JEGYZŐKÖNYV

Adatkezelés XML környezetben

Féléves feladat

XML adatkezelő rendszer és DOM API programok megvalósítása

Készítette: **Varga Bence**
Neptunkód: **HK74CE**

Dátum: 2024.12.09.

Tartalomjegyzék

Bevezetés

A feladat egy XML adatkezelő rendszer megvalósítása és a dokumentum DOM API-val történő feldolgozása. Az első részben az adatbázis tervezésére és az XML dokumentum generálására történik. A második részben DOM API segítségével végzünk adatbeolvasást, írást, lekérdezést és módosítást.

A feladat leírása:

A projekt során egy XML adatkezelő rendszert készítettünk, amely egy fiktív rendszer adatainak kezelésére szolgál. A témakör az oktatás. Az adatbázis tervezése során 5 entitást és több kapcsolatot definiáltunk: 1:1, 1:N és N:M kapcsolatok. A rendszer az alábbi lépéseket valósította meg:

1. Az ER modell megtervezése és vizualizációja szabványos szimbólumokkal.
2. Az ER modell átalakítása XDM modellre.
3. Az XDM modell alapján validált XML dokumentum generálása, többszörös előfordulási elemekkel.
4. Az XML dokumentumhoz tartozó XMLSchema elkészítése, saját típusok definiálásával.
5. DOM API segítségével négy különböző Java osztály készült, amelyek az XML adatok kezelését végzik.

Első feladat

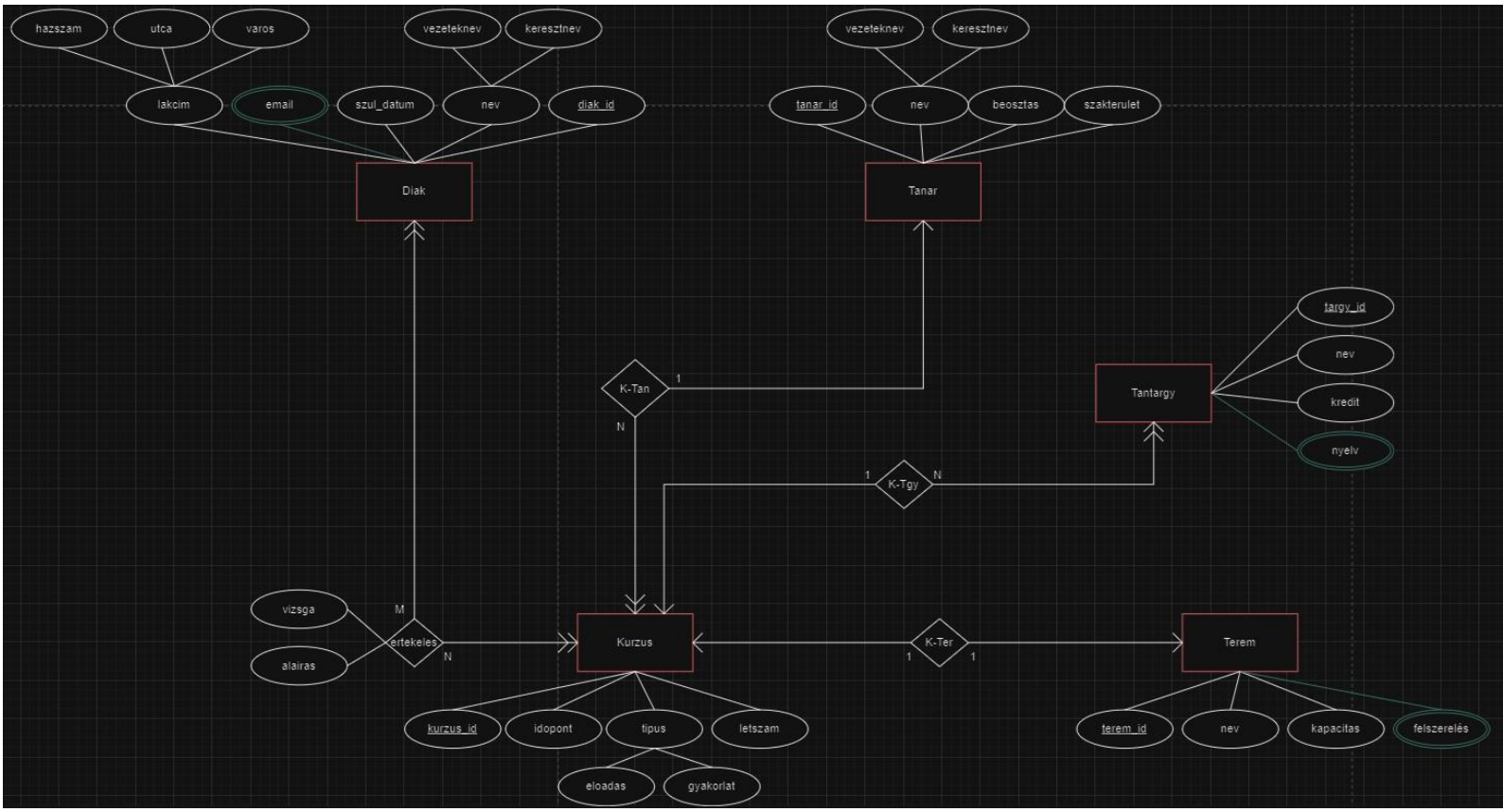
1.1 Az adatbázis ER modell tervezése

Megvalósítás:

Az ER modell 5 entitást tartalmaz: **Diák**, **Tanár**, **Tantárgy**, **Terem** és **Kurzus**. Az entitások kapcsolatai:

- **Diák-Kurzus:** N:M kapcsolat.
- **Tanár-Kurzus:** 1:N kapcsolat.
- **Kurzus-Tantárgy:** 1:N kapcsolat.
- **Kurzus-Terem:** 1:1 kapcsolat.

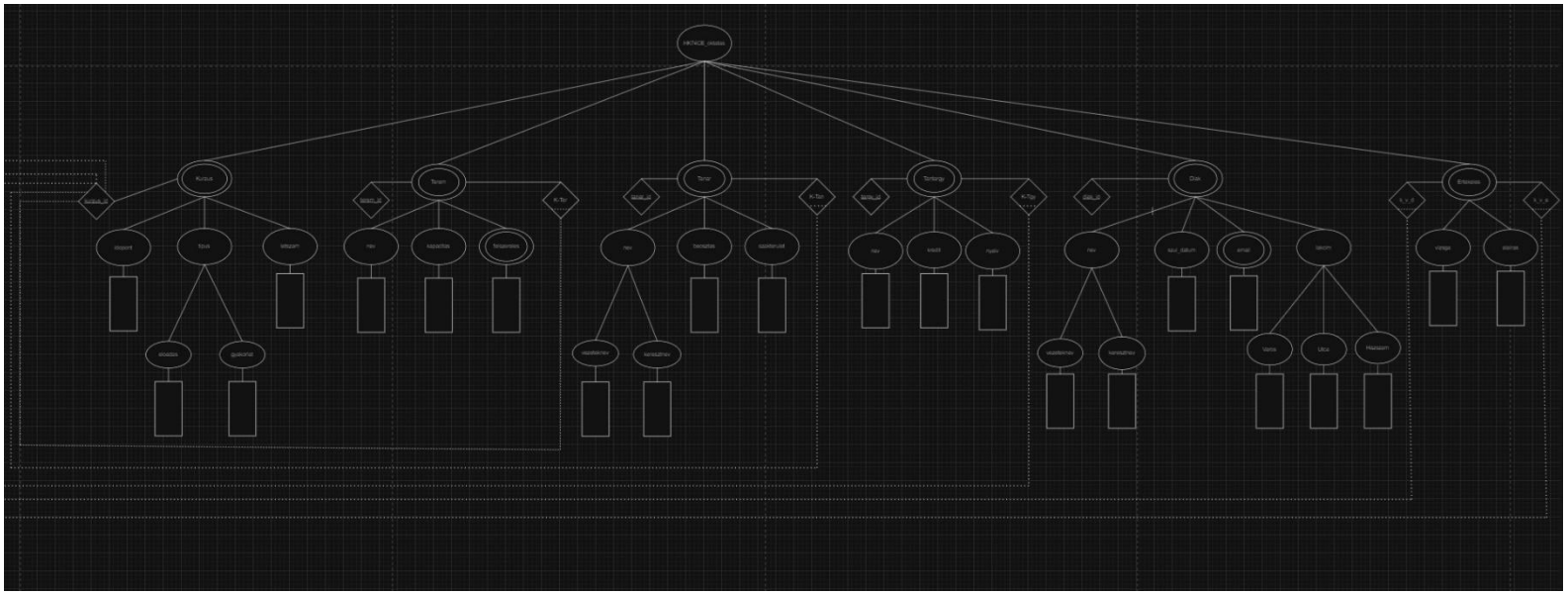
A modell tartalmaz összetett (pl. lakcím) és többértékű attribútumokat (pl.felszerelés).



1.2 Az adatbázis konvertálása XDM modellre

Megvalósítás:

Az XDM modell az ER modell hierarchikus reprezentációja. Az entitások csomópontként, az attribútumok pedig alcsomópontként jelennek meg. Az idegen kulcsok a kapcsolatok megvalósításához kerülnek felhasználásra.



1.3 Az XDM modell alapján XML dokumentum készítése

Megvalósítás:
Az XML dokumentumot a XDM modell alapján hoztuk létre. Minden entitásból legalább három példányt generáltunk, a kapcsolatok attribútumai pedig alárendelt csomópontként szerepelnek.

A kód: XML_HK74CE.xml

1.4 Az XML dokumentum alapján XMLSchema készítése

Megvalósítás:
Az XMLSchema fájl saját típusokat is definiál, amelyek tartalmazzák az entitások összetett attribútumait. Továbbá az elsődleges kulcsok és idegen kulcsok biztosítják a kapcsolatok integritását.

A kód: XMLSchemaHK74CE.xsd

Második feladat

2.1 Adatolvasás (DOMReadHK74CE.java)

Cél: Az XML dokumentum feldolgozása, az adatok blokk formájú kiírása konzolra és mentése fájlba.

kódrészlet:

```
// Az összes "Diak" elem lekérése
NodeList diakok = doc.getElementsByTagName("Diak");
// Minden diák feldolgozása
for (int i = 0; i < diakok.getLength(); i++) {
    Node node = diakok.item(i);
// Csak az elemtípusú csomópontokat dolgozza fel
    if (node.getNodeType() == Node.ELEMENT_NODE) {
        Element diak = (Element) node;


        // Diák adatainak kiolvasása (pl. DiakID, név, születési dátum)
        String diakID =
diak.getElementsByTagName("DiakID").item(0).getTextContent();
        String keresztnev =
diak.getElementsByTagName("Keresztnev").item(0).getTextContent();
        String vezeteknev =
diak.getElementsByTagName("Vezeteknev").item(0).getTextContent();
        String szulDatum =
diak.getElementsByTagName("SzulDatum").item(0).getTextContent();
        // Eredmények konzolra és fájlba írása
        String output = String.format("Diák ID: %s\nNév: %s %s\nSzületési dátum: %s\n",
diakID, keresztnev, vezeteknev, szulDatum);
        System.out.println(output);
        writer.println(output);
    }
}
```

Eredmény: Az adatok blokk formában jelennek meg a konzolon, és text fájl-ba mentésre kerül.

A kód: DomReadHK74CE.java

2.2 Adatírás (DOMWriteHK74CE.java)

Cél: Az XML dokumentum tartalmának fa struktúra formában történő konzolra írása és mentése új fájlba.

kódrészlet:

```
//TransformerFactory inicializálása
TransformerFactory transformerFactory = TransformerFactory.newInstance();
Transformer transformer = transformerFactory.newTransformer();

// A kimenet behúzásainak engedélyezése
transformer.setOutputProperty(OutputKeys.INDENT, "yes");

// Az XML dokumentum konzolra írása
StreamResult consoleResult = new StreamResult(System.out);
transformer.transform(new DOMSource(doc), consoleResult);

// Az XML dokumentum fájlba mentése
StreamResult fileResult = new StreamResult(new File("XML_HK74CE_1.xml"));
transformer.transform(new DOMSource(doc), fileResult); // Átalakítás és fájlba írás
```

Eredmény: Az XML tartalom mentve lett az XML_HK74CE_1.xml fájlba.

A kód: DOMWriteHK74CE.java

2.3 Adatlekérdezés (DOMQueryHK74CE.java)

Cél: Az XML dokumentum adatai alapján legalább 4 lekérdezés végrehajtása.

kódrészlet:

```
// Összes "Terem" elem lekérése
NodeList termek = doc.getElementsByTagName("Terem");
for (int i = 0; i < termek.getLength(); i++) { // Minden terem feldolgozása
   Node node = termek.item(i);
   if (node.getNodeType() == Node.ELEMENT_NODE) { // Csak az elemtípusú csomópontokat dolgozza fel
       Element terem = (Element) node;

// szűrés ülőhely alapján
       int kapacitas = Integer.parseInt(terem.getElementsByTagName("Kapacitas").item(0).getTextContent());

// Ha a kapacitás nagyobb, mint 100
 if (kapacitas > 100) {
           String teremID = terem.getElementsByTagName("TeremID").item(0).getTextContent();
           String nev = terem.getElementsByTagName("Nev").item(0).getTextContent();

// Szűrt adatok kiírása
           System.out.printf("Terem ID: %s, Név: %s, Kapacitás: %d\n", teremID, nev, kapacitas);
       }
   }
}
```

Eredmény: A konzolon megjelenik a szűrt adatok listája (pl. 100 ülőhelynél nagyobb kapacitású termek listája).

A kód: DOMQueryHK74CE.java

2.4 Adatmódosítás (DOMModifyHK74CE.java)

Cél: Az XML dokumentum módosítása legalább 4 adatnál, majd mentés új fájlba.

kód:

```
// Összes "Kurzus" elem lekérése
NodeList kurzusok = doc.getElementsByTagName("Kurzus");
for (int i = 0; i < kurzusok.getLength(); i++) { // Minden kurzus feldolgozása
    Node node = kurzusok.item(i);

// Csak az elemtípusú csomópontokat dolgozza fel
    if (node.getNodeType() == Node.ELEMENT_NODE) {
        Element kurzus = (Element) node;

        // Ha az ID megegyezik, növeljük a létszámot
        if (kurzus.getElementsByTagName("KurzusID").item(0).getTextContent().equals("KURZ001")) {
            int jelenlegiLetszam = Integer.parseInt(kurzus.getElementsByTagName("Letszam").item(0).getTextContent());

// Létszám növelése
kurzus.getElementsByTagName("Letszam").item(0).setTextContent(String.valueOf(jelenlegiLetszam + 5));
// Módosítás kiírása
            System.out.println("Kurzus KURZ001 létszáma növelve: " + (jelenlegiLetszam + 5));
            break;
        }
    }
}
```

Eredmény: A módosított XML tartalom mentve lett az XML_HK74CE_Modified.xml fájlba.

A kód: DOMModifyHK74CE.java