JEGYZŐKÖNYV

Adatkezelés XML környezetben

Virágbolt vevőkör nyilvántartó rendszer

Készítette: Tóth Gergő

Neptunkód: FOQH34

1) Feladat leírása:

Az adatbázis rendszerem alapja egy virágbolt láncolat nyilvántartási rendszere melyben egyed szerepet tölt be a

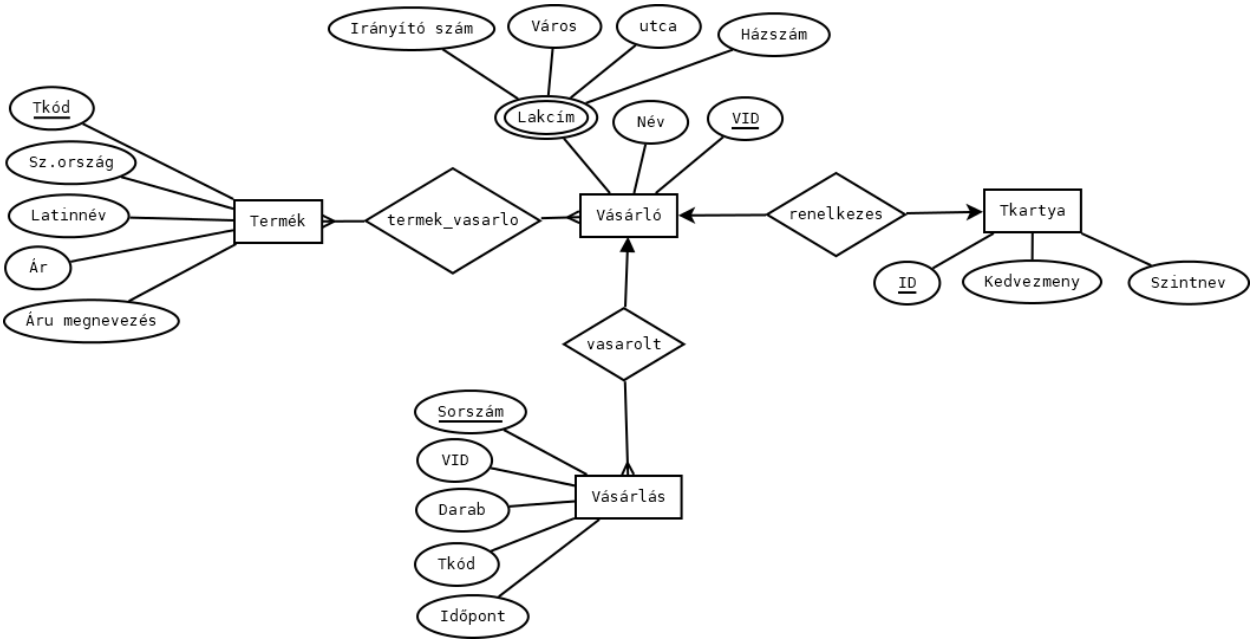
-törzsvásárlói kártya melynek tulajdonságai az azonosítója, a törszvásárlói szint megnevezése és a hozzátartozó kedvezmény mértéke.

-bolt árukészlete tulajdonságai az áru megnevezése, az áru latin neve, ára, termék kódja valamint a származási ország neve.

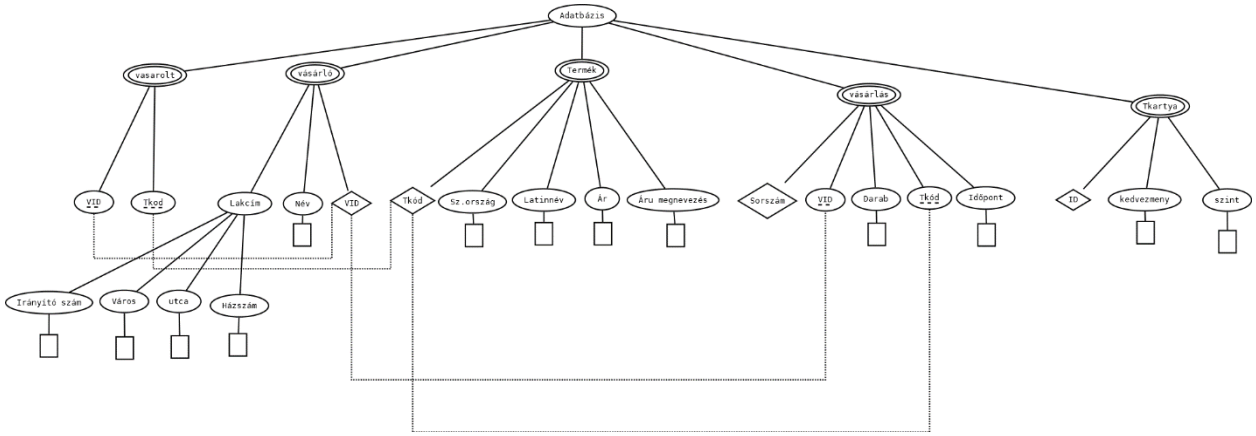
-vásárlók amik a nevéből,lakcíméből, törzsvásárlói kártyájukból, illetve egyedi vásárló azonosítójából áll.

illetve a vásárlás adatai ami egy egyéni sorszámmal rendelkezik valamint egy vásárló azonosítóval, egy dátummal, a termék kódjával és a vásárolt termék darabszámával.

1a) Az adatbázis ER modell:



1b) Az adatbázis konvertálása XDM modellre:



1c) Az XDM modell alapján XML dokumentum készítése:

```xml
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<adatbazis xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="XMLSchemaFOQH34.xsd">

  <vasarlo id="V12">
    <lakcim>
        <iranyitoSzam>3535</iranyitoSzam>
        <varos>Miskolc</varos>
        <utca>Hegyalja utca</utca>
        <hazszam>15</hazszam>
    </lakcim>
    <nev>Tóth Gergő</nev>
  </vasarlo>

  <vasarlo id="V13">
    <lakcim>
      <iranyitoSzam>3516</iranyitoSzam>
      <varos>Miskolc</varos>
      <utca>Csermak Antal utca</utca>
      <hazszam>41</hazszam>
    </lakcim>
    <nev>Toth Levente</nev>
  </vasarlo>

  <termek id="T01">
    <szarmazasiOrszag>Hollandia</szarmazasiOrszag>
    <latinNev>Rosa</latinNev>
    <ar>850</ar>
    <aruMegnevezes>Voros rozsa</aruMegnevezes>
  </termek>

  <termek id="T02">
    <szarmazasiOrszag>Magyar</szarmazasiOrszag>
    <latinNev>-</latinNev>
    <ar>1200</ar>
    <aruMegnevezes>illat gyertya</aruMegnevezes>
  </termek>

```

```xml

  <vasarlas id="S01">
    <vid>V12</vid>
    <darab>10</darab>
    <tKod>T01</tKod>
    <idopont>2022-12-05</idopont>
  </vasarlas>

  <tkartya id="TK01">
    <szint>platina</szint>
    <kedvezmeny>10</kedvezmeny>
  </tkartya>
</adatbazis>
```

1d) Az XML dokumentum alapján XMLSchema készítése (saját típusok):

```xml
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns:tns="http://www.example.org/XMLSchemaFOQH34" elementFormDefault="qualified">
    <xs:element name="adatbazis">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="vasarlo" type="vasarloTipus" minOccurs="0" maxOccurs="unbounded"/>
                <xs:element name="termek" type="termekTipus" maxOccurs="unbounded"/>
                <xs:element name="vasarlas" type="vasarlasTipus" maxOccurs="unbounded"/>
                <xs:element name="tkartya" type="tkartyaTipus" maxOccurs="unbounded"/>
            </xs:sequence>
        </xs:complexType>

        <xs:key name="vasarloKulcs">
        <xs:selector xpath="vasarlo"/>
        <xs:field xpath="@id" />
        </xs:key>
        <xs:key name="tkartyaKulcs">
        <xs:selector xpath="tkartya"/>
        <xs:field xpath="@id" />
        </xs:key>
        <xs:key name="termekKulcs">
        <xs:selector xpath="termek"/>
        <xs:field xpath="@id" />
        </xs:key>
        <xs:key name="vasarlasKulcs">
        <xs:selector xpath="vasarlas"/>
        <xs:field xpath="@id" />
        </xs:key>

        <xs:keyref name="vasarlas-termek" refer="termekKulcs">
        <xs:selector xpath="vasarlas/tKod"/>
        <xs:field xpath="."/>
        </xs:keyref>
        <xs:keyref name="vasarlas-vasarlo" refer="vasarloKulcs">
        <xs:selector xpath="vasarlas/vid"/>
        <xs:field xpath="."/>
        </xs:keyref>
        <xs:keyref name="vasarlo-tkartya" refer="tkartyaKulcs">
```

```xml
        <xs:keyref name="vasarlo-tkartya" refer="tkartyaKulcs">
        <xs:selector xpath="vasarlo/tkartyaid"/>
        <xs:field xpath="."/>
        </xs:keyref>
    </xs:element>
    <xs:complexType name="vasarloTipus">
        <xs:sequence>
            <xs:element name="lakcim" type="lakcimTipus" />
            <xs:element name="nev" type="nev"/>
        </xs:sequence>
        <xs:attribute name="id" type="xs:ID" use="required" />
    </xs:complexType>

    <xs:complexType name="lakcimTipus">
        <xs:sequence>
            <xs:element name="iranyitoSzam" type="iranyitoSzam"/>
            <xs:element name="varos" type="varos"/>
            <xs:element name="utca" type="utca"/>
            <xs:element name="hazszam" type="hazszam"/>
        </xs:sequence>
    </xs:complexType>

    <xs:simpleType name="iranyitoSzam">
        <xs:restriction base="xs:positiveInteger"/>
    </xs:simpleType>

    <xs:simpleType name="varos">
        <xs:restriction base="xs:string"/>
    </xs:simpleType>

    <xs:simpleType name="utca">
        <xs:restriction base="xs:string"/>
    </xs:simpleType>

    <xs:simpleType name="hazszam">
        <xs:restriction base="xs:positiveInteger"/>
    </xs:simpleType>
```

```
    </xs:simpleType>

    <xs:simpleType name="nev">
        <xs:restriction base="xs:string"/>
    </xs:simpleType>

    <xs:complexType name="termekTipus">
        <xs:sequence>
            <xs:element name="szarmazasiOrszag" type="xs:string"/>
            <xs:element name="latinNev" type="xs:string"/>
            <xs:element name="ar" type="xs:positiveInteger"/>
            <xs:element name="aruMegnevezes" type="xs:string"/>
        </xs:sequence>
        <xs:attribute name="id" type="xs:ID" use="required" />
    </xs:complexType>
    <xs:complexType name="tkartyaTipus">
        <xs:sequence>
            <xs:element name="szint" type="xs:string"/>
            <xs:element name="kedvezmeny" type="xs:positiveInteger"/>
        </xs:sequence>
        <xs:attribute name="id" type="xs:ID" use="required" />
    </xs:complexType>
    <xs:complexType name="vasarlasTipus">
        <xs:sequence>
            <xs:element name="vid" type="xs:string"/>
            <xs:element name="darab" type="xs:string"/>
            <xs:element name="tKod" type="xs:string"/>
            <xs:element name="idopont" type="xs:date"/>
        </xs:sequence>
        <xs:attribute name="id" type="xs:ID" use="required" />
    </xs:complexType>
</xs:schema>
```

2. feladat:

2a) adatolvasás

```
package hu.domparse.FOQH34;

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 DomReadFOQH34 {public static void main(String argv[]) throws SAXException, IOException, ParserConfigurationException {
    File xmlFile = new File("XMLfoqh34.xml");

    //DOMFA felépítése
    DocumentBuilderFactory factory = DocumentBuilderFactory.newInstance();
    DocumentBuilder dBuilder = factory.newDocumentBuilder();
    Document doc =dBuilder.parse(xmlFile);
    doc.getDocumentElement().normalize();

    //Adattagokon végig megyünk
     System.out.println("Root element: "+doc.getDocumentElement().getNodeName());
    String[] tagNames= {"vasarlo" ,"termek", "vasarlas","tkartya"};
    for(String tagName :tagNames) {
        NodeList nList = doc.getElementsByTagName(tagName);


    for(int i=0; i< nList.getLength();i++) {
        Node nNode = nList.item(i);
        System.out.println("\nCurrent Element: " + nNode.getNodeName());

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

```
            Element elem =(Element) nNode;

            //Azonosító kiírása
            String id = elem.getAttribute("id");
            System.out.println("    ID: " +id);
            //Tulajdonságok kiírása

            String nodeContent="";
            NodeList childNodes =elem.getChildNodes();

            for(int j =0; j< childNodes.getLength();j++) {
                if(childNodes.item(j).getTextContent().trim() !="") {
                    nodeContent =normalizeText(childNodes.item(j).getTextContent().trim());
                    System.out.println("    "+childNodes.item(j).getNodeName()+": "+nodeContent);
                }
            }

            }
        System.out.println();
        }



    }
}
    private static String normalizeText(String text) {
        text=text.replaceAll("\\n", ", ");
        text=text.replaceAll("\\s+", " ");
        return text;
    }
}

```

2b) Adatmódosítás

V12 ID-vel rendelkező vásárló nevének módositása Kertész László-ra

T02 ID-vel rendelkező termek nevének módositása Pink Rose-ra

T02 ID-vel rendelkező termek származási országának módositása Kubára

```java
package hu.domparse.FOQH34;

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

import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.DocumentBuilderFactory;
import javax.xml.parsers.ParserConfigurationException;
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.Document;
import org.w3c.dom.Node;
import org.w3c.dom.NodeList;
import org.xml.sax.SAXException;

public class DomModifyFOQH34 {
    public static void main(String[] args) {
        try {
        File xmlFile = new File("XMLfoqh34.xml");
        DocumentBuilderFactory factory = DocumentBuilderFactory.newInstance();
        DocumentBuilder dBuilder = factory.newDocumentBuilder();
        Document doc =dBuilder.parse(xmlFile);
        doc.getDocumentElement().normalize();

        //V12 ID-vel rendelkező vásárló nevének módosítása Kertész László-ra
        NodeList nodes =doc.getElementsByTagName("vasarlo");
        for(int i =0;i<nodes.getLength();i++) {
            Node node = nodes.item(i);
            if(node.getNodeType()== Node.ELEMENT_NODE) {
                if(node.getAttributes().getNamedItem("id").getTextContent().equals("V12")) {
                    NodeList childNodes = node.getChildNodes();
                    for(int j=0 ; j < childNodes.getLength();j++) {
                        Node childNode = childNodes.item(j);


                        Node childNode = childNodes.item(j);
                        if(childNode.getNodeName().equals("nev")) {
                            childNode.setTextContent("Kertesz Laszlo");
                        }
                    }
                }
            }
        }
        //T02 ID-vel rendelkező termek nevének módosítása Pink Rose-ra
                nodes =doc.getElementsByTagName("termek");
                for(int i =0;i<nodes.getLength();i++) {
                    Node node = nodes.item(i);
                    if(node.getNodeType()== Node.ELEMENT_NODE) {
                        if(node.getAttributes().getNamedItem("id").getTextContent().equals("T02")) {
                            NodeList childNodes = node.getChildNodes();
                            for(int j=0 ; j < childNodes.getLength();j++) {
                                Node childNode = childNodes.item(j);
                                if(childNode.getNodeName().equals("aruMegnevezes")) {
                                    childNode.setTextContent("Pink Rose");
                                }
                            }
                        }
                    }
                }
                //T02 ID-vel rendelkező termek származási országának módositása Kubára
                nodes =doc.getElementsByTagName("termek");
                for(int i =0;i<nodes.getLength();i++) {
                    Node node = nodes.item(i);
                    if(node.getNodeType()== Node.ELEMENT_NODE) {
                        if(node.getAttributes().getNamedItem("id").getTextContent().equals("T02")) {
                            NodeList childNodes = node.getChildNodes();
                            for(int j=0 ; j < childNodes.getLength();j++) {
                                Node childNode = childNodes.item(j);
                                if(childNode.getNodeName().equals("szarmazasiOrszag")) {
                                    childNode.setTextContent("Kuba");
                                }
                            }
                        }


                        }
                    }
                }

        File file =new File("XMLFOQH34.xml");
        writeXml(doc,file);

        }
        catch(ParserConfigurationException | IOException | SAXException | TransformerException ex){
            System.out.println("Some error happened:\n"+ex.getMessage());
            ex.printStackTrace();
        }
    }
    private static void writeXml(Document doc, File output) throws TransformerException,UnsupportedEncodingException{
        TransformerFactory transformerFactory= TransformerFactory.newInstance();
        Transformer transf =transformerFactory.newTransformer();
        transf.setOutputProperty(OutputKeys.ENCODING, "UTF-8");
        transf.setOutputProperty(OutputKeys.INDENT, "yes");
        transf.setOutputProperty("{http://xml.apache.org/xslt}indent-amount", "2");
        DOMSource source = new DOMSource(doc);

        StreamResult console = new StreamResult(System.out);
        StreamResult file = new StreamResult(output);

        transf.transform(source, console);
        transf.transform(source, file);
    }
}
```

3c) Lekérdezés:

```java
package hu.domparse.FOQH34;

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 DomQueryFOQH34 {

    public static void main(String[] args) {
        try {
        File xmlFile = new File("XMLfoqh34.xml");
        DocumentBuilderFactory factory = DocumentBuilderFactory.newInstance();
        DocumentBuilder dBuilder = factory.newDocumentBuilder();
        Document doc =dBuilder.parse(xmlFile);
        doc.getDocumentElement().normalize();

        String feladat="Ezer forintnál drágább termékek:";
        System.out.println(feladat +"\n\n");
        Lekerdezes1(doc);

        String feladat2="V13 kódú vásárló adatai: ";
        System.out.println(feladat2 +"\n\n");
        Lekerdezes2(doc);




        }
        catch(ParserConfigurationException | IOException | SAXException ex){
            System.out.println("Some error happened:\n"+ex.getMessage());
            System.out.println("Some error happened:\n"+ex.getMessage());
            ex.printStackTrace();
        }
    }

    private static String normalizeText(String text) {
        text=text.replaceAll("\\n", ", ");
        text=text.replaceAll("\\s+", " ");
        return text;
    }

    private static void Lekerdezes1(Document doc) {
        NodeList termekek = doc.getElementsByTagName("termek");
        for(int i = 0; i < termekek.getLength(); i++) {
            Element termek =(Element)termekek.item(i);
            NodeList childNodes = termek.getChildNodes();
            for(int j =0;j<childNodes.getLength();j++) {
                Node childNode = childNodes.item(j);
                if(childNode.getNodeName().equals("ar")) {
                    if(Integer.parseInt(childNode.getTextContent())>1000) {
                        printElement(termek);
                    }
                }
            }
        }
        }

    private static void Lekerdezes2(Document doc) {
        NodeList vevok = doc.getElementsByTagName("vasarlo");
        for (int i = 0; i < vevok.getLength(); i++) {
            Element vevo = (Element)vevok.item(i);
            if (vevo.getAttributeNode("id").getValue().equals("V13"))
                printElement(vevo);
        }
    }


    }

    private static void printElement(Element elem) {
        String id = elem.getAttribute("id");
        System.out.println("    ID: "+ id);

        String nodeContent="";
        NodeList childNodes = elem.getChildNodes();
        for(int j =0;j<childNodes.getLength() ; j++) {
            if(childNodes.item(j).getTextContent().trim()!="") {
                nodeContent = normalizeText(childNodes.item(j).getTextContent().trim());
                System.out.println(childNodes.item(j).getNodeName()+": "+nodeContent);
            }
        }


    }
}
```