JEGYZŐKÖNYV

Adatkezelés XML környezetben

Féléves feladat

Készítette: **Paulin Krisztián**

Neptunkód: **WF8H4F**

A feladat leírása:

Ebben a feladatban videojátékokat áruló cégek, azok boltjaik ,bennük fellelhető videojátékok és az ott vásárolt felhasználók adatainak tárolását tűztem ki célul. Felépítésének köszönhetően több céget képes tárolni, amelyeknek egy vagy több boltja lehet, és a felhasználók egy vagy több boltból is vásárolhatnak videojátékokat kedvük szerint.

1 feladat

1a) Az adatbázis ER modell:
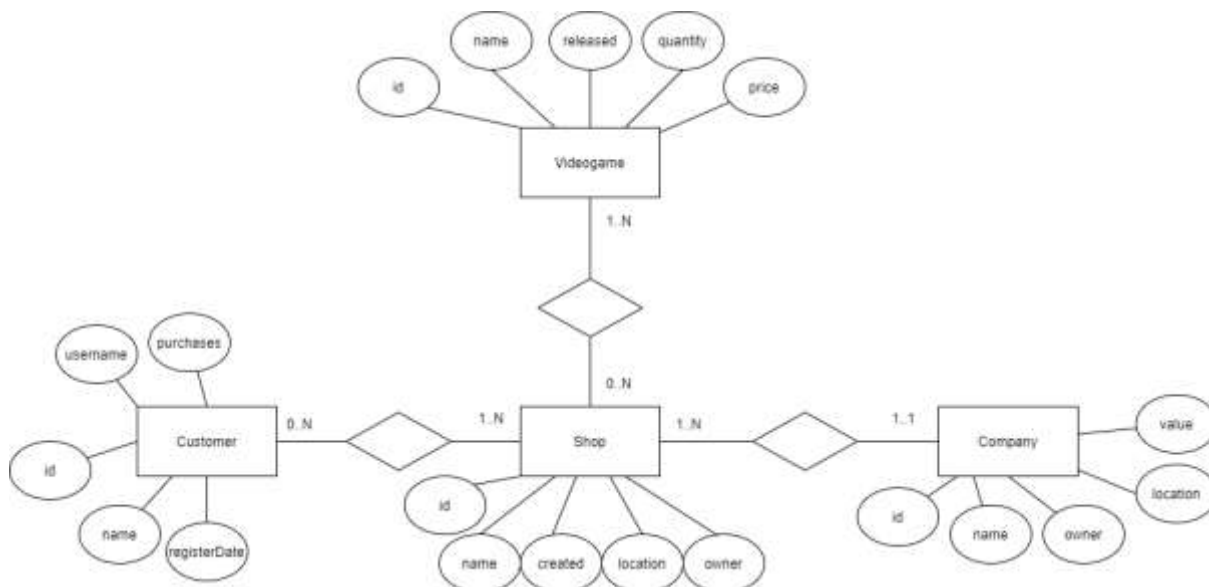
1b) Az adatbázis konvertálása XDM modellre:
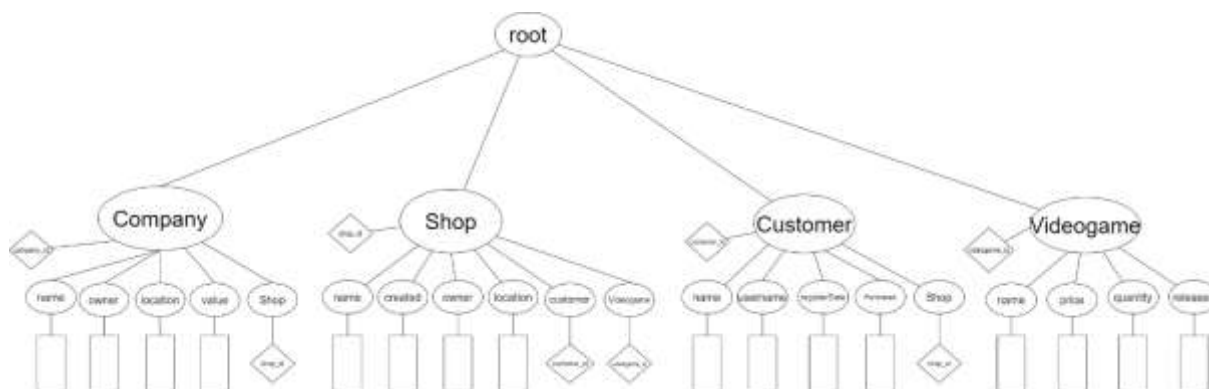
1c) Az XDM modell alapján XML dokumentum készítése:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="XMLSchemaWF8H4F.xsd"?>

<root>

<videogame videogame_id="1">
      <name>DOOM (2016)</name>
      <price>18999</price>
      <quantity>69</quantity>
      <released>2016-06-09</released>
</videogame>

 <videogame videogame_id="2">
      <name>DOOM Eternal</name>
```

```xml
        <price>18999</price>
        <quantity>20</quantity>
        <released>2020-04-20</released>
</videogame>

 <videogame videogame_id="3">
        <name>Dark Souls REMASTERED</name>
        <price>10000</price>
        <quantity>14</quantity>
        <released>2019-03-15</released>
</videogame>

 <customer customer_id="1">
        <name>Minta Béla</name>
        <username>MenoMano2000</username>
        <registerDate>2015-08-26</registerDate>
        <purchased>5</purchased>
        <shop shop_id="1"/>
 </customer>

 <shop shop_id="1">
        <name>Steam</name>
        <created>2003-11-10</created>
        <owner>Gabe Newell</owner>
        <location>Washington</location>
        <customer customer_id="1"/>
        <videogame videogame_id="1"/>
 </shop>

 <company company_id="1">
        <name>Valve</name>
        <owner>Gabe Newell</owner>
        <location>Washington</location>
        <value>10000</value>
        <shop shop_id="1"/>

 </company>

</root>
```

1d) Az XML dokumentum alapján XMLSchema készítése:

```xml
<?xml version="1.0" encoding="UTF-8"?>

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

<xs:complexType name="videogameType">
      <xs:sequence>
            <xs:element name="name" type="xs:string"/>
            <xs:element name="price" type="xs:integer"/>
            <xs:element name="quantity" type="xs:integer"/>
            <xs:element name="released" type="xs:date"/>
      </xs:sequence>
      <xs:attribute name="videogame_id" type="xs:integer" use="required"/>
</xs:complexType>

<xs:complexType name="customerType">
```

```xml
        <xs:sequence>
                <xs:element name="name" type="xs:string"/>
                <xs:element name="username" type="xs:string"/>
                <xs:element name="registerDate" type="xs:date"/>
                <xs:element name="purchased" type="xs:integer"/>
                <xs:element name="shop" maxOccurs="unbounded">
                <xs:complexType>
                    <xs:attribute name="shop_id" type="xs:integer" use="required"/>
                </xs:complexType>
            </xs:element>
        </xs:sequence>
        <xs:attribute name="customer_id" type="xs:integer" use="required"/>
</xs:complexType>

<xs:complexType name="shopType">
        <xs:sequence>
                <xs:element name="name" type="xs:string"/>
                <xs:element name="created" type="xs:date"/>
                <xs:element name="owner" type="xs:string"/>
                <xs:element name="location" type="xs:string"/>
                <xs:element name="customer" maxOccurs="unbounded">
                <xs:complexType>
                    <xs:attribute name="customer_id" type="xs:integer"
use="required"/>
                </xs:complexType>
            </xs:element>
            <xs:element name="videogame" maxOccurs="unbounded">
                <xs:complexType>
                    <xs:attribute name="videogame_id" type="xs:integer"
use="required"/>
                </xs:complexType>
            </xs:element>
        </xs:sequence>
        <xs:attribute name="shop_id" type="xs:integer" use="required"/>
</xs:complexType>

<xs:complexType name="companyType">
        <xs:sequence>
                <xs:element name="name" type="xs:string"/>
                <xs:element name="owner" type="xs:string"/>
                <xs:element name="location" type="xs:string"/>
                <xs:element name="value" type="xs:integer"/>
                <xs:element name="shop" maxOccurs="unbounded">
                <xs:complexType>
                    <xs:attribute name="shop_id" type="xs:integer" use="required"/>
                </xs:complexType>
            </xs:element>
        </xs:sequence>
        <xs:attribute name="company_id" type="xs:integer" use="required"/>
</xs:complexType>

<xs:element name="root">
        <xs:complexType>
                <xs:sequence>
                        <xs:element name="videogame" type="videogameType"
maxOccurs="unbounded"></xs:element>
                        <xs:element name="customer" type="customerType"
maxOccurs="unbounded"></xs:element>
```

```xml
                        <xs:element name="shop" type="shopType"
maxOccurs="unbounded"></xs:element>
                        <xs:element name="company" type="companyType"
maxOccurs="unbounded"></xs:element>
                </xs:sequence>
        </xs:complexType>

        <xs:key name="key1">
          <xs:selector xpath="videogame"/>
          <xs:field xpath="@videogame_id"/>
    </xs:key>

    <xs:key name="key2">
        <xs:selector xpath="customer"/>
        <xs:field xpath="@customer_id"/>
    </xs:key>

    <xs:key name="key3">
        <xs:selector xpath="shop"/>
        <xs:field xpath="@shop_id"/>
    </xs:key>

     <xs:key name="key4">
        <xs:selector xpath="company"/>
        <xs:field xpath="@company_id"/>
    </xs:key>

      <xs:keyref name="keyref1" refer="key1">
        <xs:selector xpath="shop/videogame"/>
        <xs:field xpath="@videogame_id"/>
    </xs:keyref>

    <xs:keyref name="keyref2" refer="key2">
        <xs:selector xpath="shop/customer"/>
        <xs:field xpath="@customer_id"/>
    </xs:keyref>

    <xs:keyref name="keyref3" refer="key3">
        <xs:selector xpath="company/shop"/>
        <xs:field xpath="@shop_id"/>
    </xs:keyref>

    <xs:keyref name="keyref4" refer="key3">
        <xs:selector xpath="customer/shop"/>
        <xs:field xpath="@shop_id"/>
    </xs:keyref>

</xs:element>

</xs:schema>
```

2 feladat

2a)

package hu.domparse.wf8h4f;

```java
import java.io.File;

import javax.xml.parsers.DocumentBuilderFactory;

import javax.xml.parsers.DocumentBuilder;

import org.w3c.dom.Document;

import org.w3c.dom.NodeList;

import org.w3c.dom.Node;

import org.w3c.dom.NamedNodeMap;


public class DOMReadWF8H4F {

public static void main(String[] args) {

    try {

        DocumentBuilderFactory dbFactory = DocumentBuilderFactory.newInstance();

        DocumentBuilder dBuilder = dbFactory.newDocumentBuilder();

        Document doc = dBuilder.parse(new File("src/XMLWF8H4F.xml"));

        doc.getDocumentElement().normalize();


        printNode(doc.getDocumentElement(), "+");


    } catch (Exception e) {

        e.printStackTrace();

    }
  }


public static void printNode(Node rootNode, String spacer) {
if (rootNode.getNodeType() == Node.ELEMENT_NODE) {
```

```java
    System.out.print(spacer + rootNode.getNodeName() + " -> " + (rootNode.hasAttributes()?
printAttributes(rootNode)+"\n" : rootNode.getFirstChild().getNodeValue().replace("\n",
"")+"\n"));



}
  NodeList nl = rootNode.getChildNodes();

  for (int i = 0; i < nl.getLength(); i++)

    printNode(nl.item(i), spacer + "   ");
}


public static String printAttributes(Node mainNode)
  {
    NamedNodeMap attributeList = mainNode.getAttributes();

    String attributes = " " + attributeList.item(0).getNodeName() + ":"
+attributeList.item(0).getNodeValue();

    for(int i = 1; i < attributeList.getLength(); i++)

    {

      Node attribute = attributeList.item(i);

      attributes += (", " + attribute.getNodeName() + ":" +attribute.getNodeValue());

    }
    return attributes;

  }



}
```

2b) adatmódosítás

package hu.domparse.wf8h4f;

import java.io.File;

```java
import javax.xml.parsers.DocumentBuilder;

import javax.xml.parsers.DocumentBuilderFactory;

import javax.xml.transform.TransformerFactory;

import javax.xml.transform.dom.DOMSource;

import javax.xml.transform.stream.StreamResult;

import javax.xml.transform.Transformer;

import javax.xml.transform.TransformerException;


import org.w3c.dom.Document;

import org.w3c.dom.Element;

import org.w3c.dom.Node;

import org.w3c.dom.NodeList;



public class DOMModifyWF8H4F {

public static void main(String[] args) {

try {

        DocumentBuilderFactory dbFactory = DocumentBuilderFactory.newInstance();

    DocumentBuilder dBuilder = dbFactory.newDocumentBuilder();

    Document doc = dBuilder.parse(new File("src/XMLWF8H4F.xml"));

    doc.getDocumentElement().normalize();


    Node root = doc.getDocumentElement();


    search(root, "videogame", "name", "xd", "1");
```

```java
            modifyDocument(doc, new File("src/XMLWF8H4F.xml"));


    }catch (Exception e) {

        e.printStackTrace();

    }


}


public static void search(Node root, String parent, String child, String replace, String id) {


    NodeList list = root.getChildNodes();

    for (int i = 0; i < list.getLength(); i++) {

            Node node = list.item(i);

            if (node.getNodeType() == Node.ELEMENT_NODE) {

                Element eElement = (Element) node;

                if (parent.equals(eElement.getNodeName()) &&
eElement.getAttributes().getNamedItem(parent+"_id").getNodeValue().equals(id)) {

                    Element foundElement = (Element)
eElement.getElementsByTagName(child).item(0);

                    foundElement.setTextContent(replace);

                }

            }

        }


}


public static void modifyDocument(Document document, File xmlFile) throws
TransformerException

{
```

```java
TransformerFactory transformerFactory = TransformerFactory.newInstance();

Transformer transformer = transformerFactory.newTransformer();

DOMSource source = new DOMSource(document);

StreamResult result = new StreamResult(xmlFile);


transformer.transform(source, new StreamResult(System.out));

transformer.transform(source, result);
}


}
```