, azok típusai, dolgozók, vásárlók, valamint a rendelések. A dokumentumban minden többszörös előfordulású elemről legalább három példány készült, így jól szemlélteti az adatkapcsolatok sokféleségét.

Az XML dokumentumban az egyes részeket megjegyzésekkel láttam el, hogy az entitások és azok attribútumainak szerepe egyértelmű legyen. Minden kapcsolat, például egy rendelés és a hozzá tartozó vásárló, dolgozó, vagy termék, logikusan van modellezve. Az XML formátum lehetővé teszi az adatok további feldolgozását és felhasználását például XSLT vagy XPath segítségével.

```xml
<?xml version="1.0" encoding="utf-8"?>
<Adatbazis xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="XMLSchemaCNGDZ3.xsd">
    <!-- Típusok -->
    <Tipusok>
        <Tipus TipKod="1" TNev="Bútor" />
        <Tipus TipKod="2" TNev="Elektromos eszköz" />
        <Tipus TipKod="3" TNev="Konyhai felszerelés" />
    </Tipusok>

    <!-- Termékek -->
    <Termekek>
        <Termek TerKod="101" TerNev="Szék" TermekAr="5000" TipKod="1" />
        <Termek TerKod="102" TerNev="Asztal" TermekAr="15000" TipKod="1" />
        <Termek TerKod="201" TerNev="Hűtő" TermekAr="80000" TipKod="2" />
        <Termek TerKod="202" TerNev="Sütő" TermekAr="45000" TipKod="3" />
```

```
        <Termek TerKod="203" TerNev="Fazék" TermekAr="12000" TipKod="3" />
    </Termekek>

    <!-- Rendelések -->
    <Rendelesek>
        <Rendeles RSzam="R001" Datum="2024-01-15" Bevetel="20000" DID="1"
VID="1001">
            <Kiszallitas Ker-e="true" SzallitasiAr="1500" />
        </Rendeles>
        <Rendeles RSzam="R002" Datum="2024-01-20" Bevetel="5000" DID="2"
VID="1002">
            <Kiszallitas Ker-e="false" />
        </Rendeles>
        <Rendeles RSzam="R003" Datum="2024-01-25" Bevetel="12500" DID="3"
VID="1003">
            <Kiszallitas Ker-e="true" SzallitasiAr="2000" />
        </Rendeles>
    </Rendelesek>

    <!-- Dolgozók -->
    <Dolgozok>
        <Dolgozo DID="1" Nev="Kovács Béla" TelSzam="+36123456789"
Elerheto="true" />
        <Dolgozo DID="2" Nev="Nagy Anna" TelSzam="+36198765432"
Elerheto="false" />
        <Dolgozo DID="3" Nev="Szabó Péter" TelSzam="+36201234567"
Elerheto="true" />
    </Dolgozok>

    <!-- Vásárlók -->
    <Vasarlok>
        <Vasarlo VID="1001" Nev="Szabó János" SzuletesiIdo="1980-05-15"
VTelSzam="+36201234567">
            <Lakcim>
                <Iranyitoszam>1011</Iranyitoszam>
                <Telepules>Budapest</Telepules>
                <UtcaesHazszam>Fő utca 10</UtcaesHazszam>
                <Emelet>2</Emelet>
            </Lakcim>
        </Vasarlo>
        <Vasarlo VID="1002" Nev="Kiss Éva" SzuletesiIdo="1992-03-22"
VTelSzam="+36207654321">
            <Lakcim>
                <Iranyitoszam>6720</Iranyitoszam>
                <Telepules>Szeged</Telepules>
                <UtcaesHazszam>Kossuth utca 5</UtcaesHazszam>
                <Emelet>1</Emelet>
            </Lakcim>
        </Vasarlo>
```

```
        <Vasarlo VID="1003" Nev="Tóth Gábor" SzuletesiIdo="1985-08-30"
VTelSzam="+36203456789">
            <Lakcim>
                <Iranyitoszam>8200</Iranyitoszam>
                <Telepules>Veszprém</Telepules>
                <UtcaesHazszam>Petőfi utca 12</UtcaesHazszam>
                <Emelet>3</Emelet>
            </Lakcim>
        </Vasarlo>
    </Vasarlok>
</Adatbazis>
```

### 1.4 Az XML dokumentum alapján XMLSchema készítése

Az XML dokumentumhoz készített XMLSchema célja az adatok szerkezetének és integritásának biztosítása. Az XMLSchema a következő funkciókat valósítja meg:

1. **Saját típusok létrehozása**: Az egyes adatelemekhez megfelelő típusok (pl. sztring, szám, dátum) definiálása a pontos érvényesítés érdekében.
2. **key és keyref használata**: Az XML dokumentumban a kulcs- és kulcshivatkozások megvalósítása, például a termékek, dolgozók és vásárlók azonosítói között.
3. **Referenciaelemek használata (`ref`)**: Az elemek struktúrájának újra felhasználhatósága érdekében referenciaelemeken keresztül is meghatározom a mezőket.
4. **Speciális elemek**: Attribútumok, összetett típusok, opcionális és kötelező elemek kezelése, valamint egyedi tartományok beállítása (pl. árak pozitív számok legyenek).

Az XMLSchema megjegyzésekkel van ellátva, amelyek segítenek az egyes részek feladatának megértésében.

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

    <!-- Típusok -->
    <xs:element name="Adatbazis">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="Tipusok">
                    <xs:complexType>
                        <xs:sequence>
                            <xs:element name="Tipus" maxOccurs="unbounded">
                                <xs:complexType>
                                    <xs:attribute name="TipKod"
type="xs:string" use="required"/>
                                    <xs:attribute name="TNev" type="xs:string"
use="required"/>
```

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

            <!-- Termékek -->
            <xs:element name="Termekek">
                <xs:complexType>
                    <xs:sequence>
                        <xs:element name="Termek" maxOccurs="unbounded">
                            <xs:complexType>
                                <xs:attribute name="TerKod"
type="xs:string" use="required"/>
                                <xs:attribute name="TerNev"
type="xs:string" use="required"/>
                                <xs:attribute name="TermekAr"
type="xs:integer" use="required"/>
                                <xs:attribute name="TipKod"
type="xs:string" use="required"/>
                            </xs:complexType>
                        </xs:element>
                    </xs:sequence>
                </xs:complexType>
            </xs:element>

            <!-- Rendelések -->
            <xs:element name="Rendelesek">
                <xs:complexType>
                    <xs:sequence>
                        <xs:element name="Rendeles" maxOccurs="unbounded">
                            <xs:complexType>
                                <xs:sequence>
                                    <xs:element name="Kiszallitas">
                                        <xs:complexType>
                                            <xs:attribute name="Ker-e"
type="xs:boolean" use="required"/>
                                            <xs:attribute
name="SzallitasiAr" type="xs:integer"/>
                                        </xs:complexType>
                                    </xs:element>
                                </xs:sequence>
                                <xs:attribute name="RSzam"
type="xs:string" use="required"/>
                                <xs:attribute name="Datum" type="xs:date"
use="required"/>
                                <xs:attribute name="Bevetel"
type="xs:integer" use="required"/>
```

```
                                    <xs:attribute name="DID" type="xs:string"
use="required"/>
                                    <xs:attribute name="VID" type="xs:string"
use="required"/>
                                </xs:complexType>
                            </xs:element>
                        </xs:sequence>
                    </xs:complexType>
                </xs:element>

                <!-- Dolgozók -->
                <xs:element name="Dolgozok">
                    <xs:complexType>
                        <xs:sequence>
                            <xs:element name="Dolgozo" maxOccurs="unbounded">
                                <xs:complexType>
                                    <xs:attribute name="DID" type="xs:string"
use="required"/>
                                    <xs:attribute name="Nev" type="xs:string"
use="required"/>
                                    <xs:attribute name="TelSzam"
type="xs:string" use="required"/>
                                    <xs:attribute name="Elerheto"
type="xs:boolean" use="required"/>
                                </xs:complexType>
                            </xs:element>
                        </xs:sequence>
                    </xs:complexType>
                </xs:element>

                <!-- Vásárlók -->
                <xs:element name="Vasarlok">
                    <xs:complexType>
                        <xs:sequence>
                            <xs:element name="Vasarlo" maxOccurs="unbounded">
                                <xs:complexType>
                                    <xs:sequence>
                                        <xs:element name="Lakcim">
                                            <xs:complexType>
                                                <xs:sequence>
                                                    <xs:element
name="Iranyitoszam" type="xs:string"/>
                                                    <xs:element
name="Telepules" type="xs:string"/>
                                                    <xs:element
name="UtcaesHazszam" type="xs:string"/>
                                                    <xs:element name="Emelet"
type="xs:integer"/>
                                                </xs:sequence>
```

```
                                        </xs:complexType>
                                    </xs:element>
                                </xs:sequence>
                                <xs:attribute name="VID" type="xs:string"
use="required"/>
                                <xs:attribute name="Nev" type="xs:string"
use="required"/>
                                <xs:attribute name="SzuletesiIdo"
type="xs:date" use="required"/>
                                <xs:attribute name="VTelSzam"
type="xs:string" use="required"/>
                            </xs:complexType>
                        </xs:element>
                    </xs:sequence>
                </xs:complexType>
            </xs:element>
        </xs:sequence>
    </xs:complexType>

    <!-- Key és Keyref deklarációk -->
    <xs:key name="TipusKey">
        <xs:selector xpath="Tipusok/Tipus"/>
        <xs:field xpath="@TipKod"/>
    </xs:key>

    <xs:keyref name="TermekTipusRef" refer="TipusKey">
        <xs:selector xpath="Termekek/Termek"/>
        <xs:field xpath="@TipKod"/>
    </xs:keyref>

    <xs:key name="DolgozoKey">
        <xs:selector xpath="Dolgozok/Dolgozo"/>
        <xs:field xpath="@DID"/>
    </xs:key>

    <xs:keyref name="RendelesDolgozoRef" refer="DolgozoKey">
        <xs:selector xpath="Rendelesek/Rendeles"/>
        <xs:field xpath="@DID"/>
    </xs:keyref>

    <xs:key name="VasarloKey">
        <xs:selector xpath="Vasarlok/Vasarlo"/>
        <xs:field xpath="@VID"/>
    </xs:key>

    <xs:keyref name="RendelesVasarloRef" refer="VasarloKey">
        <xs:selector xpath="Rendelesek/Rendeles"/>
        <xs:field xpath="@VID"/>
    </xs:keyref>
```

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

# 2. Feladat

## 2a)

Az alábbi Java kód egy XML dokumentum DOM (Document Object Model) alapú beolvasását és feldolgozását végzi. Az alkalmazás célja az XML dokumentumban található adatok hierarchikus struktúrájának beolvasása, és az egyes elemek értékeinek kiírása.

```
3. package hu.domparse.cngdz3;
4.
5. import java.io.File;
6. import javax.xml.parsers.DocumentBuilder;
7. import javax.xml.parsers.DocumentBuilderFactory;
8. import org.w3c.dom.Document;
9. import org.w3c.dom.Element;
10.import org.w3c.dom.Node;
11.import org.w3c.dom.NodeList;
12.
13.public class DOMReadCNGDZ3 {
14.
15.    public static void main(String[] args) {
16.        try {
17.            // Letrehozzuk a DocumentBuilderFactory-t es a
   DocumentBuilder-t
18.            DocumentBuilderFactory factory =
   DocumentBuilderFactory.newInstance();
19.            DocumentBuilder builder = factory.newDocumentBuilder();
20.
21.            // Beolvassuk az XML fajlt
22.            File file = new File("XMLCNGDZ3.xml");
23.            Document document = builder.parse(file);
24.
25.            // Normalizaljuk az XML strukturat
26.            document.getDocumentElement().normalize();
27.
28.            // Kiirjuk a gyokerelem nevet
29.            System.out.println("Root element: " +
   document.getDocumentElement().getNodeName());
30.
31.            // Beolvassuk es kiirjuk a Tipusok elemeket
32.            NodeList tipusokList =
   document.getElementsByTagName("Tipus");
33.            System.out.println("Tipusok:");
34.            for (int i = 0; i < tipusokList.getLength(); i++) {
35.                Node node = tipusokList.item(i);
```

```
36.                 if (node.getNodeType() == Node.ELEMENT_NODE) {
37.                     Element elem = (Element) node;
38.                     System.out.println("TipKod: " +
   elem.getAttribute("TipKod"));
39.                     System.out.println("TNev: " +
   elem.getAttribute("TNev"));
40.                 }
41.             }
42.
43.             // Beolvassuk es kiirjuk a Termekek elemeket
44.             NodeList termekekList =
   document.getElementsByTagName("Termek");
45.             System.out.println("Termekek:");
46.             for (int i = 0; i < termekekList.getLength(); i++) {
47.                 Node node = termekekList.item(i);
48.                 if (node.getNodeType() == Node.ELEMENT_NODE) {
49.                     Element elem = (Element) node;
50.                     System.out.println("TerKod: " +
   elem.getAttribute("TerKod"));
51.                     System.out.println("TerNev: " +
   elem.getAttribute("TerNev"));
52.                     System.out.println("TermekAr: " +
   elem.getAttribute("TermekAr"));
53.                     System.out.println("TipKod: " +
   elem.getAttribute("TipKod"));
54.                 }
55.             }
56.
57.             // Beolvassuk es kiirjuk a Rendelesek elemeket
58.             NodeList rendelesekList =
   document.getElementsByTagName("Rendeles");
59.             System.out.println("Rendelesek:");
60.             for (int i = 0; i < rendelesekList.getLength(); i++) {
61.                 Node node = rendelesekList.item(i);
62.                 if (node.getNodeType() == Node.ELEMENT_NODE) {
63.                     Element elem = (Element) node;
64.                     System.out.println("RSzam: " +
   elem.getAttribute("RSzam"));
65.                     System.out.println("Datum: " +
   elem.getAttribute("Datum"));
66.                     System.out.println("Bevetel: " +
   elem.getAttribute("Bevetel"));
67.                     System.out.println("DID: " +
   elem.getAttribute("DID"));
68.                     System.out.println("VID: " +
   elem.getAttribute("VID"));
69.
70.                     // Beolvassuk es kiirjuk a Kiszallitas elemeket
```

```
71.                     Element kiszallitas = (Element)
   elem.getElementsByTagName("Kiszallitas").item(0);
72.                     System.out.println("Kiszallitas Ker-e: " +
   kiszallitas.getAttribute("Ker-e"));
73.                     if (kiszallitas.hasAttribute("SzallitasiAr")) {
74.                         System.out.println("SzallitasiAr: " +
   kiszallitas.getAttribute("SzallitasiAr"));
75.                     }
76.                 }
77.             }
78.
79.             // Beolvassuk es kiirjuk a Dolgozok elemeket
80.             NodeList dolgozokList =
   document.getElementsByTagName("Dolgozo");
81.             System.out.println("Dolgozok:");
82.             for (int i = 0; i < dolgozokList.getLength(); i++) {
83.                 Node node = dolgozokList.item(i);
84.                 if (node.getNodeType() == Node.ELEMENT_NODE) {
85.                     Element elem = (Element) node;
86.                     System.out.println("DID: " +
   elem.getAttribute("DID"));
87.                     System.out.println("Nev: " +
   elem.getAttribute("Nev"));
88.                     System.out.println("TelSzam: " +
   elem.getAttribute("TelSzam"));
89.                     System.out.println("Elerheto: " +
   elem.getAttribute("Elerheto"));
90.                 }
91.             }
92.
93.             // Beolvassuk es kiirjuk a Vasarlok elemeket
94.             NodeList vasarlokList =
   document.getElementsByTagName("Vasarlo");
95.             System.out.println("Vasarlok:");
96.             for (int i = 0; i < vasarlokList.getLength(); i++) {
97.                 Node node = vasarlokList.item(i);
98.                 if (node.getNodeType() == Node.ELEMENT_NODE) {
99.                     Element elem = (Element) node;
100.                        System.out.println("VID: " +
   elem.getAttribute("VID"));
101.                        System.out.println("Nev: " +
   elem.getAttribute("Nev"));
102.                        System.out.println("SzuletesiIdo: " +
   elem.getAttribute("SzuletesiIdo"));
103.                        System.out.println("VTelSzam: " +
   elem.getAttribute("VTelSzam"));
104.
105.                        // Beolvassuk es kiirjuk a Lakcim elemeket
```

```
                        Element lakcim = (Element)
   elem.getElementsByTagName("Lakcim").item(0);
                        System.out.println("Iranyitoszam: " +
   lakcim.getElementsByTagName("Iranyitoszam").item(0).getTextContent());
                        System.out.println("Telepules: " +
   lakcim.getElementsByTagName("Telepules").item(0).getTextContent());
                        System.out.println("UtcaesHazszam: " +
   lakcim.getElementsByTagName("UtcaesHazszam").item(0).getTextContent());
                        System.out.println("Emelet: " +
   lakcim.getElementsByTagName("Emelet").item(0).getTextContent());
                    }
                }

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

## 2b)

Ez a Java program egy meglévő XML fájlt módosít DOM használatával. Beolvasom az XML-t, és végig megyek az összes "Rendeles" elemen. Ha találok egy adott feltételnek megfelelő elemet (pl. "RSzam" attribútum értéke "R001"), módosítom annak egy attribútumát, jelen esetben a "Bevetel" értékét. A módosításokat egy új fájlba mentem el "XMLCNGDZ3_modified.xml" néven. A célom, hogy az XML tartalmát egyszerűen és hatékonyan szerkesszem a program segítségével.

```
package hu.domparse.cngdz3;

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

public class DOMReadCNGDZ3 {

    public static void main(String[] args) {
        try {
            // Letrehozzuk a DocumentBuilderFactory-t es a DocumentBuilder-t
            DocumentBuilderFactory factory =
DocumentBuilderFactory.newInstance();
            DocumentBuilder builder = factory.newDocumentBuilder();

            // Beolvassuk az XML fajlt
            File file = new File("XMLCNGDZ3.xml");
```

```
            Document document = builder.parse(file);

            // Normalizaljuk az XML strukturat
            document.getDocumentElement().normalize();

            // Kiirjuk a gyokerelem nevet
            System.out.println("Root element: " +
document.getDocumentElement().getNodeName());

            // Beolvassuk es kiirjuk a Tipusok elemeket
            NodeList tipusokList = document.getElementsByTagName("Tipus");
            System.out.println("Tipusok:");
            for (int i = 0; i < tipusokList.getLength(); i++) {
                Node node = tipusokList.item(i);
                if (node.getNodeType() == Node.ELEMENT_NODE) {
                    Element elem = (Element) node;
                    System.out.println("TipKod: " +
elem.getAttribute("TipKod"));
                    System.out.println("TNev: " + elem.getAttribute("TNev"));
                }
            }

            // Beolvassuk es kiirjuk a Termekek elemeket
            NodeList termekekList = document.getElementsByTagName("Termek");
            System.out.println("Termekek:");
            for (int i = 0; i < termekekList.getLength(); i++) {
                Node node = termekekList.item(i);
                if (node.getNodeType() == Node.ELEMENT_NODE) {
                    Element elem = (Element) node;
                    System.out.println("TerKod: " +
elem.getAttribute("TerKod"));
                    System.out.println("TerNev: " +
elem.getAttribute("TerNev"));
                    System.out.println("TermekAr: " +
elem.getAttribute("TermekAr"));
                    System.out.println("TipKod: " +
elem.getAttribute("TipKod"));
                }
            }

            // Beolvassuk es kiirjuk a Rendelesek elemeket
            NodeList rendelesekList =
document.getElementsByTagName("Rendeles");
            System.out.println("Rendelesek:");
            for (int i = 0; i < rendelesekList.getLength(); i++) {
                Node node = rendelesekList.item(i);
                if (node.getNodeType() == Node.ELEMENT_NODE) {
                    Element elem = (Element) node;
```

```
                    System.out.println("RSzam: " +
elem.getAttribute("RSzam"));
                    System.out.println("Datum: " +
elem.getAttribute("Datum"));
                    System.out.println("Bevetel: " +
elem.getAttribute("Bevetel"));
                    System.out.println("DID: " + elem.getAttribute("DID"));
                    System.out.println("VID: " + elem.getAttribute("VID"));

                    // Beolvassuk es kiirjuk a Kiszallitas elemeket
                    Element kiszallitas = (Element)
elem.getElementsByTagName("Kiszallitas").item(0);
                    System.out.println("Kiszallitas Ker-e: " +
kiszallitas.getAttribute("Ker-e"));
                    if (kiszallitas.hasAttribute("SzallitasiAr")) {
                        System.out.println("SzallitasiAr: " +
kiszallitas.getAttribute("SzallitasiAr"));
                    }
                }
            }

            // Beolvassuk es kiirjuk a Dolgozok elemeket
            NodeList dolgozokList = document.getElementsByTagName("Dolgozo");
            System.out.println("Dolgozok:");
            for (int i = 0; i < dolgozokList.getLength(); i++) {
                Node node = dolgozokList.item(i);
                if (node.getNodeType() == Node.ELEMENT_NODE) {
                    Element elem = (Element) node;
                    System.out.println("DID: " + elem.getAttribute("DID"));
                    System.out.println("Nev: " + elem.getAttribute("Nev"));
                    System.out.println("TelSzam: " +
elem.getAttribute("TelSzam"));
                    System.out.println("Elerheto: " +
elem.getAttribute("Elerheto"));
                }
            }

            // Beolvassuk es kiirjuk a Vasarlok elemeket
            NodeList vasarlokList = document.getElementsByTagName("Vasarlo");
            System.out.println("Vasarlok:");
            for (int i = 0; i < vasarlokList.getLength(); i++) {
                Node node = vasarlokList.item(i);
                if (node.getNodeType() == Node.ELEMENT_NODE) {
                    Element elem = (Element) node;
                    System.out.println("VID: " + elem.getAttribute("VID"));
                    System.out.println("Nev: " + elem.getAttribute("Nev"));
                    System.out.println("SzuletesiIdo: " +
elem.getAttribute("SzuletesiIdo"));
```

```
                    System.out.println("VTelSzam: " +
elem.getAttribute("VTelSzam"));

                    // Beolvassuk es kiirjuk a Lakcim elemeket
                    Element lakcim = (Element)
elem.getElementsByTagName("Lakcim").item(0);
                    System.out.println("Iranyitoszam: " +
lakcim.getElementsByTagName("Iranyitoszam").item(0).getTextContent());
                    System.out.println("Telepules: " +
lakcim.getElementsByTagName("Telepules").item(0).getTextContent());
                    System.out.println("UtcaesHazszam: " +
lakcim.getElementsByTagName("UtcaesHazszam").item(0).getTextContent());
                    System.out.println("Emelet: " +
lakcim.getElementsByTagName("Emelet").item(0).getTextContent());
                }
            }

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

## 2c)

Az alábbi program egy XML fájlból hajt végre lekérdezéseket DOM segítségével. Beolvasom az XML tartalmát, majd az összes "Rendeles" elemen végigmegyek, hogy megtaláljam azokat, amelyeknél a "Kiszallitas" elem "Ker-e" attribútuma "true". Ha ilyen rendelést találok, akkor kiírom az adott rendelés részleteit, beleértve az azonosítót, a dátumot, a bevételt, a dolgozó és vásárló ID-ját, valamint a szállítási árat. A célom ezzel a programmal az, hogy egy adott feltétel alapján kiemeljem és megjelenítsem az XML releváns adatait.

```
package hu.domparse.cngdz3;

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

public class DOMQueryCNGDZ3 {

    public static void main(String[] args) {
        try {
            // Letrehozzuk a DocumentBuilderFactory-t es a DocumentBuilder-t
            DocumentBuilderFactory factory =
DocumentBuilderFactory.newInstance();
            DocumentBuilder builder = factory.newDocumentBuilder();
```

```
            // Beolvassuk az XML fajlt
            File file = new File("XMLCNGDZ3.xml");
            Document document = builder.parse(file);

            // Normalizaljuk az XML strukturat
            document.getDocumentElement().normalize();

            // Kiirjuk a gyokerelem nevet
            System.out.println("Root element: " +
document.getDocumentElement().getNodeName());

            // Beolvassuk az osszes "Rendeles" elemet
            NodeList rendelesekList =
document.getElementsByTagName("Rendeles");

            // Vegigmegyunk az osszes "Rendeles" elemen
            System.out.println("Rendelesek, amelyek kiszallitasa igen:");
            for (int i = 0; i < rendelesekList.getLength(); i++) {
                Node node = rendelesekList.item(i);

                if (node.getNodeType() == Node.ELEMENT_NODE) {
                    Element elem = (Element) node;

                    // Beolvassuk a "Kiszallitas" elemet
                    Element kiszallitas = (Element)
elem.getElementsByTagName("Kiszallitas").item(0);
                    String kerE = kiszallitas.getAttribute("Ker-e");

                    // Ha a "Kiszallitas" eleme "true", akkor kiirjuk a
rendeles adatait
                    if ("true".equals(kerE)) {
                        System.out.println("RSzam: " +
elem.getAttribute("RSzam"));
                        System.out.println("Datum: " +
elem.getAttribute("Datum"));
                        System.out.println("Bevetel: " +
elem.getAttribute("Bevetel"));
                        System.out.println("DID: " +
elem.getAttribute("DID"));
                        System.out.println("VID: " +
elem.getAttribute("VID"));
                        System.out.println("SzallitasiAr: " +
kiszallitas.getAttribute("SzallitasiAr"));
                        System.out.println("----------------------------------
-");
                    }
                }
            }
```

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

## 2d)

Az alábbi program a DOM API használatával beolvassa egy XML fájl tartalmát, majd rekurzívan kiírja az XML fa struktúráját a konzolra, beleértve az elemek nevét és szöveges értékeit. Az XML dokumentumot az eredeti struktúrával együtt elmentem egy új fájlba, "XMLCNGDZ31.xml" néven. A program hasznos az XML fájlok tartalmának megértéséhez és másolásához, miközben rekurzív megközelítést alkalmaz az XML hierarchia feldolgozására.

```
package hu.domparse.cngdz3;

import java.io.File;
import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.DocumentBuilderFactory;
import javax.xml.transform.Transformer;
import javax.xml.transform.TransformerFactory;
import javax.xml.transform.dom.DOMSource;
import javax.xml.transform.stream.StreamResult;
import org.w3c.dom.Document;
import org.w3c.dom.Node;
import org.w3c.dom.NodeList;

public class DOMWriteCNGDZ3 {

    public static void main(String[] args) {
        try {
            // Letrehozzuk a DocumentBuilderFactory-t es a DocumentBuilder-t
            DocumentBuilderFactory factory =
DocumentBuilderFactory.newInstance();
            DocumentBuilder builder = factory.newDocumentBuilder();

            // Beolvassuk az XML fajlt
            File file = new File("XMLCNGDZ3.xml");
            Document document = builder.parse(file);

            // Normalizaljuk az XML strukturat
            document.getDocumentElement().normalize();

            // Kiirjuk a gyokerelem nevet
            System.out.println("Root element: " +
document.getDocumentElement().getNodeName());
```

```
            // Kiirjuk az XML fa strukturajat
            printNode(document.getDocumentElement(), "");

            // Az XML fajl tartalmanak mentese egy uj fajlba
            TransformerFactory transformerFactory =
TransformerFactory.newInstance();
            Transformer transformer = transformerFactory.newTransformer();
            DOMSource source = new DOMSource(document);
            StreamResult result = new StreamResult(new
File("XMLCNGDZ31.xml"));
            transformer.transform(source, result);

            System.out.println("Az XML fajl tartalma elmentve az
XMLCNGDZ31.xml fajlba.");

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

    // Rekurziv fuggveny az XML fa strukturajanak kiirasahoz
    private static void printNode(Node node, String indent) {
        System.out.println(indent + "Node: " + node.getNodeName() + ", Value:
" + node.getTextContent().trim());

        NodeList nodeList = node.getChildNodes();
        for (int i = 0; i < nodeList.getLength(); i++) {
            Node childNode = nodeList.item(i);
            if (childNode.getNodeType() == Node.ELEMENT_NODE) {
                printNode(childNode, indent + "  ");
            }
        }
    }
}
```