

JEGYZŐKÖNYV

Webes adatkezelő környezetek

Féléves feladat

Könyvtári rendszer

Készítette: **Bucskó Gábor**

Neptunkód: **PSD1T8**

Dátum: **2025 december**

Miskolc, 2025

Tartalomjegyzék

1. [Bevezető](#)
2. [A feladat leírása](#)
3. [ER modell készítése](#)
 - [3.1 Az adatbázis ER modell tervezése](#)
 - [3.2 Az entitások és kapcsolatok definiálása](#)
4. [XML és XSD készítése](#)
 - [4.1 XML dokumentum létrehozása](#)
 - [4.2 XML séma \(XSD\)](#)
 - [4.3 Validálás](#)
5. [DOM parse implementáció](#)
 - [5.1 DomRead osztály](#)
 - [5.2 DomQuery osztály](#)
 - [5.3 DomModify osztály](#)
6. [Összegzés](#)

1. Bevezető

Ez a jegyzőkönyv a **Webes adatkezelő környezetek** tárgy féléves feladatának részletes dokumentációja. A feladat célja egy könyvtári rendszer adatmodelljének megtervezése, XML formátumban való megvalósítása, valamint Java DOM parser használatával történő adatkezelés implementálása.

A projekt több fázisból áll:

- **ER (Entity-Relationship) modell** tervezése a könyvtári rendszerhez
- **XML séma (XSD)** és XML dokumentum készítése
- **Java alkalmazás** fejlesztése DOM parserrel az XML adatok olvasásához, lekérdezéséhez és módosításához

2. A feladat leírása

2.1 A feladat általános célja

A feladat célja egy **könyvtári rendszer (KÖNYVTÁR)** adatmodelljének létrehozása, amely képes kezelni:

- Szerzőket (többértékű email címekkel)
- Kiadókat (összetett címmel)
- Könyveket (könyv-szerző kapcsolat)
- Könyvtárosokat (mentor hierarchiával - 1:1 kapcsolat)
- Olvasókat (többértékű telefonszámokkal)
- Kölcsönzéseket (olvasó-könyv M:N kapcsolat)

2.2 Követelmények

- **Minimum 5 entitás** különböző kapcsolati típusokkal (1:1, 1:N, M:N)
- **Elsődleges kulcsok**, összetett attribútumok és többértékű attribútumok használata
- **XML séma validáció** XSD használatával
- **Java DOM parser** implementáció legalább 3 lekérdezéssel és 3 módosítással

3. ER modell készítése

3.1 Az adatbázis ER modell tervezése

A könyvtári rendszer adatmodellje az alábbi entitásokat tartalmazza:

Entitások:

1. **KONYV** (Könyv)
 - Konyv_ID (elsődleges kulcs)
 - Cim
 - Kiadasi_ev
 - Oldalszam
2. **SZERZO** (Szerző)
 - Szerzo_ID (elsődleges kulcs)
 - Nev
 - Szuletesi_datum
 - Email_cimek (többértékű attribútum)
3. **KIADO** (Kiadó)
 - Kiado_ID (elsődleges kulcs)
 - Nev
 - Cim (összetett attribútum: Varos, Utca)
 - Kapcsolat
4. **OLVASO** (Olvasó)
 - Olvaso_ID (elsődleges kulcs)
 - Nev
 - Szuletesi_datum
 - Telefonszamok (többértékű attribútum)
5. **KONYVTAROS** (Könyvtáros)
 - Konyvtaros_id (elsődleges kulcs)
 - Nev
 - Befosztas
 - Elsosztas
 - Mentor_ref (1:1 kapcsolat saját magával)

3.2 Az entitások és kapcsolatok definiálása

Kapcsolatok:

1. **IRTA** (Szerző - Könyv) - **M:N kapcsolat**

- Egy szerzőnek több könyve is lehet
- Egy könyvnek több szerzője is lehet
- Szerep attribútummal (pl. Főszerző, Társszerző)

2. **KIADJA** (Kiadó - Könyv) - **1:N kapcsolat**

- Egy kiadó több könyvet adhat ki
- Egy könyvet egy kiadó ad ki

3. **KOLCSONZES** (Olvasó - Könyv) - **M:N kapcsolat**

- Egy olvasó több könyvet is kölcsönözhet
- Egy könyvet több olvasó is kölcsönözhet (időben)
- Attribútumok: Kolcsonzes_datuma, Visszahozas_hatarido, Visszahozva

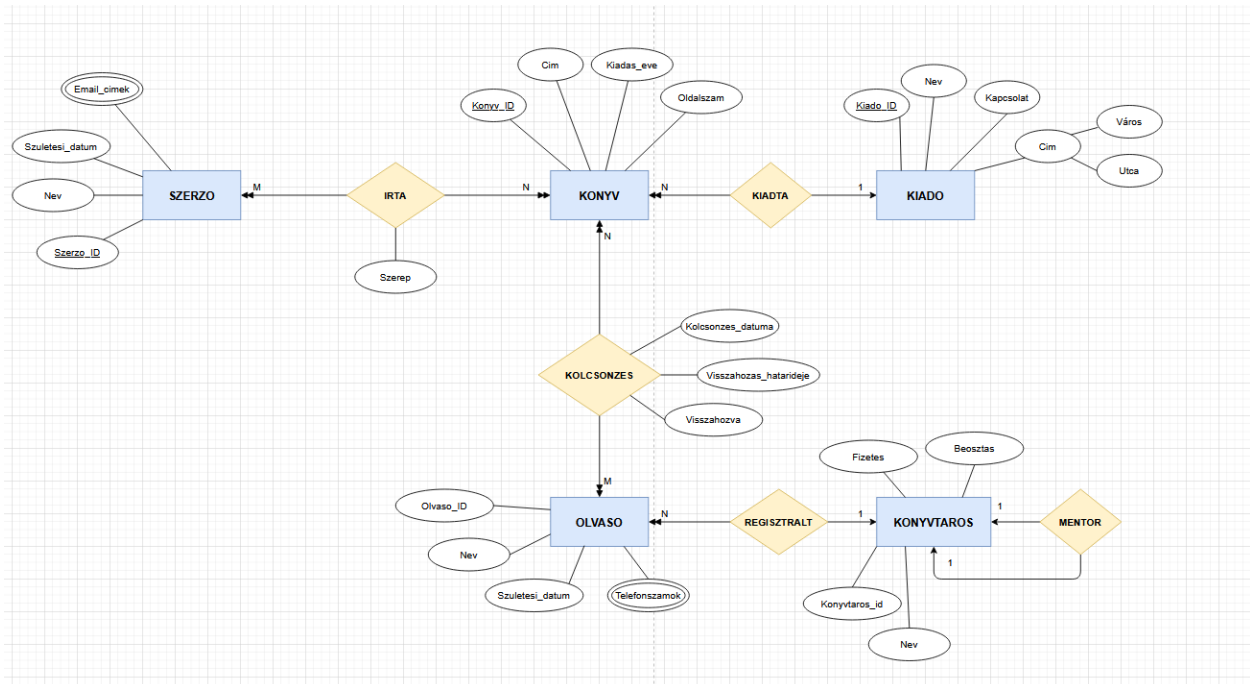
4. **REGISZTRACIO** (Könyvtáros - Olvasó) - **1:N kapcsolat**

- Egy könyvtáros több olvasót is regisztrálhat
- Egy olvasót egy könyvtáros regisztrál

5. **MENTOR** (Könyvtáros - Könyvtáros) - **1:1 rekurzív kapcsolat**

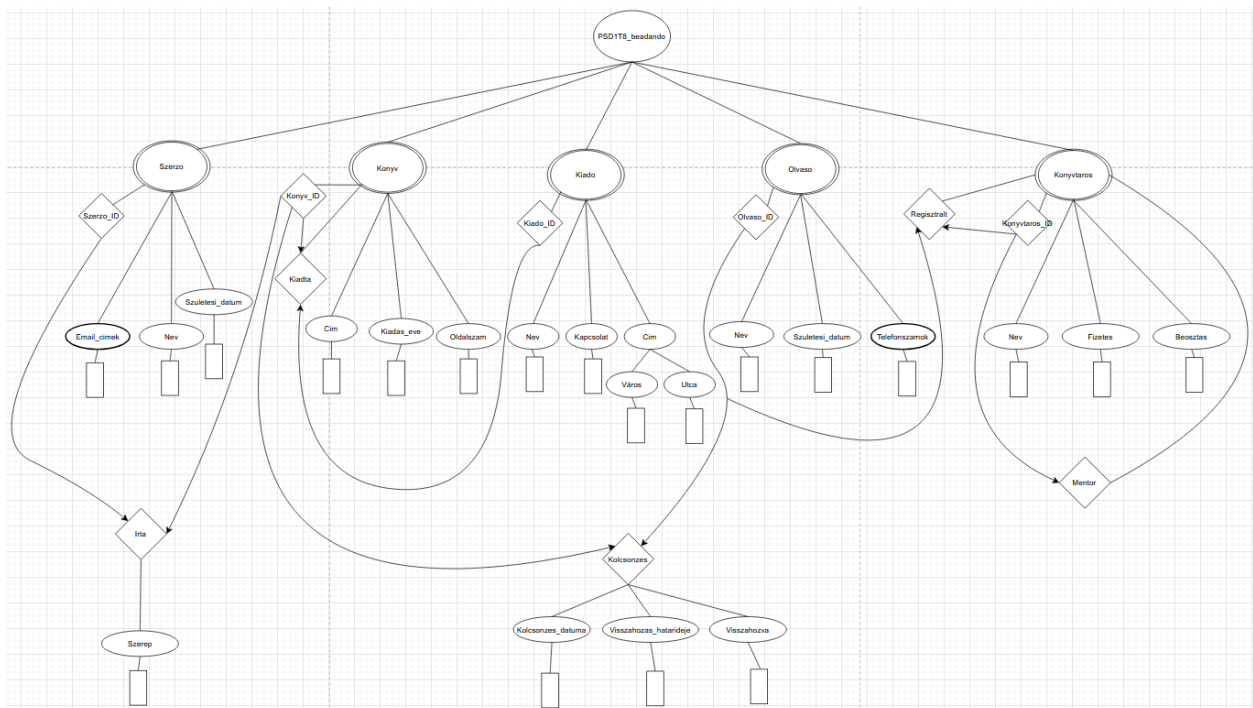
- Egy könyvtárosnak egy mentora van
- Egy könyvtáros több beosztott mentorálója lehet

Modell:



3.3 XDM Modell készítése

A gyakorlatokon és órán tanult módszerrel és példák segítségével konvertáltam az ER modell-t XDM modell-re:



4. XML és XSD készítése

4.1 XML dokumentum létrehozása

XML dokumentum:

PSD1T8_XML.xml

Az XML fájl egy könyvtári rendszer teljes adatstruktúráját tartalmazza XML formátumban:

- **Szerzők** (3 szerző többértékű email címekkel)
 - Molnár János (SZ001) - 2 email címmel
 - Kovács Anna (SZ002) - 1 email címmel
 - Nagy Péter (SZ003) - 2 email címmel
- **Kiadók** (3 kiadó összetett címmel)
 - Móra Könyvkiadó (K001) - Budapest, Váci utca
 - Libri Kiadó (K002) - Budapest, Páva utca
 - Helikon Kiadó (K003) - Budapest, Nagymező utca
- **Könyvek** (4 könyv kiadó és szerző referenciákkal)
 - A magyar nyelv története, Programozási alapok, Webfejlesztés XML alapokkal, Adatbázis-kezelés
- **Könyvtárosok** (3 könyvtáros mentor hierarchiával)
 - Farkas Éva (KT001) - Főkönyvtáros, mentor
 - Szabó Gábor (KT002) - Gyerekkönyvtár, mentora: KT001
 - Tóth Mária (KT003) - Kölcsönzési részleg, mentora: KT001
- **Olvasók** (3 olvasó többértékű telefonszámokkal)
 - Kiss Tamás (O001), Varga Eszter (O002), Horváth László (O003)
- **Kölcsönzések** (4 kölcsönzés aktív státusszal)
 - Különböző olvasók könyveinek kölcsönzése dátumokkal és határidőkkel

Az XML dokumentum jól strukturált, hierarchikus felépítésű, és használ referenciákat (IDREF) az entitások közötti kapcsolatok megvalósítására.

4.2 XML séma (XSD)

XML séma:

PSD1T8_XMLSchema.xsd

Az XSD séma definiálja az XML dokumentum struktúráját és validációs szabályait:

Főbb XSD elemek:

- Gyökérelem típusának meghatározása (konyvtar)
- Összetett típusok definiálása (pl. CimType kompozit elem Város és Utca alelemekkel)
- Többértékű attribútumok kezelése (Email_cimek, Telefonszámok)
- Referencia integritás biztosítása IDREF használatával
- Adattípus megkötések (pl. dátum formátumok, számok)

Az XSD séma szigorúan ellenőrzi a dokumentum struktúráját és adattípusait, biztosítva az adatok konzisztenciáját.

A következő megvalósítása során az XSD sémát használok validációra.

4.3 Adatvalidálás és konzisztencia

Az XML dokumentum az XSD séma ellen validálva lett, amely garantálja:

- A megfelelő hierarchikus struktúrát
- Az adattípusok helyességét
- A referenciális integritást az ID és IDREF használatával
- A kötelező mezők jelenlétét

5. A HARMADIK feladat címe - DOM parse implementáció

5.1 DomRead osztály

A DomRead osztály felelős az XML dokumentum beolvasásáért és konzolra történő kiírásáért. A főbb metódusok:

Implementált funkciók:

- main() - XML fájl betöltése és DOM fa feldolgozása
- printSzerzok() - Szerzők listázása emailekkel
- printKiadok() - Kiadók listázása címekkel
- printKonyvek() - Könyvek listázása kapcsolódó adatokkal
- printKonyvtarosok() - Könyvtárosok listázása mentor hierarchiával
- printOlvasok() - Olvasók listázása telefonszámokkal
- printKolcsonzesek() - Kölcsönzések listázása
- getElementText() - Segédfüggvény szöveges tartalom kiolvasásához

A konzolra írás során minden entitás részletesen megjelenik, beleértve az összetett és többértékű attribútumokat is.

5.2 DomQuery osztály

ADomQuery osztály interaktív lekérdezéseket valósít meg konzolos felhasználói bemenettel. Az osztály 5 különböző lekérdezési funkciót tartalmaz:

1. Lekérdezés - Könyvek szerző szerint (konyvekSzerzoSzerint)

- Bemenet: Szerző ID (pl. SZ001, SZ002)
- Kimenet: Az adott szerző által írt összes könyv címe, kiadási éve, oldalszáma
- Használt DOM módszer: getElementByTagName(), getAttribute()

2. Lekérdezés - Kölcsönzések olvasó szerint (kolcsonzesekOlvasoSzerint)

- Bemenet: Olvasó ID (pl. O001, O002)
- Kimenet: Az olvasó összes kölcsönzése (könyv címe, dátum, határidő, státusz)
- Kapcsolt adatok: könyv címek referencia alapján

3. Lekérdezés - Könyvtárosok és mentoraik (konyvtarosokEsMentoraik)

- Bemenet: nincs (automatikus listázás)
- Kimenet: Könyvtárosok neve, beosztása és mentorának neve
- Rekurzív kapcsolat megjelenítése

4. Lekérdezés - Kiadók és könyveik (kiadokEsKonyveik)

- Bemenet: nincs (automatikus listázás)
- Kimenet: Minden kiadó és az általuk kiadott könyvek listája
- 1:N kapcsolat megjelenítése

5. Lekérdezés - Aktív kölcsönzések (aktivKolcsonzesek)

- Bemenet: nincs (automatikus lista)
- Kimenet: Még vissza nem hozott könyvek (Visszahozva=false)
- Szűrés boolean érték alapján

Segédfüggvények:

- getKonyvCim() - Könyv címének lekérése ID alapján
- getOlvasoNev() - Olvasó nevének lekérése ID alapján
- getKonyvtarosNev() - Könyvtáros nevének lekérése ID alapján
- getElementText() - Elem szöveges tartalmának kinyerése

A lekérdezések interaktív módon működnek, a felhasználó a konzolról adja meg a paramétereket.

5.3 DomModify osztály

A DomModify osztály az XML dokumentum módosítását végzi interaktív konzolos felülettel. Az osztály 5 különböző módosítási műveletet támogat:

1. Módosítás - Új szerző hozzáadása (ujSzerzoHozzaadasa)

- Bemenet: Szerző ID, Név, Születési dátum, Email címek (vesszővel elválasztva)
- Művelet: Új <szerzo> elem létrehozása többértékű email címekkel
- DOM metódus: createElement(), appendChild()

2. Módosítás - Kölcsönzés visszahozása (kolcsonzesVisszahozasa)

- Bemenet: Olvasó ID, Könyv ID
- Művelet: Kölcsönzés <Visszahozva> elemének értéke "false"-ról "true"-ra
- DOM metódus: `getElementsByTagName()`, `setTextContent()`

3. Módosítás - Olvasó telefonszámának módosítása (olvasoTelefonModositasa)

- Bemenet: Olvasó ID, Új telefonszám
- Művelet: Olvasó első telefonszámának frissítése
- DOM metódus: `getElementsByTagName()`, `replaceChild()`

4. Módosítás - Új könyv hozzáadása (ujKonyvHozzaadasa)

- Bemenet: Könyv ID, Cím, Kiadási év, Oldalszám, Kiadó ref, Szerző ref, Szerep
- Művelet: Új <konyv> elem létrehozása összes kapcsolódó adattal
- Összetett struktúra: Szerzők referenciák külön elemekkel

5. Módosítás - Szerző törlése (szerzoTorlese)

- Bemenet: Szerző ID
- Művelet: Szerző elem teljes eltávolítása a dokumentumból
- DOM metódus: `removeChild()`, `getParentNode()`

Fájlkezelés:

- `mentes()` - Módosított XML dokumentum fájlba írása formázott kimenettel
- Transformer használata indent beállítással a jól olvasható output érdekében

Az összes módosítás után a program menti a módosított XML-t új fájlba, így az eredeti dokumentum nem sérül.

Összegzés

A féléves feladat során elkészítettem egy teljes körű könyvtári rendszer adatmodelljét és implementációját:

1. **ER modell** - 5 entitás, különböző kapcsolati típusokkal (1:1, 1:N, M:N)
2. **XML és XSD** - Validálható XML dokumentum összetett struktúrával
3. **Java DOM parser** - Teljes CRUD funkcionalitással:
 - **Olvasás** (DomRead) - Teljes dokumentum kiírása
 - **Lekérdezés** (DomQuery) - 5 interaktív lekérdezés
 - **Módosítás** (DomModify) - 5 különböző módosítási művelet

A projekt Maven alapú, tiszta kódstruktúrával és konzolos interaktív felülettel.