JEGYZŐKÖNYV

Adatkezelés XML környezetben

Féléves feladat

Banki tranzakciós adatbázis

Készítette: **Répási Gábor**

Neptunkód: **D51MXC**

Dátum: **2023.12.04**

Tartalomjegyzék

[Bevezetés – A feladat leírása, ER-modell](#1a) 3

[Átrajzolás XDM-típusú modellre](#1b) 4

[XML-fájl létrehozása](#1c) 5

[XSD-sémafájl létrehozása](#1d) 6

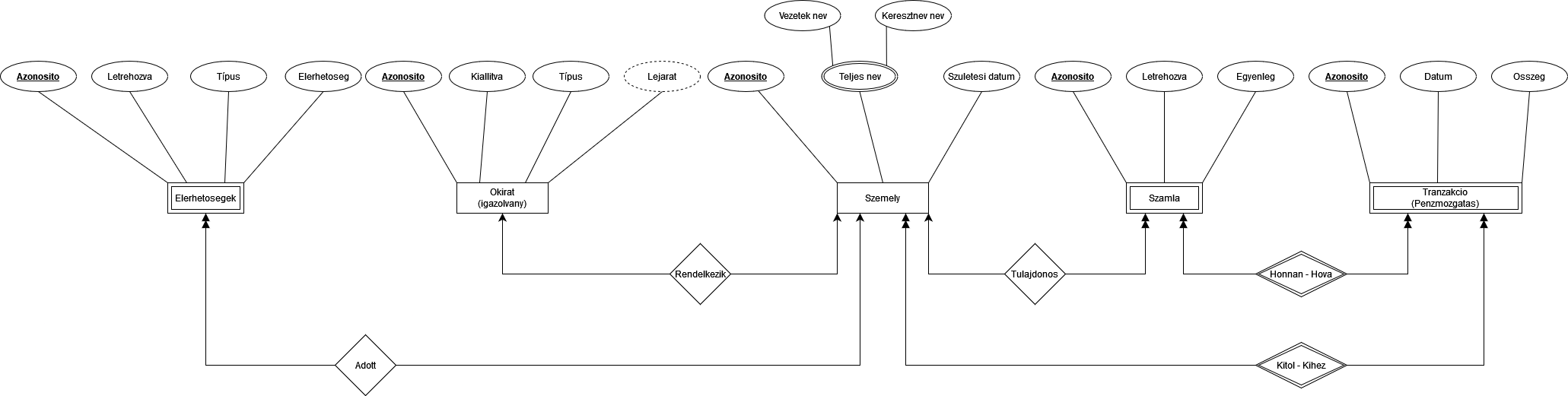
[DOMRead – kiolvasó funkciók](#2a) 7

[DOMModify – node módosítások](#2b) 8

[DOMQuery – lekérdezések](#2c) 9

[DOMWrite – módosítások kiírása fájlba](#2d) 10

Bevezetés



**1a. feladat**

A feladatom egy banki adatbázis menedzselése. Mint minden banknak vannak ügyfelei. Az ügyfelek általában többen vannak, és a számlákból is több van. Ami a hétköznapi életben is előszokott fordulni, hogy egy természetes személy egyszer fordul elő egy banknál, de ennek a személynek lehet több számlája is, ez valósítja meg az **1:N** kapcsolatot.

Szintén egy ügyfélnek kell lennie egy hivatalos elérhetőségnek, amelyen mindig mindenkor utolérhető, ez is megvalósítja az **1:N** kapcsolatot. Opcionális, lehet nulla, egy vagy több is.

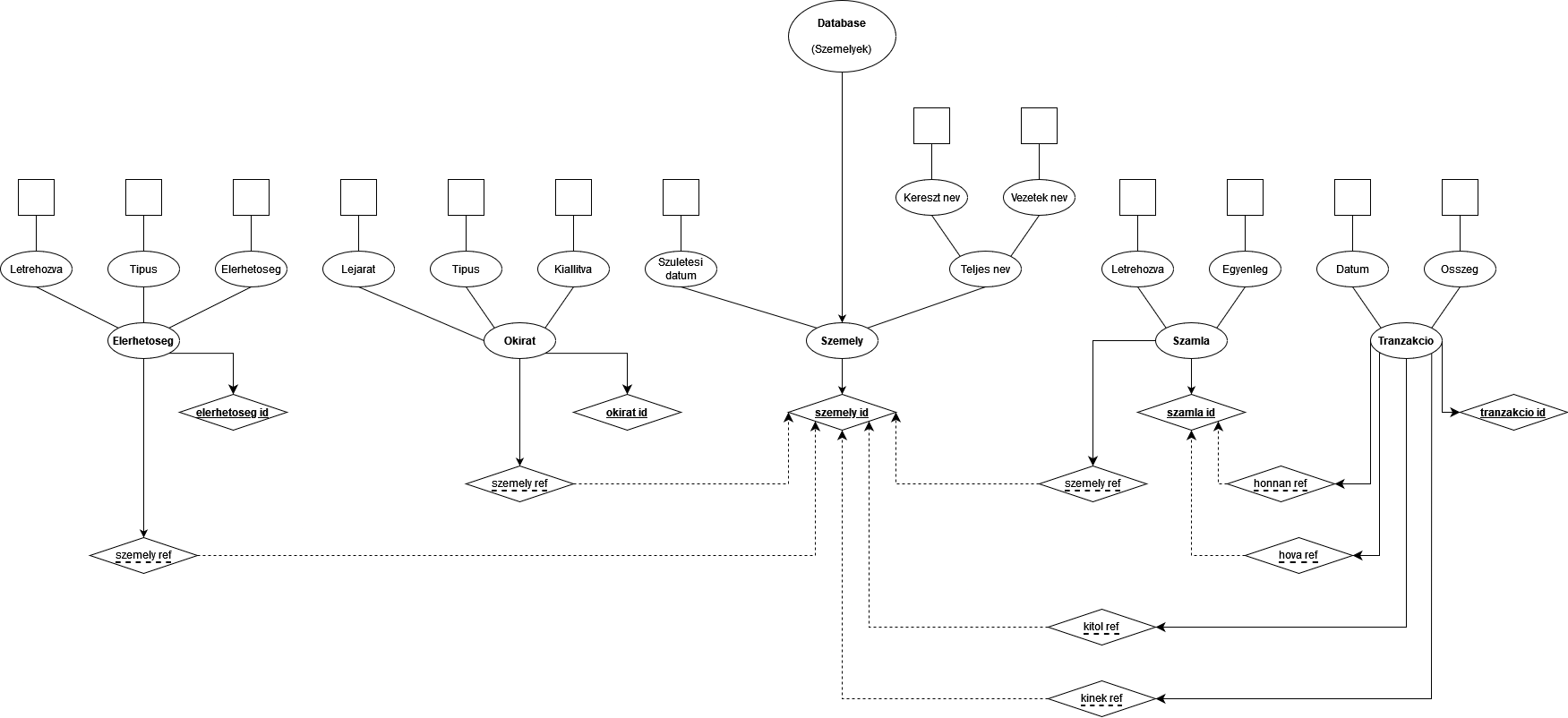
Egy személynek csak egyféle okirata lehet, például nem lehet több személyi igazolványa, csak egy, amely csak egy személyhez tartozhat, ez lesz az **1:1** kapcsolat. A lejárati mező opcionális, ugyanis nem minden okiratnak van lejárati ideje, ez jelöli a rajz is.

Nem utolsó sorban pedig maga a tranzakciót megvalósító, azt dokumentáló, illetve naplózó bejegyzések, ahol több személyhez is kapcsolódhat, attól függően, hogy kitől és kihez kerül a pénz, valamint melyik számláról melyikre, amely így több-több **N:M** kapcsolat.

A pénzmozgások nyomon követhetőek, ezáltal kimutatásokat is lehet belőle készíteni, amelyet szemléltetni is fogok a második feladat lekérdezési és szűrési feltételével.

**Átrajzolás XDM típusú modellre**

**1b. feladat**



*(Kép és szerkeszthető terv-sablon mellékelve.)*

*Az ER modellből megrajzoltam az XDM modell-t.*

Tartalmazza a fő „egyedülálló” egyedszerű bejegyzést, amely maga az adatbázis és a neve, amely az én esetemben csak a „database” elnevezést kapott.

A vonalak nem keresztezik egymást.

Az egyedek felett helyezkednek el a tulajdonságaik. Ezt követi az elsődleges kulcsmezőjük a rá következő sorban. A második sortól kezdve pedig az idegenkulcsok, szintén egymást követő sorban, elkülönülten és modell-szerű definiált sorrendben. Az egyedek mutatnak az elsődleges és idege kulcsikra is, az idegenkulcsok mutatnak más egyedek elsődleges kulcsaira.

**XML fájl létrehozása**

**1c. feladat**

Létrehoztam az XML fájlt.

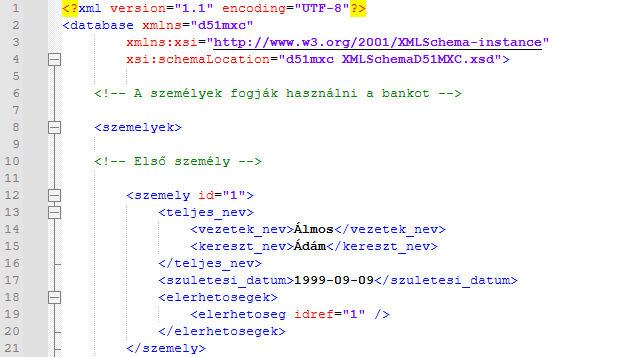
Tartalmaz 5 egyedet:

* Személyek
* Elérhetosegek
* Okiratok
* Számlak
* Tranzakciók

Valamint az ehhez szükséges:

* Elsődleges kulcsok definicióját
* Idegenkulcsok definicíóját

*Részlet a fájl elejéből:*

****

**XSD fájl létrehozása**

**1d. feladat**

Létrehoztam az XSD-sémafájlt. Létrehoztam a komplex-típusú egyedeket.

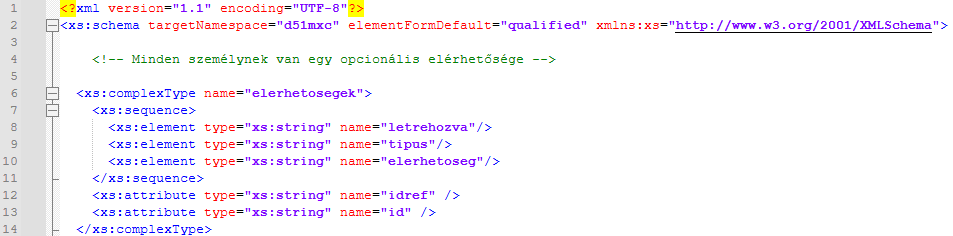
Tartalmaz 5 egyedet, amelyeknek mind van egy elsődleges kulcsa, kivétel nélkül.

A az első „Személyek” egyedeken kívül mind tartalmaz legalább egy idegenkulcsot.

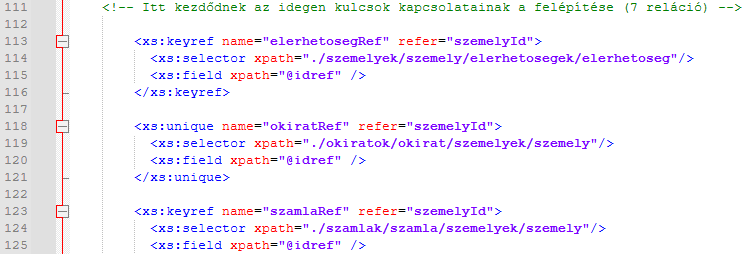
A tranzakciókat naplózó „Tranzakciók” egyed 4 idegenkulcsot tartalmaz, amelyből 2-2 egy egyedre mutat.

Összeségében 7 db. Idegenkulcs definíciót tartalmaz.

*Részlet az XSD-fájl elejéből:*



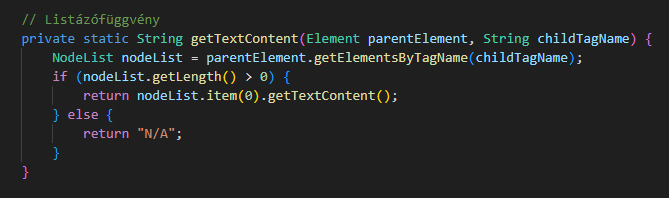
*Részlet az XSD-fájl idegenkulcs definícióiból:*

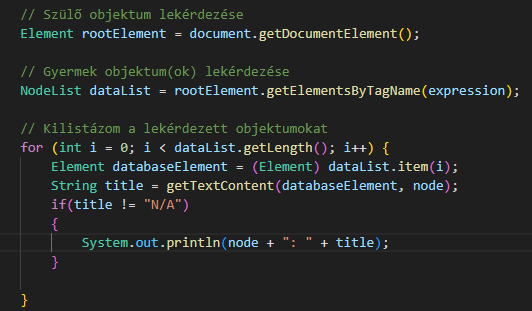


**DOMRead**

**2a. feladat**

A „DOMReadD51mxc.java” fájl nem nyitja meg közvetlen az XML fájlt, hanem átveszi a DOMQuery-től. Ez csak a fő funkciókat tartalmazza, elvégzi az adatkinyerést, majd visszaadja a query-nek a lekérdezett adatokat a „printAdat” függvénnyel.



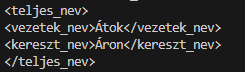


**DOMModify**

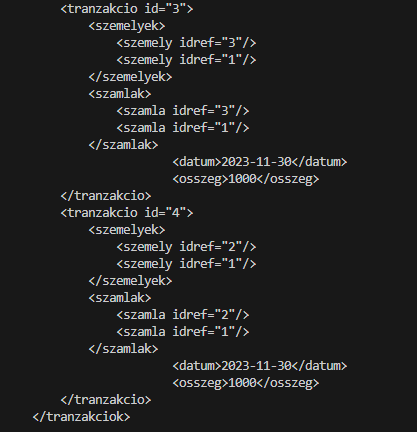
**2b. feladat**

A DomModifyD51mxc.java megnyitja az XML-fájlt és abból felépít a DOM fát, ebben végez módosításokat majd mentésre átadja a DOMWrite-nak amely az előzőkben leírtak szerint menti, majd kilistázza.





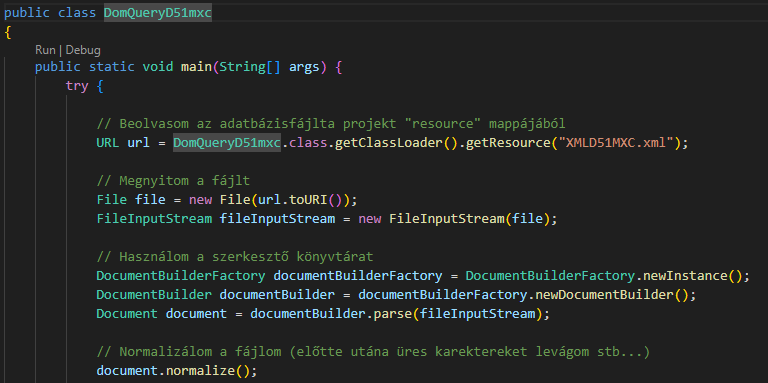
Részlet a fa strúktúra listázásból (behúzásnak 4 db. szóköz karaktert állítottam be):



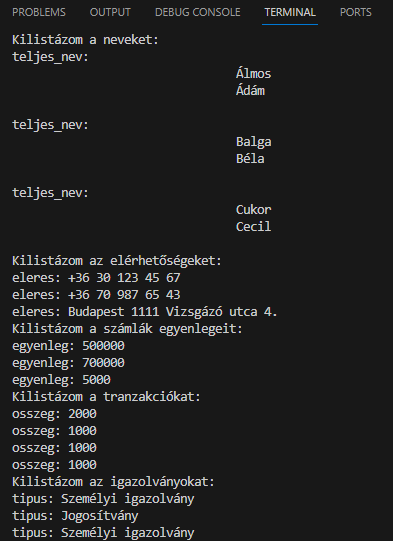
**DOMQuery**

**2c. feladat**

A DomQueryD51mxc.java megnyitja az XML-fájlt és abból felépít a DOM fát. Majd feldolgozásra továbbítja a DOMRead osztálynak aki visszaadja kilistázott formában. Így működik a DOMRead és DOMQuery kéz a kézben.



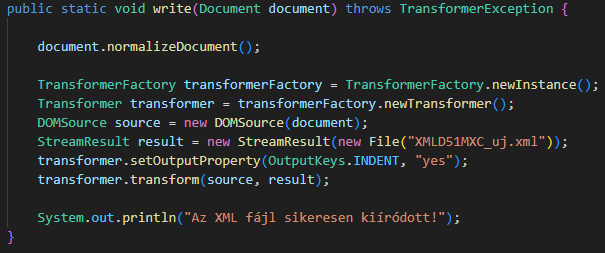
Az 5 db. Lekérdezésem eredménye:



**DOMWrite**

**2d. feladat**

A DOMWriteD51mxc.java a DOMRead-hez hasonlóan funkciókkal bővíti ki a hozzá tartozó DOMModify osztályt. Feladata a módosítás utáni fájlba mentés és kiíratás: XMLD51MXC\_uj.xml



Adatok kilistázására való funkció (fa struktúrába):

