JEGYZŐKÖNYV

Adatkezelés XML-ben

Féléves feladat

Cukrászda

Készítette: **Kelemen Beáta**

Neptunkód: **XBG3S6**

Dátum: **2025. december**

Miskolc, 2025

Tartalomjegyzék

Bevezetés 3

1. feladat 4

1.1 Az adatbázis ER modell tervezése 4

1.2 Az adatbázis konvertálása XDM modellre 6

1.3 Az XDM modell alapján XML dokumentum készítése 7

1.4 Az XML dokumentum alapján XMLSchema készítése 11

2. feladat 20

2.1 Adatolvasás 20

2.2 Adatlekérdezés 21

2.3 Adatmódosítás 25

Bevezetés

A feladat leírása:

A beadandó feladata egy olyan relációs adatbázis tervezése és megvalósítása, amely egy cukrászda működését és mindennapi folyamatait kezeli. A rendszer célja, hogy a cukrászda teljes működését átláthatóan és rendszerezetten tárolja, beleértve a rendelések kezelését, a vevők adatait, a futárok munkáját és a beszállítókkal való kapcsolatokat. Az adatbázis lefedi mind a hagyományos, pultos értékesítést, mind pedig az online és telefonos rendeléseket, ezáltal egy modern működést modellez. A rendszerben lehetőség van különböző adatok lekérdezésére és összekapcsolására. Lekérdezhetők például a cukrászdában dolgozó futárok adatai, a hozzájuk tartozó rendelések és kiszállítások, valamint a beszállító cégek, akiktől a cukrászda a hozzávalókat rendeli. Emellett a vevők adatai, például nevük, elérhetőségük, címeik és rendeléseik, is tárolásra kerülnek, így könnyen visszakereshető, hogy egy adott vevő mikor és milyen süteményeket rendelt. Az adatbázis kialakításánál fontos szempont volt, hogy ne legyenek fölösleges ismétlődések, és az adatok között egyértelmű kapcsolatok legyenek. Az egyedek között többféle kapcsolat is megjelenik, például egy cukrászdának több futára lehet, egy futár több helyen is dolgozhat, vagy egy beszállító több cukrászdát is elláthat alapanyagokkal. A modell így jól tükrözi a valós működési környezetet. A rendszer segítségével a működés hatékonyabbá és átláthatóbbá válik, így gyorsan elérhetők a rendelési adatok, egyszerűen nyomon követhetők a futárok teljesítményei, és könnyen kezelhetők a beszállítások. Összességében az adatbázis hozzájárul a cukrászda informatikai folyamataihoz.

1. feladat

1.1 Az adatbázis ER modell tervezése

- **A Beszállítók egyed tulajdonságai:**
  - *BeszállítókID:* A beszállítók egyed azonosítója, elsődleges kulcsa.
  - *Név:* A beszállító cég neve.
  - *Telefonszám:* A beszállító cég telefonszáma, többértékű tulajdonság.
  - *Cím:* A beszállító cég címe, összetett tulajdonság.
- **A Cukrászda egyed tulajdonságai:**
  - *CukrászdaID:* A Cukrászda egyed azonosítója, elsődleges kulcsa.
  - *Név:* A cukrászda neve.
  - *Nyitva tartás:* A cukrászda nyitva tartási ideje.
  - *Elérhetőség:* A cukrászda elérhetősége, ide tartozik a weboldal és a telefonszám, összetett tulajdonság.
- **A Futár egyed tulajdonságai:**
  - *FutárID:* A Futár egyed azonosítója, elsődleges kulcsa.
  - *Név:* A futár neve.
  - *Telefonszám:* A futár telefonszáma.
- **A Sütemények egyed tulajdonságai:**
  - *RendelésID:* A Sütemények egyed rendelési azonosítója, elsődleges kulcsa.
  - *Dátum:* A rendelés dátuma.
  - *Ár:* A rendelt sütemények teljes ára, származtatott tulajdonság. Ami a megvásárolt termékek teljes árát összegzi.
  - *Mennyiség:* A megrendelt termékből vásárolt mennyiség.
  - *Termék:* Egy bizonyos sütemény, amit a vevő vásárolt.
- **A Vevő egyed tulajdonságai:**
  - *VevőID:* A Vevő egyed azonosítója, elsődleges kulcsa.
  - *Telefonszám:* A vevő telefonszáma, többértékű tulajdonság.
  - *Cím:* A vevő címe, összetett tulajdonság.
  - *Név:* A vevő neve.
  - *E-mail:* A vevő e-mail címe.
- **A Bankkártya egyed tulajdonságai:**
  - *Kártyaszám:* A Bankkártya egyed azonosítója, elsődleges kulcsa.
  - *Lejárati dátum:* A bankkártya lejárati dátuma.

- *Cím:* A bankkártyát kiállító bank neve, amelyhez tartozik.

Az Egyedek közötti kapcsolatok:

- **Cukrászda és Futár (Szállítás):**

  A *Cukrászda* és a *Futár* egyedek között több-több (N:M) kapcsolat van, mivel a cukrászda alkalmazhat több futárt is a szállítási feladatok elvégzésére, illetve egy futár több helyen is végezheti ezt a munkakört.

- **Beszállítók és Cukrászda (Beszállítás):**

  A *Beszállítók* és a *Cukrászda* egyedek között több-több (N:M) kapcsolat van, hiszen egy cukrászda rendelhet egyszerre több beszállítótól, valamint egy beszállító szállíthat több vendéglátóhelynek, cukrászdának is. A kapcsolat paraméterei: a Hozzávállalók, amely a beszállító által beszállított hozzávalókat jelenti, illetve a Dátum a beszállítás dátumát.

  adott sütemény csak egy cukrászdához tartozhat.

- **Vevő és Sütemények (Rendelés):**

  A Vevő és a *Sütemények* egyedek között több-több (N:M) kapcsolat van, mivel egy vevő rendelhet többféle süteményt, de egy sütemény, vagy terméktípus több vevő rendelésében is szerepelhet.

- **Cukrászda és Sütemények (Készítés):**

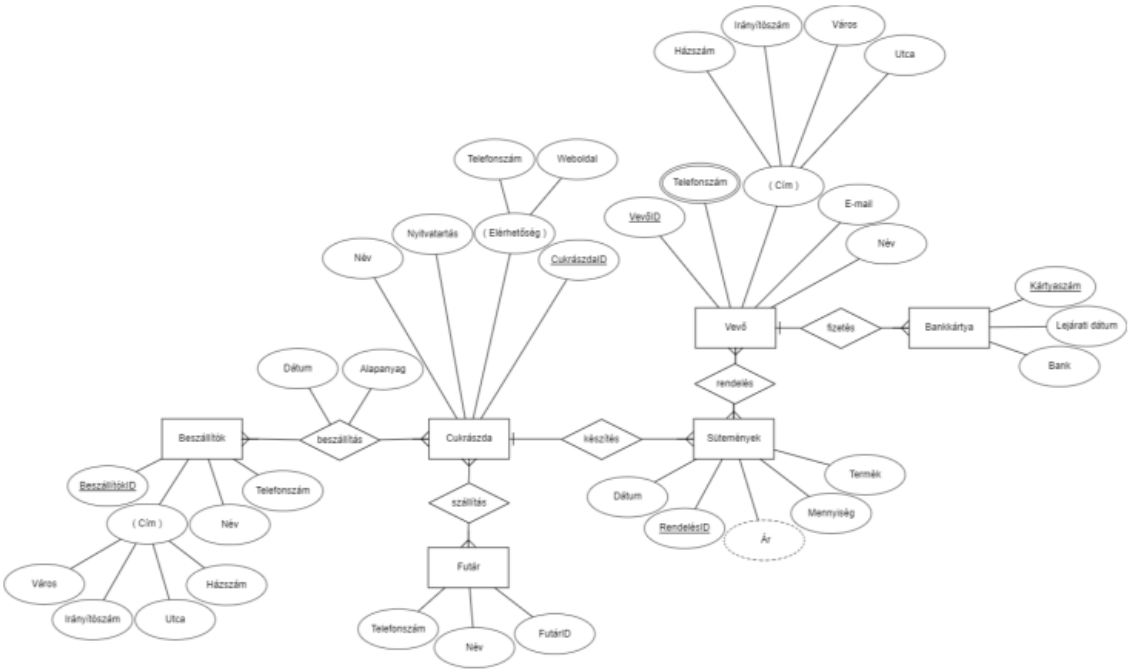
  A *Cukrászda* és a *Sütemények* egyedek között egy-több (1:N) kapcsolat van, ugyanis egy cukrászdának többféle sütemény kínálata van, de egy adott sütemény csak egy cukrászdához tartozhat.

- **Vevő és Bankkártya (Fizetés):**

  A *Vevő* és a *Bankkártya* egyedek között egy-több (1:N) kapcsolat van, mivel egy vevőnek lehet több bankkártyája is, de egy bankkártyának nem lehet több tulajdonosa.

- Ebben a környezetben az 1:1 kapcsolatot nem lehet „értelmesen” alkalmazni.

ER modell:



1. ábra: A Cukrászda adatbázis ER modellje

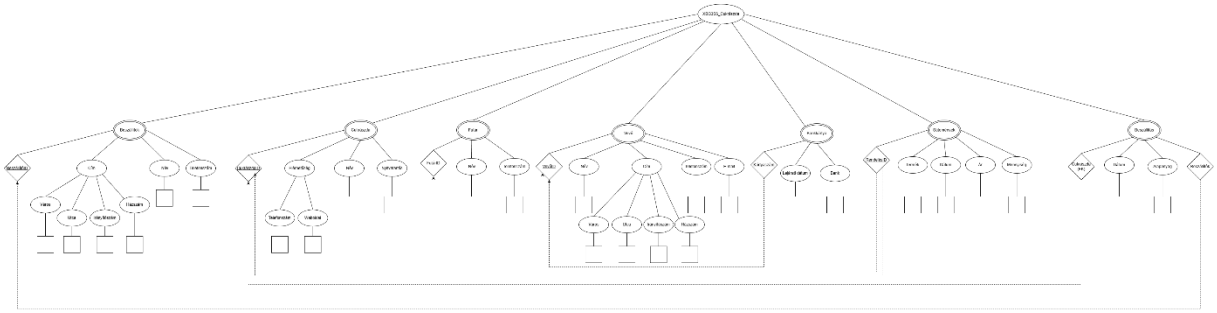
1.2 Az adatbázis konvertálása XDM modellre

Az XDM modell készítésénél egy egységes jelölésrendszert használunk, hogy áttekinthető legyen. A különböző elemeket eltérő alakzatokkal jelöljük.

- **Elemek**: Az elemeket egyszerű **ellipszissel** ábrázoljuk. Ide tartoznak az egyedek és azok tulajdonságai is, amelyekből később XML elemek lesznek.
- **Attribútumok**: Az attribútumokat **rombusz** jelöli. Ezek a kulcstulajdonságokból (például az azonosítókból vagy idegen kulcsokból) származnak, és az XML-ben majd attribútumként fognak megjelenni.
- **Konkrét szöveges érték**: A ténylegesen megjelenő szöveget, amelyet az XML elemek tartalmaznak (például egy név vagy cím értéke), **téglalappal** jelöljük.
- **Többször előforduló elemek**: Azokat az elemeket, amelyekből az XML-ben több példány is lehet, **kettős ellipszis** jelöli. Ez mutatja, hogy adott elem ismétlődhet (például több vevő, több rendelés stb.).
- **Kapcsolat kulcs és idegen kulcs között**: Az idegen kulcs és a hozzá tartozó elsődleges kulcs közötti kapcsolatot **szaggatott vonallal és nyíllal** jelöljük. A nyíl az idegen

kulcsot tartalmazó elem felől mutat az elsődleges kulcsot jelölő elemre, ez érzékelteti a hivatkozást.

XDM Modell:



2. ábra: A Cukrászda XDM modellje

1.3 Az XDM modell alapján XML dokumentum készítése

Az XDM modell alapján elkészítettem az XML dokumentumot, melynél root, azaz gyökér element-ként "XBG3S6_Cukrászda" szerepel. A gyermek elemekből létrehoztam a példányokat, ezeknek az attribútumai tartalmazzák a kulcsokat és idegenkulcsokat is. Illetve az elemekhez további gyermek elemeket is kerültek.

Az XML forráskód:

```
<?xml version="1.0" encoding="UTF-8"?>
<XBG3S6_Cukraszda xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                  xsi:noNamespaceSchemaLocation="XBG3S6_XMLSchema.xsd">

    <!-- Beszállítók -->

    <!-- elso beszallito peldany -->
    <beszallito BeszallitoID="B01">
        <nev>Liszt Kft.</nev>
        <cim>
            <varos>Miskolc</varos>
            <iranyitoszam>3525</iranyitoszam>
            <utca>Kenyérgyár utca</utca>
            <hazszam>12</hazszam>
        </cim>
        <telefonszam>36301234567</telefonszam>
        <telefonszam>36701234567</telefonszam>
    </beszallito>
```

```xml
    <!-- masodik beszallito peldany -->
    <beszallito BeszallitoID="B02">
        <nev>Cukorvilág Bt.</nev>
        <cim>
            <varos>Debrecen</varos>
            <iranyitoszam>4026</iranyitoszam>
            <utca>Cukrász tér</utca>
            <hazszam>3</hazszam>
        </cim>
        <telefonszam>36201239876</telefonszam>
    </beszallito>


    <!-- Cukrászdák -->

    <!-- elso cukraszda peldany -->
    <cukraszda CukraszdaID="C01">
        <nev>Kisgergely Cukrászda</nev>
        <nyitvatartas>H-P 9:00-20:00</nyitvatartas>
        <elerhetoseg>
            <weboldal>https://kisgergely.hu</weboldal>
            <telefonszam>36303037697</telefonszam>
        </elerhetoseg>
    </cukraszda>

    <!-- masodik cukraszda peldany -->
    <cukraszda CukraszdaID="C02">
        <nev>Gyöngy Cukrászda</nev>
        <nyitvatartas>H-V 10:00-19:00</nyitvatartas>
        <elerhetoseg>
            <weboldal>https://kisgergely.hu</weboldal>
            <telefonszam>3646531241</telefonszam>
        </elerhetoseg>
    </cukraszda>


    <!-- Futárok -->

    <!-- elso futar peldany -->
    <futar FutarID="F01">
        <nev>Nagy Péter</nev>
        <telefonszam>36701234598</telefonszam>
    </futar>

    <!-- masodik futar peldany -->
    <futar FutarID="F02">
        <nev>Kiss Réka</nev>
        <telefonszam>36709876543</telefonszam>
    </futar>
```

```xml
<!-- Vevők  -->

<!-- elso vevo peldany -->
<vevo VevoID="V01">
    <nev>Kovács Anna</nev>
    <cim>
        <varos>Miskolc</varos>
        <iranyitoszam>3527</iranyitoszam>
        <utca>Hegyalja ut</utca>
        <hazszam>7</hazszam>
    </cim>
    <telefonszam>36201234567</telefonszam>
    <telefonszam>36701234567</telefonszam>
    <email>anna.kovacs@example.com</email>
</vevo>

<!-- masodik vevo peldany-->
<vevo VevoID="V02">
    <nev>Szabó Bence</nev>
    <cim>
        <varos>Eger</varos>
        <iranyitoszam>3300</iranyitoszam>
        <utca>Kazinczy ter</utca>
        <hazszam>8</hazszam>
    </cim>
    <telefonszam>36205556666</telefonszam>
    <email>bence.szabo@example.com</email>
</vevo>


<!-- Bankkártyák -->

<!-- elso bankkartya peldany -->
<bankkartya Kartyaszam="4444333322221111" VevoID="V01">
    <lejarat>2026-09</lejarat>
    <bank>OTP Bank</bank>
</bankkartya>

<!-- masodik bankkartya peldany -->
<bankkartya Kartyaszam="5555444433332222" VevoID="V02">
    <lejarat>2027-03</lejarat>
    <bank>Erste Bank</bank>
</bankkartya>


<!-- Sütemények  -->
```

```xml
<!-- elso sutemeny peldany -->
<sutemeny RendelesID="R01" CukraszdaID="C01">
    <termek>Pannonhalmi karamellas torta szelet</termek>
    <mennyiseg>2</mennyiseg>
    <ar>1400</ar>
</sutemeny>

<!-- masodik sutemeny peldany-->
<sutemeny RendelesID="R02" CukraszdaID="C01">
    <termek>Dobos torta szelet</termek>
    <mennyiseg>1</mennyiseg>
    <ar>1100</ar>
</sutemeny>

<!-- harmadik sutemeny peldany-->
<sutemeny RendelesID="R03" CukraszdaID="C02">
    <termek>Rákóczi túrós</termek>
    <mennyiseg>6</mennyiseg>
    <ar>1500</ar>
</sutemeny>


<!-- Kapcsolat (Beszállítók–Cukrászda) -->

<!-- elso beszallitas -->
<beszallitas BeszallitasID="BSZ01" BeszallitoID="B01" CukraszdaID="C01">
    <datum>2025-12-20</datum>
    <hozzavallalok>Finomliszt, vaj, cukor</hozzavallalok>
</beszallitas>

<!-- masodik beszallitas -->
<beszallitas BeszallitasID="BSZ02" BeszallitoID="B02" CukraszdaID="C02">
    <datum>2025-12-05</datum>
    <hozzavallalok>Csokolade, tojas</hozzavallalok>
</beszallitas>


<!--Rendelés kapcsolat (Vevő–Sütemények N:M)  -->

<!-- elso rendeles -->
<rendeles RendelesTetelID="RT01" VevoID="V01" RendelesID="R01">
    <datum>2025-12-01</datum>
</rendeles>

<!-- masodik rendeles -->
<rendeles RendelesTetelID="RT02" VevoID="V01" RendelesID="R02">
    <datum>2025-12-02</datum>
</rendeles>
```

```
    <!-- harmadik rendeles -->
    <rendeles RendelesTetelID="RT03" VevoID="V02" RendelesID="R03">
        <datum>2025-12-03</datum>
    </rendeles>


    <!--  Szállítás kapcsolat (Cukrászda–Futár–Rendelés)
====================== -->

    <!-- elso szallitas -->
    <szallitas SzallitasID="SZ01" RendelesID="R01" FutarID="F01"
CukraszdaID="C01">
        <mod>Házhoz szállítás</mod>
        <szallitasi_koltseg>1500</szallitasi_koltseg>
        <megjegyzes>A rendelés 12:00 és 14:00 között érkezik.</megjegyzes>
    </szallitas>

    <!-- masodik szallitas -->
    <szallitas SzallitasID="SZ02" RendelesID="R03" FutarID="F02"
CukraszdaID="C02">
        <mod>Házhoz szállítás</mod>
        <szallitasi_koltseg>1800</szallitasi_koltseg>
        <megjegyzes>Csengessenek.</megjegyzes>
    </szallitas>


    <!-- Fizetés kapcsolat (Vevő–Bankkártya) -->

    <!-- elso fizetes -->
    <fizetes FizetesID="FZ01" VevoID="V01" Kartyaszam="4444333322221111"
RendelesID="R01">
        <osszeg>3500</osszeg>
        <datum>2025-12-01</datum>
    </fizetes>

    <!-- masodik fizetes -->
    <fizetes FizetesID="FZ02" VevoID="V02" Kartyaszam="5555444433332222"
RendelesID="R03">
        <osszeg>3900</osszeg>
        <datum>2025-12-05</datum>
    </fizetes>

</XBG3S6_Cukraszda>
```

1.4 Az XML dokumentum alapján XMLSchema készítése

A séma készítése során a teljes adatstruktúrát a gyökérelembe (XBG3S6_Cukraszda) szerveztem, ez fogja össze az összes olyan elemet, amelyek a cukrászdák működésével

kapcsolatosak. Ehhez több összetett típust definiáltam, például a beszállítókra, vevőkre, futárokra, rendelésre, fizetésre és a többi entitásra. Ezek a komplex típusok határozzák meg, hogy az adott elem milyen al elemeket tartalmazhat, illetve milyen attribútumai vannak.

A séma elején először az egyszerű típusokat gyűjtöttem össze (név, város, termék, ár), majd létrehoztam egy saját típust is a telefonszámhoz, amely szabályozza, hogy 11 számjegyből álljon. Erre építve készültek el a komplex típusok, amik már a különböző szereplők és adatok teljes szerkezetét írják le.

Miután ezek megvoltak, a gyökérelem alatt határoztam meg, hogy az egyes típusokból több példány is előfordulhat az XML-ben. Ezután létrehoztam az elsődleges kulcsokat (xs:key), amelyek egyértelműen azonosítják a megfelelő elemeket, például a vevőt, rendelést vagy fizetést. A kapcsolatok biztosításához idegen kulcsokat (xs:keyref) is megadtam, így a séma ellenőrzi, hogy minden hivatkozott azonosító valóban létezik.

Végül az 1:1 kapcsolatot is elkészítettem, ebben az esetben azt, hogy egy vevőhöz csak egy bankkártya tartozhasson. Ezt az xs:unique megadásával oldottam meg. Összességében a séma így nemcsak az XML felépítését határozza meg, hanem a kapcsolatok helyességét is ellenőrzi.

Az XMLSchema forráskód:

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
elementFormDefault="qualified">

    <!--saját típusok-->

    <!--stringek -->
    <xs:element name="nev" type="xs:string"/>
    <xs:element name="varos" type="xs:string"/>
    <xs:element name="utca" type="xs:string"/>
    <xs:element name="iranyitoszam" type="xs:integer"/>
    <xs:element name="hazszam" type="xs:integer"/>
    <xs:element name="email" type="xs:string"/>
    <xs:element name="weboldal" type="xs:string"/>
    <xs:element name="nyitvatartas" type="xs:string"/>
    <xs:element name="bank" type="xs:string"/>
    <xs:element name="termek" type="xs:string"/>
    <xs:element name="hozzavallalok" type="xs:string"/>
    <xs:element name="megjegyzes" type="xs:string"/>
    <xs:element name="mennyiseg" type="xs:integer"/>
    <xs:element name="ar" type="xs:integer"/>
    <xs:element name="osszeg" type="xs:integer"/>
    <xs:element name="szallitasi_koltseg" type="xs:integer"/>
```

```
    <!--dátum-->
    <xs:element name="datum" type="xs:date"/>
    <xs:element name="lejarat" type="xs:gYearMonth"/>

    <!--szállítási mód-->
    <xs:element name="mod" type="xs:string"/>

    <!-- saját típus(egyszerű)-telefonszám (11 számjegy) -->
    <xs:simpleType name="TelefonszamTipus">
        <xs:restriction base="xs:string">
            <xs:pattern value="\d{11}"/>
        </xs:restriction>
    </xs:simpleType>

    <xs:element name="telefonszam" type="TelefonszamTipus"/>


    <!--Összetett típusok-->

    <!-- Beszállító: név, cím, telefon(ok) -->
    <xs:complexType name="beszallitoTipus">
        <xs:sequence>
            <xs:element ref="nev"/>
            <xs:element name="cim">
                <xs:complexType>
                    <xs:sequence>
                        <xs:element ref="varos"/>
                        <xs:element ref="iranyitoszam"/>
                        <xs:element ref="utca"/>
                        <xs:element ref="hazszam"/>
                    </xs:sequence>
                </xs:complexType>
            </xs:element>
            <xs:element ref="telefonszam" maxOccurs="unbounded"/>
        </xs:sequence>
        <xs:attribute name="BeszallitoID" type="xs:string" use="required"/>
    </xs:complexType>

    <!-- Cukrászda: név, nyitvatartás, elérhetőség (weboldal + telefonszám) --
>
    <xs:complexType name="cukraszdaTipus">
        <xs:sequence>
            <xs:element ref="nev"/>
            <xs:element ref="nyitvatartas"/>
            <xs:element name="elerhetoseg">
                <xs:complexType>
                    <xs:sequence>
                        <xs:element ref="weboldal"/>
                        <xs:element ref="telefonszam"/>
```

```
                </xs:sequence>
            </xs:complexType>
        </xs:element>
    </xs:sequence>
    <xs:attribute name="CukraszdaID" type="xs:string" use="required"/>
</xs:complexType>

<!-- Futár: név + telefonszám -->
<xs:complexType name="futarTipus">
    <xs:sequence>
        <xs:element ref="nev"/>
        <xs:element ref="telefonszam"/>
    </xs:sequence>
    <xs:attribute name="FutarID" type="xs:string" use="required"/>
</xs:complexType>

<!-- Vevő: név, cím, 1..n telefonszám, email -->
<xs:complexType name="vevoTipus">
    <xs:sequence>
        <xs:element ref="nev"/>
        <xs:element name="cim">
            <xs:complexType>
                <xs:sequence>
                    <xs:element ref="varos"/>
                    <xs:element ref="iranyitoszam"/>
                    <xs:element ref="utca"/>
                    <xs:element ref="hazszam"/>
                </xs:sequence>
            </xs:complexType>
        </xs:element>
        <xs:element ref="telefonszam" maxOccurs="unbounded"/>
        <xs:element ref="email"/>
    </xs:sequence>
    <xs:attribute name="VevoID" type="xs:string" use="required"/>
</xs:complexType>

<!-- Bankkártya: lejárat, bank; Kartyaszam + VevoID -->
<xs:complexType name="bankkartyaTipus">
    <xs:sequence>
        <xs:element ref="lejarat"/>
        <xs:element ref="bank"/>
    </xs:sequence>
    <xs:attribute name="Kartyaszam" type="xs:string" use="required"/>
    <xs:attribute name="VevoID" type="xs:string" use="required"/>
</xs:complexType>

<!-- Sütemény: termék, mennyiség, ár; FK: RendelesID, CukraszdaID -->
<xs:complexType name="sutemenyTipus">
    <xs:sequence>
```

```
            <xs:element ref="termek"/>
            <xs:element ref="mennyiseg"/>
            <xs:element ref="ar"/>
        </xs:sequence>
        <xs:attribute name="RendelesID" type="xs:string" use="required"/>
        <xs:attribute name="CukraszdaID" type="xs:string" use="required"/>
    </xs:complexType>

    <!-- Beszállítás: dátum, hozzávalók; FK: BeszallitoID, CukraszdaID -->
    <xs:complexType name="beszallitasTipus">
        <xs:sequence>
            <xs:element ref="datum"/>
            <xs:element ref="hozzavallalok"/>
        </xs:sequence>
        <xs:attribute name="BeszallitasID" type="xs:string" use="required"/>
        <xs:attribute name="BeszallitoID" type="xs:string" use="required"/>
        <xs:attribute name="CukraszdaID" type="xs:string" use="required"/>
    </xs:complexType>

    <!-- Rendelés tétel: dátum; FK: VevoID, RendelesID -->
    <xs:complexType name="rendelesTipus">
        <xs:sequence>
            <xs:element ref="datum"/>
        </xs:sequence>
        <xs:attribute name="RendelesTetelID" type="xs:string" use="required"/>
        <xs:attribute name="VevoID" type="xs:string" use="required"/>
        <xs:attribute name="RendelesID" type="xs:string" use="required"/>
    </xs:complexType>

    <!-- Szállítás: mód, költség, megjegyzés; FK: RendelesID, FutarID,
CukraszdaID -->
    <xs:complexType name="szallitasTipus">
        <xs:sequence>
            <xs:element ref="mod"/>
            <xs:element ref="szallitasi_koltseg"/>
            <xs:element ref="megjegyzes"/>
        </xs:sequence>
        <xs:attribute name="SzallitasID" type="xs:string" use="required"/>
        <xs:attribute name="RendelesID" type="xs:string" use="required"/>
        <xs:attribute name="FutarID" type="xs:string" use="required"/>
        <xs:attribute name="CukraszdaID" type="xs:string" use="required"/>
    </xs:complexType>

    <!-- Fizetés: összeg, dátum; FK: VevoID, Kartyaszam, RendelesID -->
    <xs:complexType name="fizetesTipus">
        <xs:sequence>
            <xs:element ref="osszeg"/>
            <xs:element ref="datum"/>
        </xs:sequence>
```

```
        <xs:attribute name="FizetesID" type="xs:string" use="required"/>
        <xs:attribute name="VevoID" type="xs:string" use="required"/>
        <xs:attribute name="Kartyaszam" type="xs:string" use="required"/>
        <xs:attribute name="RendelesID" type="xs:string" use="required"/>
    </xs:complexType>


    <!--Gyökérelemek-->

    <xs:element name="XBG3S6_Cukraszda">
        <xs:complexType>
            <xs:sequence>
                <xs:element
name="beszallito"  type="beszallitoTipus"  maxOccurs="unbounded"/>
                <xs:element
name="cukraszda"   type="cukraszdaTipus"   maxOccurs="unbounded"/>
                <xs:element
name="futar"       type="futarTipus"       maxOccurs="unbounded"/>
                <xs:element
name="vevo"        type="vevoTipus"        maxOccurs="unbounded"/>
                <xs:element
name="bankkartya"  type="bankkartyaTipus"  maxOccurs="unbounded"/>
                <xs:element
name="sutemeny"    type="sutemenyTipus"    maxOccurs="unbounded"/>
                <xs:element name="beszallitas" type="beszallitasTipus"
maxOccurs="unbounded"/>
                <xs:element
name="rendeles"    type="rendelesTipus"    maxOccurs="unbounded"/>
                <xs:element
name="szallitas"   type="szallitasTipus"   maxOccurs="unbounded"/>
                <xs:element
name="fizetes"     type="fizetesTipus"     maxOccurs="unbounded"/>
            </xs:sequence>
        </xs:complexType>

        <!--Elsődleges kulcsok (PK)-->

        <xs:key name="beszallitoPK">
            <xs:selector xpath="beszallito"/>
            <xs:field xpath="@BeszallitoID"/>
        </xs:key>

        <xs:key name="cukraszdaPK">
            <xs:selector xpath="cukraszda"/>
            <xs:field xpath="@CukraszdaID"/>
        </xs:key>

        <xs:key name="futarPK">
            <xs:selector xpath="futar"/>
```

```
        <xs:field xpath="@FutarID"/>
    </xs:key>

    <xs:key name="vevoPK">
        <xs:selector xpath="vevo"/>
        <xs:field xpath="@VevoID"/>
    </xs:key>

    <xs:key name="bankkartyaPK">
        <xs:selector xpath="bankkartya"/>
        <xs:field xpath="@Kartyaszam"/>
    </xs:key>

    <!-- azonositok rendelesnel -->
    <xs:key name="rendelesAzonositoPK">
        <xs:selector xpath="sutemeny"/>
        <xs:field xpath="@RendelesID"/>
    </xs:key>

    <xs:key name="beszallitasPK">
        <xs:selector xpath="beszallitas"/>
        <xs:field xpath="@BeszallitasID"/>
    </xs:key>

    <xs:key name="rendelesTetelPK">
        <xs:selector xpath="rendeles"/>
        <xs:field xpath="@RendelesTetelID"/>
    </xs:key>

    <xs:key name="szallitasPK">
        <xs:selector xpath="szallitas"/>
        <xs:field xpath="@SzallitasID"/>
    </xs:key>

    <xs:key name="fizetesPK">
        <xs:selector xpath="fizetes"/>
        <xs:field xpath="@FizetesID"/>
    </xs:key>


    <!-- Idegen kulcsok (FK) -->

    <!-- beszállítás-beszállító -->
    <xs:keyref name="beszallitasBeszallitoFK" refer="beszallitoPK">
        <xs:selector xpath="beszallitas"/>
        <xs:field xpath="@BeszallitoID"/>
    </xs:keyref>

    <!-- beszállítás-cukrászda -->
```

```
        <xs:keyref name="beszallitasCukraszdaFK" refer="cukraszdaPK">
            <xs:selector xpath="beszallitas"/>
            <xs:field xpath="@CukraszdaID"/>
        </xs:keyref>

        <!-- rendelés tétel-vevő -->
        <xs:keyref name="rendelesVevoFK" refer="vevoPK">
            <xs:selector xpath="rendeles"/>
            <xs:field xpath="@VevoID"/>
        </xs:keyref>

        <!-- rendelés tétel-rendelés azonosító (sütemény RendelesID) -->
        <xs:keyref name="rendelesSutemenyFK" refer="rendelesAzonositoPK">
            <xs:selector xpath="rendeles"/>
            <xs:field xpath="@RendelesID"/>
        </xs:keyref>

        <!-- sütemény-cukrászda -->
        <xs:keyref name="sutemenyCukraszdaFK" refer="cukraszdaPK">
            <xs:selector xpath="sutemeny"/>
            <xs:field xpath="@CukraszdaID"/>
        </xs:keyref>

        <!-- szállítás-rendelés -->
        <xs:keyref name="szallitasRendelesFK" refer="rendelesAzonositoPK">
            <xs:selector xpath="szallitas"/>
            <xs:field xpath="@RendelesID"/>
        </xs:keyref>

        <!-- szállítás-cukrászda -->
        <xs:keyref name="szallitasCukraszdaFK" refer="cukraszdaPK">
            <xs:selector xpath="szallitas"/>
            <xs:field xpath="@CukraszdaID"/>
        </xs:keyref>

        <!-- szállítás-futár -->
        <xs:keyref name="szallitasFutarFK" refer="futarPK">
            <xs:selector xpath="szallitas"/>
            <xs:field xpath="@FutarID"/>
        </xs:keyref>

        <!-- fizetés-vevő -->
        <xs:keyref name="fizetesVevoFK" refer="vevoPK">
            <xs:selector xpath="fizetes"/>
            <xs:field xpath="@VevoID"/>
        </xs:keyref>

        <!-- fizetés-bankkártya -->
        <xs:keyref name="fizetesKartyaFK" refer="bankkartyaPK">
```

```
            <xs:selector xpath="fizetes"/>
            <xs:field xpath="@Kartyaszam"/>
        </xs:keyref>

        <!-- 1:1 kapcsolat -->
        <!-- egy vevőhöz legfeljebb egy bankkártya tartozhat -->
        <xs:unique name="vevoBankkartyaUnique">
            <xs:selector xpath="bankkartya"/>
            <xs:field xpath="@VevoID"/>
        </xs:unique>
    </xs:element>
</xs:schema>
```

2. feladat

2.1 Adatolvasás

A DOMRead osztály feladata az XML dokumentum teljes beolvasása, fává alakítása és ennek strukturált kiíratása.

A ReadXMLDocument() metódus egy DocumentBuilder példány segítségével tölti be a megadott XML fájlt, majd a dokumentum normalizálásával egységessé teszi a DOM-struktúrát.

A beolvasott dokumentumot a printDocument() metódus dolgozza fel: kiírja az XML deklarációt, a gyökérelem nevét és attribútumait, majd a getChildNodes() segítségével sorban bejárja a gyermekelemeket.

A bejárás a rekurzív printNode() metóduson keresztül történik, amely kezeli az elemek nevét, attribútumait, a szöveges csomópontokat és a többszintű hierarchiát is.

A program a dokumentumot nemcsak a konzolra írja ki, hanem egy kimeneti fájlba (XBG3S6XML_output.xml) is elmenti.

Az XBG3S6DOMRead forráskód részletei:

```
    //XML beolvasása
    public static void ReadXMLDocument(String filePath) {
        try {
            //fájlbeolvasás
            File xmlFile = new File(filePath);

            DocumentBuilderFactory dbFactory =
DocumentBuilderFactory.newInstance();
            //példányosítás
            DocumentBuilder dBuilder = dbFactory.newDocumentBuilder();
            Document doc = dBuilder.parse(xmlFile);
            doc.getDocumentElement().normalize();

            //kiírás konzolra és fájlba
            printDocument(doc);

        } catch (ParserConfigurationException | IOException | SAXException e)
{
            e.printStackTrace();
        }
    }

    //dokumentum kiírása és mentése
    private static void printDocument(Document doc) {
        PrintWriter writer = null;
        try {
```

```
            //mentés fájlba
            File outputFile = new File("XBG3S6XML_output.xml");
            writer = new PrintWriter(new FileWriter(outputFile, false));

            //deklaráció kiírása
            System.out.print("<?xml version=\"1.0\" encoding=\"UTF-8\"?>\n");
            writer.print("<?xml version=\"1.0\" encoding=\"UTF-8\"?>\n");

            //gyökérelem
            Element rootElement = doc.getDocumentElement();
            String rootName = rootElement.getTagName();

            //gyökérelem attribútumai
            StringJoiner rootAttributes = new StringJoiner(" ");
            NamedNodeMap rootAttributeMap = rootElement.getAttributes();
            for (int i = 0; i < rootAttributeMap.getLength(); i++) {
                Node attribute = rootAttributeMap.item(i);
                rootAttributes.add(attribute.getNodeName() + "=\""
                        + attribute.getNodeValue() + "\"");
            }

            System.out.print("<" + rootName + " " + rootAttributes.toString()
+ ">\n");
            writer.print("<" + rootName + " " + rootAttributes.toString() +
">\n");

            //gyökérelem alatti gyermek elem
            NodeList children = rootElement.getChildNodes();
            for (int i = 0; i < children.getLength(); i++) {
                printNode(children.item(i), 1, writer);
            }

            System.out.println("</" + rootName + ">");
            writer.println("</" + rootName + ">");
        } catch (IOException e) {
            e.printStackTrace();
        } finally {
            if (writer != null) {
                writer.close();
            }
        }
    }
```

2.2 Adatlekérdezés

A DOMQuery osztály különféle lekérdezéseket hajt végre az XML adatai alapján. A dokumentum beolvasása után a szükséges elemeket getElementsByTagName() segítségével

gyűjtöm ki, majd a NodeList-eken végigiterálva készítem el a lekérdezéseket. A keresésekhez egy segédmetódust is használok, amely attribútum alapján adja vissza a megfelelő elemet, így könnyen össze tudom kapcsolni például a rendeléseket a hozzájuk tartozó vevőkkel vagy szállításokkal. A lekérdezések olyan információkat gyűjtenek ki, mint a vevők részletes adatai, rendelések és sütemények párosítása, nagyobb összegű házhoz szállítások futárokkal, illetve a beszállítások kapcsolata a cukrászdákkal. Az eredményeket a program áttekinthető formában a konzolra írja.

Az XBG3S6DOMQuery forráskód részlete:

```
    public static void QueryPrescribedDetails(String filePath) {

        Document doc = null;

        try {
            //fájlbeolvasás
            File inputFile = new File(filePath);

            //példányosítás
            DocumentBuilderFactory dbFactory =
DocumentBuilderFactory.newInstance();

            DocumentBuilder dBuilder = dbFactory.newDocumentBuilder();

            doc = dBuilder.parse(inputFile);

            //normalizálása
            doc.getDocumentElement().normalize();

        } catch (Exception e) {
            e.printStackTrace();
            return;
        }

        //1. lekérdezés, összes vevő adatai
        System.out.println();
        System.out.println("1. Lekérdezés:");
        System.out.println("Összes vevő adatainak kiírása:");
        System.out.println();

        NodeList vevoList = doc.getElementsByTagName("vevo");

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

            if (node.getNodeType() == Node.ELEMENT_NODE) {
```

```
                Element vevo = (Element) node;

                String vevoID = vevo.getAttribute("VevoID");
                String nev =
vevo.getElementsByTagName("nev").item(0).getTextContent();

                Element cimElem = (Element)
vevo.getElementsByTagName("cim").item(0);
                String varos =
cimElem.getElementsByTagName("varos").item(0).getTextContent();
                String irsz =
cimElem.getElementsByTagName("iranyitoszam").item(0).getTextContent();
                String utca =
cimElem.getElementsByTagName("utca").item(0).getTextContent();
                String hazszam =
cimElem.getElementsByTagName("hazszam").item(0).getTextContent();

                NodeList telefonok = vevo.getElementsByTagName("telefonszam");
                String email =
vevo.getElementsByTagName("email").item(0).getTextContent();

                System.out.println("VevoID: " + vevoID);
                System.out.println("Név: " + nev);
                System.out.println("Cím: " + varos + " " + irsz + ", " + utca
+ " " + hazszam);

                for (int j = 0; j < telefonok.getLength(); j++) {
                    System.out.println("Telefonszám: " +
telefonok.item(j).getTextContent());
                }

                System.out.println("E-mail: " + email);
                System.out.println("----------------------------------------
");
            }
        }
```

A lekérdezett adatok:

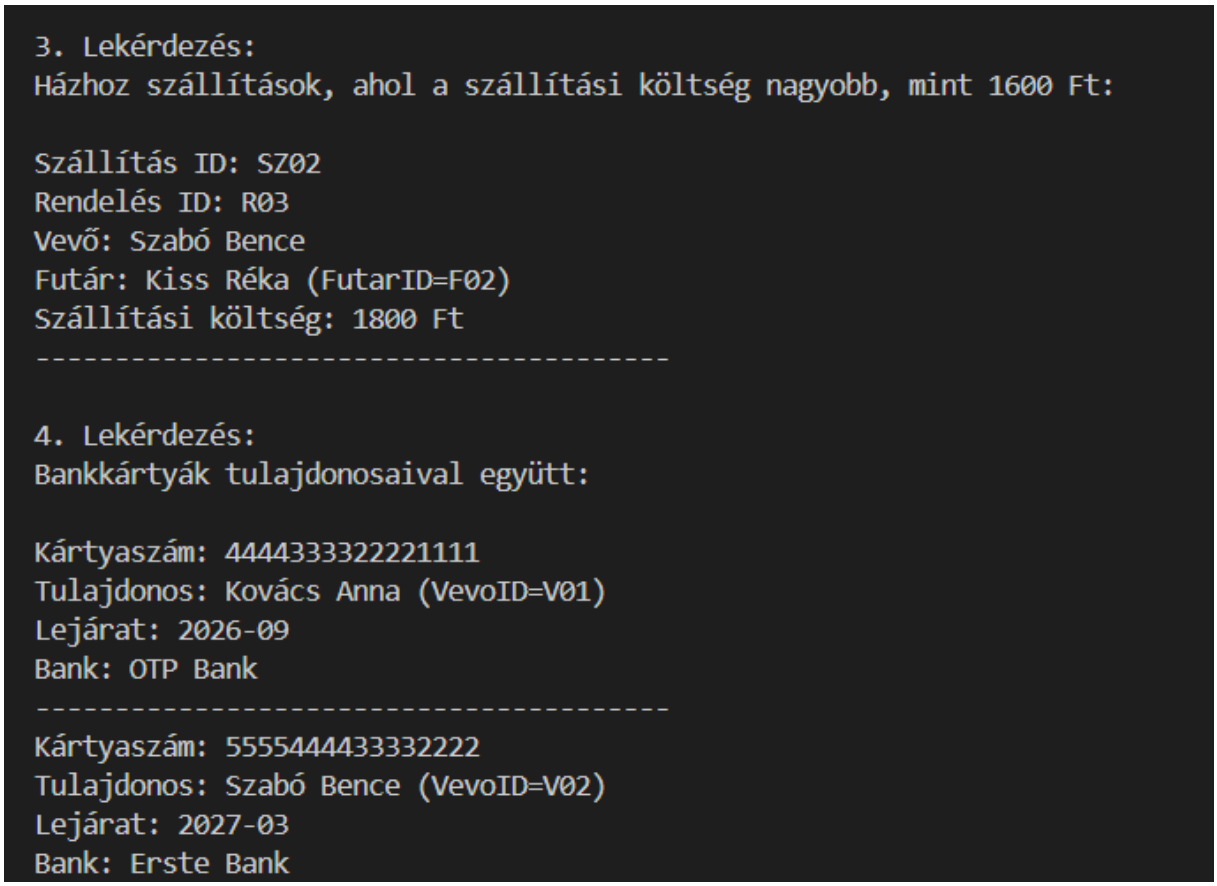
```
1. Lekérdezés:
Összes vevő adatainak kiírása:

VevoID: V01
Név: Kovács Anna
Cím: Miskolc 3527, Hegyalja ut 7
Telefonszám: 36201234567
Telefonszám: 36701234567
E-mail: anna.kovacs@example.com
----------------------------------------
VevoID: V02
Név: Szabó Bence
Cím: Eger 3300, Kazinczy ter 8
Telefonszám: 36205556666
E-mail: bence.szabo@example.com
----------------------------------------

2. Lekérdezés:
Rendelések vevőkkel és rendelt süteményekkel:

Rendelés ID: R01 (2025-12-01)
Vevő: Kovács Anna (VevoID=V01)
Cukrászda: Kisgergely Cukrászda (CukraszdaID=C01)
Termék: Pannonhalmi karamellas torta szelet, mennyiség: 2, ár: 1400 Ft
----------------------------------------
Rendelés ID: R02 (2025-12-02)
Vevő: Kovács Anna (VevoID=V01)
Cukrászda: Kisgergely Cukrászda (CukraszdaID=C01)
Termék: Dobos torta szelet, mennyiség: 1, ár: 1100 Ft
----------------------------------------
Rendelés ID: R03 (2025-12-03)
Vevő: Szabó Bence (VevoID=V02)
Cukrászda: Gyöngy Cukrászda (CukraszdaID=C02)
Termék: Rákóczi túrós, mennyiség: 6, ár: 1500 Ft
----------------------------------------
```

1. kép: A lekérdezett adatok megjelenítése



2. kép: A lekérdezett adatok megjelenítése

2.3 Adatmódosítás

A DOMModify osztály feladata az XML dokumentum előre meghatározott elemeinek módosítása DOM műveletekkel.

A ModifyElement() metódus beolvassa a teljes XML fájlt egy DocumentBuilder segítségével, majd a dokumentum normalizálása után meghívja a ModifyPrescribedElements() metódust, amely elvégzi a konkrét változtatásokat.

A módosítási folyamat a gyökérelemtől indul: a getElementsByTagName() metódussal kikeresem a szükséges csomópontokat (például beszállító, cukrászda, vevő, futár, sütemény vagy fizetés), majd a megfelelő al-elemek szövegét a setTextContent() hívással átírom.

Így módosítható például egy beszállító neve, egy cukrászda nyitvatartása, egy vevő címe vagy egy sütemény ára.

A frissített dokumentumot nem külön fájlba mentem, hanem a printDocument() metódus segítségével egy Transformer írja ki formázott XML-ként a konzolra, ami lehetővé teszi a módosítások gyors és átlátható ellenőrzését.

Az XBG3S6DOMModify forráskód részlete:

```
    //XML elemek módosítása
    public static void ModifyElement(String filePath) {

        try {
            //fájlbeolvasás
            File inputFile = new File(filePath);

            if (!inputFile.exists()) {
                System.out.println("Nem található XML fájlt: " +
inputFile.getAbsolutePath());
                return;
            }

            DocumentBuilderFactory dbFactory =
DocumentBuilderFactory.newInstance();

            //példányosítás
            DocumentBuilder dBuilder = dbFactory.newDocumentBuilder();

            //beolvasás
            Document doc = dBuilder.parse(inputFile);

            //normalizálás
            doc.getDocumentElement().normalize();

            //módosítások elvégzése
            ModifyPrescribedElements(doc);

        } catch (Exception e) {
            e.printStackTrace();
        }
    }

    //módosítások
    private static void ModifyPrescribedElements(Document doc) throws
TransformerException {

        //Gyökérelem
        Element root = doc.getDocumentElement();
        if (root == null) {
            System.out.println("Nincs gyökérelem a dokumentumban");
            return;
        }

        //az első beszállító nevének módosítása
        NodeList beszallitoList = root.getElementsByTagName("beszallito");
        if (beszallitoList.getLength() > 0) {
```

```
            Element beszallito = (Element) beszallitoList.item(0);
            beszallito.getElementsByTagName("nev")
                    .item(0)
                    .setTextContent("Prémium Liszt Kft.");
        }
```