JEGYZŐKÖNYV

Adatkezelés XML környezetben

Féléves feladat

Készítette: Rőthi János
Neptunkód: NDRQ1

Tartalomjegyzék

A feladat leírása

Az én témám egy város tömegközlekedési vállalatának a modellezése. Az adatbázisának ER-modellezése, konvertálása XDM modellre, majd az XDM modell alapján XML dokumentum és XMLSchema készítés, majd DOM programok készítése az XML dokumentum adatainak olvasására és módosítására.

Az ER-modell az alábbi jellemzőkkel épül fel:

A **Jármű** egyedhez tartoznak a *Rendszámtábla*, a *Férőhely*, a *Gyártó* és a *Típus* tulajdonságok. A *Rendszámtábla* kulcsként funkcionál, mert egyértelműen azonosítja az adott **Jármű**vet.

A **Járművezető** egyednek a *Mikor vezeti lesz* a kulcsa. A **Járművezető**t a **Jármű** egyeddel összekötő kapcsolat a VEZETI. A **Járművezető**nek az *Életkor* attribútuma származtatott, ugyanis az kiszámítható a *Születési dátum* tulajdonságából. A *Jogosítványok* egy többértékű tulajdonság, hiszen egy **Járművezető**nek több, különböző jogosítványa is lehet. A *Név* egy összetett attribútum, amelyet a *Vezetéknév* és a *Utónév* altulajdonságok alkotnak.

A **Jármű** és a **Járművezető** egyedek közötti kapcsolat kötelező, mert valakinek vezetnie kell az adott járművet és a járművezetőnek is kell vezetnie egy járművet (ekkor kerül adatbázisba). 1:1 típusú, hiszen egy járművet egy ember vezethet és egy ember is egy járművet képes vezetni.

A **Menetrend** egyed tulajdonságai a *Menetrend azonosító*, az *Indulás időpontja* és az *Érkezés időpontja*. A *Menetrend azonosító* szolgál kulcsként.

A **Menetrend** és a **Jármű** egyedek közötti KÖZLEKEDIK kapcsolat 1:N típusú, több **Jármű** is közlekedhet ugyanazon menetrend szerint. Kötelező kapcsolat.

Az **Ellenőr** egyed tulajdonsága az *Ellenőr azonosító*, amely a kulcs szerepét is betölti. Az **Ellenőr** egyed továbbá jelentkezik egy *Név* összetett tulajdonságból, amely a *Vezetéknév* és *Utónév* tagokból áll össze.

Az **Ellenőr** és a **Jármű** egyedek között a TARTÓZKODIK kapcsolat áll fenn, amely a **Jármű** felé kötelező, az **Ellenőr** felé opcionális (az ellenőrnek járművön kell lennie, de nem kell minden járművon ellenőrnek lennie) és 1:1 típusú (egy járművön egy ellenőr lehet).

Az **Útvonal** egyedhez tartozó tulajdonságok a *Vonalszám*, a *Távolság*, a *Kilométerár* és a *Viteldíj*. A *Vonalszám* kulcs, amely egyértelműen azonosítja az adott **Útvonal**at. A *Viteldíj* egy származtatott tulajdonság, a *Távolság* és a *Kilométerár*ból számolható.

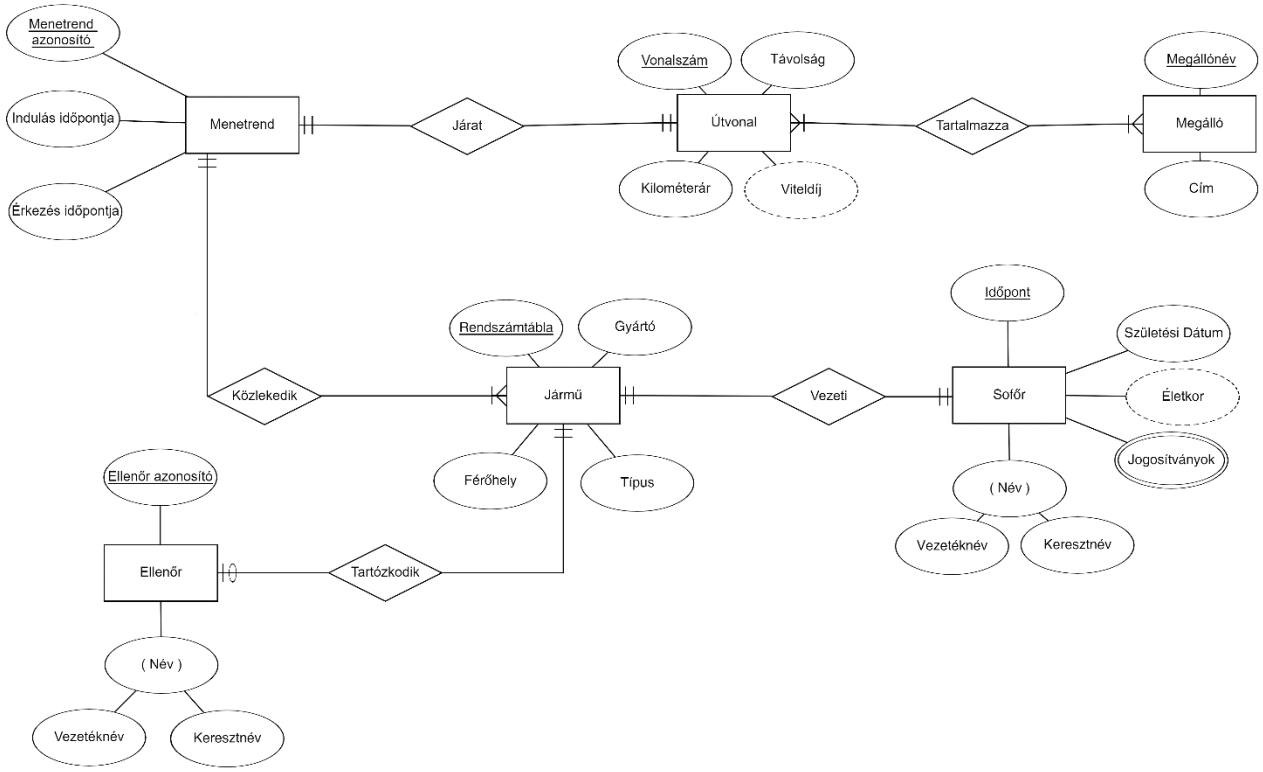
Az **Útvonal** és a **Menetrend** egyedek között lévő JÁRAT kapcsolat kötelező (minden útvonalon van menetrend) és 1:1 típusú (egy útvonalon egyféle menetrend).

A **Megálló** is egy egyed, amelyben a *Megállónév* tulajdonsága kulcsként funkcionál. A **Megálló** egyednek tulajdonsága továbbá a *Cím*.

Az **Útvonal** és a **Megálló** egyedek között a TARTALMAZZA kapcsolat van, amely N:N típusú, ugyanis egy útvonalhaz több megálló is tartozhat és egy megállóhoz is tartozhat többféle útvonal. Kötelező kapcsolat, hiszen megállók nélkül nincs útvonal és útvonal nélkül nincs értelme egy megállónak sem.

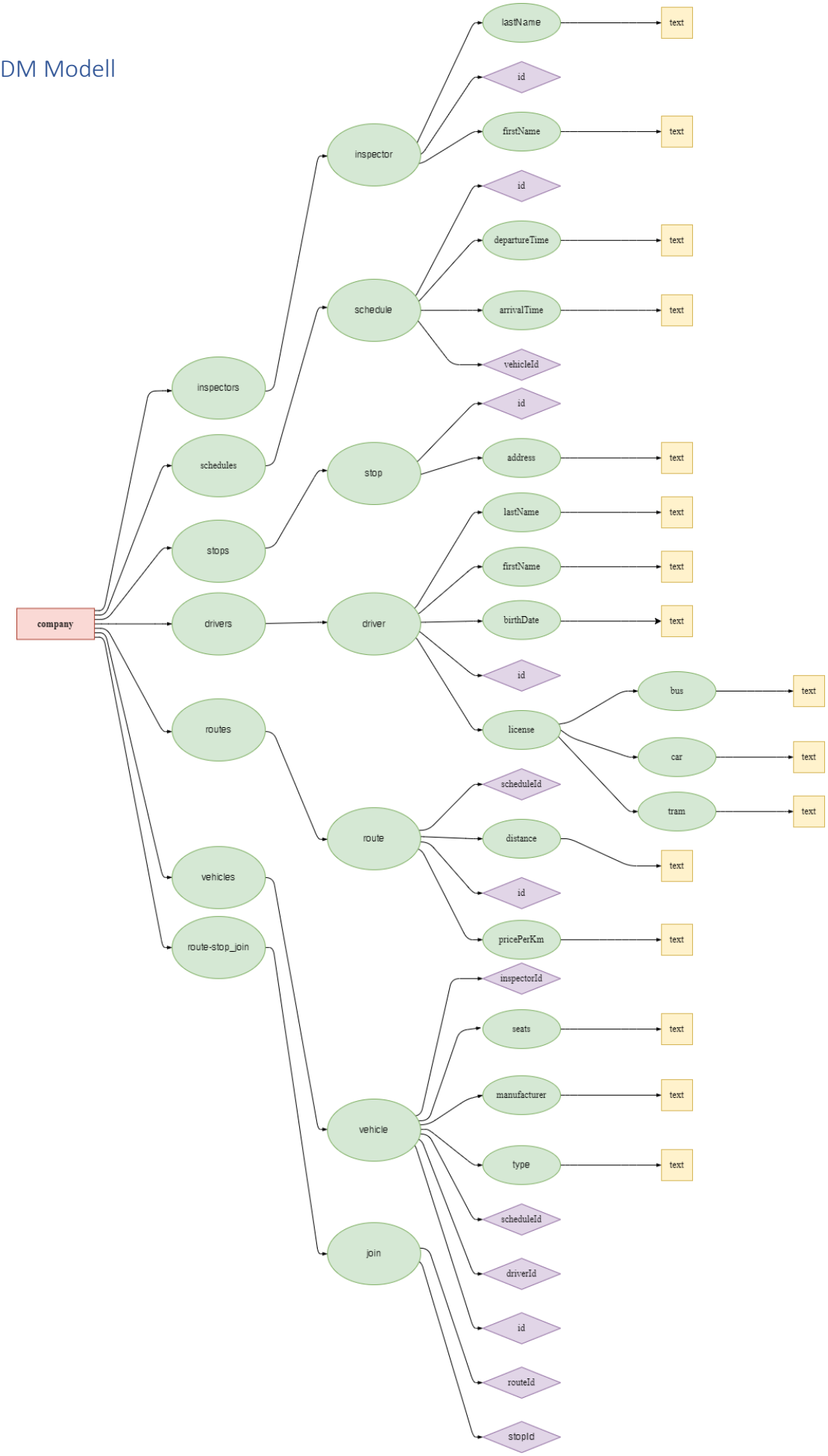
1. Feladat

1.1: ER Modell



Az ER-modellt az ERDplus programmal készítettem el (https://erdplus.com/)

1.2: XDM Modell



Az XDM modellt a draw.io programmal szerkesztettem (https://app.diagrams.net/).

Az egyedekből elemek keletkeztek, az elsődleges és idegen kulcsokból attribútumokká alakultak. A többértékű tulajdonságból egy külön elemet készítünk ahol a megadható értékeket gyerekelemekként rögzítjük. Az 1:1 és 1:N típusú kapcsolatokat csak idegen kulcsként rögzítjük, az N:M kapcsolatokat pedig kapcsolótáblával kezeljük. Az összetett tulajdonságokat elhagyjuk, csak azokat a tulajdonságokat rögzítjük elemként amikből az összetett tulajdonság előáll. A származtatott tulajdonságokat is szintén el- hagyjuk hiszen azt ki tudjuk mi is számolni a programunkban.

1.3: Az XML kód

```xml
<?xml version="1.0" encoding="UTF-8"?>
<company xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="XMLSchemaNDRQ1Y.xsd">
    <inspectors>
        <inspector id="2">
            <lastName>Jean</lastName>
            <firstName>Todt</firstName>
        </inspector>
        <inspector id="4">
            <lastName>Laboda</lastName>
            <firstName>Dániel</firstName>
        </inspector>
        <inspector id="6">
            <lastName>Elek</lastName>
            <firstName>Zoltán</firstName>
        </inspector>
    </inspectors>
    <schedules>
        <schedule id="1" vehicleId="ASD-123">
            <arrivalTime>05:50:00</arrivalTime>
            <departureTime>06:15:00</departureTime>
        </schedule>
        <schedule id="3" vehicleId="QWE-968">
            <arrivalTime>08:30:00</arrivalTime>
            <departureTime>08:35:00</departureTime>
        </schedule>
        <schedule id="5" vehicleId="ADC-666">
            <arrivalTime>10:17:00</arrivalTime>
            <departureTime>10:50:00</departureTime>
        </schedule>
        <schedule id="7" vehicleId="BOT-404">
            <arrivalTime>12:40:00</arrivalTime>
            <departureTime>13:00:00</departureTime>
        </schedule>
        <schedule id="8" vehicleId="TOP-282">
            <arrivalTime>20:22:00</arrivalTime>
            <departureTime>20:50:00</departureTime>
        </schedule>
    </schedules>
```

```
    <stops>
        <stop id="Bacsó Béla Utca">
            <address>Bacsó Béla út 40.</address>
        </stop>
        <stop id="Soltész Nagy Kálmán">
            <address>Soltész Nagy Kálmán út 8.</address>
        </stop>
        <stop id="Vízügyi igazgatóság">
            <address>Vörösmarty Mihály utca. 77</address>
        </stop>
        <stop id="Miskolctapolca-Barlangfürdő">
            <address>Pazár István Sétány 1.</address>
        </stop>
        <stop id="Római part">
            <address>Római Part 30</address>
        </stop>
    </stops>
    <drivers>
        <driver id="1111-03-22">
            <birthDate>1969-01-03</birthDate>
            <lastName>Mischael</lastName>
            <firstName>Schumacher</firstName>
            <license>
                <car>true</car>
                <bus>true</bus>
                <tram>true</tram>
            </license>
        </driver>
        <driver id="1111-03-23">
            <birthDate>1960-03-21</birthDate>
            <lastName>Ayrton</lastName>
            <firstName>Senna</firstName>
            <license>
                <car>true</car>
                <bus>true</bus>
                <tram>false</tram>
            </license>
        </driver>
        <driver id="1111-03-24">
            <birthDate>1955-02-24</birthDate>
            <lastName>Alain</lastName>
            <firstName>Prost</firstName>
            <license>
                <car>false</car>
                <bus>false</bus>
                <tram>true</tram>
            </license>
        </driver>
```

```
        <driver id="1111-03-25">
            <birthDate>1987-07-03</birthDate>
            <lastName>Sebastian</lastName>
            <firstName>Vettel</firstName>
            <license>
                <car>false</car>
                <bus>true</bus>
                <tram>true</tram>
            </license>
        </driver>
        <driver id="1111-03-26">
            <birthDate>1981-07-29</birthDate>
            <lastName>Fernando</lastName>
            <firstName>Alonso</firstName>
            <license>
                <car>true</car>
                <bus>true</bus>
                <tram>false</tram>
            </license>
        </driver>
    </drivers>
    <routes>
        <route id="1A" scheduleId="1">
            <distance>12</distance>
            <pricePerKm>30.2</pricePerKm>
        </route>
        <route id="20B" scheduleId="3">
            <distance>14</distance>
            <pricePerKm>30.75</pricePerKm>
        </route>
        <route id="43" scheduleId="5">
            <distance>20</distance>
            <pricePerKm>40.25</pricePerKm>
        </route>
        <route id="1" scheduleId="7">
            <distance>17</distance>
            <pricePerKm>25.0</pricePerKm>
        </route>
        <route id="12G" scheduleId="8">
            <distance>21</distance>
            <pricePerKm>35.0</pricePerKm>
        </route>
    </routes>
    <vehicles>
        <vehicle id="ASD-123" driverId="1111-03-22" scheduleId="1" inspectorId="2">
            <seats>48</seats>
            <manufacturer>MAN</manufacturer>
            <type>bus</type>
        </vehicle>
```

```xml
        <vehicle id="QWE-968" driverId="1111-03-23" scheduleId="3" inspectorId="4">
            <seats>34</seats>
            <manufacturer>Ikarus</manufacturer>
            <type>bus</type>
        </vehicle>
        <vehicle id="ADC-666" driverId="1111-03-24" scheduleId="5" inspectorId="6">
            <seats>70</seats>
            <manufacturer>Renault</manufacturer>
            <type>bus</type>
        </vehicle>
        <vehicle id="BOT-404" driverId="1111-03-25" scheduleId="7">
            <seats>230</seats>
            <manufacturer>Skoda</manufacturer>
            <type>tram</type>
        </vehicle>
        <vehicle id="TOP-282" driverId="1111-03-26" scheduleId="8">
            <seats>200</seats>
            <manufacturer>Skoda</manufacturer>
            <type>tram</type>
        </vehicle>
    </vehicles>
    <route-stop_join>
        <join routeId="1A" stopId="Bacsó Béla Utca" />
        <join routeId="1A" stopId="Római part" />
        <join routeId="20B" stopId="Soltész Nagy Kálmán" />
        <join routeId="20B" stopId="Vízügyi igazgatóság" />
        <join routeId="43" stopId="Soltész Nagy Kálmán" />
        <join routeId="43" stopId="Miskolctapolca-Barlangfürdő" />
        <join routeId="43" stopId="Római part" />
        <join routeId="1" stopId="Vízügyi igazgatóság" />
        <join routeId="1" stopId="Miskolctapolca-Barlangfürdő" />
        <join routeId="12G" stopId="Bacsó Béla Utca" />
        <join routeId="12G" stopId="Vízügyi igazgatóság" />
        <join routeId="12G" stopId="Római part" />
    </route-stop_join>
</company>
```

1.4: Az XSD dokumentum

```xml
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
    <!-- gyökér csomópont és az abból származó komplex típusú elemek -->
    <xs:element name="company">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="inspectors">
                    <xs:complexType>
                        <xs:sequence>
                            <xs:element name="inspector" type="inspector" minOccurs="0" maxOccurs="unbounded" />
                        </xs:sequence>
                    </xs:complexType>
                </xs:element>
                <xs:element name="schedules">
                    <xs:complexType>
                        <xs:sequence>
                            <xs:element name="schedule" type="schedule" minOccurs="0" maxOccurs="unbounded" />
                        </xs:sequence>
                    </xs:complexType>
                </xs:element>
                <xs:element name="stops">
                    <xs:complexType>
                        <xs:sequence>
                            <xs:element name="stop" type="stop" minOccurs="0" maxOccurs="unbounded" />
                        </xs:sequence>
                    </xs:complexType>
                </xs:element>
                <xs:element name="drivers">
                    <xs:complexType>
                        <xs:sequence>
                            <xs:element name="driver" type="driver" minOccurs="0" maxOccurs="unbounded" />
                        </xs:sequence>
                    </xs:complexType>
                </xs:element>
                <xs:element name="routes">
                    <xs:complexType>
                        <xs:sequence>
                            <xs:element name="route" type="route" minOccurs="0" maxOccurs="unbounded" />
                        </xs:sequence>
                    </xs:complexType>
                </xs:element>
                <xs:element name="vehicles">
                    <xs:complexType>
                        <xs:sequence>
                            <xs:element name="vehicle" type="vehicle" minOccurs="0" maxOccurs="unbounded" />
                        </xs:sequence>
                    </xs:complexType>
                </xs:element>
                <xs:element name="route-stop_join">
```

```
                    <xs:complexType>
                        <xs:sequence>
                            <xs:element name="join" type="route-stop" minOccurs="0" maxOccurs="unbounded" />
                        </xs:sequence>
                    </xs:complexType>
                </xs:element>
            </xs:sequence>
        </xs:complexType>

        <!-- key - keyref azonossági megszorítások -->
        <xs:unique name="inspectorId">
            <xs:selector xpath=".//vehicle" />
            <xs:field xpath="@inspectorId" />
        </xs:unique>
        <xs:keyref name="inspectorIdref" refer="inspectorId">
            <xs:selector xpath=".//inspector" />
            <xs:field xpath="@id" />
        </xs:keyref>

        <xs:key name="routeId">
            <xs:selector xpath=".//route" />
            <xs:field xpath="@id" />
        </xs:key>
        <xs:keyref name="routeIdref" refer="routeId">
            <xs:selector xpath=".//join" />
            <xs:field xpath="@routeId" />
        </xs:keyref>

        <xs:key name="scheduleId">
            <xs:selector xpath=".//schedule" />
            <xs:field xpath="@id" />
        </xs:key>
        <xs:keyref name="scheduleIdref" refer="scheduleId">
            <xs:selector xpath=".//route | .//vehicle" />
            <xs:field xpath="@scheduleId" />
        </xs:keyref>

        <xs:key name="stopId">
            <xs:selector xpath=".//stop" />
            <xs:field xpath="@id" />
        </xs:key>
        <xs:keyref name="stopIdref" refer="stopId">
            <xs:selector xpath=".//join" />
            <xs:field xpath="@stopId" />
        </xs:keyref>

        <xs:key name="driverId">
            <xs:selector xpath=".//driver" />
```

```
            <xs:field xpath="@id" />
        </xs:key>
        <xs:keyref name="driverIdref" refer="driverId">
            <xs:selector xpath=".//vehicle" />
            <xs:field xpath="@driverId" />
        </xs:keyref>
    </xs:element>

    <!-- egyedileg definiált komplex típusok -->
    <xs:complexType name="inspector">
        <xs:sequence>
            <xs:element name="lastName" type="xs:string" />
            <xs:element name="firstName" type="xs:string" />
        </xs:sequence>
        <xs:attribute name="id" type="xs:int" use="required" />
    </xs:complexType>

    <xs:complexType name="schedule">
        <xs:sequence>
            <xs:element name="arrivalTime" type="xs:time" />
            <xs:element name="departureTime" type="xs:time" />
        </xs:sequence>
        <xs:attribute name="id" type="xs:int" use="required" />
        <xs:attribute name="vehicleId" type="licensePlateNumber" use="required" />
    </xs:complexType>

    <xs:complexType name="stop">
        <xs:sequence>
            <xs:element name="address" type="xs:string" />
        </xs:sequence>
        <xs:attribute name="id" type="xs:string" use="required" />
    </xs:complexType>

    <xs:complexType name="driver">
        <xs:sequence>
            <xs:element name="birthDate" type="xs:date" />
            <xs:element name="lastName" type="xs:string" />
            <xs:element name="firstName" type="xs:string" />
            <xs:element name="license" type="license" />
        </xs:sequence>
        <xs:attribute name="id" type="xs:date" use="required" />
    </xs:complexType>

    <xs:complexType name="license">
        <xs:sequence>
            <xs:element name="car" type="xs:boolean" />
            <xs:element name="bus" type="xs:boolean" />
            <xs:element name="tram" type="xs:boolean" />
        </xs:sequence>
```

```
147     </xs:complexType>
148
149     <xs:complexType name="route">
150         <xs:sequence>
151             <xs:element name="distance" type="xs:int" />
152             <xs:element name="pricePerKm" type="xs:decimal" />
153         </xs:sequence>
154         <xs:attribute name="id" type="xs:string" use="required" />
155         <xs:attribute name="scheduleId" type="xs:int" use="required" />
156     </xs:complexType>
157
158     <xs:complexType name="vehicle">
159         <xs:sequence>
160             <xs:element name="seats" type="xs:int" />
161             <xs:element name="manufacturer" type="xs:string" />
162             <xs:element name="type">
163                 <xs:simpleType>
164                     <xs:restriction base="xs:string">
165                         <xs:enumeration value="bus" />
166                         <xs:enumeration value="tram" />
167                     </xs:restriction>
168                 </xs:simpleType>
169             </xs:element>
170         </xs:sequence>
171         <xs:attribute name="id" type="licensePlateNumber" use="required" />
172         <xs:attribute name="driverId" type="xs:date" use="required" />
173         <xs:attribute name="scheduleId" type="xs:int" use="required" />
174         <xs:attribute name="inspectorId" type="xs:int" use="optional" />
175     </xs:complexType>
176
177     <xs:complexType name="route-stop">
178         <xs:attribute name="routeId" type="xs:string" use="required" />
179         <xs:attribute name="stopId" type="xs:string" use="required" />
180     </xs:complexType>
181
182     <!-- egyedileg definiált egyszerű típusok -->
183     <xs:simpleType name="licensePlateNumber">
184         <xs:restriction base="xs:string">
185             <xs:pattern value="[A-Z]{3}-[0-9]{3}" />
186         </xs:restriction>
187     </xs:simpleType>
188 </xs:schema>
189
```

2.Feladat

2.1: Az Java beolvasó és kiíró program.

```java
package XMLFeladatNDRQ1Y;

import java.io.*;

public class DOMReadNDRQ1Y {
    public static void main(String[] args) throws ParserConfigurationException, SAXException, IOException {
        // XML fájl beolvasása
        File xml = new File("XMLNDRQ1Y.xml");

        // XML fájl DOM document való formába való alakítása
        DocumentBuilderFactory factory = DocumentBuilderFactory.newInstance();
        DocumentBuilder builder = factory.newDocumentBuilder();
        Document document = builder.parse(xml);

        // DOM document átalakítása DOM DocumentTraversal formába
        DocumentTraversal traversal = (DocumentTraversal) document;

        // DOM TreeWalker inicializálása
        // a gyökérelemtől kezdve bejárhatjuk az összes elemet és szöveget tartalmazó
        // csomópontot
        TreeWalker walker = traversal.createTreeWalker(document.getDocumentElement(),
                NodeFilter.SHOW_ELEMENT | NodeFilter.SHOW_TEXT, null, true);

        // a DOM bejárása és kiíratása rekurzívan
        DomTraverser.traverseLevel(walker, "");
    }

    private static class DomTraverser {
        public static void traverseLevel(TreeWalker walker, String indent) {
            // kimentjük az aktuális csomópontot
            Node node = walker.getCurrentNode();

            // kiíratjuk a megfelelő metódussal
            if (node.getNodeType() == Node.ELEMENT_NODE) {
                printElementNode(node, indent);
            } else {
                printTextNode(node, indent);
            }

            // rekurzívan meghívjuk a bejárást a DOM fa eggyel mélyebben lévő csomópontjára,
            // majd azok testvér csomópontjaira
            for (Node n = walker.firstChild(); n != null; n = walker.nextSibling()) {
                traverseLevel(walker, indent + "    ");
```

```
47                  traverseLevel(walker, indent + "    ");
48              }
49
50              walker.setCurrentNode(node);
51          }
52
53          private static void printElementNode(Node node, String indent) {
54              System.out.print(indent + node.getNodeName());
55
56              printElementAttributes(node.getAttributes());
57          }
58
59          private static void printElementAttributes(NamedNodeMap attributes) {
60              int length = attributes.getLength();
61
62              if (length > 0) {
63                  System.out.print(" (");
64
65                  for (int i = 0; i < length; i++) {
66                      Node attribute = attributes.item(i);
67
68                      System.out.printf("%s=%s%s", attribute.getNodeName(), attribute.getNodeValue(),
69                              i != length - 1 ? ", " : "");
70                  }
71
72                  System.out.println(")");
73              } else {
74                  System.out.println();
75              }
76          }
77
78          private static void printTextNode(Node node, String indent) {
79              String content_trimmed = node.getTextContent().trim();
80
81              if (content_trimmed.length() > 0) {
82                  System.out.print(indent);
83                  System.out.printf("%s%n", content_trimmed);
84              }
85          }
86      }
87  }
```

Kimenet:

```
company (xmlns:xsi=http://www.w3.org/2001/XMLSchema-instance, xsi:noNamespaceSchemaLocation=XMLSchemaNDRQ1Y.xsd)
    inspectors
        inspector (id=2)
            lastName
                Jean
            firstName
                Todt
        inspector (id=4)
            lastName
                Laboda
            firstName
                Dániel
        inspector (id=6)
            lastName
                Elek
            firstName
                Zoltán
    schedules
        schedule (id=1, vehicleId=ASD-123)
            arrivalTime
                05:50:00
            departureTime
                06:15:00
        schedule (id=3, vehicleId=QWE-968)
            arrivalTime
                08:30:00
            departureTime
                08:35:00
        schedule (id=5, vehicleId=ADC-666)
            arrivalTime
                10:17:00
            departureTime
                10:50:00
        schedule (id=7, vehicleId=BOT-404)
            arrivalTime
                12:40:00
            departureTime
                13:00:00
        schedule (id=8, vehicleId=TOP-282)
            arrivalTime
                20:22:00
            departureTime
```

```
            departureTime
                20:50:00
    stops
        stop (id=Bacsó Béla Utca)
            address
                Bacsó Béla út 40.
        stop (id=Soltész Nagy Kálmán)
            address
                Soltész Nagy Kálmán út 8.
        stop (id=Vízügyi igazgatóság)
            address
                Vörösmarty Mihály utca. 77
        stop (id=Miskolctapolca-Barlangfürdő)
            address
                Pazár István Sétány 1.
        stop (id=Római part)
            address
                Római Part 30
    drivers
        driver (id=1111-03-22)
            birthDate
                1969-01-03
            lastName
                Michael
            firstName
                Schumacher
            license
                car
                    true
                bus
                    true
                tram
                    true
        driver (id=1111-03-23)
            birthDate
                1960-03-21
            lastName
                Ayrton
            firstName
                Senna
            license
                car
                    true
```

```
                true
            bus
                true
            tram
                false
    driver (id=1111-03-24)
        birthDate
            1955-02-24
        lastName
            Alain
        firstName
            Prost
        license
            car
                false
            bus
                false
            tram
                true
    driver (id=1111-03-25)
        birthDate
            1987-07-03
        lastName
            Sebastian
        firstName
            Vettel
        license
            car
                false
            bus
                true
            tram
                true
    driver (id=1111-03-26)
        birthDate
            1981-07-29
        lastName
            Fernando
        firstName
            Alonso
        license
            car
                true
```

```
                true
            bus
                true
            tram
                false
routes
    route (id=1A, scheduleId=1)
        distance
            12
        pricePerKm
            30.2
    route (id=20B, scheduleId=3)
        distance
            14
        pricePerKm
            30.75
    route (id=43, scheduleId=5)
        distance
            20
        pricePerKm
            40.25
    route (id=1, scheduleId=7)
        distance
            17
        pricePerKm
            25.0
    route (id=12G, scheduleId=8)
        distance
            21
        pricePerKm
            35.0
vehicles
    vehicle (driverId=1111-03-22, id=ASD-123, inspectorId=2, scheduleId=1)
        seats
            48
        manufacturer
            MAN
        type
            bus
    vehicle (driverId=1111-03-23, id=QWE-968, inspectorId=4, scheduleId=3)
        seats
            34
                true
```

```
            seats
                34
            manufacturer
                Ikarus
            type
                bus
        vehicle (driverId=1111-03-24, id=ADC-666, inspectorId=6, scheduleId=5)
            seats
                70
            manufacturer
                Renault
            type
                bus
        vehicle (driverId=1111-03-25, id=BOT-404, scheduleId=7)
            seats
                230
            manufacturer
                Skoda
            type
                tram
        vehicle (driverId=1111-03-26, id=TOP-282, scheduleId=8)
            seats
                200
            manufacturer
                Skoda
            type
                tram
    route-stop_join
        join (routeId=1A, stopId=Bacsó Béla Utca)
        join (routeId=1A, stopId=Római part)
        join (routeId=20B, stopId=Soltész Nagy Kálmán)
        join (routeId=20B, stopId=Vízügyi igazgatóság)
        join (routeId=43, stopId=Soltész Nagy Kálmán)
        join (routeId=43, stopId=Miskolctapolca-Barlangfürdő)
        join (routeId=43, stopId=Római part)
        join (routeId=1, stopId=Vízügyi igazgatóság)
        join (routeId=1, stopId=Miskolctapolca-Barlangfürdő)
        join (routeId=12G, stopId=Bacsó Béla Utca)
        join (routeId=12G, stopId=Vízügyi igazgatóság)
        join (routeId=12G, stopId=Római part)
```

2.2: Az Java módosító program.

```java
package XMLFeladatNDRQ1Y;

import java.io.*;

public class DOMModifyNDRQ1Y {
    public static void main(String[] args) throws ParserConfigurationException, SAXException, IOException,
            XPathExpressionException, DOMException, ParseException {
        // XML fájl beolvasása
        File xml = new File("XMLNDRQ1Y.xml");

        // XML fájl DOM document való formába való alakítása
        DocumentBuilderFactory factory = DocumentBuilderFactory.newInstance();
        DocumentBuilder builder = factory.newDocumentBuilder();
        Document document = builder.parse(xml);

        // a DOM document lévő adatok módosítása
        DomModifier.modifyDom(document);

        // DOM document átalakítása DOM DocumentTraversal formába
        DocumentTraversal traversal = (DocumentTraversal) document;

        // DOM TreeWalker inicializálása
        // a gyökérelemtől kezdve bejárhatjuk az összes elemet és szöveget tartalmazó
        // csomópontot
        TreeWalker walker = traversal.createTreeWalker(document.getDocumentElement(),
                NodeFilter.SHOW_ELEMENT | NodeFilter.SHOW_TEXT, null, true);

        // a DOM bejárása rekurzívan
        DomTraverser.traverseLevel(walker, "");
    }

    private static class DomModifier {
        public static void modifyDom(Document document) throws XPathExpressionException, DOMException, ParseException {
            XPathFactory factory = XPathFactory.newInstance();
            XPath xpath = factory.newXPath();

            // 1.)  Michael Schumacher  megszervezte a villamoshoz a jogosítványt,
            // és ezt kellene hozzáadnunk az adatbázisunkhoz
            // XPath segítségével lekérdezzük a megfelelő elemet/csomópontot a DOM documentből
            Node driver = (Node) xpath.evaluate("//driver[./lastName='Michael' and ./firstName='Schumacher']/license/bus",
                    document, XPathConstants.NODE);

            driver.setTextContent("true");

            // 2.) Egy városi rendelet szerint a 15 km-nél rövidebb útvonalaknak 20%-kal nő
            // a kilométerára
            NodeList routes = (NodeList) xpath.evaluate("//route[./distance<15]/pricePerKm", document,
                    XPathConstants.NODESET);

            for (int i = 0; i < routes.getLength(); i++) {
                Node route = routes.item(i);

                double pricePerKm = Double.parseDouble(route.getTextContent());
                route.setTextContent(Double.toString(pricePerKm * 1.20));
            }

            // 3.) A 08:35-kor beérkező járatnak 15 perccel nőtt a menetideje, ezért később fog beérkezni
            Node schedule = (Node) xpath.evaluate("//schedule[./departureTime='08:35:00']/departureTime", document,
                    XPathConstants.NODE);

            DateTimeFormatter formatter = DateTimeFormatter.ofPattern("HH:mm:ss");
            LocalTime departureTime = LocalTime.parse(schedule.getTextContent(), formatter);
            departureTime = departureTime.plusMinutes(15);
            schedule.setTextContent(formatter.format(departureTime));

            // 4.) A 43-es járat át lett nevezve, mivel ez egy ID ezért az ehhez kapcsolódó
            // elemekben is módosítanunk kell a kapcsolatot
            NodeList nodes = (NodeList) xpath.evaluate("//route[@id='43']/@id | //*[@routeId='43']/@routeId", document,
                    XPathConstants.NODESET);
            String newId = "666";

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

                node.setTextContent(newId);
            }
        }
    }

    private static class DomTraverser {
        public static void traverseLevel(TreeWalker walker, String indent) {
            // kimentjük az aktuális csomópontot
            Node node = walker.getCurrentNode();

            // kiíratjuk a megfelelő metódussal
            if (node.getNodeType() == Node.ELEMENT_NODE) {
```

```
96            if (node.getNodeType() == Node.ELEMENT_NODE) {
97                printElementNode(node, indent);
98            } else {
99                printTextNode(node, indent);
100           }
101
102           // rekurzívan meghívjuk a bejárást a DOM fa eggyel mélyebben lévő csomópontjára,
103           // majd azok testvér csomópontjaira
104           for (Node n = walker.firstChild(); n != null; n = walker.nextSibling()) {
105               traverseLevel(walker, indent + "    ");
106           }
107
108           walker.setCurrentNode(node);
109       }
110
111       private static void printElementNode(Node node, String indent) {
112           System.out.print(indent + node.getNodeName());
113
114           printElementAttributes(node.getAttributes());
115       }
116
117       private static void printElementAttributes(NamedNodeMap attributes) {
118           int length = attributes.getLength();
119
120           if (length > 0) {
121               System.out.print(" (");
122
123               for (int i = 0; i < length; i++) {
124                   Node attribute = attributes.item(i);
125
126                   System.out.printf("%s=%s%s", attribute.getNodeName(), attribute.getNodeValue(),
127                           i != length - 1 ? ", " : "");
128               }
129
130               System.out.println(")");
131           } else {
132               System.out.println();
133           }
134       }
135
136       private static void printTextNode(Node node, String indent) {
137           String content_trimmed = node.getTextContent().trim();
```

```
136       private static void printTextNode(Node node, String indent) {
137           String content_trimmed = node.getTextContent().trim();
138
139           if (content_trimmed.length() > 0) {
140               System.out.print(indent);
141               System.out.printf("%s%n", content_trimmed);
142           }
143       }
144   }
145 }
```

Kimenet:

```
schedules
    schedule (id=1, vehicleId=ASD-123)
        arrivalTime
            05:50:00
        departureTime
            06:15:00
    schedule (id=3, vehicleId=QWE-968)
        arrivalTime
            08:30:00
        departureTime
            08:50:00
    schedule (id=5, vehicleId=ADC-666)
        arrivalTime
            10:17:00
        departureTime
            10:50:00
    schedule (id=7, vehicleId=BOT-404)
        arrivalTime
            12:40:00
        departureTime
            13:00:00
    schedule (id=8, vehicleId=TOP-282)
        arrivalTime
            20:22:00
        departureTime
            20:50:00
```

```
    route (id=1A, scheduleId=1)
        distance
            12
        pricePerKm
            36.2399999999999995
    route (id=20B, scheduleId=3)
        distance
            14
        pricePerKm
            36.9
    route (id=666, scheduleId=5)
        distance
            20
        pricePerKm
            40.25
    route (id=1, scheduleId=7)
        distance
            17
        pricePerKm
            25.0
    route (id=12G, scheduleId=8)
        distance
            21
        pricePerKm
            35.0
```

```
    driver (id=1111-03-22)
        birthDate
            1969-01-03
        lastName
            Michael
        firstName
            Schumacher
        license
            car
                true
            bus
                true
            tram
                true
```

2.3: Az Java lekérdező program.

```java
package XMLFeladatNDRQ1Y;

import java.io.File;

public class DOMQueryNDRQ1Y {
    public static void main(String[] args) throws ParserConfigurationException, SAXException, IOException,
    XPathExpressionException, DOMException, ParseException {
// XML fájl beolvasása
File xml = new File("XMLNDRQ1Y.xml");

// XML fájl DOM document való formába való alakítása
DocumentBuilderFactory factory = DocumentBuilderFactory.newInstance();
DocumentBuilder builder = factory.newDocumentBuilder();
Document document = builder.parse(xml);


// DOM document átalakítása DOM DocumentTraversal formába
DocumentTraversal traversal = (DocumentTraversal) document;


NodeList vehicles = document.getElementsByTagName("vehicle");

//Lekérdezzük azokat a járműveket amelyeknek a gyártója Skoda.
for (int i = 0; i < vehicles.getLength(); i++) {
    Node temp = vehicles.item(i);
    NodeList tempchildren = temp.getChildNodes();
    for (int j = 0; j < tempchildren.getLength(); j++) {
        Node temp2 = tempchildren.item(j);
            if (temp2.getNodeName().equals("manufacturer") && temp2.getTextContent().equals("Skoda")) {
                TreeWalker walker = traversal.createTreeWalker(temp,
                        NodeFilter.SHOW_ELEMENT | NodeFilter.SHOW_TEXT, null, true);
                DomTraverser.traverseLevel(walker, "");
            }
    }
}

//Lekérdezzük azokat a sofőröket akiknek van jogosítványuk villamosra.
NodeList drivers = document.getElementsByTagName("driver");

for (int i = 0; i < drivers.getLength(); i++) {
    Node temp = drivers.item(i);
    NodeList tempchildren = temp.getChildNodes();
    for (int j = 0; j < tempchildren.getLength(); j++) {
```

```
        Node temp2 = tempchildren.item(j);
            if (temp2.getNodeName().equals("license")) {
                NodeList temp2children = temp2.getChildNodes();
                for (int k = 0; k < temp2children.getLength(); k++) {
                    Node temp3 = temp2children.item(k);

                    if (temp3.getNodeName().equals("tram") && temp3.getTextContent().equals("true")) {

                        TreeWalker walker = traversal.createTreeWalker(temp,
                                NodeFilter.SHOW_ELEMENT | NodeFilter.SHOW_TEXT, null, true);
                        DomTraverser.traverseLevel(walker, "");
                    }
                }
            }


    }
}




}
    private static class DomTraverser {
        public static void traverseLevel(TreeWalker walker, String indent) {
            // kimentjük az aktuális csomópontot
            Node node = walker.getCurrentNode();

            // kiíratjuk a megfelelő metódussal
            if (node.getNodeType() == Node.ELEMENT_NODE) {
                printElementNode(node, indent);
            } else {
                printTextNode(node, indent);
            }

            // rekurzívan meghívjuk a bejárást a DOM fa eggyel mélyebben lévő csomópontjára,
            // majd azok testvér csomópontjaira
            for (Node n = walker.firstChild(); n != null; n = walker.nextSibling()) {
                traverseLevel(walker, indent + "    ");
            }

            walker.setCurrentNode(node);
        }
```

```
104
105         private static void printElementNode(Node node, String indent) {
106             System.out.print(indent + node.getNodeName());
107
108             printElementAttributes(node.getAttributes());
109         }
110
111         private static void printElementAttributes(NamedNodeMap attributes) {
112             int length = attributes.getLength();
113
114             if (length > 0) {
115                 System.out.print(" (");
116
117                 for (int i = 0; i < length; i++) {
118                     Node attribute = attributes.item(i);
119
120                     System.out.printf("%s=%s%s", attribute.getNodeName(), attribute.getNodeValue(),
121                             i != length - 1 ? ", " : "");
122                 }
123
124                 System.out.println(")");
125             } else {
126                 System.out.println();
127             }
128         }
129
130         private static void printTextNode(Node node, String indent) {
131             String content_trimmed = node.getTextContent().trim();
132
133             if (content_trimmed.length() > 0) {
134                 System.out.print(indent);
135                 System.out.printf("%s%n", content_trimmed);
136             }
137         }
138     }
139 }
```

Kimenet:

```
vehicle (driverId=1111-03-25, id=BOT-404, scheduleId=7)
    seats
        230
    manufacturer
        Skoda
    type
        tram
vehicle (driverId=1111-03-26, id=TOP-282, scheduleId=8)
    seats
        200
    manufacturer
        Skoda
    type
        tram
driver (id=1111-03-22)
    birthDate
        1969-01-03
    lastName
        Michael
    firstName
        Schumacher
    license
        car
            true
        bus
            true
        tram
            true
```

```
driver (id=1111-03-24)
    birthDate
        1955-02-24
    lastName
        Alain
    firstName
        Prost
    license
        car
            false
        bus
            false
        tram
            true
driver (id=1111-03-25)
    birthDate
        1987-07-03
    lastName
        Sebastian
    firstName
        Vettel
    license
        car
            false
        bus
            true
        tram
            true
```