

Jegyzőkönyv

Adatkezelés XML környezetben

Féléves feladat

Szabó Bálint

EJX162

Pizzéria hálózat modellezése

Tartalomjegyzék

Pizzéria hálózat modellezése.....	1
A feladat leírása:.....	1
ER Modell:	3
XDM Modell:.....	3
XML Dokumentum:	4
XML Schema:	9
DOM Read:	12
Dom Read Output:.....	13
DOM Modify Output:	19
DOM Query:	22
DOM Query Output:	23

1. Feladat

A feladat leírása:

A feladatomban egy 4 egyedből álló adatbázis modellt szeretnék bemutatni, amely a tulajdonosok, pizzériák, pizzák és vevők kapcsolatáról szól.

A **Pizzéria** egyedhez tartozik a **Pikód** tulajdonság, amely az elsődleges kulcsa az egyednek. Ezen kívül még a **Telefonszám**, amely egy többértékű tulajdonsága, hiszem egy Pizzériának lehet több telefonszáma is, például egy vezetékes és egy

mobil. Továbbá a **Név** tulajdonság és a **Cím**, amely egy összetett tulajdonság, hiszen az Irszám, a Város, utca és házszám elemei vannak.

A **Pizza** egyed **Pkód** tulajdonsága kulcsként funkcionál, másik két tulajdonsága a **Név** és az **Ár**, valamint a **Feltét**, amely többértékű tulajdonság, hiszen egy pizzán lehet egyszerre sajt is a feltét meg szalámi is.

A **Tulajdonos** egyed **Tkód** tulajdonsága egyedi, számjegyekből álló azonosítója, azaz kulcsként funkcionál. Van egy származtatott tulajdonsága, az **Életkor**, amelyet a **Szül.idő** tulajdonságból származtathatunk hiszen, ha a jelenlegi dátumból kivonjuk a születési dátumot, akkor megkapjuk a Tulajdonos életkorát.

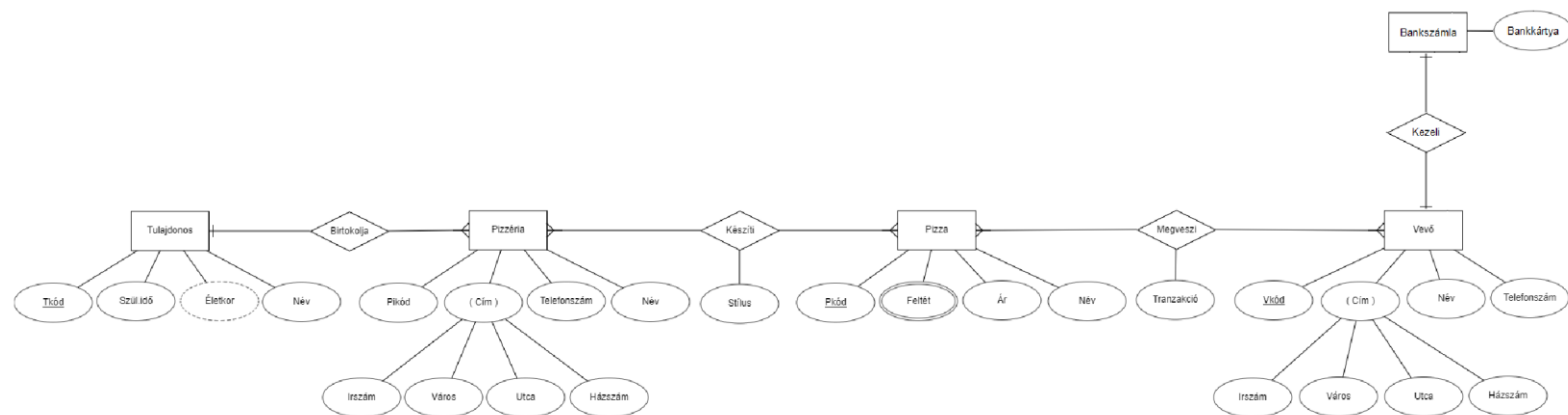
A **Vevő** egyed elsődleges kulcsa a **Vkód**, amely egyértelműen azonosítja a vevőt. Másik két tulajdonsága a **Név** és a **Telefonszám**, ezek egyszerű tulajdonságok. Végül pedig a **Cím** tulajdonság, amely egy összetett tulajdonság, az Irszámból, Városból, Utcából és a Házámból tevődik össze.

A **Pizzéria** és a **Pizza** egyedek között a **Készíti** kapcsolat van, amely egy egy **több a többhöz kapcsolat (N:M)**, hiszen egy pizzéria több pizzát is készíthet, és egy pizzát több pizzéria is el tud készíteni.

A **Pizzéria** és a **Tulajdonos** között **egy a többhöz (1:N)** kapcsolat van, amit a **Birtokolja** jelöl, mert egy pizzériának több tulajdonosa is lehet, viszont az én modellemben egy pizzériának csak egy tulajdonosa lehet.

A **Pizza** és a **Vevő** egyedek között **Megveszi** kapcsolat van, ami szintén egy **több a többhöz kapcsolat (N:M)**, mivel egy féle pizzát több vevő is megvehet, valamint egy vevő több pizzériától is vásárolhat.

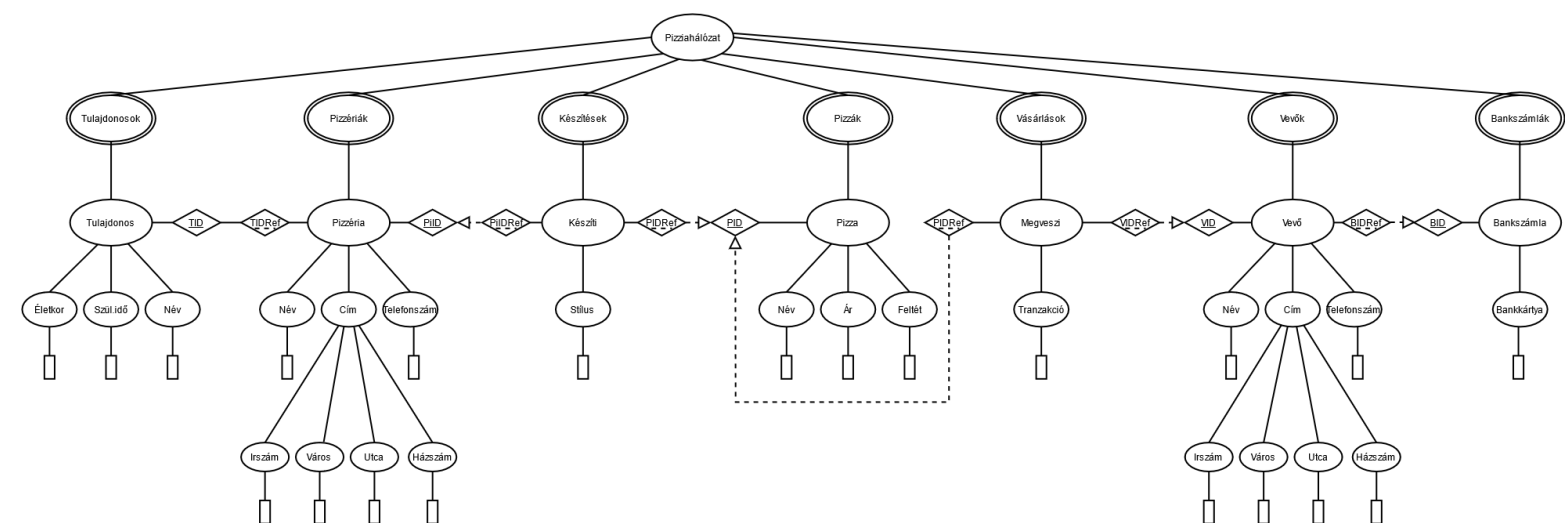
ER Modell:



XDM Modell:

Az ER modell XDM modellre való konvertálása után létrejöttek idegen kulcsok, ezek a TIDRef, PIDRef, PiIDRef és a VIDRef, ezek a hozzájuk tartozó elsődleges kulcsra mutatnak.

Ezen kívül a két több-több (N:M) kapcsolatnál létrejött két objektum. A Pizzériák és a Pizzák objektum közötti kapcsolatnál létrejött a Készítések objektum, valamint a Pizzák és a Vevők között a Vásárlások objektum.



Az XDM modell alapján XML Dokumentum készítése:

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <Pizzahalozat xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
3   xsi:noNamespaceSchemaLocation="XMLSchemaEJX162.xsd">
4
5   <Tulajdonosok>
6
7     <Tulajdonos TID="1">
8       <Nev>Nagy Béla</Nev>
9       <Eletkor>21</Eletkor>
10      <Szulido>1997</Szulido>
11    </Tulajdonos>
12
13    <Tulajdonos TID="2">
14      <Nev>Lőre István</Nev>
15      <Eletkor>27</Eletkor>
16      <Szulido>1998</Szulido>
17    </Tulajdonos>
18
19    <Tulajdonos TID="3">
20      <Nev>Kiss Imre</Nev>
21      <Eletkor>45</Eletkor>
22      <Szulido>2000</Szulido>
23    </Tulajdonos>
24  </Tulajdonosok>
25
26  <Pizzeriak>
27    <Pizzeria PiID="1" TIDRef="1">
28      <Nev>Italian Stallion</Nev>
29      <Telefonszam>012432123</Telefonszam>
30      <Cim>
31        <Irszam>1256</Irszam>
32        <Varos>Budapest</Varos>
33        <Utca>Ferencziek útja</Utca>
34        <Hazszam>14</Hazszam>
35      </Cim>
36    </Pizzeria>
37  </Pizzeriak>
38</Pizzahalozat>
```

```
37         </Pizzeria>
38
39     <Pizzeria PiID="2" TIDRef="2">
40         <Nev>New York Style</Nev>
41         <Telefonszam>05618478</Telefonszam>
42     <Cim>
43         <Irszam>3525</Irszam>
44         <Varos>Miskolc</Varos>
45         <Utca>Széchenyi István út</Utca>
46         <Hazzsam>26</Hazzsam>
47     </Cim>
48 </Pizzeria>
49
50 <Pizzeria PiID="3" TIDRef="3">
51     <Nev>Lábas</Nev>
52     <Telefonszam>035637923</Telefonszam>
53 <Cim>
54     <Irszam>3540</Irszam>
55     <Varos>Alsózsolca</Varos>
56     <Utca>Fő út</Utca>
57     <Hazzsam>6</Hazzsam>
58 </Cim>
59 </Pizzeria>
60 </Pizzeriak>
61
62 <Keszitesek>
63     <Kesziti PIDRef="1" PiIDRef="1">
64         <Stilus>Al Dente</Stilus>
65     </Kesziti>
66
67     <Kesziti PIDRef="2" PiIDRef="2">
68         <Stilus>Közepes</Stilus>
```

```
69         </Kesziti>
70
71     <Kesziti PIDRef="3" PiIDRef="3">
72         <Stilus>Pirított</Stilus>
73     </Kesziti>
74 </Keszitesek>
75
76 <Pizzak>
77     <Pizza PID="1">
78         <Nev>Margareta</Nev>
79         <Ar>1200</Ar>
80         <Feltet>Sajt</Feltet>
81     </Pizza>
82
83     <Pizza PID="2">
84         <Nev>Sonkás</Nev>
85         <Ar>1400</Ar>
86         <Feltet>Sajt</Feltet>
87         <Feltet>Sonka</Feltet>
88     </Pizza>
89
90     <Pizza PID="3">
91         <Nev>Szalámis</Nev>
92         <Ar>1400</Ar>
93         <Feltet>Sajt</Feltet>
94         <Feltet>Szalámi</Feltet>
95     </Pizza>
96
97     <Pizza PID="4">
98         <Nev>Sonkás-Kukoricás</Nev>
99         <Ar>1600</Ar>
100        <Feltet>Sajt</Feltet>
101        <Feltet>Sonka</Feltet>
```

```
102         <Feltet>Kukorica</Feltet>
103     </Pizza>
104
105     <Pizza PID="5">
106         <Nev>Lábas Speciál</Nev>
107         <Ar>2100</Ar>
108         <Feltet>Sajt</Feltet>
109         <Feltet>BBQ szósz</Feltet>
110         <Feltet>Pulled Pork</Feltet>
111         <Feltet>Bacon</Feltet>
112     </Pizza>
113 </Pizzak>
114
115 <Vasarlasok>
116     <Megveszi PIDRef="1" VIDRef="1">
117         <Tranzakcio>123</Tranzakcio>
118     </Megveszi>
119
120     <Megveszi PIDRef="2" VIDRef="2">
121         <Tranzakcio>456</Tranzakcio>
122     </Megveszi>
123
124     <Megveszi PIDRef="3" VIDRef="3">
125         <Tranzakcio>789</Tranzakcio>
126     </Megveszi>
127 </Vasarlasok>
128
129 <Vevok>
130     <Vevo VID="1">
131         <Nev>Nagy Ferenc</Nev>
132         <Telefonszam>77895412</Telefonszam>
133     <Cim>
134         <Irszam>3519</Irszam>
```

```
135         <Varos>Miskolc</Varos>
136         <Utca>Fenyő utca</Utca>
137         <Hazzsam>23</Hazzsam>
138     </Cim>
139 </Vevo>
140
141 <Vevo VID="2">
142     <Nev>Novák Balázs</Nev>
143     <Telefonszam>078954412</Telefonszam>
144 <Cim>
145     <Irszam>1149</Irszam>
146     <Varos>Budapest</Varos>
147     <Utca>Vezér utca</Utca>
148     <Hazzsam>149</Hazzsam>
149 </Cim>
150 </Vevo>
151
152 <Vevo VID="3">
153     <Nev>Trab Antal</Nev>
154     <Telefonszam>012258</Telefonszam>
155 <Cim>
156     <Irszam>3530</Irszam>
157     <Varos>Miskolc</Varos>
158     <Utca>Pattantyús utca</Utca>
159     <Hazzsam>14</Hazzsam>
160 </Cim>
161 </Vevo>
162 </Vevok>
163 </Pizzahalozat>
```


Az XML dokumentum alapján XML Schema készítése:

```
1 <?xml version="1.0" encoding="UTF-8"?>
2◉ <xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
3   elementFormDefault="qualified"
4   attributeFormDefault="qualified">
5
6   <xs:attribute name="TID" type="xs:integer"/>
7   <xs:attribute name="TIDRef" type="xs:integer"/>
8   <xs:attribute name="PiID" type="xs:integer"/>
9   <xs:attribute name="PiIDRef" type="xs:integer"/>
10  <xs:attribute name="PID" type="xs:integer"/>
11  <xs:attribute name="PIDRef" type="xs:integer"/>
12  <xs:attribute name="VID" type="xs:integer"/>
13  <xs:attribute name="VIDRef" type="xs:integer"/>
14
15◉ <xs:complexType name="Tulajdonos_tipus">
16◉   <xs:sequence>
17     <xs:element name="Nev" type="xs:string"/>
18     <xs:element name="Eletkor" type="xs:integer"/>
19     <xs:element name="Szulido" type="xs:integer"/>
20
21   </xs:sequence>
22   <xs:attribute ref="TID" use="required"/>
23 </xs:complexType>
24
25◉ <xs:complexType name="Pizzeria_tipus">
26◉   <xs:sequence>
27     <xs:element name="Nev" type="xs:string"/>
28     <xs:element name="Telefonszam" type="xs:integer"/>
29     <xs:element name="Cim" type="Cim_tipus" maxOccurs="1"/>
30   </xs:sequence>
31   <xs:attribute ref="PiID" use="required"/>
32   <xs:attribute ref="TIDRef" use="required"/>
33 </xs:complexType>
34
35◉ <xs:complexType name="Cim_tipus">
36◉   <xs:sequence>
```

```

37         <xs:element name="Irszam" type="xs:string"/>
38         <xs:element name="Varos" type="xs:string"/>
39         <xs:element name="Utca" type="xs:string"/>
40         <xs:element name="Hazzsam" type="xs:integer"/>
41     </xs:sequence>
42 </xs:complexType>
43
44 <xs:complexType name="Kesziti_tipus">
45     <xs:sequence>
46         <xs:element name="Stilus" type="xs:string"/>
47     </xs:sequence>
48     <xs:attribute ref="PIDRef" use="required"/>
49     <xs:attribute ref="PiIDRef" use="required"/>
50 </xs:complexType>
51
52 <xs:complexType name="Pizza_tipus">
53     <xs:sequence>
54         <xs:element name="Nev" type="xs:string"/>
55         <xs:element name="Ar" type="xs:integer"/>
56         <xs:element name="Feltet" type="xs:string" maxOccurs="unbounded"/>
57     </xs:sequence>
58     <xs:attribute ref="PID" use="required"/>
59 </xs:complexType>
60
61 <xs:complexType name="Megveszi_tipus">
62     <xs:sequence>
63         <xs:element name="Tranzakcio" type="xs:integer"/>
64     </xs:sequence>
65     <xs:attribute ref="PIDRef" use="required"/>
66     <xs:attribute ref="VIDRef" use="required"/>
67 </xs:complexType>
68
69 <xs:complexType name="Vevo_tipus">
70     <xs:sequence>
71         <xs:element name="Nev" type="xs:string"/>
72         <xs:element name="Telefonszam" type="xs:integer"/>
73         <xs:element name="Cim" type="Cim_tipus" maxOccurs="1"/>
74     </xs:sequence>
75     <xs:attribute ref="VID" use="required"/>
76 </xs:complexType>
77
78 <xs:complexType name="Tulajdonosok_tipus">
79     <xs:sequence>
80         <xs:element name="Tulajdonos" type="Tulajdonos_tipus" maxOccurs="unbounded"/>
81     </xs:sequence>
82 </xs:complexType>
83
84 <xs:complexType name="Pizzeriak_tipus">
85     <xs:sequence>
86         <xs:element name="Pizzeria" type="Pizzeria_tipus" maxOccurs="unbounded"/>
87     </xs:sequence>
88 </xs:complexType>
89
90 <xs:complexType name="Keszitesek_tipus">
91     <xs:sequence>
92         <xs:element name="Kesziti" type="Kesziti_tipus" maxOccurs="unbounded"/>
93     </xs:sequence>
94 </xs:complexType>
95
96 <xs:complexType name="Pizzak_tipus">
97     <xs:sequence>
98         <xs:element name="Pizza" type="Pizza_tipus" maxOccurs="unbounded"/>
99     </xs:sequence>
100 </xs:complexType>
101
102 <xs:complexType name="Vasarlasok_tipus">
103     <xs:sequence>
104         <xs:element name="Megveszi" type="Megveszi_tipus" maxOccurs="unbounded"/>

```

```

104     <xs:element name="Megveszi" type="Megveszi_tipus" maxOccurs="unbounded"/>
105   </xs:sequence>
106 </xs:complexType>
107
108 <xs:complexType name="Vevok_tipus">
109   <xs:sequence>
110     <xs:element name="Vevo" type="Vevo_tipus" maxOccurs="unbounded"/>
111   </xs:sequence>
112 </xs:complexType>
113
114 <xs:element name="Pizzahalozat">
115   <xs:complexType>
116     <xs:sequence>
117       <xs:element name="Tulajdonosok" type="Tulajdonosok_tipus"/>
118       <xs:element name="Pizzeriak" type="Pizzeriak_tipus"/>
119       <xs:element name="Keszitesek" type="Keszitesek_tipus"/>
120       <xs:element name="Pizzak" type="Pizzak_tipus"/>
121       <xs:element name="Vasarlasok" type="Vasarlasok_tipus"/>
122       <xs:element name="Vevok" type="Vevok_tipus"/>
123     </xs:sequence>
124   </xs:complexType>
125
126 <xs:key name="Tulajdonos_EKulcs">
127   <xs:selector xpath="Tulajdonosok/Tulajdonos"/>
128   <xs:field xpath="@TID"/>
129 </xs:key>
130
131 <xs:key name="Pizzeria_EKulcs">
132   <xs:selector xpath="Pizzeriak/Pizzeria"/>
133   <xs:field xpath="@PiID"/>
134 </xs:key>
135
136 <xs:key name="Pizza_EKulcs">
137   <xs:selector xpath="Pizzak/Pizza"/>

```

```

138     <xs:field xpath="@PID"/>
139   </xs:key>
140
141 <xs:key name="Vevo_EKulcs">
142   <xs:selector xpath="Vevok/Vevo"/>
143   <xs:field xpath="@VID"/>
144 </xs:key>
145
146 <xs:keyref name="Tulajdonos_IKulcs" refer="Tulajdonos_EKulcs">
147   <xs:selector xpath="Pizzeriak/Pizzeria"/>
148   <xs:field xpath="@TIDRef"/>
149 </xs:keyref>
150
151 <xs:keyref name="Pizzeria_IKulcs" refer="Pizzeria_EKulcs">
152   <xs:selector xpath="Reszvetelek/Reszvetel"/>
153   <xs:field xpath="@PiIDRef"/>
154 </xs:keyref>
155
156 <xs:keyref name="Pizza_IKulcs" refer="Pizza_EKulcs">
157   <xs:selector xpath="Vasarlasok/Megveszi"/>
158   <xs:field xpath="@PIDRef"/>
159 </xs:keyref>
160
161 <xs:keyref name="Vevo_IKulcs" refer="Vevo_EKulcs">
162   <xs:selector xpath="Vasarlasok/Megveszi"/>
163   <xs:field xpath="@VIDRef"/>
164 </xs:keyref>
165 </xs:element>
166 </xs:schema>

```

DOM Read:

A feladat egy DOM program készítése XML dokumentum adatainak adminisztrálása alapján (comment-tel együtt). A DOM Read kiírja az XML dokumentumom adatait.

```
1 package hu.domparse.ejx162;
2
3 import java.io.File;
4 import java.io.IOException;
5
6 import javax.xml.parsers.DocumentBuilder;
7 import javax.xml.parsers.DocumentBuilderFactory;
8 import javax.xml.parsers.ParserConfigurationException;
9
10 import org.w3c.dom.Document;
11 import org.w3c.dom.NamedNodeMap;
12 import org.w3c.dom.Node;
13 import org.w3c.dom.traversal.DocumentTraversal;
14 import org.w3c.dom.traversal.NodeFilter;
15 import org.w3c.dom.traversal.TreeWalker;
16 import org.xml.sax.SAXException;
17
18 public class DomReadEJX162 {
19
20     public static void main(String[] args) throws ParserConfigurationException, SAXException, IOException {
21         File xml = new
22 File("C:\\Users\\Bálint\\Desktop\\EGYETEM\\2022-23. 5.félév\\xml_beadando\\DOMParseEJX162"
23 + "\\src\\hu\\domparse\\ejx162\\XMLEJX162.xml");
24
25
26         // XML fájl DOM document alakítása
27         DocumentBuilderFactory factory = DocumentBuilderFactory.newInstance();
28         DocumentBuilder builder = factory.newDocumentBuilder();
29         Document document = builder.parse(xml);
30
31         // DOM document átalakítása DOM DocumentTraversal formába
32         DocumentTraversal traversal = (DocumentTraversal) document;
33
34         //TreeWalker inicializálása
35         TreeWalker walker = traversal.createTreeWalker(document.getDocumentElement(),
36             NodeFilter.SHOW_ELEMENT | NodeFilter.SHOW_TEXT, null, true);
37
38         //DOM bejárása és kiírása
```

```

39     DomTraverser.traverseLevel(walker, "");
40 }
41
42 private static class DomTraverser {
43     public static void traverseLevel(TreeWalker walker, String indent) {
44         // Aktuális csomópont
45         Node node = walker.getCurrentNode();
46
47         if (node.getNodeType() == Node.ELEMENT_NODE) {
48             printElementNode(node, indent);
49         } else {
50             printTextNode(node, indent);
51         }
52
53         // Rekurzívan meghívjuk a bejárást a DOM fában
54         for (Node n = walker.firstChild(); n != null; n = walker.nextSibling()) {
55             traverseLevel(walker, indent + "    ");
56         }
57
58         walker.setCurrentNode(node);
59     }
60
61     private static void printElementNode(Node node, String indent) {
62         System.out.print(indent + node.getNodeName());
63
64         printElementAttributes(node.getAttributes());
65     }
66
67     private static void printElementAttributes(NamedNodeMap attributes) {
68         int length = attributes.getLength();
69
70         if (length > 0) {
71             System.out.print(" [ ");
72
73             for (int i = 0; i < length; i++) {
74                 Node attribute = attributes.item(i);
75
76                 System.out.printf("%s=%s%s", attribute.getNodeName(), attribute.getNodeValue(),
77                     i != length - 1 ? ", " : "");
78             }
79
80             System.out.println(" ]");
81         } else {
82             System.out.println();
83         }
84     }
85
86     private static void printTextNode(Node node, String indent) {
87         String content_trimmed = node.getTextContent().trim();
88
89         if (content_trimmed.length() > 0) {
90             System.out.print(indent);
91             System.out.printf("{ %s }%n", content_trimmed);
92         }
93     }
94 }
95 }

```

Dom Read Output:

```

model [ xmlns:xsi=http://www.w3.org/2001/XMLSchema-instance,
xsi:noNamespaceSchemaLocation=XMLSchemaH7PG8U.xsd ]

```

Tulajdonosok

Tulajdonos [Tkod=1]

Nev

{ Nagy Béla }

Szulido

Ev

{ 1987 }

Honap

{ 03 }

Nap

{ 12 }

Tulajdonos [Tkod=2]

Nev

{ Lőre István }

Szulido

Ev

{ 1967 }

Honap

{ 10 }

Nap

{ 25 }

Tulajdonos [Tkod=3]

Nev

{ Kiss Imre }

Szulido

Ev

{ 1980 }

Honap

{ 08 }

Nap

{ 20 }

Pizzeriak

Pizzeria [Pikod=1]

Nev

{ Italian Stallion }

Telefonszam

{ 012432123 }

Cim

Irszam

{ 1256 }

Varos

{ Budapest }

Utca

{ Ferencziek útja }

Hazszam

{ 14 }

Pizzeria [Pikod=2]

Nev

{ New York Style }

Telefonszam

{ 05618478 }

Cim

Irszam

{ 3525 }

Varos

{ Miskolc }
 Utca
 { Széchenyi István út }
 Hatszám
 { 26 }
Pizzeria [Pikod=3]
 Név
 { László }
 Telefonszám
 { 035637923 }
 Cím
 Irszám
 { 3540 }
 Város
 { Alsózsoltca }
 Utca
 { Fő út }
 Hatszám
 { 6 }
Pizzák
 Pizza [Pkod=1]
 Név
 { Margareta }
 Ár
 { 1200 }
 Feltet
 { Sajt }
 Pizza [Pkod=2]
 Név
 { Sonkás }
 Ár
 { 1400 }
 Feltet
 { Sajt }
 Feltet
 { Sonka }
 Pizza [Pkod=3]
 Név
 { Szalámis }
 Ár
 { 1400 }
 Feltet
 { Sajt }
 Feltet
 { Szalámi }
 Pizza [Pkod=4]
 Név
 { Sonkás-Kukoricás }
 Ár
 { 1600 }
 Feltet
 { Sajt }
 Feltet
 { Sonka }
 Feltet

```

        { Kukorica }
Pizza [ Pkod=5 ]
    Nev
        { Lásbas Speciál }
    Ar
        { 2100 }
    Feltet
        { Sajt }
    Feltet
        { BBQ szósz }
    Feltet
        { Pulled Pork }
    Feltet
        { Bacon }
Vevok
    Vevo [ Vkod=1 ]
        Nev
            { Nagy Ferenc }
        Telefonszam
            { 77895412 }
        Cim
            Irszam
                { 3519 }
            Varos
                { Miskolc }
            Utca
                { Fenyő utca }
        Hazszam
            { 23 }
    Vevo [ Vkod=2 ]
        Nev
            { Novák Balázs }
        Telefonszam
            { 078954412 }
        Cim
            Irszam
                Tulajdonos_Pizzeria [ Pizzeriaref=1, Tulajdonosref=1 ]
                Tulajdonos_Pizzeria [ Pizzeriaref=2, Tulajdonosref=2 ]
                Tulajdonos_Pizzeria [ Pizzeriaref=3, Tulajdonosref=2 ]
Pizzeria_Pizza_kapcsolok
    Pizzeria_Pizza [ Pizzaref=1, Pizzeriaref=1 ]
    Pizzeria_Pizza [ Pizzaref=3, Pizzeriaref=1 ]
    Pizzeria_Pizza [ Pizzaref=2, Pizzeriaref=2 ]
    Pizzeria_Pizza [ Pizzaref=4, Pizzeriaref=2 ]
    Pizzeria_Pizza [ Pizzaref=5, Pizzeriaref=3 ]
Pizza_Vevo_kapcsolok
    Pizza_Vevo [ Pizzaref=2, Vevoref=1 ]
    Pizza_Vevo [ Pizzaref=4, Vevoref=2 ]
    Pizza_Vevo [ Pizzaref=5, Vevoref=3 ]

```

DOM Modify:

A feladat egy DOM program készítése, amely módosít néhány elemet az XML dokumentumomban. Én Kiss Imre nevét írtam át Nagyobb Imrére, valamint a 1300 Ft-nál drágább pizzák árát csökkentem 10%-kal.

```
1 package hu.domparse.ejx162;
2
3 import java.io.File;
4 import java.io.IOException;
5 import java.text.ParseException;
6
7 import javax.xml.parsers.DocumentBuilder;
8 import javax.xml.parsers.DocumentBuilderFactory;
9 import javax.xml.parsers.ParserConfigurationException;
10 import javax.xml.xpath.XPath;
11 import javax.xml.xpath.XPathConstants;
12 import javax.xml.xpath.XPathExpressionException;
13 import javax.xml.xpath.XPathFactory;
14
15 import org.w3c.dom.DOMException;
16 import org.w3c.dom.Document;
17 import org.w3c.dom.NamedNodeMap;
18 import org.w3c.dom.Node;
19 import org.w3c.dom.NodeList;
20 import org.w3c.dom.traversal.DocumentTraversal;
21 import org.w3c.dom.traversal.NodeFilter;
22 import org.w3c.dom.traversal.TreeWalker;
23 import org.xml.sax.SAXException;
24
25 public class DomModifyEJX162 {
26
27     public static void main(String[] args) throws ParserConfigurationException, SAXException, IOException,
28         XPathExpressionException, DOMException, ParseException {
29
30         File xml = new
31 File("C:\\Users\\Bálint\\Desktop\\EGYETEM\\2022-23. 5.félév\\xml_beadando"
32 + "\\DOMParseEJX162\\src\\hu\\domparse\\ejx162\\XMLEJX162.xml");
33
34         DocumentBuilderFactory factory = DocumentBuilderFactory.newInstance();
35         DocumentBuilder builder = factory.newDocumentBuilder();
36         Document document = builder.parse(xml);
37
38         // a DOM document módosítása
39
```

```

39 DomModifier.modifyDom(document);
40
41 // DOM document átalakítása DOM DocumentTraversal formába
42 DocumentTraversal traversal = (DocumentTraversal) document;
43
44 //TreeWalker inicializálása
45 TreeWalker walker = traversal.createTreeWalker(document.getDocumentElement(),
46     NodeFilter.SHOW_ELEMENT | NodeFilter.SHOW_TEXT, null, true);
47
48 //DOM bejárása
49 DomTraverser.traverseLevel(walker, "");
50
51 }
52
53 private static class DomModifier {
54     public static void modifyDom(Document document) throws XPathExpressionException, DOMException, ParseException {
55         XPathFactory factory = XPathFactory.newInstance();
56         XPath xpath = factory.newXPath();
57
58         // 1.) Kiss Imre nevenek megváltoztatása Nagyobb Imrére
59         Node owner = (Node) xpath.evaluate("//Tulajdonos[./Nev='Kiss Imre']",
60             document, XPathConstants.NODE);
61
62         owner.setTextContent("Nagyobb Imre");
63
64         // 2.) Minden 1300 forintnál drágább pizzának az ára csökken 10%-al
65         NodeList pizzas = (NodeList) xpath.evaluate("//Pizza[./Ar>1300]/Ar", document, XPathConstants.NODESET);
66         System.out.println(pizzas);
67         for (int i = 0; i < pizzas.getLength(); i++) {
68             Node pizza = pizzas.item(i);
69
70             double price = Double.parseDouble(pizza.getTextContent());
71             pizza.setTextContent(Double.toString(price * 0.9));
72
73         }
74     }
75 }
76
77 private static class DomTraverser {
78     public static void traverseLevel(TreeWalker walker, String indent) {
79         //Aktuális csomópont
80         Node node = walker.getCurrentNode();
81
82         if (node.getNodeType() == Node.ELEMENT_NODE) {
83             printElementNode(node, indent);
84         } else {
85             printTextNode(node, indent);
86         }
87
88         // Rekurzívan meghíviuk a bejárást a DOM fában
89         for (Node n = walker.firstChild(); n != null; n = walker.nextSibling()) {
90             traverseLevel(walker, indent + "    ");
91         }
92
93         walker.setCurrentNode(node);
94     }
95
96     private static void printElementNode(Node node, String indent) {
97         System.out.print(indent + node.getNodeName());
98
99         printElementAttributes(node.getAttributes());
100     }
101
102     private static void printElementAttributes(NamedNodeMap attributes) {
103         int length = attributes.getLength();
104
105         if (length > 0) {
106             System.out.print(" [ ");
107
108             for (int i = 0; i < length; i++) {
109                 Node attribute = attributes.item(i);

```

```

111         System.out.printf("%s=%s", attribute.getNodeName(), attribute.getNodeValue(),
112             i != length - 1 ? ", " : "");
113     }
114
115     System.out.println(" ]");
116 } else {
117     System.out.println();
118 }
119 }
120
121 private static void printTextNode(Node node, String indent) {
122     String content_trimmed = node.getTextContent().trim();
123
124     if (content_trimmed.length() > 0) {
125         System.out.print(indent);
126         System.out.printf("{ %s }%n", content_trimmed);
127     }
128 }
129 }
130
131 }

```

DOM Modify Output:

model [xmlns:xsi=http://www.w3.org/2001/XMLSchema-instance,
xsi:noNamespaceSchemaLocation=XMLSchemaH7PG8U.xsd]

Tulajdonosok

Tulajdonos [Tkod=1]

Nev

{ Nagy Béla }

Szulido

Ev

{ 1987 }

Honap

{ 03 }

Nap

{ 12 }

Tulajdonos [Tkod=2]

Nev

{ Lőre István }

Szulido

Ev

{ 1967 }

Honap

{ 10 }

Nap

{ 25 }

Tulajdonos [Tkod=3]

{ Nagyobb Imre }

Pizzeriak

Pizzeria [Pikod=1]

Nev

{ Italian Stallion }

Telefonszam

{ 012432123 }

Cim

Irszam
 { 1256 }
Varos
 { Budapest }
Utca
 { Ferenciek útja }
Haszam
 { 14 }
Pizzeria [Pikod=2]
Nev
 { New York Style }
Telefonszam
 { 05618478 }
Cim
 Irszam
 { 3525 }
 Varos
 { Miskolc }
 Utca
 { Széchenyi István út }
 Haszam
 { 26 }
Pizzeria [Pikod=3]
Nev
 { Lábás }
Telefonszam
 { 035637923 }
Cim
 Irszam
 { 3540 }
 Varos
 { Alsózsolca }
 Utca
 { Fő út }
Haszam
 { 6 }
Pizzak
 Pizza [Pkod=1]
 Nev
 { Margareta }
 Ar
 { 1200 }
 Feltet
 { Sajt }
 Pizza [Pkod=2]
 Nev
 { Sonkás }
 Ar
 { 1260.0 }
 Feltet

```
        { Sajt }
    Feltet
        { Sonka }
    Pizza [ Pkod=3 ]
    Nev
        { Szalámis }
    Ar
        { 1260.0 }
    Feltet
        { Sajt }
    Feltet
        { Szalámi }
    Pizza [ Pkod=4 ]
    Nev
        { Sonkás-Kukoricás }
    Ar
        { 1440.0 }
    Feltet
        { Sajt }
    Feltet
        { Sonka }
    Feltet
        { Kukorica }
    Pizza [ Pkod=5 ]
    Nev
        { Lábás Speciál }
    Ar
        { 1890.0 }
    Feltet
        { Sajt }
    Feltet
        { BBQ szósz }
    Feltet
        { Pulled Pork }
    Feltet
        { Bacon }
    Vevok
    Vevo [ Vkod=1 ]
    Nev
        { Nagy Ferenc }
    Telefonszam
        { 77895412 }
    Cim
    Irszam
        { 3519 }
    Varos
        { Miskolc }
    Utca
    { Fenyő utca }
    Hazszam
```

```

        { 23 }
Vevo [ Vkod=2 ]
    Nev
        { Novák Balázs }
    Telefonszam
        { 078954412 }
    Cim
    Irszam
        { 1149 }
    Varos
Pizzeria_Pizza [ Pizzaref=1, Pizzeriaref=1 ]
Pizzeria_Pizza [ Pizzaref=3, Pizzeriaref=1 ]
Pizzeria_Pizza [ Pizzaref=2, Pizzeriaref=2 ]
Pizzeria_Pizza [ Pizzaref=4, Pizzeriaref=2 ]
Pizzeria_Pizza [ Pizzaref=5, Pizzeriaref=3 ]
Pizza_Vevo_kapcsolok
    Pizza_Vevo [ Pizzaref=2, Vevoref=1 ]
    Pizza_Vevo [ Pizzaref=4, Vevoref=2 ]
    Pizza_Vevo [ Pizzaref=5, Vevoref=3 ]

```

DOM Query:

A DOM Query file-ban lekérdezem az XML file-ból, az összes Pizzéria attributumát, valamint annak a Pizzériának a nevét, amely Pesten van.

```

1 package hu.domparse.ejx162;
2
3 import java.io.File;
15
16 public class DomQueryEJX162 {
17
18     public static void main(String[] args) throws ParserConfigurationException, SAXException, IOException {
19         File file = new File("C:\\Users\\Bálint\\Desktop\\EGYETEM\\2022-23. 5.félév\\xml_beadando"
20             + "\\DOMParseEJX162\\src\\hu\\domparse\\ejx162\\XMLEJX162.xml");
21         // Parse-olás
22         DocumentBuilderFactory dbFactory = DocumentBuilderFactory.newInstance();
23         DocumentBuilder dBuilder = dbFactory.newDocumentBuilder();
24
25         Document doc = dBuilder.parse(file);
26         doc.getDocumentElement().normalize();
27         // Root element kiírása
28         System.out.println("Gyoker element: ");
29         System.out.println(doc.getDocumentElement().getNodeName());
30         NodeList nList = doc.getElementsByTagName("Pizzeria");
31
32         // Minden pizzeria attribútum kiírása
33
34         System.out.println("PIZZERIAK");
35         for (int i = 0; i < nList.getLength(); i++) {
36             Node node = nList.item(i);
37             System.out.println("\nElement nev : " + node.getNodeName());
38             if (node.getNodeType() == Node.ELEMENT_NODE) {
39                 Element elem = (Element) node;
40                 System.out.println("ID: " + elem.getAttribute("Pikod"));
41                 NodeList nList2 = elem.getChildNodes();
42                 for (int j = 0; j < nList2.getLength(); j++) {
43                     Node node2 = nList2.item(j);
44                     if (node2.getNodeType() == Node.ELEMENT_NODE) {
45                         Element elem2 = (Element) node2;
46                         if (!node2.getNodeName().equals("Cim")) {
47                             System.out.println(node2.getNodeName() + " : " + node2.getTextContent());

```

```

48         } else {
49             System.out.println("Cim:");
50
51             NodeList nList3 = elem2.getChildNodes();
52             for (int k = 0; k < nList3.getLength(); k++) {
53                 Node node3 = nList3.item(k);
54
55                 if (node3.getNodeType() == Node.ELEMENT_NODE) {
56
57                     System.out.println("    " + node3.getNodeName() + " : " + node3.getTextContent());
58                 }
59             }
60         }
61     }
62 }
63
64 }
65 //Kírja annak a pizzeriának a nevet, ami Pesten van
66 System.out.println("\nPESTI PIZZERIA\n");
67 for (int i = 0; i < nList.getLength(); i++) {
68     Node node = nList.item(i);
69     if (node.getNodeType() == Node.ELEMENT_NODE) {
70         Element elem = (Element) node;
71         NodeList nList2 = elem.getChildNodes();
72         for (int j = 0; j < nList2.getLength(); j++) {
73             Node node2 = nList2.item(j);
74             if (node2.getNodeType() == Node.ELEMENT_NODE) {
75                 Element elem2 = (Element) node2;
76                 NodeList nList3 = elem2.getChildNodes();
77                 for (int k = 0; k < nList3.getLength(); k++) {
78                     Node node3 = nList3.item(k);
79                     if (node3.getNodeType() == Node.ELEMENT_NODE) {
80                         if (node3.getNodeName().equals("Varos")) {
81                             if (node3.getTextContent().equals("Budapest")) {
82                                 node2 = nList2.item(1);
83                                 System.out.println(node2.getNodeName() + " : " + node2.getTextContent());
84                             }
85                         }
86                     }
87                 }
88             }
89         }
90     }
91 }
92 }
93 }
94 }
95 }
96 }

```

DOM Query Output:

Gyökér element: model
PIZZERIAK

Element nev : Pizzeria
ID:1
Nev : Italian Stallion
Telefonszam : 012432123 Cim:
Irszam : 1256
Varos : Budapest
Utca : Ferencziek útja
Hatszam : 14

Element nev : Pizzeria
ID:2
Nev : New York Style
Telefonszam : 05618478 Cim:

Irszam : 3525
Varos : Miskolc
Utca : Széchenyi István út
Hazsam : 26

Element nev : Pizzeria
ID:3
Nev : Lábas Telefonszam
: 035637923 Cim:
Irszam : 3540
Varos : Alsózsolca
Utca : Fő út
Hazsam : 6

PESTI PIZZERIA

Nev: Italian Stallion