Jegyzőkönyv

**Adatkezelés XML környezetben**

**Féléves feladat**

**Bolt nyilvántartó redszere**

Készítette: **Szivós Ádám**

Nepunkód: **B9ZL3Q**

1. **A feladat leírása:**

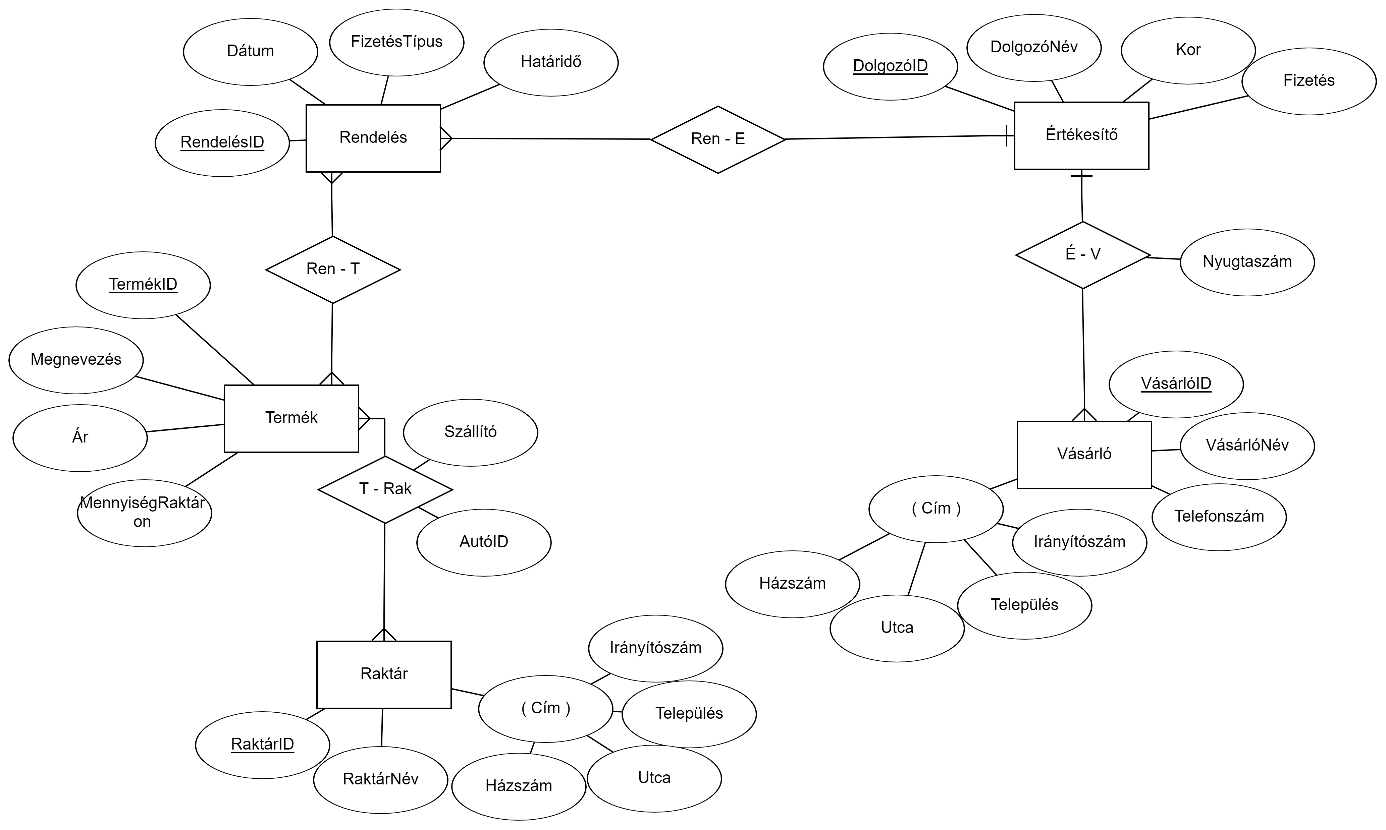
Én egy bolt nyilvántartó rendszeréről készítettem a beadandómat.

5 objektumból áll, köztük a *vásárló, értékesítő, rendelés, termék, raktár*. Mind több attribútummal rendelkeznek. A vásárló elemnek van egy neve, telefonszáma és címe, ezek alapján tudja a bolt azonosítani a vevőt. Az értékesítőnek is szerepelnek az alapdatai, amik a cégnek kellenek. Ezek a név, a kor és a fizetés. A vásárlót és az értékesítőt egy 1:N kapcsolat köt össze, aminek van egy attribútuma, a nyugtaszám, ezzel könnyen be lehet azonosítani a tranzakciót. A rendelés elemnek van dátuma, teljesítési ideje és egy fizetés típusa attribútuma. A dátumot String-ként tárolom és csak informatív jellegű. A fizetés típusa a szállítónak hasznos, hogy a kiszállítás végén számítania kell-e valamilyen tranzakcióra. Ezt az elemet az értékesítőve szintén egy 1:N kapcsolat köti össze. A terméknek is van neve, azon kívül egy ára és egy darabszáma. Ezt egy N:N kapcsolat köti össze a rendelés elemmel. Végül a raktár elem, aminek van egy neve és egy címe. Ezt is egy N:N kapcsolat köti össze a termékkel, de ennek a kapcsolatnak van 2 attribútuma: egy autó ID és egy szállító név.

Soron lehet követni egy tranzakció lefolyását a vásárlótól a kiszállításig.

**1. feladat**

**1a, Az adatbázis ER modell:**



**1b, Az adatbázis konvertálása XDM modellre:**

**1c, Az XDM modell alapján XML dokumentum készítése:**

<?xml version="1.0" encoding="UTF-8" standalone="no"?><?xml-model href="XMLSchemaB9ZL3Q.xsd" type="application/xml" schematypens="http://www.w3.org/2001/XMLSchema"?><bolt>

    <vasarlok>

        <vasarlo vid="v1">

            <vnev>Vas Alex</vnev>

            <telefonszam>06201234566</telefonszam>

            <lakcim>

                <iranyitoszam>3333</iranyitoszam>

                <telepules>Abaujteleki</telepules>

                <utca>Petofi</utca>

                <hazszam>2</hazszam>

            </lakcim>

        </vasarlo>

        <vasarlo vid="v2">

            <vnev>Kis Béla</vnev>

            <telefonszam>06704446667</telefonszam>

            <lakcim>

                <iranyitoszam>1463</iranyitoszam>

                <telepules>Peterujfalva</telepules>

                <utca>Fo</utca>

                <hazszam>33</hazszam>

            </lakcim>

        </vasarlo>

        <vasarlo vid="v3">

            <vnev>Kovacs Nora</vnev>

            <telefonszam>06307653486</telefonszam>

            <lakcim>

                <iranyitoszam>3325</iranyitoszam>

                <telepules>Felsoignac</telepules>

                <utca>Pesti</utca>

                <hazszam>13</hazszam>

            </lakcim>

        </vasarlo>

    </vasarlok>

    <ertekesitok>

        <ertekesito eid="e1">

            <enev>Petting Elek</enev>

            <kor>34</kor>

            <fizetes>210000</fizetes>

        </ertekesito>

        <ertekesito eid="e2">

            <enev>Kiss Tivadar</enev>

            <kor>55</kor>

            <fizetes>270000</fizetes>

        </ertekesito>

        <ertekesito eid="e3">

            <enev>Laszlo Evelin</enev>

            <kor>21</kor>

            <fizetes>190000</fizetes>

        </ertekesito>

    </ertekesitok>

    <rendelesek>

        <rendeles rid="r1">

            <datum>2021.10.12.</datum>

            <hatarido>2022.02.01.</hatarido>

            <fiztip>KP</fiztip>

        </rendeles>

        <rendeles rid="r2">

            <datum>2021.12.15.</datum>

            <hatarido>2022.01.08.</hatarido>

            <fiztip>utalas</fiztip>

        </rendeles>

        <rendeles rid="r2">

            <datum>2021.09.01.</datum>

            <hatarido>2022.12.23.</hatarido>

            <fiztip>kartya</fiztip>

        </rendeles>

    </rendelesek>

    <termekek>

        <termek tid="t1">

            <tnev>Zold 3-lyuku feltegla</tnev>

            <raktaron>166</raktaron>

            <ar>33</ar>

        </termek>

        <termek tid="t2">

            <tnev>Piros 8-lyuku feltegla</tnev>

            <raktaron>300</raktaron>

            <ar>553</ar>

        </termek>

        <termek tid="t3">

            <tnev>Ekru 4-lyuku feltegla</tnev>

            <raktaron>29</raktaron>

            <ar>99999</ar>

        </termek>

    </termekek>

    <raktarak>

        <raktar raid="ra1">

            <rnev>Zsolcai telephely</rnev>

            <cim>

                <iranyitoszam>2263</iranyitoszam>

                <telepules>Alsozsolca</telepules>

                <utca>Miskolci</utca>

                <hazszam>43</hazszam>

            </cim>

        </raktar>

        <raktar raid="ra2">

            <rnev>Kapolnai raktar</rnev>

            <cim>

                <iranyitoszam>4325</iranyitoszam>

                <telepules>Kapolna</telepules>

                <utca>Ersek</utca>

                <hazszam>1</hazszam>

            </cim>

        </raktar>

        <raktar raid="ra2">

            <rnev>Pesti telephely</rnev>

            <cim>

                <iranyitoszam>1111</iranyitoszam>

                <telepules>Budapest</telepules>

                <utca>Ipari park 2</utca>

                <hazszam>2</hazszam>

            </cim>

        </raktar>

    </raktarak>

    <t\_rak>

        <kapcsolat trakid="ka1">

            <idtermek refID="t1"/>

            <idraktar refID="ra1"/>

            <autoid>rgh-324</autoid>

            <szallito>Kis Tivadar</szallito>

        </kapcsolat>

        <kapcsolat trakid="ka2">

            <idtermek refID="t2"/>

            <idraktar refID="ra2"/>

            <autoid>frg-123</autoid>

            <szallito>Nagy Peter</szallito>

        </kapcsolat>

        <kapcsolat trakid="ka3">

            <idtermek refID="t3"/>

            <idraktar refID="ra3"/>

            <autoid>mnz-734</autoid>

            <szallito>Toth Laszlo</szallito>

        </kapcsolat>

    </t\_rak>

    <e\_v>

        <kapcsolat evid="kb1">

            <idertekesito refID="e1"/>

            <idvasarlo refID="v1"/>

            <nyugtaszam>756396428</nyugtaszam>

        </kapcsolat>

        <kapcsolat evid="kb2">

            <idertekesito refID="e2"/>

            <idvasarlo refID="v2"/>

            <nyugtaszam>5375932366</nyugtaszam>

        </kapcsolat>

        <kapcsolat evid="kb3">

            <idertekesito refID="e3"/>

            <idvasarlo refID="v3"/>

            <nyugtaszam>2753927429</nyugtaszam>

        </kapcsolat>

    </e\_v>

</bolt>

**1d, Az XML dokumentum alapján XMLSchema készítése:**

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

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

    <xs:element name="bolt">

        <xs:complexType>

            <xs:sequence>

                <xs:element name="vasarlok" type="vasarloktype" maxOccurs="1"/>

                <xs:element name="ertekesitok" type="ertekesitoktype" maxOccurs="1"/>

                <xs:element name="rendelesek" type="rendelesektype" maxOccurs="1"/>

                <xs:element name="termekek" type="termekektype" maxOccurs="1"/>

                <xs:element name="raktarak" type="raktaraktype" maxOccurs="1"/>

                <xs:element name="t\_rak" type="t\_raktype" maxOccurs="1"/>

                <xs:element name="e\_v" type="e\_vtype" maxOccurs="1"/>

            </xs:sequence>

        </xs:complexType>

    </xs:element>

    <xs:complexType name="vasarloktype">

        <xs:sequence>

            <xs:element name="vasarlo" maxOccurs="unbounded">

                <xs:complexType>

                    <xs:sequence>

                        <xs:element name="vnev" type="xs:string"/>

                        <xs:element name="telefonszam" type="xs:string"/>

                        <xs:element name="cim" type="cimtype"/>

                    </xs:sequence>

                    <xs:attribute name="vid" type="xs:ID" use="required"/>

                </xs:complexType>

            </xs:element>

        </xs:sequence>

    </xs:complexType>

    <xs:complexType name="ertekesitoktype">

        <xs:sequence>

            <xs:element name="ertekesito" maxOccurs="unbounded">

                <xs:complexType>

                    <xs:sequence>

                        <xs:element name="nev" type="xs:string"/>

                        <xs:element name="kor" type="xs:integer"/>

                        <xs:element name="fizetes" type="xs:integer"/>

                    </xs:sequence>

                    <xs:attribute name="eid" type="xs:ID" use="required"/>

                </xs:complexType>

            </xs:element>

        </xs:sequence>

    </xs:complexType>

    <xs:complexType name="rendelesektype">

        <xs:sequence>

            <xs:element name="rendeles" maxOccurs="unbounded">

                <xs:complexType>

                    <xs:sequence>

                        <xs:element name="datum" type="xs:string"/>

                        <xs:element name="hatarido" type="xs:string"/>

                        <xs:element name="fiztip" type="xs:string"/>

                    </xs:sequence>

                    <xs:attribute name="rid" type="xs:ID" use="required"/>

                </xs:complexType>

            </xs:element>

        </xs:sequence>

    </xs:complexType>

    <xs:complexType name="termekektype">

        <xs:sequence>

            <xs:element name="termek" maxOccurs="unbounded">

                <xs:complexType>

                    <xs:sequence>

                        <xs:element name="tnev" type="xs:string"/>

                        <xs:element name="raktaron" type="xs:integer"/>

                        <xs:element name="ar" type="xs:integer"/>

                    </xs:sequence>

                    <xs:attribute name="tid" type="xs:ID" use="required"/>

                </xs:complexType>

            </xs:element>

        </xs:sequence>

    </xs:complexType>

    <xs:complexType name="raktaraktype">

        <xs:sequence>

            <xs:element name="raktar" maxOccurs="unbounded">

                <xs:complexType>

                    <xs:sequence>

                        <xs:element name="rnev" type="xs:string"/>

                        <xs:element name="cim" type="cimtype"/>

                    </xs:sequence>

                    <xs:attribute name="raid" type="xs:ID" use="required"/>

                </xs:complexType>

            </xs:element>

        </xs:sequence>

    </xs:complexType>

    <xs:complexType name="cimtype">

        <xs:sequence>

            <xs:element name="iranyitoszam" type="xs:string"/>

            <xs:element name="varos" type="xs:string"/>

            <xs:element name="utca" type="xs:string"/>

            <xs:element name="hazszam" type="xs:string"/>

        </xs:sequence>

    </xs:complexType>

    <xs:complexType name="t\_rak">

        <xs:sequence>

            <xs:element name="kapcsolat" maxOccurs="unbounded">

                <xs:complexType>

                    <xs:sequence>

                        <xs:element name="idtermek">

                            <xs:complexType>

                                <xs:attribute name="refID" type="xs:IDREF"/>

                            </xs:complexType>

                        </xs:element>

                        <xs:element name="idraktar">

                            <xs:complexType>

                                <xs:attribute name="refID" type="xs:IDREF"/>

                            </xs:complexType>

                        </xs:element>

                        <xs:element name="autoid" type="xs:string"/>

                        <xs:element name="szallito" type="xs:string"/>

                    </xs:sequence>

                    <xs:attribute name="id" type="xs:ID" use="required"/>

                </xs:complexType>

            </xs:element>

        </xs:sequence>

    </xs:complexType>

    <xs:complexType name="e\_v">

        <xs:sequence>

            <xs:element name="kapcsolat" maxOccurs="unbounded">

                <xs:complexType>

                    <xs:sequence>

                        <xs:element name="idertekesito">

                            <xs:complexType>

                                <xs:attribute name="refID" type="xs:IDREF"/>

                            </xs:complexType>

                        </xs:element>

                        <xs:element name="idvasarlo">

                            <xs:complexType>

                                <xs:attribute name="refID" type="xs:IDREF"/>

                            </xs:complexType>

                        </xs:element>

                        <xs:element name="nyugtaszam" type="xs:string"/>

                    </xs:sequence>

                    <xs:attribute name="id" type="xs:ID" use="required"/>

                </xs:complexType>

            </xs:element>

        </xs:sequence>

    </xs:complexType>

</xs:schema>

**2. feladat**

**2a, Adatolvasás:**

package hu.domparse.b9zl3q;

import java.io.\*;

import javax.xml.parsers.\*;

import org.xml.sax.SAXException;

import org.w3c.dom.\*;

public class DOMReadB9ZL3Q {

    public static void main(String[] args) {

        // DOMBuilder-rel  DOM letrheozasa

        DocumentBuilderFactory documentBuilderFactory = DocumentBuilderFactory.newInstance();

        try {

            DocumentBuilder documentBuilder = documentBuilderFactory.newDocumentBuilder();

            Document document = documentBuilder.parse(new File("XMLB9ZL3Q.xml"));

            document.getDocumentElement().normalize();

            //root elem megadasa

            Node root = document.getDocumentElement();

            listAll(root);

        } catch (ParserConfigurationException | SAXException | IOException e) {

            e.printStackTrace();

        }

    }

    public static void listAll(Node root) {

        //a child elemek megadasa

        NodeList nodeList = root.getChildNodes();

        String vasarlo = "Vasarlok\n\n";

        String ertekesito = "Ertekesitok\n\n";

        String rendeles = "Rendelesek\n\n";

        String termek = "Termekek\n\n";

        String raktar = "Raktarak\n\n";

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

            Node node = nodeList.item(i);

            if(node.getNodeType() == Node.ELEMENT\_NODE) {

                //a child elemek child-jainak kivalassztasa

                NodeList subNodeList = nodeList.item(i).getChildNodes();

                for (int j = 0; j < subNodeList.getLength(); j++) {

                    Node subNode = subNodeList.item(j);

                    //kigyujtes a definialt string-ekbe

                    if(subNode.getNodeType() == Node.ELEMENT\_NODE) {

                        vasarlo += oneNode(subNode,"vasarlo");

                        ertekesito += oneNode(subNode,"ertekesito");

                        rendeles += oneNode(subNode,"rendeles");

                        termek += oneNode(subNode,"termek");

                        raktar += oneNode(subNode,"raktar");

                    }

                }

            }

        }

        //kiiratas

        System.out.println(vasarlo);

        System.out.println(ertekesito);

        System.out.println(rendeles);

        System.out.println(termek);

        System.out.println(raktar);

    }

    public static String oneNode(Node subNode, String x) {

        String out = "";

        if(subNode.getNodeName().equals(x)) {

            if(subNode.getAttributes().getLength()>0)

            //ha a node megegyezik x-el és van attribute-ja, akkor kiiroik

            {

                out +=x + " ID : " + subNode.getAttributes().item(0).getTextContent()+"\n";

            }

            NodeList subSubNodeList = subNode.getChildNodes();

            for (int k = 0; k < subSubNodeList.getLength(); k++) {

                Node subSubNode = subSubNodeList.item(k);

                if(subSubNode.getNodeType() == Node.ELEMENT\_NODE) {

                    //id alapjan elkuloniti

                    switch(subSubNode.getNodeName()) {

                    case "vid":

                    case "eid":

                    case "rid":

                    case "tid":

                    case "raid":

                        //az elso attribute-omokat rakja bele a String-be

                        out +=subSubNode.getNodeName() + " : " + subSubNode.getAttributes().item(0).getTextContent()+"\n";

                    break;

                    case "cim":

                        NodeList subSubSubNode = subSubNode.getChildNodes();

                        for (int l = 0; l < subSubSubNode.getLength(); l++) {

                            if (subSubSubNode.item(l).getNodeType() == Node.ELEMENT\_NODE) {

                                out += subSubNode.getNodeName() + "-" + subSubSubNode.item(l).getNodeName() + " : " + subSubSubNode.item(l).getTextContent()+"\n";

                            }

                        }

                    break;

                    default:

                    //az elem nevet es a kontextust String-be irja

                    out += subSubNode.getNodeName() + " : " + subSubNode.getTextContent()+"\n";

                    }

                }

            }

            out+="\n";

        }

        return out;

    }

}

**2b, Adatmódosítás:**

package hu.domparse.b9zl3q;

import java.io.\*;

import java.util.\*;

import javax.xml.parsers.\*;

import javax.xml.transform.\*;

import javax.xml.transform.dom.DOMSource;

import javax.xml.transform.stream.StreamResult;

import org.w3c.dom.\*;

import org.xml.sax.SAXException;

public class DOMModifyB9ZL3Q {

    public static void main(String[] args){

        // DOMBuilder-rel  DOM letrheozasa

        DocumentBuilderFactory documentBuilderFactory = DocumentBuilderFactory.newInstance();

        try {

            DocumentBuilder documentBuilder = documentBuilderFactory.newDocumentBuilder();

            Document document = documentBuilder.parse(new File("XMLB9ZL3Q.xml"));

            //normalizalas

            document.getDocumentElement().normalize();

            Scanner scanner = new Scanner(System.in);

            boolean exit=true;

            while(exit) {

                //switch case segitsegevel menu

                System.out.println("0: Kilepes\n1: Termek ar mododsitas\n2: Vasarlo telefonszam mododsitas\n3: Raktar nev mododsitas");

                switch(scanner.nextInt()) {

                case 1 : termAr(document, scanner);break;

                case 2 : vasTel(document, scanner);break;

                case 3 : rakNev(document, scanner);break;

                default : exit=false;

                }

            }

            scanner.close();

            //kiirja az uj doksiba

            writeToXml(document);

        } catch (ParserConfigurationException | SAXException | IOException | TransformerException | InputMismatchException e) {

            e.printStackTrace();

        }

    }

    public static void termAr(Document document, Scanner scanner) {

        //kikeresi tag alapjan

        NodeList nodeList = document.getElementsByTagName("termek");

        System.out.println("Termek ID: ");

        String input = scanner.next();

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

            Node node = nodeList.item(i);

            if (node.getNodeType() == Node.ELEMENT\_NODE) {

                //kikeresi id alapjan

                if (node.getAttributes().getNamedItem("tid").getTextContent().equals(input)) {

                    //lekeri a child node-okat

                    NodeList subNodeList = node.getChildNodes();

                    for (int j = 0; j < subNodeList.getLength(); j++) {

                        Node subNode = subNodeList.item(j);

                        if (subNode.getNodeName().equals("ar")) {

                            System.out.println("Uj ar: ");

                            subNode.setTextContent(scanner.next());

                        }

                    }

                }

            }

        }

    }

    public static void vasTel(Document document, Scanner scanner) {

        //kikeresi tag alapjan

        NodeList nodeList = document.getElementsByTagName("vasarlo");

        System.out.println("Vasarlo ID: ");

        String input = scanner.next();

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

            Node node = nodeList.item(i);

            if (node.getNodeType() == Node.ELEMENT\_NODE) {

                //kikeresi id alapjan

                if (node.getAttributes().getNamedItem("vid").getTextContent().equals(input)) {

                    //lekeri a child node-okatv

                    NodeList subNodeList = node.getChildNodes();

                    for (int j = 0; j < subNodeList.getLength(); j++) {

                        Node subNode = subNodeList.item(j);

                        if (subNode.getNodeName().equals("telefonszam")) {

                            System.out.println("Uj telefonszam: ");

                            subNode.setTextContent(scanner.next());

                        }

                    }

                }

            }

        }

    }

    public static void rakNev(Document document, Scanner scanner) {

        //kikeresi tag alapjan

        NodeList nodeList = document.getElementsByTagName("raktar");

        System.out.println("Raktar ID: ");

        String input = scanner.next();

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

            Node node = nodeList.item(i);

            if (node.getNodeType() == Node.ELEMENT\_NODE) {

                //kikeresi id alapjan

                if (node.getAttributes().getNamedItem("raid").getTextContent().equals(input)) {

                    //lekeri a child node-okat

                    NodeList subNodeList = node.getChildNodes();

                    for (int j = 0; j < subNodeList.getLength(); j++) {

                        Node subNode = subNodeList.item(j);

                        if (subNode.getNodeName().equals("rnev")) {

                            System.out.println("Uj nev: ");

                            String name = scanner.next();

                            name +=scanner.nextLine();

                            subNode.setTextContent(name);

                        }

                    }

                }

            }

        }

    }

    //kiiras xml-be

    public static void writeToXml(Document document) throws TransformerException, UnsupportedEncodingException {

        TransformerFactory transformerFactory = TransformerFactory.newInstance();

        Transformer transf = transformerFactory.newTransformer();

        DOMSource source = new DOMSource(document);

        StreamResult console = new StreamResult(System.out);

        StreamResult file = new StreamResult(new File("XMLB9ZL3Q.out.xml"));

        transf.transform(source, console);

        transf.transform(source, file);

    }

}

**2c, Adat lekérdezés:**

package hu.domparse.b9zl3q;

import java.io.\*;

import java.util.\*;

import javax.xml.parsers.\*;

import org.w3c.dom.\*;

import org.xml.sax.SAXException;

public class DOMQueryB9ZL3Q {

    public static void main(String[] args){

        // DOMBuilder-rel  DOM letrheozasa

        DocumentBuilderFactory documentBuilderFactory = DocumentBuilderFactory.newInstance();

        try {

            DocumentBuilder documentBuilder = documentBuilderFactory.newDocumentBuilder();

            Document document = documentBuilder.parse(new File("XMLB9ZL3Q.xml"));

            //normalizalas

            document.getDocumentElement().normalize();

            Scanner scanner = new Scanner(System.in);

            boolean exit=true;

            while(exit) {

                //switch case segitsegevel menu

                System.out.println("0: Kilepes\n1: Bizonyos ar feletti termek \n2: Rendeles egyezo hatarido \n3: Adott kornal idosebb elado ");

                switch(scanner.nextInt()) {

                case 1 : termAr(document, scanner);break;

                case 2 : rendHat(document, scanner);break;

                case 3 : ertKor(document, scanner);break;

                default : exit=false;

                }

            }

            scanner.close();

        } catch (ParserConfigurationException | SAXException | IOException e) {

            e.printStackTrace();

        }

    }

    //termek aranak megvaltoztatasa

    public static void termAr(Document document,Scanner scanner) {

        //kikeresi tag alapjan

        NodeList nodeList = document.getElementsByTagName("termek");

        System.out.println("Adja meg minel nagyobb: ");

        Long input = scanner.nextLong();

        System.out.println("\n");

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

            Node node = nodeList.item(i);

            if(node.getNodeType() == Node.ELEMENT\_NODE) {

                //lekeri a child node-okat

                NodeList subNodeList = node.getChildNodes();

                for (int j = 0; j < subNodeList.getLength(); j++) {

                Node subNode = subNodeList.item(j);

                if(subNode.getNodeType() == Node.ELEMENT\_NODE) {

                    if (subNode.getNodeName().equals("ar") )

                        if (Long.parseLong(subNode.getTextContent()) > input)

                            listSub(node, "Termek");

                    }

                }

            }

        }

    }

    public static void rendHat(Document document,Scanner scanner) {

        //kikeresi tag alapjan

        NodeList nodeList = document.getElementsByTagName("rendeles");

        System.out.println("Adja meg a hataridot: ");

        String input = scanner.next();

        System.out.println("\n");

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

            Node node = nodeList.item(i);

            if(node.getNodeType() == Node.ELEMENT\_NODE) {

                //lekeri a child node-okat

                NodeList subNodeList = node.getChildNodes();

                for (int j = 0; j < subNodeList.getLength(); j++) {

                Node subNode = subNodeList.item(j);

                if(subNode.getNodeType() == Node.ELEMENT\_NODE) {

                    if (subNode.getNodeName().equals("hatarido") )

                        if (subNode.getTextContent().equals(input))

                            listSub(node, "Rendeles");

                    }

                }

            }

        }

    }

    public static void ertKor(Document document,Scanner scanner) {

        //kikeresi tag alapjan

        NodeList nodeList = document.getElementsByTagName("ertekesito");

        System.out.println("Adja meg a kort: ");

        int input = scanner.nextInt();

        System.out.println("\n");

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

            Node node = nodeList.item(i);

            if(node.getNodeType() == Node.ELEMENT\_NODE) {

                //lekeri a child node-okat

                NodeList subNodeList = node.getChildNodes();

                for (int j = 0; j < subNodeList.getLength(); j++) {

                Node subNode = subNodeList.item(j);

                if(subNode.getNodeType() == Node.ELEMENT\_NODE) {

                    if (subNode.getNodeName().equals("kor") )

                        if (Integer.parseInt(subNode.getTextContent()) > input)

                            listSub(node, "Ertekesito");

                    }

                }

            }

        }

    }

    private static void listSub(Node node, String tag) {

        //id alapjan az elemek kilistazasa

        String out="";

        out +=tag + " ID : " + node.getAttributes().item(0).getTextContent()+"\n";

            Element element = (Element) node;

                switch(tag) {

                case "Termek":

                    out += element.getElementsByTagName("tnev").item(0).getNodeName() + " : " +element.getElementsByTagName("tnev").item(0).getTextContent()+"\n";

                    out += element.getElementsByTagName("ar").item(0).getNodeName() + " : " +element.getElementsByTagName("ar").item(0).getTextContent()+"\n";

                break;

                case "Rendeles":

                    out += element.getElementsByTagName("datum").item(0).getNodeName() + " : " +element.getElementsByTagName("datum").item(0).getTextContent()+"\n";

                    out += element.getElementsByTagName("hatarido").item(0).getNodeName() + " : " +element.getElementsByTagName("hatarido").item(0).getTextContent()+"\n";

                break;

                case "Ertekesito":

                    out += element.getElementsByTagName("enev").item(0).getNodeName() + " : " +element.getElementsByTagName("enev").item(0).getTextContent()+"\n";

                    out += element.getElementsByTagName("kor").item(0).getNodeName() + " : " +element.getElementsByTagName("kor").item(0).getTextContent()+"\n";

                break;

            }

        System.out.println(out);

    }

}