JEGYZŐKÖNYV

Adatkezelés XML-ben
Féléves feladat

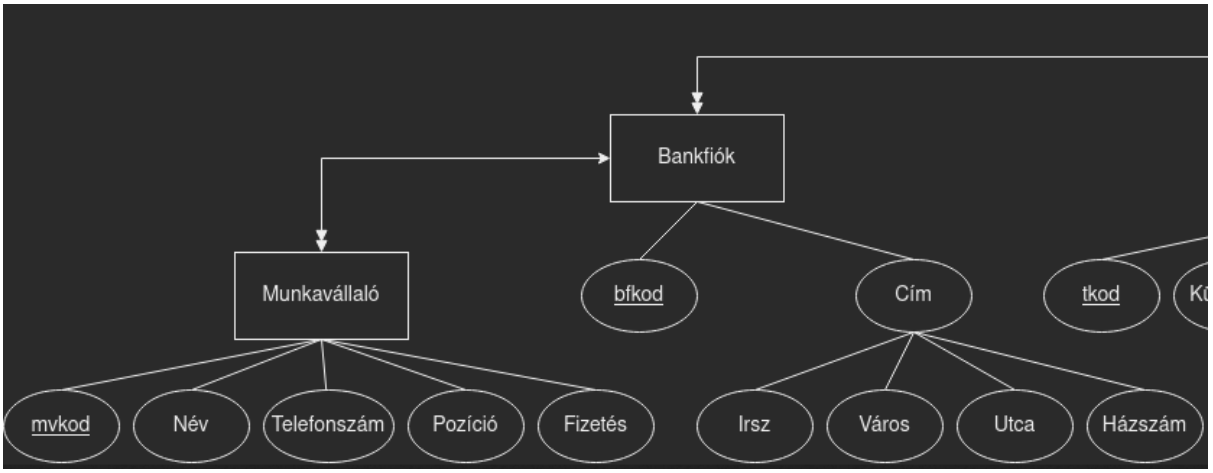
Készítette: Horváth Ákos Zsigmond
Neptun kód: R3SZY2

TARTALOMJEGYZÉK

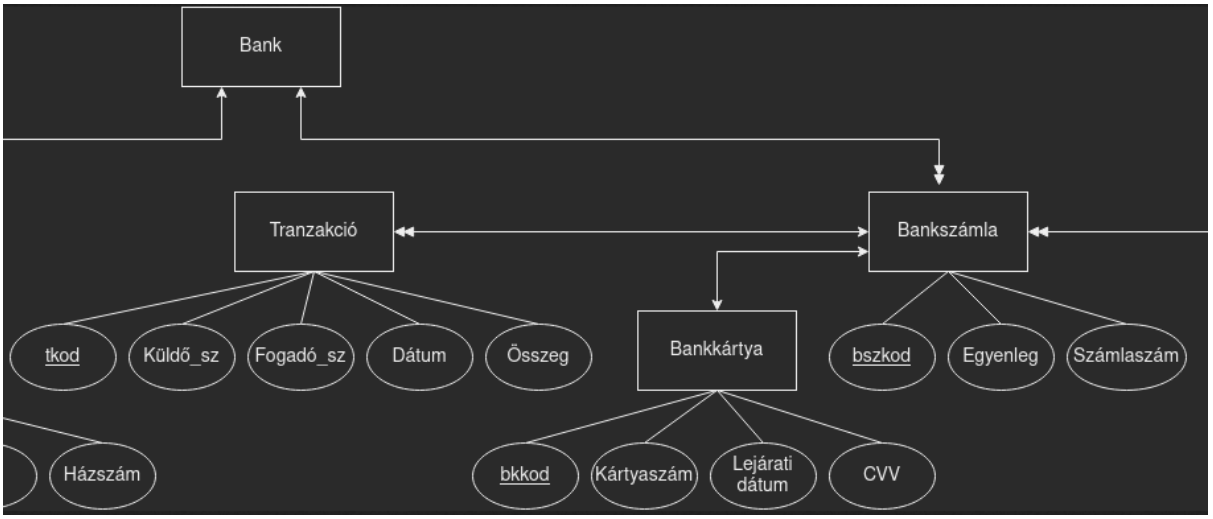
1.feladat

1a) Az adatbázis ER modell (Legyen legalább 5 egyed, többféle kapcsolat (1:1; 1:N; M:N), minden tulajdonság (normál, kulcs, összetett, többértékű).

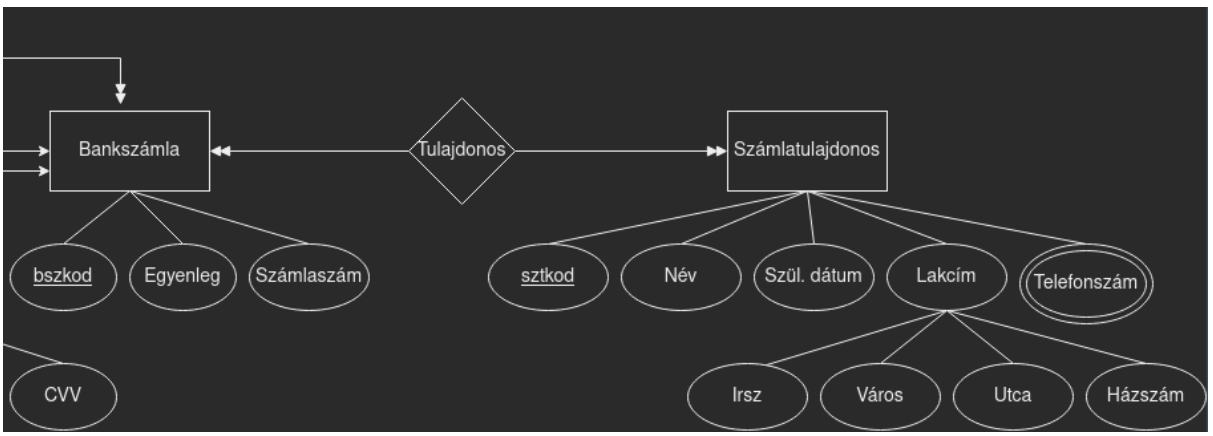
Ebben a feladatban létrehoztam az eltárolt adatokat reprezentáló Entity-Relationship modellt. Az ábrák létrehozásához az https://app.diagrams.net/ programot használtam.
A tárolt adatok egy bank adatai.



A képen látható a Munkavállaló és Bankfiók egyedek, amik között 1:N kapcsolat van.



Ezen a képen a Bank root egyed, és a Tranzakció, Bankszámla, Bankkártya egyedek láthatók. A Bankszámla és Tranzakció egyedek között 1:N, és a Bankszámla és Bankkártya egyedek között 1:1 kapcsolat van.

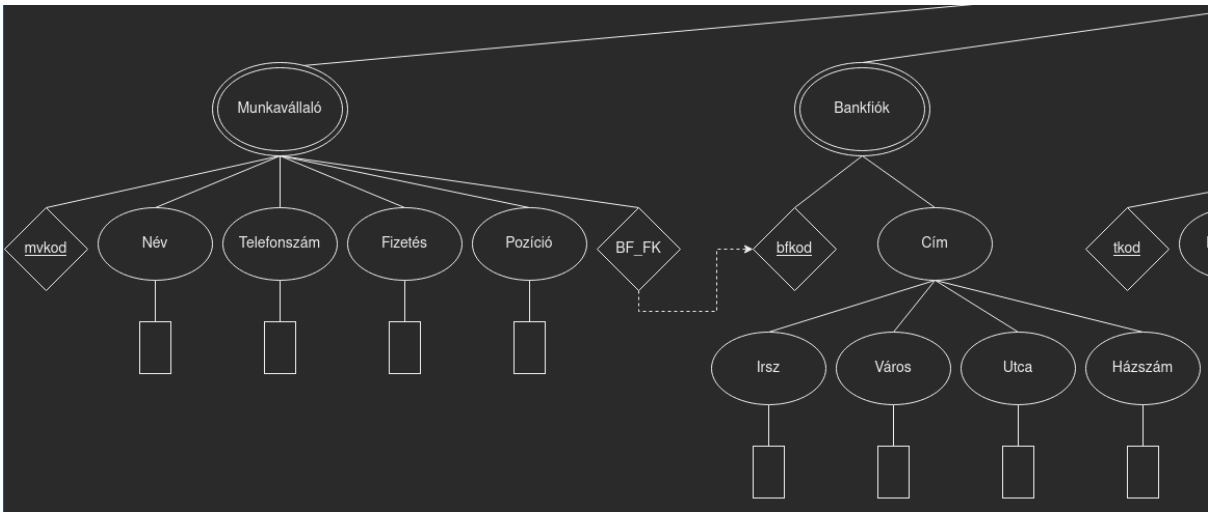


A Bankszámla és Számlatulajdonos között N:M kapcsolat van, és ehhez szükséges a Tulajdonos egyed.

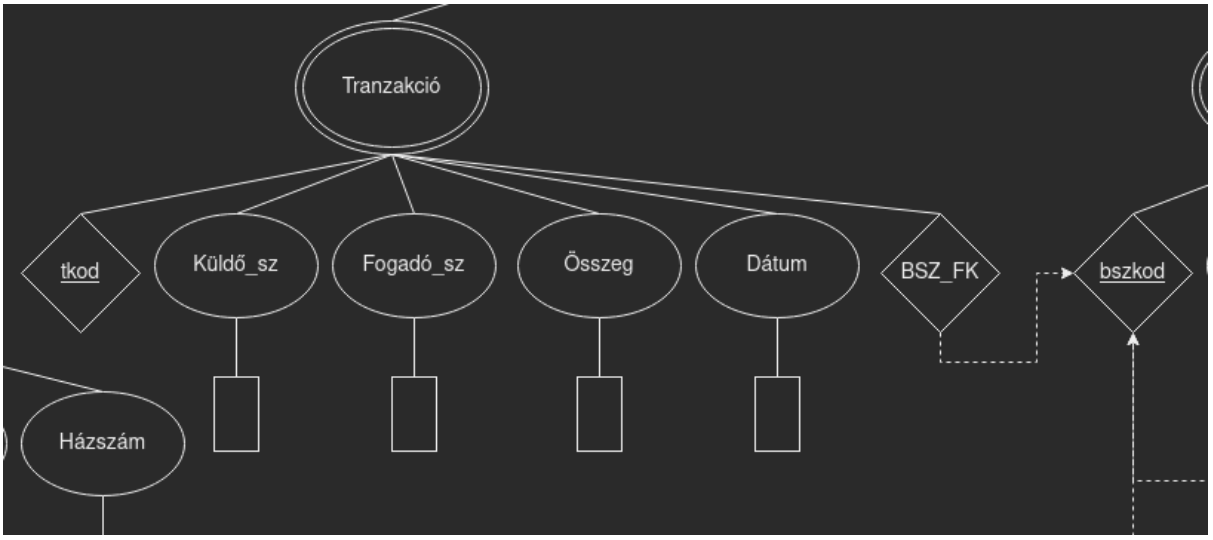
1b) Az adatbázis konvertálása XDM modellre

Az XDM, vagyis az XML Data Model elkészítéséhez is a https://app.diagrams.net/ weboldalt használtam.

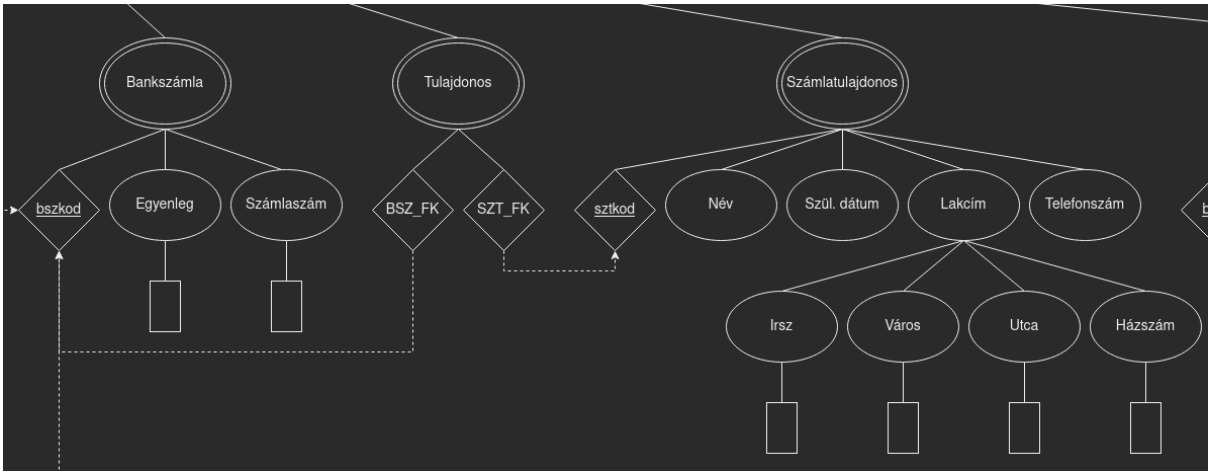
A Munkavállaló és Bankfiók egyedek:



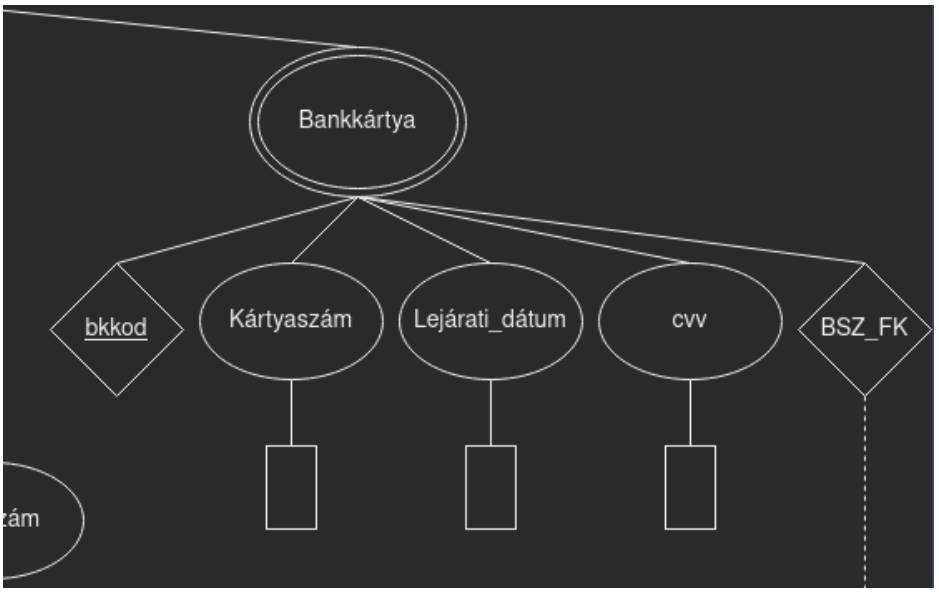
Tranzakció egyed:



Bankszámla, Tulajdonos, és Számlatulajdonos egyedek:



És a Bankkártya egyed:



1c) Az XDM modell alapján XML dokumentum készítése

Minden egyedből létrehoztam három darab XML elemet, a jegyzőkönyvben minden egyedből csak egyet fogok bemutatni.
A kulcsok és foreign key-k attribútumként szerepelnek.

Bankszámla:

```
<!-- Bankszamla peldanyai  -->
<bankszamla bszkod="1">
  <szamlaszam>111-222-333</szamlaszam>
  <egyenleg>120000</egyenleg>
</bankszamla>
```

Számlatulajdonos:

```
<!-- Szamlatulajdonos peldanyai  -->
<szamlatulajdonos sztkod="1">
   <nev>Kiss István</nev>
   <szuldatum>1980-5-11</szuldatum>
   <lakcim>
     <irsz>1033</irsz>
     <varos>Budapest</varos>
     <utca>Petőfi utca</utca>
     <hazszam>1</hazszam>
   </lakcim>
   <telefonszam>062011223344</telefonszam>
   <telefonszam>062011225566</telefonszam>
</szamlatulajdonos>
```

Tulajdonos:

```
<!-- Tulajdonos, a Bankszamla es Szamlatulajdonos
     egyedeket osszekoto egyed peldanyai  -->
<tulajdonos bsz_fk="1" szt_fk="2"/>
```

Bankkártya:

```
<!-- Bankkartya peldanyai  -->
<bankkartya bkkod="1" bsz_fk="2">
  <kartyaszam>1111-2222-3333-4444</kartyaszam>
  <lejarati_datum>2024-11-01</lejarati_datum>
  <cvv>111</cvv>
</bankkartya>
```

Tranzakció:

```
<!-- Tranzakcio peldanyai  -->
<tranzakcio tkod="1" bsz_fk="3">
  <kuldo_sz>888-999-001</kuldo_sz>
  <fogado_sz>111-222-333</fogado_sz>
  <osszeg>15000</osszeg>
  <datum>2022-09-23</datum>
</tranzakcio>
```

Bankfiók:

```
<!-- Bankfiok peldanyai  -->
<bankfiok bfkod="1">
  <cim>
    <irsz>1234</irsz>
    <varos>Budapest</varos>
    <utca>Pest utca</utca>
    <hazszam>12</hazszam>
  </cim>
</bankfiok>
```

Munkavállaló:

```
<!-- Munkavallalo peldanyai  -->
<munkavallalo mvkod="1" bf_fk="2">
  <nev>Lantos Ferenc</nev>
  <telefonszam>0620123456123</telefonszam>
  <fizetes>250000</fizetes>
  <pozicio>Ügyfélszolgálat</pozicio>
</munkavallalo>
```

1d) Az XML dokumentum alapján XMLSchema készítése

Minden egyedhez létrehoztam egy saját típust.

Bankszámla:

```
<xs:complexType name="bankszamlaType">
  <xs:sequence>
    <xs:element type="xs:string" name="szamlaszam"/>
    <xs:element type="xs:float" name="egyenleg"/>
  </xs:sequence>
  <xs:attribute type="xs:int" name="bszkod"
use="required"/>
</xs:complexType>
```

Lakcím (nem egyed, de többször is előfordul):

```
<xs:complexType name="lakcimType">
  <xs:sequence>
    <xs:element type="xs:string" name="irsz"/>
    <xs:element type="xs:string" name="varos"/>
    <xs:element type="xs:string" name="utca"/>
    <xs:element type="xs:unsignedShort" name="hazszam"/>
  </xs:sequence>
</xs:complexType>
```

Számlatulajdonos:

```
<xs:complexType name="szamlatulajdonosType">
  <xs:sequence>
    <xs:element type="xs:string" name="nev"/>
    <xs:element type="xs:string" name="szuldatum"/>
    <xs:element type="lakcimType" name="lakcim"/>
    <xs:element type="xs:long" name="telefonszam"/>
  </xs:sequence>
  <xs:attribute type="xs:int" name="sztkod"
use="required"/>
</xs:complexType>
```

Tulajdonos:

```
<xs:complexType name="tulajdonosType">
  <xs:simpleContent>
    <xs:extension base="xs:string">
      <xs:attribute type="xs:int" name="bsz_fk"
use="required"/>
       <xs:attribute type="xs:int" name="szt_fk"
use="required"/>
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>
```

Bankkártya:

```
<xs:complexType name="bankkartyaType">
  <xs:sequence>
    <xs:element type="xs:string" name="kartyaszam"/>
    <xs:element type="xs:date" name="lejarati_datum"/>
    <xs:element type="xs:string" name="cvv"/>
  </xs:sequence>
  <xs:attribute type="xs:int" name="bkkod"
use="required"/>
  <xs:attribute type="xs:int" name="bsz_fk"
use="required"/>
</xs:complexType>
```

Tranzakció:

```
<xs:complexType name="tranzakcioType">
  <xs:sequence>
    <xs:element type="xs:string" name="kuldo_sz"/>
    <xs:element type="xs:string" name="fogado_sz"/>
    <xs:element type="xs:int" name="osszeg"/>
    <xs:element type="xs:date" name="datum"/>
```

```
  </xs:sequence>
  <xs:attribute type="xs:int" name="tkod"
use="required"/>
     <xs:attribute type="xs:int" name="bsz_fk"
use="required"/>
</xs:complexType>
```

Bankfiók:

```
<xs:complexType name="bankfiokType">
  <xs:sequence>
    <xs:element type="lakcimType" name="cim"/>
  </xs:sequence>
  <xs:attribute type="xs:int" name="bfkod"
use="required"/>
</xs:complexType>
```

Munkavállaló:

```
<xs:complexType name="munkavallaloType">
  <xs:sequence>
    <xs:element type="xs:string" name="nev"/>
    <xs:element type="xs:string" name="telefonszam"/>
    <xs:element type="xs:float" name="fizetes"/>
    <xs:element type="xs:string" name="pozicio"/>
  </xs:sequence>
  <xs:attribute type="xs:int" name="mvkod"
use="required"/>
  <xs:attribute type="xs:int" name="bf_fk"
use="required"/>
</xs:complexType>
```

2. feladat

A feladat egy DOM program készítése az XML dokumentum - XMLNeptunkod.xml – adatainak adminisztrálása alapján

2a) adatolvasás

A DOM olvasás során kiválasztom az összes egyedet:

```
ArrayList<NodeList> nodeLists = new
ArrayList<NodeList>();
nodeLists.add(document.getElementsByTagName("bankszamla")
);
nodeLists.add(document.getElementsByTagName("szamlatulajd
onos"));
...
```

Majd a listán átiterálva, egy switch-blokkban kiválasztom egyenként az egyedek tulajdonságait.

```
switch (node.getNodeName()) {
case "bankszamla":
System.out.println("\tbszkod: " +
     el.getAttribute("bszkod"));
System.out.println("\tszamlaszam: " +
           DomReadR3SZY2.getElement(el, "szamlaszam"));
System.out.println("\tegyenleg: " +
          DomReadR3SZY2.getElement(el, "egyenleg"))
break;
```

Ennek a kimenete:

```
current element: bankszamla
    bszkod: 1
    szamlaszam: 111-222-333
    egyenleg: 120000
```

2b) adatmódosítás

Adatmódosításként a “2” illetve “3” id-vel rendelkező Bankszámla egyedek egyenlegét módosítom:

```
if ("egyenleg".equals(el.getNodeName())) {
     Element parent = (Element) el.getParentNode();
     if ((parent.getAttribute("bszkod").equals("2"))) {
          el.setTextContent("30000");
     }
     if ((parent.getAttribute("bszkod").equals("3"))) {
          el.setTextContent("195000");
     }
}
```

És kitörlöm az első Bankkártya elemet:

```
NodeList childNodes = root.getChildNodes();

for (int i = 0; i < childNodes.getLength(); i++) {
     Node node = childNodes.item(i);

     // delete first "bankkartya" node
     if ("bankkartya".equals(node.getNodeName())) {
          root.removeChild(node);
          break;
     }
}
```

2c) adatlekérdezés

A DomQueryR3SZY2 osztály query metódusának paramétere egy fájl, és egy sztring, amilyen nevű elemek értékét kérdezzük le.

Az adatolvasáshoz hasonlóan, a kiválasztott elemeken végigiterálok, és ha egyezik az elem neve a kért stringgel, akkor kiíratom a konzolra:

```
for (int j = 0; j < nodeList.getLength(); j++) {
  Node node = nodeList.item(j);
  if (node.getNodeType() == Node.ELEMENT_NODE) {
    Element el = (Element) node;
    NodeList nodeList1  =
el.getElementsByTagName(elementName);
    if (nodeList1.getLength() > 0) {
      System.out.println("found element \"" + elementName
+ "\": " + nodeList1.item(0).getTextContent());
      found = true;
}
```

Ha az elem nem található, akkor pedig az lesz kiírva:

```
if (!found) {
  System.out.println("did not find element \"" +
elementName + "\"");
}
```