Jegyzőkönyv

Webes adatkezelő környezetek

Féléves feladat

Egyetemi Rendszer

Egyed Martin CLZPRE 2025. december 01.

Tartalom

Bevezetés

A jegyzőkönyv célja egy **Egyetemi Rendszer** adatszerkezetének megtervezése XML-alapú környezetben. A tervezési folyamat során először egy **ER (Entity-Relationship) modell** \ készül el, amely fogalmi szinten definiálja az egyedeket (pl. Hallgató, Tantárgy, Oktató) és azok kapcsolatait.

Ezt követi az ER modell konvertálása **XDM (XQuery and XPath Data Model)** \ modellre, ami a hierarchikus, faszerkezetű ábrázolást biztosítja az XML dokumentum számára . Az XDM modell alapján létrehozott XML dokumentum konkrét adatokkal szemlélteti a rendszert.

Az adatstruktúra formai leírása az **XML séma (XSD)** segítségével történik, ami rögzíti a szerkezetet, típusokat, és biztosítja az adatintegritást (pl. kulcsok és idegen kulcsok) .

A második rész a **DOM (Document Object Model)** szabványra fókuszál:

- **Adatolvasás:** A teljes XML dokumentum strukturált megjelenítése, a rövidített XML tagnevek magyar nyelvű megfelelőkre fordításával.
- **Adatlekérdezés:** Legalább négy, összefüggő adatokra vonatkozó hasznos információ kigyűjtése.
- **Adatmódosítás:** Legalább négy, adatintegritást megőrző módosítás végrehajtása az XML fában, a műveletek és az érintett csomópontok blokkos kiírásával.

1. Az ELSŐ feladat címe

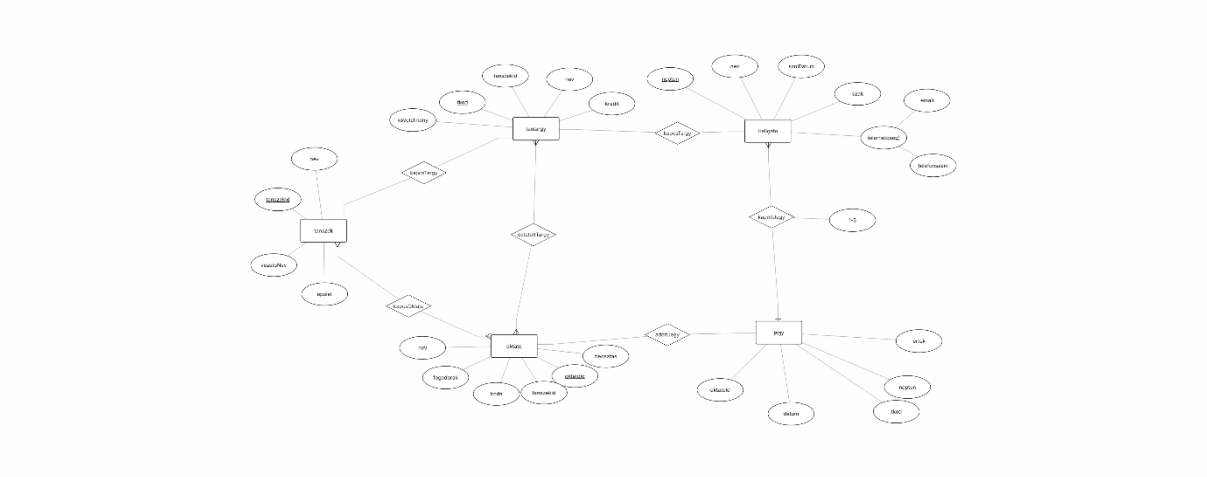
Fogalmi és logikai tervezés, majd XML és XSD létrehozása.

1.1 Az adatbázis ER modell tervezése megvalósítása

Az Egyetemi Rendszer logikai felépítését az ER modell ábrázolja, amely a főbb entitásokat és azok kapcsolatait mutatja be.

- **Kulcsfontosságú entitások:** Hallgató, Tantárgy, Oktató, Jegy, Tanszék.
- **Kapcsolatok:** Egy Hallgató több Tantárgyat vehet fel, és egy Tantárgyat több Hallgató vehet fel (több-a-több). Egy Tantárgyat több Oktató oktathat, és egy Oktató több Tantárgyat oktathat (több-a-több). A **Jegy** entitás a Hallgató és a Tantárgy közötti kapcsolatot valósítja meg.

Az ER modell alapot biztosít a hatékony adatbázis-tervezéshez és az adatok későbbi kezeléséhez, lekérdezéséhez.



Egyedek és tulajdonságok:

1.1.1. Hallgató (hallgato)

Tulajdonságok:

- **neptun** - egyedi azonosító (primary key)
- **nev** - a hallgató teljes neve
- **szuletesiDatum** - születési dátum (ÉÉÉÉ-HH-NN formátumban)
- **elerhetoseg**:
    - **email** - e-mail cím
    - **telefonszam** - telefonszám
- **szak** - hallgatói szak (pl. mérnökinformatikus)

Magyarázat: Ez az entitás reprezentálja az egyetemre beiratkozott hallgatókat. Az elérhetőségi adatok egy al-elemen belül vannak összefoglalva.

1.1.2. Tantárgy (tantargy)

Tulajdonságok:

- **tkod** - tantárgy kódja (primary key)
- **nev** - tantárgy neve
- **kredit** - kreditpont értéke
- **tanszekId** - a tantárgyat gondozó tanszék azonosítója (foreign key)
- **kovetelmeny** - követelmény típusa (pl. vizsga, aláírás)

Magyarázat: Ez az entitás a különböző, az egyetemen oktatott tantárgyakat tartalmazza. A **tanszekId** segítségével kapcsolódik a Tanszék entitáshoz.

1.1.3. Oktató (oktato)

Tulajdonságok:

- **oktatoId** - egyedi azonosító (primary key)
- **nev** - oktató teljes neve
- **beosztas** - beosztás (pl. professzor, adjunktus)
- **tanszekId** - az oktató tanszékének azonosítója (foreign key)
- **iroda** - iroda száma
- **fogadorak** - fogadóórák ideje

Magyarázat: Az oktatók entitása tartalmazza az oktatási feladatot ellátó személyek adatait. Egy oktató több tantárgyat is oktathat.

1.1.4. Jegy (jegy)

Tulajdonságok:

- **neptun** - hallgató Neptun kódja (foreign key)
- **tkod** - tantárgy kódja (foreign key)
- **ertek** - a szerzett jegy (1-5 pont közötti értékelés)
- **datum** - a jegy beírásának dátuma
- **oktatoId** - az értékelést beíró oktató azonosítója (foreign key)

Magyarázat: Ez az entitás rögzíti a Hallgató és Tantárgy közötti N:M kapcsolatot, valamint a szerzett jegyet.

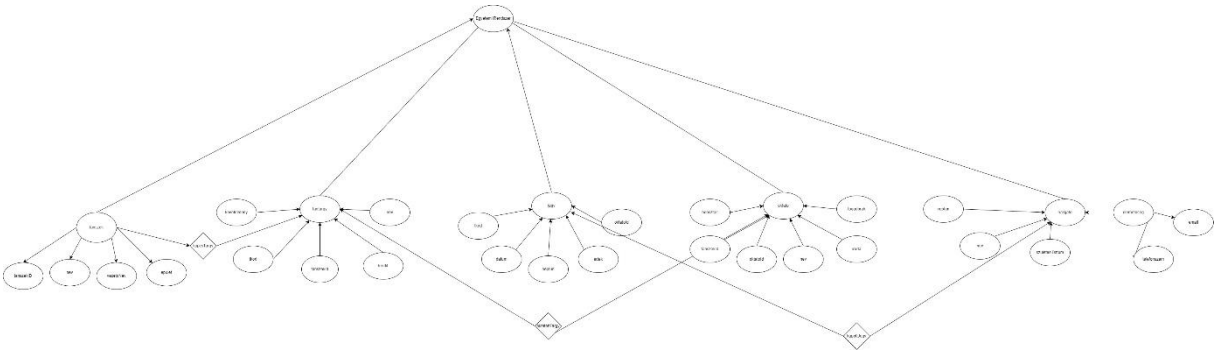
1.1.5. Tanszék (tanszek)

Tulajdonságok:

- **tanszekId** - egyedi azonosító (primary key)
- **nev** - tanszék neve
- **vezetoNev** - tanszékvezető neve
- **epulet** - elhelyezkedés (épület és szobaszám)

Magyarázat: A Tanszék entitás a szervezeti egységeket írja le, amelyek a tantárgyak gondozásáért és az oktatók foglalkoztatásáért felelnek.

1.2 Az adatbázis konvertálása XDM modellre



Az XDM modell a fogalmi ER modell faszerkezetű ábrázolása, amely kijelöli a gyökérelemet (pl. EgyetemiRendszer).

- Csomópont típusok: A modell elemekre (ellipszis) és attribútumokra (rombusz) bontja az adatokat . A téglalap a tényleges szöveges tartalom helyét jelöli.
- Ismétlődő elemek: Azok az elemek, amelyek többször is előfordulhatnak (pl. több hallgato a gyökér alatt, vagy több jegy egy hallgato-nál).
- Kapcsolatok: A kulcsok és idegen kulcsok közötti kapcsolatok szaggatott vonalas nyíllal vannak ábrázolva (pl. hallgato.neptun $\to$ jegy.neptun).
- Hierarchia: A gyökérelem alatt helyezkednek el a fő entitások (pl. hallgato, tantargy, oktato), amelyek tartalmazzák a hozzájuk kapcsolódó gyermekelemeket (pl. jegy).

1.3 Az XDM modell alapján XML dokumentum készítése

Az XML dokumentum felépítését az XDM modell alapján kezdtem, a EgyetemiRendszer gyökérelem létrehozásával.

- **Struktúra:** Minden fő entitásból (Hallgató, Tantárgy, Oktató stb.) legalább kettő-kettő példányt készítettem.
- **Azonosítók és Hivatkozások:** Az entitások közötti kapcsolódást attribútumokkal oldottam meg. Az elsődleges kulcsok (pl. neptun, tkod) azonosítóként szerepelnek, míg az idegen kulcsok (pl. jegy elem neptun attribútuma) hivatkozásokként .
- **Összetett Tulajdonságok:** Az összetett tulajdonságokat (pl. elerhetoseg a Hallgatónál) al-elemekre bontottam (email, telefonszam) az áttekinthetőség érdekében.
- **Többértékű Elemek:** Ahol egy elem többször is előfordulhat (pl. több jegy egy Hallgatóhoz), ott biztosítottam a megfelelő számú előfordulást.

XML-részlet:

```xml
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">
    <xs:element name="EgyetemiRendszer">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="hallgato" type="HallgatoTipus" maxOccurs="unbounded"/>
                <xs:element name="tantargy" type="TantargyTipus" maxOccurs="unbounded"/>
                </xs:sequence>
        </xs:complexType>

        <xs:key name="hallgatoKey">
            <xs:selector xpath="hallgato"/>
            <xs:field xpath="@neptun"/>
        </xs:key>
        <xs:keyref name="jegy_to_hallgato" refer="hallgatoKey">
            <xs:selector xpath="jegy"/>
            <xs:field xpath="@neptun"/>
        </xs:keyref>
        </xs:element>

    <xs:complexType name="HallgatoTipus">
        <xs:sequence>
            <xs:element name="nev" type="xs:string"/>
            <xs:element name="elerhetoseg" type="ElerhetosegTipus"/>
            </xs:sequence>
        <xs:attribute name="neptun" type="xs:string" use="required"/>
    </xs:complexType>

    <xs:complexType name="JegyTipus">
        <xs:sequence>
            <xs:element name="ertek" type="JegyErtekTipus"/>
            <xs:element name="datum" type="xs:date"/>
        </xs:sequence>
        <xs:attribute name="neptun" type="xs:string" use="required"/>
        <xs:attribute name="tkod" type="xs:string" use="required"/>
        <xs:attribute name="oktatoId" type="xs:string" use="required"/>
    </xs:complexType>

    <xs:simpleType name="JegyErtekTipus">
        <xs:restriction base="xs:integer">
            <xs:minInclusive value="1"/>
            <xs:maxInclusive value="5"/>
        </xs:restriction>
    </xs:simpleType>
</xs:schema>
```

2. A MÁSODIK feladat címe

XML dokumentum feldolgozása DOM szabvánnyal Java nyelven.

2.1 Adatolvasás (CLZPREDomRead)

A program célja az XML fájl tartalmának beolvasása és egy olvasható, magyar nyelvű formátumba történő kiírása.

- **Nyelvi Térképek (Map):** Létrehoztam két statikus Map-et, az egyiket az elemek, a másikat az attribútumok nevének magyarítására.

```
private static final Map<String, String> adatNevek = new HashMap<>();
private static final Map<String, String> kulcsNevek = new HashMap<>
();
// Példa:
adatNevek.put("tantargy", "Tantárgy");
kulcsNevek.put("neptun", "Neptunkód");
```

- **XML Beolvasása:** A fájl betöltését **DocumentBuilderFactory** és **DocumentBuilder** segítségével végeztem, majd a gyökérelemet normalizáltam.
- **Kimenet Struktúra:** A kimenetet egy TXT fájlba irányítottam. Minden elemet és attribútumot külön sorba írtam ki, és a hierarchiát behúzással (indentációval) jelöltem. Az angol/rövidített XML neveket a **kulcsNevek** és **adatNevek** Map-ek alapján fordítottam magyarra, vagy ha nem volt találat, az eredeti nevet használtam.

2.2. Adat-lekérdezés (CLZPREDomQuery)

A lekérdezés célja, hogy a Hallgatókhoz tartozó felvett Tantárgyak adatait, a szerzett Jegyekkel, a tantárgyat oktató Oktató nevével és a Tanszék nevével együtt összekapcsoljam és megjelenítsem.

1. **DOM Struktúra Létrehozása:** Betöltöttem az XML fájlt és létrehoztam a DOM fastruktúrát.
2. **Adatok Előfeldolgozása (Map-ek):**
    - **Hallgatók, Tantárgyak, Oktatók, Tanszékek:** Kinyertem a fő entitások adatait (ID-t és Neveket/Főbb Tulajdonságokat), és Map-ekben tároltam el *(pl. hallgatoNevek.put("CLZPRE", "Egyed Martin")).*

3. **Összekapcsolás a Jegy (Kapcsolattábla) Elemeken Keresztül:**
    - Végigmentem az összes **<jegy>** elemen.
    - Minden jegyhez kinyertem a **neptun, tkod, oktatoId** attribútumokat.
    - Ezeket a kulcsokat felhasználva lekérdeztem a Map-ekből a Hallgató nevét, a Tantárgy nevét és kreditjét, az Oktató nevét és a hozzá tartozó Tanszék nevét.
    - Az összefűzött adatokat egy saját adatstruktúrában (**FelvettTargy**) tároltam el, amely a Hallgatókhoz lett rendelve egy **Map<String, List<FelvettTargy>>** szerkezetben.
4. **Konzolra Kiírás:** A végeredményt a konzolra írtam ki. Minden Hallgatót külön blokkban jelenítettem meg, majd alatta felsoroltam az általa felvett Tantárgyakat és a hozzájuk tartozó összes kapcsolódó információt.

Lekérdezési példák (4 hasznos információ):

1. **Hallgatói teljesítmény:** Melyik Hallgató, melyik Tantárgyból, milyen jegyet szerzett, és ki volt az értékelő Oktató?
2. **Tantárgyak és Kreditjeik:** Egy Hallgató mely Tantárgyakat vette fel, hány kredit értékben?
3. **Oktatói feladatok:** Melyik Oktató adott Jegyet egy adott Tantárgyhoz, és mely Hallgatók részére?
4. **Szervezeti Egységek:** Melyik Tanszékhez tartozó Tantárgyat vette fel egy Hallgató?

```
// Példa a Jegy feldolgozásra:
NodeList jegyLista = doc.getElementsByTagName("jegy");
for (int i = 0; i < jegyLista.getLength(); i++) {
    Element jegyElem = (Element) jegyLista.item(i);
    String neptun = jegyElem.getAttribute("neptun");
    String tkod = jegyElem.getAttribute("tkod");
    String oktatoId = jegyElem.getAttribute("oktatoId");

    // Adatok lekérése a Map-ekből
    String hallgatoNev = hallgatoNevek.getOrDefault(neptun, "Ismeretlen Hallgató");
    String targyNev = tantargyNevek.getOrDefault(tkod, "Ismeretlen Tantárgy");
    String oktatoNev = oktatoNevek.getOrDefault(oktatoId, "Ismeretlen Oktató");
    // ... további adatok

    // Adatok eltárolása a Hallgatóhoz rendelt listában
    // ...
}
```

2.3. Adatmódosítás (CLZPREDomModify)

A program az XML fájl beolvasása, módosítása és a módosított tartalom kiírása a konzolra, csak a módosított elemekre fókuszálva.

1. **XML Beolvasása és Normalizálása:**
2. **Módosítások Végrehajtása:**
    - **1. módosítás: Hallgató szakának frissítése.**
        - Lekérdeztem egy adott Neptun kódú Hallgatót.
        - Módosítottam a ***<szak>*** gyermekelem tartalmát.
    - **2. módosítás: Tantárgy kreditpontjának növelése.**
        - Lekérdeztem egy adott Tantárgy kódú Tantárgyat.
        - Módosítottam a ***<kredit>*** gyermekelem értékét.
    - **3. módosítás: Oktató beosztásának módosítása.**
        - Lekérdeztem egy adott Oktató ID-val rendelkező Oktatót.
        - Módosítottam a ***<beosztas>*** gyermekelem tartalmát.
    - **4. módosítás: Egy Hallgató Jegyének átírása.**
        - Lekérdeztem egy adott Hallgató Neptun kódjához és Tantárgy kódjához tartozó ***<jegy>*** elemet.
        - Átírtam az ***<ertek>*** gyermekelem szöveges tartalmát.
3. **Konzolra Kiírás (Módosítások Dokumentálása):**
    - Minden módosítás után blokkos kiírás készült a konzolra, feltüntetve a műveletet és az érintett elem magyarított nevét és attribútumait.
    - ***kulcsnevek*** **Map** használata a kiírásban a magyar nyelvű attribútumnevek megjelenítéséhez.

Módosítási példa (Jegy átírása):

```
// Módosítás példa (jegy értékének átírása)
NodeList jegyLista = doc.getElementsByTagName("jegy");
for (int i = 0; i < jegyLista.getLength(); i++) {
    Element jegy = (Element) jegyLista.item(i);
    // Neptun kód és Tantárgy kód ellenőrzése
    if ("FJYXPC".equals(jegy.getAttribute("neptun")) && "AB1234".equals(jegy.getAttribute("tkod")))
{       Node ertek = jegy.getElementsByTagName("ertek").item(0);
        if (ertek != null) {
            System.out.println("--- Jegy módosítása ---");
            System.out.println("Korábbi érték: " + ertek.getTextContent());
            ertek.setTextContent("4"); // Jegy átírása 4-re
            System.out.println("Új érték: 4");
            // ... kiírás a Hallgató és Tantárgy nevével ...
        }
        break;
    }
}
```