Sveučilište Jurja Dobrile u Puli

Fakultet ekonomije i turizma

„Dr. Mijo Mirković“

Kristijan Prižmić

XML TEHNOLOGIJA BAZA PODATAKA

Završni rad

Pula, 2015.

Sveučilište Jurja Dobrile u Puli
Fakultet ekonomije i turizma
„Dr. Mijo Mirković“

Kristijan Prižmić

XML TEHNOLOGIJA BAZA PODATAKA

Završni rad

JMBAG: 0303033687, redoviti student
Studijski smjer: Informatika

Predmet: Baze podatka 2
Mentor: Dr. sc. Vanja Bevanda

Pula, svibanj 2015.

IZJAVA O AKADEMSKOJ ČESTITOSTI

Ja, dolje potpisani _____, kandidat za prvostupnika _____ovime izjavljujem da je ovaj Završni rad rezultat isključivo mojega vlastitog rada, da se temelji na mojim istraživanjima te da se oslanja na objavljenu literaturu kao što to pokazuju korištene bilješke i bibliografija. Izjavljujem da niti jedan dio Završnog rada nije napisan na nedozvoljen način, odnosno da je prepisan iz kojega necitiranog rada, te da ikoji dio rada krši bilo čija autorska prava. Izjavljujem, također, da nijedan dio rada nije iskorišten za koji drugi rad pri bilo kojoj drugoj visokoškolskoj, znanstvenoj ili radnoj ustanovi.

Student:

U Puli, __. __. 2015. _____

Sadržaj

SAŽETAK

XML (Extensible Markup Language) je markup jezik koji definira skup pravila za kodiranje dokumenata u formatu koji je i ljudski čitljiv i strojno čitljiv.

U ovom radu će se definirati pojmovi XML-a i njegove sintakse, prikazat će se osnove DTD-a, CDATA sekcija i XML Sheme. Ovaj rad će također dati kratki pregled na moguće načine programske obrade XML dokumenata i mogućih primjena XML tehnologije. Također ćemo malo pojasniti osnove baza podataka i tipove XML baza podataka.

ABSTRACT

XML (Extensible Markup Language) is a markup language that defines a set of rules for encoding documents in a format that is both human readable and machine readable.

This paper will define the concepts od XML and its syntax, also the basics od DTD, CDATA sections and XML scheme will be shown. This paper will also give a brief overview of the possible ways of program processing XML documents and potential aplications od XML tehnology. We will also clarify the basics od databases and types od XML databases.

UVOD

XML je zamišljen kao jezik za opisivanje dokumenata i podataka. Pod izrazom „dokumenti i podaci“ podrazumijevamo tekstualne dokumente ili skupove podataka kakvi se obično pohranjuju u baze podataka. XML i HTML sintaksno su slični, iako su razvijeni sa različitim namjenama. XML je prvenstveno razvijen za opisivanje podataka. Oznake kod XML-a su slobodne i korisnici ih moraju sami smisliti i kreirati. U HTML-a postoji predefinirani skup oznaka koje uglavnom služe za prikazivanje sadržaja u Internet pregledniku na odgovarajući način. Sintaksna pravila XML-a su stroga i ako dokument nije formatiran u skladu s njima, računalni program neće moći pročitati XML dokument.

U prvom poglavlju govorim o osnovnim konceptima odnosno sadržaju i oznakama te glavnim dijelovima odnosno elementima i atributima te ostalim dijelovima kao što su to komentari, procesorske naredbe, DTD, CDATA sekcija i XML Schema.
U drugom poglavlju krećem od povijesti baze podataka dok modela odnosno relacijske baze podataka uz koju prikazujem schemu relacijske baze i samu primjenu baze podataka.
U trećem poglavlju objašnjavam tipove XML baze podataka i to flat files i relacijsku bazu podataka. U četvrtom poglavlju pojašnjavam skladištenje XML dokumenata na *data centric* i *document centric*. U petom poglavlju govorim o eksternalizaciji XML iz relacijske baze podataka. U šestom poglavlju nadovezujem se na prednosti i nedostatke te zadnje, sedmo poglavlje navodim razlike između XML i relacijskog modela.

Tijekom pisanja ovog završnog rada koristile su se deduktivna metoda, metoda analize, metoda specijalizacije i deskripcije.

1. XML TEHNOLOGIJA

XML (eng. *eXtensible Markup Language*, slika 1) je jezik za označavanje podataka.

Slika 1. XML logo

Izvor: http://akraya.com/2015/08/tbt-where-does-xml-fit-in-with-big-data/

To je jezik koji je jednostavno čitljiv i ljudima i računalnim programima (odgovarajući sadržaj treba se uokviriti ogovarajućim oznakama koje ga opisuju i imaju poznato ali lako shvatljivo značenje). XML-om zapisujemo dokumente i podatke u tekstualnom formatu. XML format čitljiv je za tekst editore (slika 2) kao što su Microsoft Office Word, WordPerfect ili OpenOffice.org. Format XML-a je sličan formatu oznaka, primjerice HTML (eng. *HyperText Markup Language*) jeziku.

Slika 2. XML tekst editori

Izvor: http://akshayam.in/wp-content/uploads/2014/06/xml_conversion_services.jpg

Danas je XML jezik vrlo raširen i koristi se za različite namjene:

- ✓ Odvajanje podataka od prezentacije
- ✓ Razmjenu podataka
- ✓ Pohranu podataka
- ✓ Povećavanje dostupnosti podataka
- ✓ Izradu novih specijaliziranih jezika za označavanje

XML je standardizirani jezik i za njegovu standardizaciju brine se World Wide Web Consortium (W3C)[1].

Primjer tekstualne datoteke u XML formatu (primjer 1):

```
<?xml version="1.0"?>
<note date=10/06/2000>
<note>
        <to>Pera</to>
        <from>Mika</from>
        <subject>pozdrav</ subject>
        <body>Puno pozdrava iz Pule</body>
</note>
```

Primjer 1. Primjer XML

Izvor: Uvod u XML i XML tehnologije

Dokument se sastoji od dijela deklaracije (osnovne i složene) i tekstualnog dijela poruke (primjer 2):

```
<?xml version="1.0"?>
```

Primjer 2. XML deklaracija

Izvor: Uvod u XML i XML tehnologije

Prvi dio zaglavlje ili XML deklaracije (primjer 2) u kojem se opisuje verzija XML-a, prema čijim pravilima je dokument napravljen.

Drugi dio je složena XML deklaracija (primjer 3). Prikazuje kodnu stranicu, odnosno, ako se ne navede ispravna kodna stranica, ukoliko programi naiđu na nestandardni jezik (primjerice hrvatska slova č, ć, ž, š).

[1] W3C, organizacija koja se bavi standardizacijom tehnologija korištenih na webu, izvor: http://www.w3.org/

<?xml version="1.0"? encoding="UTF-8"?>

Primjer 3. XML deklaracija (složena)

Izvor: Uvod u XML i XML tehnologije

Opis primjera 3:

- Oznaka **?** je oznaka za instrukciju obrade, odnosno poruka programima koji procesiraju XML dokument
- Atribut **version** određuje XML verziju
- Atribut **encoding** definira znakovni kod u kojem je XML dokument pisan, primjerice:
  UTF-8 (kompresirana verzija Unicode-a)
  UTF-16 (Unicode)

Tekstualni dio XML dokumenta (primjer 4)

```
<note>
        <to>Pera</to>
        <from>Mika</from>
        <subject>pozdrav</ subject>
        <body>Puno pozdrava iz Pule</body>
</note>
```

Primjer 4. Tekstualni dio XML

Izvor: Uvod u XML i XML tehnologije

Svaki XML dokument **mora** imati **jedan** korijenski (root) element koji zaokružuje kompletan sadržaj. Prilikom odlaska na web stranicu ispisuje se kao jedna „ovo je **poruka**".

Sam po sebi dokument je samo objašnjavajući upravo zbog oznaka koje označavaju pojedini dijelovi teksta.

XML je jedan od jezika za označavanje. Najpoznatiji i osnovni jezik je HTML koji se koristi za izradu Web stranica. Ovdje još pripadaju i SGML[2] (eng. *Standard Generalized Markup Languag*e) i RTF (eng. *Rich Text Format*).

[2] SGML je općeniti jezik za opisivanje dokumenata i specifičan za nove tagove. Složeniji je od HTML i XML.

XML je podskup SGML-a koji je složen i opsežan meta-jezik. Meta-jezici su jezici za opisivanje dozvoljenog pravopisa i gramatike drugih jezika. Jedan od temeljnih koncepata SGML je da jezici za opisivanje dokumenata i podataka mogu imati oznake. Oznake su posebne riječi u tekstu kojima određujemo vrstu teksta. U primjeru 1. oznake su <note>, <to>, <from>, <subject>, <body> i </note> (pogledati primjer 1, stranica 5).

XML je zadržao značajku SGML-a da se njime mogu opisivati novi jezici, ali je izostavljeno sve ono što je kompliciranо i teško za implementirati.

Iako u svom nazivu ima riječ „jezik“, XML nije programski jezik kao što su to C++ i Java. Dokument napisan u XML ne radi ništa, on je tekst format i kao takav je pasivan.

1.1 Struktura: sadržaj i oznake

XML je zamišljen kao jezik za opisivanje dokumenata i podataka. Pod time podrazumijevamo tekstualne dokumente ili skupove podataka korištenih za bazu podataka. XML je otvoreni standard čije su karakteristike javne i svima dostupne te nisu potrebne nikakve licence.

Oznake (eng. *tag*) koje koristimo za opisivanje podataka nisu definirane unaprijed. XML standard opisuje samo minimalni skup pravila koji dokument mora zadovoljavati. Također, korisnici XML-a moraju sami definirati dozvoljene oznake za označavanje, odnosno možda postoji definirani skup oznaka pa se mogu koristiti i te oznake.

Ako razmjenjujemo XML dokumente sa drugima, obje strane moraju poštovati zajednički dogovor o oznakama. Potrebno je definirati i dozvoljene kombinacije oznaka. To se opisuje kao DTD (eng. *Document Type Definition*) ili XML Schema.

Ono što je zajedničko XML-u i HTML-u su označavanje dijelova dokumenata oznakama. Primjerice, u HTML-u oznaka <b> koristi se za označavanja podebljanog teksta (primjer 5).

```
HTML je jezik za opisivanje <b>prikaza</b>.
```

Primjer 5. Podebljani tekst

Izvor: Kirasić D., XML tehnologija i primjena u sustavima procesne informatike (str. 2)

U Internet pretraživaču gornji HTML tekst izgleda ovako (primjer 6):

HTML je jezik za opisivanje **prikaza**.

Primjer 6. Podebljani tekst

Izvor: Kirasić D., XML tehnologija i primjena u sustavima procesne informatike (str. 2)

Važno je primijetiti da se kod XML za razliku od HTML oznake definiraju po vlastitim potrebama. HTML-ova oznaka <b> **mora** unaprijed biti definirana.

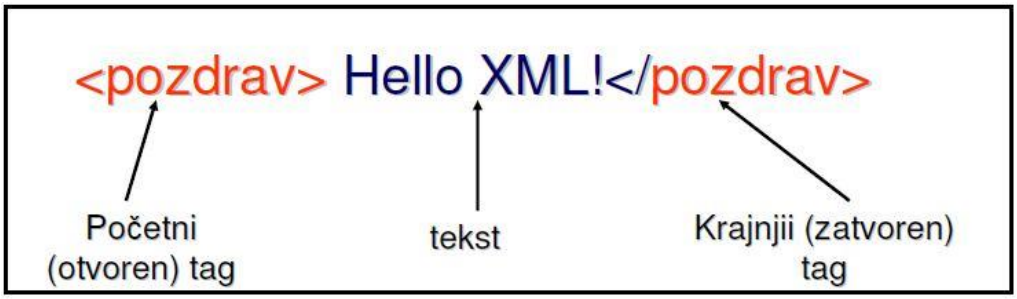
XML se usredotočuje na opis podataka i ono što oni znače, a ne na opis prikaza. To se vidi jasno u slici 1 gdje se koriste oznake za određivanje vrste podataka.

Spremanje se najčešće vrši u datoteku, ali nije nužno; mogu se dobiti preko mreže ili iz relacijske baze podataka.

1.2 Dijelovi: elementi i atributi

Na najnižoj razini XML dokument se sastoji od niza Unicode znakova. Unicode standard (http://www.unicode.org/) definira više načina kodiranja znakova i danas najvažniji znakovni standard. Za zapisivanje znakova koristi se uglavnom više okteta, za razliku od ASCII kodiranja u kojem se koristio samo jedan oktet. Unicode podržava sve grafeme najvažnijih svjetskih pisama (veliki broj kineskih grafema) pa i hrvatske grafeme č, ć, ž, š i đ.

Unicode znakovi tvore različite XML dokumente. Većinom su to elementi, podelementi i atributi. Elementi započinju *početnom oznakom* <poruka>, a završavaju *završnom oznakom* </poruka>. Između početne i završne oznake nalazi se sadržaj elementa koji može biti bilo što (slika 3). Element također može biti i prazan. Ukoliko je prazan, pišemo ga sa kosom crtom na kraju imena oznake, primjerice <porukaGotova/>.



Slika 3. XML element

Izvor: Uvod u XML i XML tehnologije

Elementi su osnovni blokovi. Složeni (kontejner) element je sa početnim i krajnjim tag sa sadržajem. Prazan element je onaj krajnji za koji se koristi skraćenica

/> „Hello XML“/>

primjerice

<poruka/>

<pozdrav tekst = “Hello XML” />

Slika 4 prikazuje uobičajeni složeni primjer XML elementa <Projects> koji počinje sa početnom i završava završnom oznakom, dok su unutar njega oznake omeđene zagradama <, > i </ ...>.

```
<?xml version= "1.0" standalone="yes"?>
    <Projects>
        <Project>
            <Name>ProductX</Name>
            <Number>1</Number>
            <Location>Bellaire</Location>
            <Dept_no>5</Dept_no>
            <Worker>
                <Ssn>123456789</Ssn>
                <Last_name>Smith</Last_name>
                <Hours>32.5</Hours>
            </Worker>
            <Worker>
                <Ssn>453453453</Ssn>
                <First_name>Joyce</First_name>
                <Hours>20.0</Hours>
            </Worker>
        </Project>
        <Project>
            <Name>ProductY</Name>
            <Number>2</Number>
            <Location>Sugarland</Location>
            <Dept_no>5</Dept_no>
            <Worker>
                <Ssn>123456789</Ssn>
                <Hours>7.5</Hours>
            </Worker>
            <Worker>
                <Ssn>453453453</Ssn>
                <Hours>20.0</Hours>
            </Worker>
            <Worker>
                <Ssn>333445555</Ssn>
                <Hours>10.0</Hours>
            </Worker>
        </Project>
    ...
    </Projects>
```

Slika 4. Elementi <Projects>

Izvor: Elmasri R., Navathe S. B., *Foundamentals of Database System*, poglavlje 12 (2011.)

Osim elemenata, imamo i atribute. Atributi se pridružuju elementima tako da pruže dodatne informacije o svojstvima elemenata (primjer 7).

<poruka **datum = "12.5.08." sala = "201"** >
<tekst>**Sastanak Katedre sutra u 10**</tekst>
</poruka>

Primjer 7. XML atribut

Izvor: Uvod u XML i XML tehnologije

Datum je naziv atributa, dok je vrijednost atributa označena navodnicima.
Možemo koristiti jednostruke ili dvostruke znake navoda

<ime=“Hrvoje“>

Ili

<ime='Hrvoje'>

Podaci se mogu spremati ili u elementima (primjer 9) ili u atributima (primjer 8). Element ima sljedeću formu.

<ime>Hrvoje</ime> dok je atribut u formi: <nesto ime="Hrvoje">

<partner **tip**="nabavljač">
<ime>Pera</ime>
<prezime>Perić</prezime>
</partner>

Primjer 8. Atribut "tip"

Izvor: Uvod u XML i XML tehnologije

<partner>nabavljač</partner>
<ime>Pera</ime>
<prezime>Perić</prezime>

Primjer 9. Element "tip"

Izvor: Uvod u XML i XML tehnologije

Uz osnovna pravila moraju biti zadovoljeni i neki minimalni uvjeti nazvani „dobro formiran“ (eng. *well formed*). Jedan od uvjeta je već spomenuti, da dokument ima samo jedan korijenski element i da postoji XML deklaracija. Drugi minimalni uvjet je da se podelementi moraju kompletno nalaziti unutar elementa roditelja (primjer 10):

 Ovo je tekst

Primjer 10. XML uvjet (ispravan)

Izvor: Uvod u XML i XML tehnologije

Nije dozvoljeno da podelement završi nakon završetka elementa-roditelja (primjer 11):

 Ovo je tekst

Primjer 11. CML uvjet (neispravan)

Izvor: Uvod u XML i XML tehnologije

Uz ove uvjete mora se paziti i na osjetljivost oznaka. Oznaka <Message> se razlikuje od oznake <message>. Ukoliko napišemo <Message> Error! </message>,

Uočimo razliku između elemenata i atributa. Na primjeru 13 vidimo datum bez navoda, a na primjeru 12 kao element. Oboje prikazuju istu informaciju.

```
<?xml version="1.0"?>
<note date=10/06/2000>
<note>
        <to>Pera</to>
         <from>Mika</from>
         <subject>pozdrav</ subject>
        <body>Puno pozdrava iz Pule</body>
</note>
```

Primjer 12. Elementi XML (neispravan)

Izvor: Uvod u XML i XML tehnologije

```
<?xml version="1.0"?>
<note date="10/06/2000">
<note>
        <to>Pera</to>
         <from>Mika</from>
         <subject>pozdrav</ subject>
        <body>Puno pozdrava iz Pule</body>
</note>
```

Primjer 13. Atributi XML (ispravno)

Izvor: Uvod u XML i XML tehnologije

1.3 Ostali dijelovi

Osim XML deklaracije, elemenata i atributa, imamo i komentare, procesorske naredbe, DTD, reference na entitete i CDATA sekcije.

1.3.1 Komentari

Njihova svrha je dodavanje napomena u XML dokument, koji se neće gledati kao podaci niti biti dio sadržaja.

<!-- Komentar -->

Primjer 14. XML komentar

Izvor: http://www.w3schools.com/xml/xml_syntax.asp

Komentar (primjer 14) započinje sa <!—a završava sa -->. Unutar komentara ne smijemo imati nove komentare. Također, mogu se proširiti u više redaka.

1.3.2 Procesorske naredbe

Upisuju se u XML kada želimo dodatne podatke za programe. Procesorska naredba izgleda ovako (primjer 15):

<? progABC inputParam="123" >

Primjer 15. Procesorska naredba (a)

Izvor: Kirasić D., XML tehnologija i primjena u sustavima procesne informatike (str. 3)

Sadržaj ovih naredbi je proizvoljan odnosno nije određen XML standardom. Mora započeti s *imenom ciljnog programa* kojem je namijenjen. U primjeru 1 to je „progABC". Drugi ciljani programi se neće obazirati na tu naredbu. *Sadržaj* naredbe mora obaviti cijeli program, inputParam="123".

U jednom XML dokumentu možemo raspolagati sa više procesnih naredbi za različite ciljane programe.

Najčešća naredba koja se koristi u praksi je „stylesheet" (primjer 16) za prikaz XML dokumenta.

```
<?xml-stylesheet type=“text/xml“
href=“beauty.xsl“ ?>
```

Primjer 16. Procesorska naredba (b)

Izvor: Kirasić D., XML tehnologija i primjena u sustavima procesne informatike (str. 3)

Ciljani program „xml-stylesheet“ iz ove naredbe poziva XSLT stroj obrađuje naredbe iz „beauty.xsl“ i primjenjuje ih na XML dokument.

1.3.3 DTD

Punim nazivom Document Type Description. Odnosi se na dio XML dokumenta koji detaljnije opisuje i ograničava dozvoljeni sadržaj XML dokumenta, ali ne mora biti prisutan u njemu.

```
<?xml version="1.0"?>
<!DOCTYPE note [
        <!ELEMENT note (to,from,subject,body)>
        <!ELEMENT to (#PCDATA)>
        <!ELEMENT from (#PCDATA)>
        <!ELEMENT subject (#PCDATA)>
        <!ELEMENT body (#PCDATA)>
]>
```

Primjer 17. Interna DTD definicija za XML

Izvor: Uvod u XML i XML tehnologije

```
<note>
        <to>Pera</to>
        <from>Mika</from>
        <subject>pozdrav</subject>
        <body>Puno pozdrava iz Beograda</body>
</note>
```

Primjer 18. Ostatak XML-a

Izvor: Uvod u XML i XML tehnologije

Na primjeru 17 prikazani su DTD interni, a na primjeru 18 ostatak XML-a. DTD određuje da XML mora započeti s korijenskim elementom <note> ispod kojeg može biti nijedan ili više elemenata <to>. Element <to> mora imati podelemente <to>, <from>, <subject> i <body>.

Ovakvi podelementi zapisuju se kao (#PCDATA). Entitete najbolje razumijemo kao par ime-vrijednost (slika 5).

```
<!DOCTYPE Projects [
        <!ELEMENT Projects (Project+)>
        <!ELEMENT Project (Name, Number, Location, Dept_no?, Workers)
            <!ATTLIST Project
                ProjId ID #REQUIRED>
            >
        <!ELEMENT Name (#PCDATA)>
        <!ELEMENT Number (#PCDATA)
        <!ELEMENT Location (#PCDATA)>
        <!ELEMENT Dept_no (#PCDATA)>
        <!ELEMENT Workers (Worker*)>
        <!ELEMENT Worker (Ssn, Last_name?, First_name?, Hours)>
        <!ELEMENT Ssn (#PCDATA)>
        <!ELEMENT Last_name (#PCDATA)>
        <!ELEMENT First_name (#PCDATA)>
        <!ELEMENT Hours (#PCDATA)>
    ]>
```

Slika 5. DTD "Projects"

Izvor: Elmasri R., Navathe S. B., *Foundamentals of Database System*, poglavlje 12 (2011.)

Objašnjenje:

„*“ (zvjezdica) označava naziv elementa koji se može i ne mora ponavljati u dokumentu (ponavljajući element)

„+“ (plus) označava naziv elementa koji se može ponavljati jednom ili više puta u dokumentu (ponavljajući element)

„?“ (upitnik) označava naziv elementa koji se može ali i ne mora ponavljati (neponavljajući element)

Element koji nema oznake mora se pojaviti točno jednom u dokumentu (neponavljajući element)

DTD ne mora biti u istoj datoteci sa XML dokumentom već se može definirati u posebnoj datoteci. Tako se više XML dokumenata može referencirati na jednu DTD datoteku.

Da bi slika 4 bila valjana, trebamo odrediti deklaraciju. To možemo napraviti promjenom prve linije slike 4 u sljedeću:

<?xml version=“1.0“ standalone=“no“?>

<!DOCTYPE Projects SYSTEM „proj.dtd“>

Kada je vrijednost standalone atributa „no“, dokument se treba provjeriti sa posebnim DTD dokumentom ili XML Schemom. DTD dokument na slici 9 biti će spremljen na isto mjesto (istu mapu) sistema kao XML dokument, ali će prikazivati naziv proj.dtd. Odnosno, možemo uključiti DTD tekstualni dokument na početak XML dokumenta radi samostalnog omogućavanja provjere.

1.3.4 CDATA sekcija

CDATA sekcija je dio XML dokumenta koja izgleda (slika 6):

```
<![CDATA[
Ovdje može biti proizvoljni sadržaj koji
uključuje i znakove <, >, &, ", '
Proizvoljni tekst unutar CDATA se neće
provjeravati niti će se na njemu obavljati
bilo kakve supstitucije teksta.
]]>
```

Slika 6. CDATA

Izvor: Kirasić D., XML tehnologija i primjena u sustavima procesne informatike (str. 3)

CDATA sekcije se koriste kada želimo imati dijelove teksta koji se neće uopće obrađivati. Tako primjerice može sadržavati naredbe:

```
if ( placa < 8000 )
        Document.writen("Hmmmmm?")
```

Primjer 19. CDATA

Izvor: Kirasić D., XML tehnologija i primjena u sustavima procesne informatike (str. 4)

1.3.5 XML Schema

XML Schema je tekstualni opis koji definira dozvoljenu strukturu XML dokumenta. Ima istu funkciju kao i DTD, ali je novija i opsežnija.

XML Schema (Bourret R., Coates A. B., Harvey B., i ostali, *Advanced XML Applications* (2007.):

- Postavlja uvjete i ograničenje na sadržaj i strukturi XML dokumenata
- Definira rječnik oznaka i gramatiku (dozvoljene sljedove XML dijelova) koji se mogu pojaviti u dokumentu
- Određuje pravila koja se moraju slijediti da bi XML dokument bio valjan.

Sljedeći primjer prikazuje XML Schemu koja odgovara bazi podataka Company (primjer 20).

```
<?xml version="1.0" encoding="UTF-8" ?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
    <xsd:annotation>
        <xsd:documentation xml:lang="en">Company Schema (Element Approach) - Prepared by Babak
            Hojabri</xsd:documentation>
    </xsd:annotation>
<xsd:element name="company">
    <xsd:complexType>
        <xsd:sequence>
            <xsd:element name="department" type="Department" minOccurs="0" maxOccurs= "unbounded" />
            <xsd:element name="employee" type="Employee" minOccurs="0" maxOccurs= "unbounded">
                <xsd:unique name="dependentNameUnique">
                    <xsd:selector xpath="employeeDependent" />
                    <xsd:field xpath="dependentName" />
                </xsd:unique>
            </xsd:element>
            <xsd:element name="project" type="Project" minOccurs="0" maxOccurs="unbounded" />
        </xsd:sequence>
    </xsd:complexType>
    <xsd:unique name="departmentNameUnique">
        <xsd:selector xpath="department" />
        <xsd:field xpath="departmentName" />
    </xsd:unique>
    <xsd:unique name="projectNameUnique">
        <xsd:selector xpath="project" />
        <xsd:field xpath="projectName" />
    </xsd:unique>
    <xsd:key name="projectNumberKey">
        <xsd:selector xpath="project" />
        <xsd:field xpath="projectNumber" />
    </xsd:key>
    <xsd:key name="departmentNumberKey">
```

```
        <xsd:selector xpath="department" />
        <xsd:field xpath="departmentNumber" />
    </xsd:key>
    <xsd:key name="employeeSSNKey">
        <xsd:selector xpath="employee" />
        <xsd:field xpath="employeeSSN" />
    </xsd:key>
    <xsd:keyref name="departmentManagerSSNKeyRef" refer="employeeSSNKey">
        <xsd:selector xpath="department" />
        <xsd:field xpath="departmentManagerSSN" />
    </xsd:keyref>
    <xsd:keyref name="employeeDepartmentNumberKeyRef"
        refer="departmentNumberKey">
        <xsd:selector xpath="employee" />

            <xsd:field xpath="employeeDepartmentNumber" />
        </xsd:keyref>
        <xsd:keyref name="employeeSupervisorSSNKeyRef" refer="employeeSSNKey">
            <xsd:selector xpath="employee" />
            <xsd:field xpath="employeeSupervisorSSN" />
        </xsd:keyref>
        <xsd:keyref name="projectDepartmentNumberKeyRef" refer="departmentNumberKey">
            <xsd:selector xpath="project" />
            <xsd:field xpath="projectDepartmentNumber" />
        </xsd:keyref>
        <xsd:keyref name="projectWorkerSSNKeyRef" refer="employeeSSNKey">
            <xsd:selector xpath="project/projectWorker" />
            <xsd:field xpath="SSN" />
        </xsd:keyref>
        <xsd:keyref name="employeeWorksOnProjectNumberKeyRef"
            refer="projectNumberKey">
            <xsd:selector xpath="employee/employeeWorksOn" />
            <xsd:field xpath="projectNumber" />
        </xsd:keyref>
    </xsd:element>
    <xsd:complexType name="Department">
        <xsd:sequence>
            <xsd:element name="departmentName" type="xsd:string" />
            <xsd:element name="departmentNumber" type="xsd:string" />
            <xsd:element name="departmentManagerSSN" type="xsd:string" />
            <xsd:element name="departmentManagerStartDate" type="xsd:date" />
            <xsd:element name="departmentLocation" type="xsd:string" minOccurs="0" maxOccurs="unbounded" />
        </xsd:sequence>
    </xsd:complexType>
    <xsd:complexType name="Employee">
```

```
    <xsd:sequence>
        <xsd:element name="employeeName" type="Name" />
        <xsd:element name="employeeSSN" type="xsd:string" />
        <xsd:element name="employeeSex" type="xsd:string" />
        <xsd:element name="employeeSalary" type="xsd:unsignedInt" />
        <xsd:element name="employeeBirthDate" type="xsd:date" />
        <xsd:element name="employeeDepartmentNumber" type="xsd:string" />
        <xsd:element name="employeeSupervisorSSN" type="xsd:string" />
        <xsd:element name="employeeAddress" type="Address" />
        <xsd:element name="employeeWorksOn" type="WorksOn" minOccurs="1" maxOccurs="unbounded" />
        <xsd:element name="employeeDependent" type="Dependent" minOccurs="0" maxOccurs="unbounded" />
    </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="Project">
    <xsd:sequence>
        <xsd:element name="projectName" type="xsd:string" />
        <xsd:element name="projectNumber" type="xsd:string" />

         <xsd:element name="projectLocation" type="xsd:string" />

         <xsd:element name="projectDepartmentNumber" type="xsd:string" />
         <xsd:element name="projectWorker" type="Worker" minOccurs="1" maxOccurs="unbounded" />
    </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="Dependent">
    <xsd:sequence>
        <xsd:element name="dependentName" type="xsd:string" />
        <xsd:element name="dependentSex" type="xsd:string" />
        <xsd:element name="dependentBirthDate" type="xsd:date" />
        <xsd:element name="dependentRelationship" type="xsd:string" />
    </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="Address">
    <xsd:sequence>
        <xsd:element name="number" type="xsd:string" />
        <xsd:element name="street" type="xsd:string" />
        <xsd:element name="city" type="xsd:string" />
        <xsd:element name="state" type="xsd:string" />
    </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="Name">
    <xsd:sequence>
        <xsd:element name="firstName" type="xsd:string" />
        <xsd:element name="middleName" type="xsd:string" />
        <xsd:element name="lastName" type="xsd:string" />
    </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="Worker">
    <xsd:sequence>
```

```
            <xsd:element name="SSN" type="xsd:string" />
            <xsd:element name="hours" type="xsd:float" />
        </xsd:sequence>
    </xsd:complexType>
    <xsd:complexType name="WorksOn">
        <xsd:sequence>
            <xsd:element name="projectNumber" type="xsd:string" />
            <xsd:element name="hours" type="xsd:float" />
        </xsd:sequence>
    </xsd:complexType>
</xsd:schema>
```

Primjer 20. XML Schema Company database

Izvor: Elmasri R., Navathe S. B., *Foundamentals of Database System*, poglavlje 12 (2011.)

Koraci za nastanak XML Scheme:

1. DTD mehanizam je prekompliciran – pojednostavimo ga!
2. W3C ustanovio radnu grupu za rješavanje tog problema
3. Nastala je XML Schema puno kompliciranija od DTD-a.

Iako je kompliciranija, pruža puno više mogućnosti. Npr. u DTD-u ne možemo odrediti da sadržaj nekog elementa bude broj, dok se to Schemi to može (slika 7).

```
<xsd:element name="mobile" type=
                    "xsd:positiveInteger"/>
```

Slika 7. XML Schema (dozvoljen samo broj)

Izvor: Kirasić D., XML tehnologija i primjena u sustavima procesne informatike (str. 4)

Druga razlika je uspoređujući DTD i XML Schemu, XML Schema pisana kao dobro formiran XML dokument, definirana vrlo logično i lako razumljivo po W3C standardima (http://www.w3schools.com/xml/xml_schema.asp).

Upravo XML Schema i DTD mehanizmi su pomoću kojih korisnici mogu točno odrediti vlastite formate zapisa podataka i dokumenata. Ti točno određeni dozvoljeni formati su ključni za strojnu obradu razmijenjenih podataka jer se mogu izbjeći potencijalne greške.

2. OSNOVE BAZE PODATAKA

Baza podataka je softverska organizirana zbirka podataka namijenjena za pohranjivanje, analizu i pretraživanje grupe sličnih i povezanih podataka. Sastoji se od jedne ili više (dvodimenzionalnih) tablica koje međusobno mogu biti povezane. Svaka tablica čuva istovrsne podatke (npr. o nekom osobi, predmetu i sl.). Svaki red u tablici predstavlja jedan slog u tablici[3], svaka kolona jedno od polja unutar tog sloga. Znači, slog može biti grupa podataka koja opisuje npr. neku osobu, a polja unutar tog sloga mogu sadržavati ime, prezime, adresu stanovanja, datum rođenja, grad te osobe. Slog se naziva i entitet, a polje se naziva atribut. Svaki slog tablice se može jedinstveno identificirati putem jedne ili kombinacijom vrijednosti nekog polja tog sloga. To polje ili kombinaciju polja tada zovemo primarni dio, primarni ili osnovni ključ. Tako se neku osobu može jedinstveno identificirati preko matičnog broja ili kombinacijom polja koji mogu biti primarni ključ. Osim identifikacije, ima ulogu povezivanja tablica. Ukoliko imamo tablicu popisa profesora, i tablicu popisa kolegija, potrebno je te dvije tablice povezati kako bismo znali koji profesor pripada kojem kolegiju. Ako u entitet ubacimo polje koje sadrži vrijednost primarnog ključa profesora, tablice smo povezali. Novo polje, koje služi samo za povezivanje dvije tabele, zovemo strani ključ, a rezultat povezivanja zovemo relacijski model baze podataka (opisan u daljnjem tekstu).

2.1 Povijest baze podataka

Početak korištenja termina „baza podataka“ seže u 60te godine 20. stoljeća kada je „*Društvo za razvoj sustava*“ uzelo pod pokroviteljstvo simpozij „*Razvoj i upravljanje računalno centriranom bazom podatak*a“.

Sam pojam baza podataka postala je uobičajena u Europi ranih 1970ih, a krajem desetljeća koristila se u glavnim američkim novinama. Prvi sustavi upravljanja bazom podataka (SUBP) razvijenu su u 1960ima od začetnika Charles Bachman-a čiji radovi pokazuju da mu je cilj bio stvaranje djelotvornije upotrebe novih uređaja s izravnim pristupom pohrane koji su bili dostupni (do tada se obrada temeljila na bušenim karticama i magnetskoj vrpci).

[3] Najmanja grupa podataka u bazi koja u potpunosti opisuje neki od koncepta koje baza modelira, izvor: http://www.seminarski-diplomski.co.rs/INFORMATIKA/BazePodataka.html

U 1990im pažnja je bila usmjerena na baze podataka orijentirane ka objektu, a u 2000im postale su XML baze podataka koje uklanjaju tradicionalnu podjelu između dokumenata i podataka, omogućujući svim organizacijskim informacijskim resursima da se drže na jednom mjestu koliko god bili visoko strukturirani.

2.2 Modeli baze podataka

Podaci su logički organizirani po nekom modelu. Model čini osnovu za osmišljanje, definiranje i implementiranje baze podataka. Današnji Sistem za upravljanje bazom podataka (SUBP ili izvorno Database Management System, DBMS) podržava tri osnovna modela:

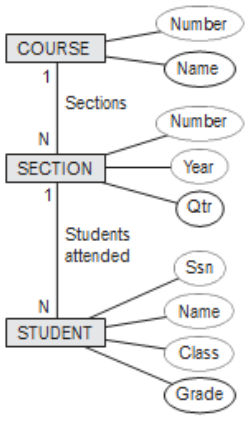
1. Hijerarhijski model
2. Mrežni model
3. Relacijski model

2.2.1 Hijerarhijski model

Zasnivaju se na hijerarhijskoj strukturi podataka koje su u obliku stabla (slika 8). Stabla imaju nivoe, s time da je prvi nivo osnovni ili korijenski. Kada se briše nadređeni segment, brišu se i svi njegov podređeni.

Nedostatci:

1. Unošenja – nije moguće unijeti neki segment ako nije poznat njegov nadređeni segment
2. Brisanja – brisanjem nadređenih se gube podaci o podređenim
3. Ažuriranja – nekad ažuriranje ovisi o ažuriranju drugih podataka.

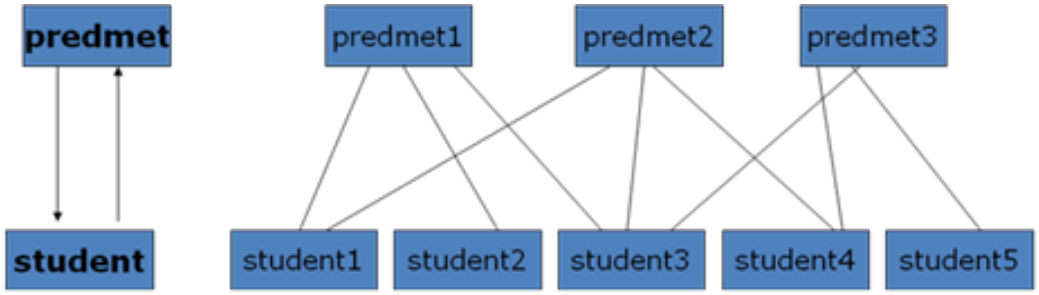


Slika 8. Hijerarhijski model (Course kao root)

Izvor: Elmasri R., Navathe S. B., *Foundamentals of Database System*, poglavlje 12 (2011.)

2.2.2 Mrežni model

Mrežni model može se predstaviti usmjerenim grafom u kojem su podaci čvorovi, a veze između podataka lukovi među čvorovima. Zasnivaju se na mreži podataka povezanih tako da ne postoji ni glavni ni podređeni segment (slika 9). Ne postoje nedostatci koje ima hijerarhijski model.



Slika 9. Mrežni model

Izvor: Jeremić M., Pojam baza (str. 7)

2.3 Relacijske baze podataka

Relacijski model je danas popularan i osnova velikog broja SUBP-a.

Karakteristike:

1. Jednostavna struktura predstavljanja
2. Određene tablice relacijskog modela mogu se tretirati kao matematičke relacije.

Ovaj model zasniva se na pojmu relacija iz teorije skupova. Koristi se relacijska algebra i relacijski računi te *upitni jezici* (SQL, QBE, DBMS).

2.3.1 Relacija i tablice

Uspostavljanje relacija između podataka može se prokazati na primjeru obavještenja o prodaji avionskih karata. Podaci se mogu predstaviti skupom PRODAJA_KARATA koji se sastoji od PUTNIK(p), BR_LETA(n) i DATUM(d), s time da je PRODAJA_KARATA=(p,n,d) odnosno putniku(p) je prodano mjesto na letu(n) za datum(d). To se zove skup, odnosno relacija PRODAJA_KARATA (p, n, d) je skup.

Relacijski model sastoji se od dva koncepta: relacija i domena.

RELACIJA ≠ TABLICA, ali u relacijskom modelu relacija je tablica.

Domena je skup vrijednosti iz kojih podaci uzimaju vrijednost, moguće vrijednosti podataka.

- U tablicama ne postoji redoslijed, u relacijama nije bitan
- U relacijskom modelu, baza je skup tablica
- Tablica ima naziv (relaciju) odnosno klasa entitet (objekt)
- Podaci su atributi u relacijskom modelu
- Kolone tablice su vrijednosti atributa
- Vrsta tablice su *n*-dio relacije.

Kako bi se izbjeglo ponavljanje podataka u tablici, svaki red mora imati primarni ključ. Primarni ključ povezan sa drugom tablicom postaje strani ključ.

2.3.2 Relacijski model

Relacijski model zasnovan je na matematičkom pojmu relacije. Podaci i veze među podacima se prikazuju preko dvodimenzionalnih tablica. Ukoliko red sadrži *n* kolona, onda je i relacija *n*-ta.

Relacija ↓ redovi →

Osobni broj	Ime	Prezime	Razred
1021/96	Ana	Petrović	III/1
1031/96	Petar	Arsić	III/5
1045/96	Maja	Nikolić	III/4
1019/96	Jelena	Jović	II/5

Tablica 1. Relacijski model

Izvor: autor

Tablica – skup podataka organiziranih po vrstama i kolonama
Polje (atribut) – jedinica ili kolona informacije u tablici. Npr. tablica student može sadržavati polja naziva osobni_broj, ime, prezime, razred.
Entitet (slog/zapis) – skup svih polja u tablicu (red po tablici). Npr. tablica student ima četiri zapisa tj. četiri imena.

STUDENT (<u>osobni_broj</u>, ime, prezime, razred)

Podataka ispred zagrade je **naziv relacije**, podaci u zagradi su nazivi **atributa**, podvučeni podaci su **ključevi** za identifikaciju redova.

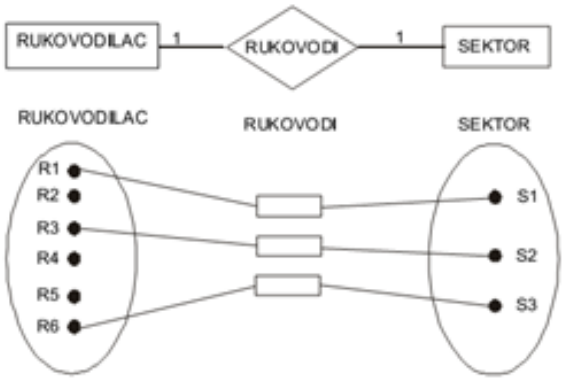
Entitet predstavlja objekt gledanja koji se može izdvojiti iz okoline i opisati. To je element o kojem se pamte informacije. **Atribut** predstavlja opis entiteta.

Za opis posebno određenog entiteta zapisuju se vrijednosti obilježja.

student (2316-E, Ana, Šverko, III/1)

Jedna baza može se sastojati od jednog ili više entiteta. Između entiteta mogu postojati sljedeće veze:

Veza 1:1 - Najjednostavniji oblik veze. Primjerice: veza rukovodilac-sektor. Na slici 10 vidimo da jedan rukovodilac može rukovoditi samo jednim sektorom, ali i obratno.

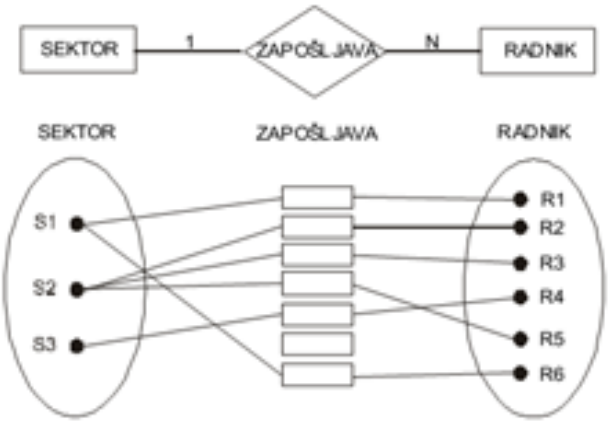


Slika 10. Veza 1:1

Izvor: Jeremić M., Pojam baza (str. 10)

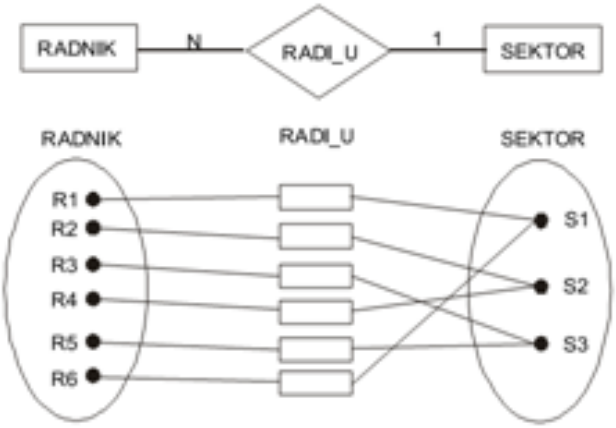
Veza 1:n (n:1) - Najčešći tip veze. Ovisno iz kojeg smjera gledamo ovisi je li 1:n ili n:1.

Primjer: sektor zapošljava radnika, gdje u jednom sektoru može biti više radnika (slika 11) dok u suprotnom smjeru, više radnika može raditi u jednom sektoru (slika 12).



Slika 11. Veza 1:n

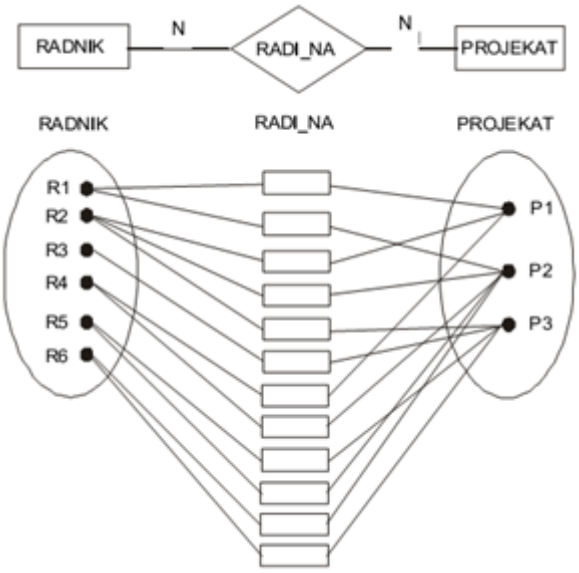
Izvor: Jeremić M., Pojam baza (str. 10)



Slika 12. Veza n:1

Izvor: Jeremić M., Pojam baza (str. 10)

Veza n:n - Veza (više prema više) (slika 13) koja se u modelima često javlja, a karakteristična je da se ne može implementirati u relacijskom modelu baze podataka jer bi dovela do nemogućnosti stroge definicije. Problem se „razbija“ vezom 1:n tako da se veza radnik:projekt (n:1) dijeli na dvije veze tj. PROJEKT:P_R(1:n) i P_R:RADNIK (n:1) gdje je P_R novi entitet (tablica u bazi) koji osigurava kontrolu modela.



Slika 13. Veza n:n

Izvor: Jeremić M., Pojam baza (str. 10)

2.3.3 Primarni i strani ključevi

Osnova relacijskog modela je da sve n-torke u relaciji budu različite. Da bi se to ostvarilo mora postojati jedan atribut ili više njih koji zajedno čine vrijednosti jedinstvene identifikacije jedne *n*-torke relacije (jedan red tablice).

Na primjer:
STUDENT(sifra_st, ime, adresa);
PREDMET(sifra_pr, naziv);
OCJENA(sifra_st, ocjena);

Da bi atribut postao ključ relacije, mora imati:

- Jedinstvenost, ne smiju postojati dvije n-torke sa istom vrijednošću k
- Osobna neponovljivost, ako se bilo koji atribut izostavi iz k, gubi se osobina jedinstvenosti

Strani ključ je atribut ili grupa atributa u relacije R1 koji u njoj nije primarni ključ, ali je primarni neke druge relacije baze podataka. Vrijednost stranog ključa koristi se za povezivanje vrijednosti primarnog ključa u određenoj relaciji R2.

2.4 Shema relacijske baze

Pojam sheme relacijske baze podataka i sama relacijska baza su različite. Shema relacijske baze definira strukturu baze. Za studentski informacijski sistem npr. *Relacijski model baze podatak*a bi se mogao postaviti ovako:

STUDENT(sifra_st, ime, adresa, sifra_sm); (sifra_st – primarni ključ, sifra_sm – strani ključ)
PREDMET (sifra_pr, naziv_pr, br_sata);
OCJENA(sifra_pr, sifra_st, ocjena); (sifra_pr, sifra_st – primarni ključevi)
SMJER(sifra_sm, naziv_sm, sifra_nast);
NASTAVNIK(sifra_nast, ime_nast, sifra_pr);

U ovome sistemu koji se odnosi na studente, objekti su:

- Student – objekt, s atributima – sifra_st, ime, adresa
- Predmet – objekt, s atributima – sifra_pr, naziv_pr, br_sata
- Ocjena – objekt, s atributom – ocjena
- Smjer – objekt, s atributima – sifra_sm, naziv_sm
- Nastavnik – objekt, s atributima – sifra_nast, ime_nast

Veze između objekata se ostvaruju preko atributa, odnosno ključeva i stranih ključeva.

Strani ključ

- Vrijednost atributa sifra_sm u relaciji student pokazuje koji smjer je student izabrao, pri čemu se pretpostavlja da student bira samo jedan smjer
- Vrijednost atributa sifra_nast u relaciji smjer pokazuje nastavnika kao rukovodioca na tom smjeru (može biti na samo jednom smjeru)
- Vrijednost atributa sifra_pr u relaciji nastavnik pokazuje koji predmet nastavnik predaje, pri čemu se pretpostavlja da nastavnik predaje samo jedan predmet.

Sama relacijska baza podataka izgleda ovako kao na sljedećim tablicama (tablica 2-tablica 6):

STUDENT

sifra_st	ime	adresa	sifra_sm
01	Dušan	Zagrebačka 1	001
02	Maja	Brodska 3	002
03	Ana	Savska 5	002
04	Darko	Pulska 7	001

Tablica 2. Relacijska BP student

Izradio: autor

PREDMET

sifra_pr	naziv_pr	br_sata
0001	Informatika	5
0002	Ekonomija	2
0003	Engleski	4

Tablica 3. Relacijska BP predmet

Izradio: autor

SMJER

sifra_sm	naziv_sm	sifra_nast
001	Društveni	00001
002	Glazbeni	00002

Tablica 4. Relacijska BP smjer

Izradio: autor

OCJENA

sifra_st	Sifra_pr	ocjena
01	0001	4
01	0002	5
02	0001	3
03	0003	5

Tablica 5. Relacijska BP ocjena

Izradio: autor

NASTAVNIK

sifra_nast	ime_nast	sifra_pr
00001	Vanja	0001
00002	Valter	0002
00003	Dean	0002
00004	Ivan	0003

Tablica 6. Relacijska BP nastavnik

Izradio: autor

Relacije u nekoj bazi mogu se podijeli ti „bazne“ i „izvedene“. Izvedene relacije (pogled) su relacije kojoj se mogu izvesti iz skupa danih baznih i izvedenih relacija, preko operacija definiranih nad relacijama.

Primjerice:
DOBRI_STUDENTI(sifra_st, ime_st, naziv_sm, sifra_pr, ocjena)

DOBRI_STUDENTI

sifra_st	ime_st	naziv_sm	sifra_pr	ocjena
01	Dusan	Ekonomija	0001	4
02	Maja	Engleski	0002	5

Tablica 7. Relacijska BP dobri_studenti

Izradio: autor

u ovoj relaciji svi studenti su oni koji iz predmeta imaju veće ocjene ili jednake određenoj ocjeni. Bazna relacija je relacija koja se ne može izvesti iz ostalih relacija baze podataka.

2.5 Primjena baze podataka

SQL omogućava puni pristup podacima u relacijskim bazama podataka (Oracle, SQL, Server, Access) tako što korisnik opiše podatke koje želi vidjeti. SQL omogućava definiranje i modificiranje izgleda tablica unutar baze. Niti jedna operacija na bazama podataka, izvršena direktno iz DMBS-a ili preko korisničke aplikacije, ne bi mogla biti izvršena bez direktne ili indirektne uporabe SQL jezika. Sam SQL nije dovoljan ukoliko je potrebno izvršiti neki upit u točno određeno vrijeme ili ga ponoviti nekoliko puta. Zbog toga svaki DBMS uz SQL podržava bar još jedan programski jezik pomoću kojeg se mogu izvršiti složenije operacije, a to su ODBC, OLE DB, JDBC, DAO, ADO i drugi.

ODBC, programeri pomoću njega mogu raditi sa tabličnim podacima, kao što su SQL baze podataka ili multidimenzionalnim podacima, kao što su OLAP kocke. Aplikacije za upravljanje bazama podataka pozivaju funkcije u ODBC-u, a ODBC putem svojih drivera za baze podataka aplikaciji vraća podatke iz baze.

OLE DB se pojavio kao rješenje problema pristupanja složenijoj organizaciji podataka, kao što su tekst fileovi, e-mail sistem i drugi. To je nova verzija ODBC-a. OLE BC nadograđuje ODBC tako da radi na isti način kao i ODBC sa relacijskom bazom podataka, omogućuje pristup drugim izvorima podataka i vrši njihovu homogenizaciju.

JDBC je ekvivalent ODBC tehnologije namijenjene upotrebi prilikom razvoja aplikacija u Java programskom jeziku.

DAO se javlja kao rješenje u izradi objektnog modela za pristup bazi podataka koji sprječava dolazak do eventualnih grešaka pri programiranju prethodno navedenih pristupa. Sastavni je dio Visual Basica, a služi za pristup MS Access bazama podataka. Također omogućava pristup ODBC izvorima podataka gdje je najveći nedostatak -> prilikom pristupa ODBC bazama podataka sve naredbe za bazu podataka i svi podaci iz nje moraju proći kroz ovaj dodatni sloj, što može ugroziti performanse aplikacije.

ADO je nova tehnologija iz Microsofta čiji je cilj zamjena DAO kao standardnog objektnog modela za pristup bazama podataka.

DBMS

- sistem za upravljanje bazom podataka
- softversko hardverski paket koji omogućava bazi podataka biti lako dostupnoj svim korisnicima.

Softverski dio sistema koji neki proizvođači nazivaju „rukovodiocem baze“ služi kao veza između korisnika i baze podataka. Ta veza :

- osigurava softverske alate potrebne za kreiranje, primjenu, pristupanje i ažuriranje baze podataka
- upravlja ulazno – izlaznim operacijama
- na većim sistemima vodi računa o tajnosti i problemima istovrsnih korisnika
- omogućava nezavisnost podataka – aplikativni program se može mijenjati bez utjecaja na memorirane podatke
- promjene se mogu događati na nekim podacima bez utjecaja na druge
- rječnik podataka (eng. *Data Dictionary*) se ugrađuje u DBMS kroz bazu podataka i dio baze podataka koju program koristi
- može biti zasnovan na jednom od tri modela (opisano pod odjeljkom modeli baze podataka)

POD DBMS SPADAJU:

1. **SQL server**

- Predstavlja proizvod koji u sebi objedinjuje snagu i fleksibilnost velikih baza podataka, iz lakoću administracije
- Nudi mogućnost prijenosa baze podataka sa jednog na više fizičkih servera koji se ponašaju kao jedan
- Dozvoljava korisniku da upite postavlja koristeći obični engleski jezik
- Dolazi u pet različitih verzija, za korisnikove potrebe
- Najbrža je i najpouzdanija baza podataka

2. **My SQL**

- Odlično rješenje za male i srednje web siteove
- ne podržava podupite, stored procedure, strane ključeve

3. **Foxpro**

- dugo postoji na svjetskom tržištu
- nije zamišljen da obuhvaća i upravlja složene i podacima pretrpane baze podataka
- brza izrada relativno jednostavnih rješenja upravljanja podacima
- donosi poboljšanja koja se odnose na pojačanu komunikacijsku i internet funkcionalnost
- zahtjeva maksimalno dvjesto MB prostora za aplikaciju i smještanje instalacije.
- Ima svoj skriptni jezik

4. **Access**

MS Access sastavio je dio Office paketa i u potpunosti integrirat sa ostatkom paketa

Osnovne osobine:

- Potpuna podrška za SQL Server bazu podataka
- Posjeduje integriranu podršku za povezivanje i korištenje SQL Server formata baze podataka
- Potpuna, dvosmjerna u program integrirana podrška za XML
- Podrška za uvoz XML
- Vođenje malih i srednje-velikih baza podataka

5. **Oracle 9**

- Namijenjen za velike ustanove i korporacije gdje od pouzdanosti baze podataka ovisi opstanak kompanija ili sigurnost država
- Relacijska baza podataka koja uz bazu podataka uključuje i cijeli skup pomoćnih alata i aplikacija (e-mail, web serveri)
- Ne ovisi od samo jednog računala
- Oracle baze podataka nisu namijenjene masovnom tržištu

6. **OLAP – Online Analytical Processing**

- Omogućava korisniku lako i selektivno pronalaženje i prikazivanje podataka iz više različitih gledišta spremljenih u tzv. OLAP kocke (multidimenzionalne tablice)
- Ugrađeni u DBMS, Oracle, SQL Server.

3. TIPOVI XML BAZA PODATAKA

3.1 Flat files

Najjednostavniji oblik XML baze podataka u kojem se dokumenti snimaju u datoteku, a njima se rukuje pomoću sučelja programske potpore, API. Takva metoda je prihvatljiva za mali skup XML dokumenata jer postoje alati za pretraživanje i modifikaciju, indeksiranje te transakcijsku obradu podataka.

3.2 Relacijske baze podataka

Skladištenje nudi niz prednosti kao što su višekorisnički pristup, sigurnost, transakcije i drugo.

Načini za smještanje:
CLOB (eng. *Character Large Object*) – pretraživanje i modifikacija se vrši nad dokumentom izvan baze podataka koja ne nudi nikakve servise za rad s XML-om. XML dokumenti nisu indeksirani što degradira performanse.

Čisto relacijski pristup (eng. ***Pure Relational***) koriste se prednosti relacijskog modela jer se podatci mapiraju na redove i stupce. Dvije najznačajnije tehnike mapiranja su table-based i object-relational. Problem je razlika između XML formata i relacijskog modela.

XML enabled relacijske baze podataka rade sa XML strukturama i u skladu s time nudi različite servise koje uključuju podršku za upitne jezike (Xpath[4]) i API-je (DOM[5], eng. *Document Object Model*). Koristeći SQL i njegove dodatke se lociraju slogovi, a baza podataka omogućuje pogleda na te podatke u XML formatu.

[4] Xpath jezik omogućuje jednostavnije pretraživanje sadržaja u XML dokumentu, (Bourret R., Coates A. B., Harvey B., i ostali, *Advanced XML Applications* (2007.))
[5] DOM, sučelje koje omogućuje računalnim programima pristup i ažuriranje sadržaja i strukture XML dokumenta

3.3 Native xml

NXD (izvorni XML baze podataka) može biti i XML dokument i vrsta XML podataka, specijalizirana baza za pohranu koja sadrži relacijske baze podataka. Možemo zaključiti da XML baza podataka je u suštini bilo koji način za pohranu XML podataka kao XML dokument.

Native XML baza podataka:

- ✓ Definira logički model XML dokumenata – pohranjuje i dohvaća dokumente. Model mora sadržavati elemente, atribute, PCDATA i redoslijedno spremljen dokument. Primjeri takvih modela su Xpath, XML infoset, DOM i SAX 1.0.
- ✓ Ima osnovni XML dokument skladištenja, baš kao relacijska baza podataka retke tablice
- ✓ Nije potrebno imati određeni temeljni model fizičke pohrane. Na primjer, može biti izgrađen na relacijskoj, hijerarhijskoj ili objektno orijentiranoj bazi podataka.

XML dokument korišten u pregledniku je u biti native XML. Uz to korištenjem XML vrste podataka u tipovima relacijskih baza podataka kao što su Oracle ili SQL čini neke relacijske baze sposobne za native XML.
U biti sve što je potrebno da se opiše neka baza podataka kao NXD ili barem kao NXD sposoban je da se XML dokument spremi kao XML dokument. Nikakvo posebno modeliranje se ne aplicira.
Isto tako bilo kakvi relacijski, objekti ili čak hijerarhijske baze podatka se mogu koristit.

4. SKLADIŠTENJE XML DOKUMENATA

4.1 Data centric

Prva kategorija XML dokumenata koja karakterizira sa regularnom strukturom, organiziranim podacima bez miješanog sadržaja koji su projektirani tako da se koriste uglavnom za obradu strojeva (primjer 21) a manje ljudsku upotrebu.

```
<meni datum='5.10.2007'>
        <jelo>
                <ime>Domaca juha</ime>
                <cijena>50.00</cijena>
                <kalorije>600</kalorije>
        </jelo>
</meni>
```

Primjer 21. Data-centric dokument

Izrada: autor

Ovakvi podaci su najčešće smješteni u relacijskoj bazi podataka i javlja se potreba za unašanjem podataka iz relacijske baze u XML dokument, iz XML dokumenta u relacijsku bazu ili u oba smjera.

4.2 Document centric

Druga kategorija XML dokumenata koja je okarakterizirana manje regularnom strukturom zbog izmiješanog sadržaja u kojoj su dokumenti izrađeni za ljudsko korištenje te redoslijed nije bitan jer su pisani ručno u XML-u (primjer 22).

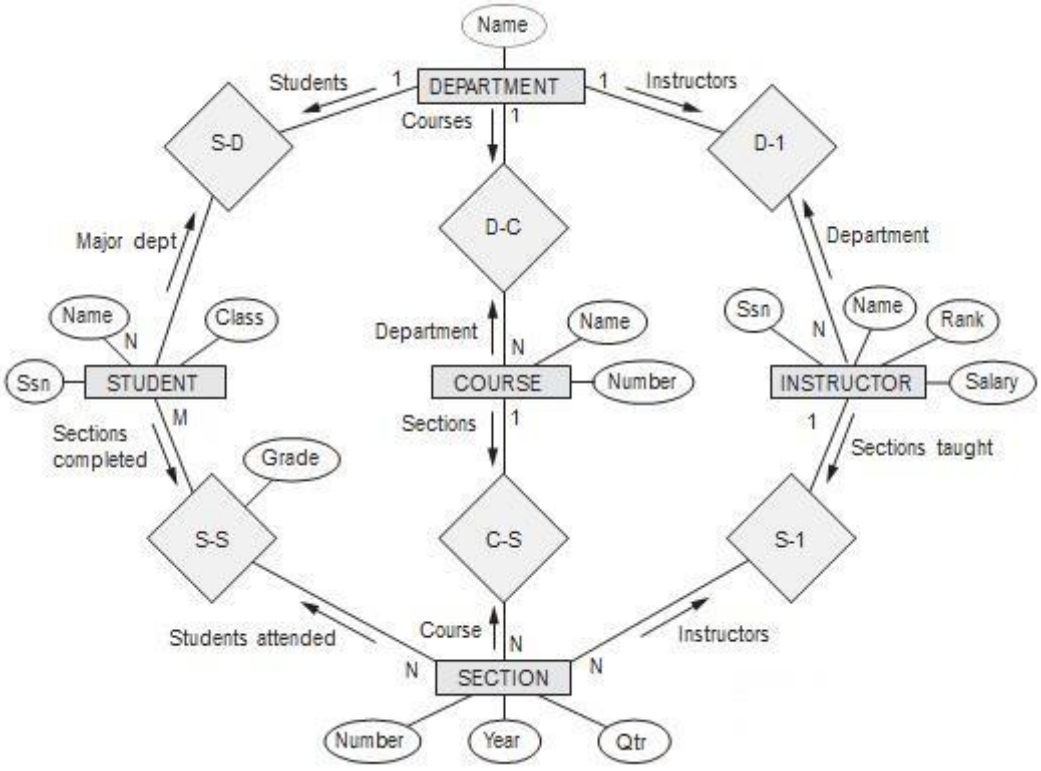
```
<Proizvod>
<Ime>KIRKOLINA – čaj za mršavljenje</Ime>
<Proizvodjac>Kirka-Pharma</ Proizvodjac>
<Opis>
<Paragraf> Predstavlja mešavinu lekovitog bilja koje kombinovanim dejstvom regulišu promet
    materija u organizmu, ubrzavaju sagorevanje masnih naslaga i utiču na smanjenje telesne
    težine. <i>Krušina, sena, zova</i> stimulišu metabolizam, podstiču probavu, smanjuju
    nadutost. <i> Breza, pirevina, rastavić </i>eliminišu nakupljene toksične materije,
    poboljšavaju cirkulaciju. <i>Matičnjak</i> oslobađa od stresa koji je često uzrok
    nekontrolisanog konzumiranja hrane. <i>Žalfija</i> kao izuzetni antiseptik štiti od mogućih
    infekcija i  utiče na jačanje organizma. </Paragraf>
<Paragraf><b>Možete:</b></Paragraf>
<List><Item><Link URL="Naruci.html">Naručiti svoj čaj za mršavljenje</Link></Item>
<Item><Link URL="Kirkolina.htm">Pročitati više o ovom proizvodu</Link></Item>
<Item><Link URL="Katalog.zip">Skinuti katalog naših proizvoda</Link></Item></List>
<Paragraf>Ovaj čaj košta <b>samo 500 dinara.</b></Paragraf>
</Opis>
</Proizvod>
```

Primjer 22. Document-centric dokument

Izvor. Tomašević J., XML baze podataka

5. EKSTRAKCIJA XML DOKUMENATA IZ RELACIJSKE BAZE PODATAKA

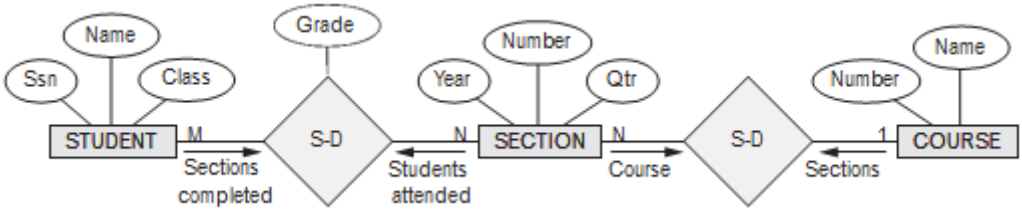
Ovdje je riječ o problemima konvertiranih podataka iz baze podataka u XML dokument.
Kako bi bilo lakše sve shvatiti, prikazati će se jednostavna University ER Schema na slici 14. Pretpostavlja se da aplikacija treba ekstrakciju XML dokumenata za informacije o students, courses i grades iz UNIVERSITY baze podataka. Podaci koji su potrebni sadržani su u atributima baze podataka entiteta tipova COURSE, SECTION i STUDENT iz slike 14 i veze S-S i C-S.



Slika 14. ER schema UNIVERSITY baze podataka

Izvor: Elmasri R., Navathe S. B., *Foundamentals of Database System*, poglavlje 12 (2011.)

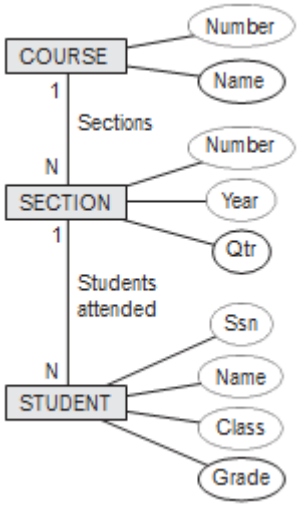
Općenito, većina ekstraktiranih dokumenata iz baze podataka se koristi samo za podešavanje atributa, entiteta i relacijskih veza baze podataka. To je prikazano na slici 15.



Slika 15. UNIVERSITY baza podataka za XML extract

Izvor: Elmasri R., Navathe S. B., *Foundamentals of Database System*, poglavlje 12 (2011.)

Ekstraktiranje je moguće na barem tri moguća načina hijerarhije. Prvo, možemo izabrati COURSE kao root (slika 16).



Slika 16. COURSE kao root

Izvor: Elmasri R., Navathe S. B., *Foundamentals of Database System*, poglavlje 12 (2011.)

Svaki course entitet je u svojoj sekciji kao podelement i svaka sekcija ima studente kao podelemente. Možemo vidjeti posljedicu modeliranja informacija u hijerarhijskoj strukturi. Ako se student u više sekcija, njegove informacije će se pojaviti višestruko, jednom u svakoj sekciji. Moguća pojednostavljena XML Schema za ovaj primjer je prikazana na primjeru 23.

```
<xsd:element name="root">
    <xsd:sequence>
    <xsd:element name="course" minOccurs="0" maxOccurs="unbounded">
        <xsd:sequence>
            <xsd:element name="cname" type="xsd:string" />
            <xsd:element name="cnumber" type="xsd:unsignedInt" />
            <xsd:element name="section" minOccurs="0" maxOccurs="unbounded">
                <xsd:sequence>
                    <xsd:element name="secnumber" type="xsd:unsignedInt" />
                    <xsd:element name="year" type="xsd:string" />
                    <xsd:element name="quarter" type="xsd:string" />

                <xsd:element name="student" minOccurs="0" maxOccurs="unbounded">
                    <xsd:sequence>
                        <xsd:element name="ssn" type="xsd:string" />
                        <xsd:element name="sname" type="xsd:string" />
                        <xsd:element name="class" type="xsd:string" />
                        <xsd:element name="grade" type="xsd:string" />
                    </xsd:sequence>
                </xsd:element>
            </xsd:sequence>
        </xsd:element>
    </xsd:sequence>
</xsd:element>
</xsd:sequence>
</xsd:element>
```

Primjer 23. XML schema COURSE kao root

Izvor: Elmasri R., Navathe S. B., *Foundamentals of Database System*, poglavlje 12 (2011.)

Atribut Ocjena baza podataka u S-S relaciji je migrirala u element STUDENT. To je zato što je STUDENT podelement SEKCIJE u ovoj hijerarhiji, tako da svaki STUDENT element unutar određenog SECTION elementa može imati posebnu ocjenu u toj sekciji.

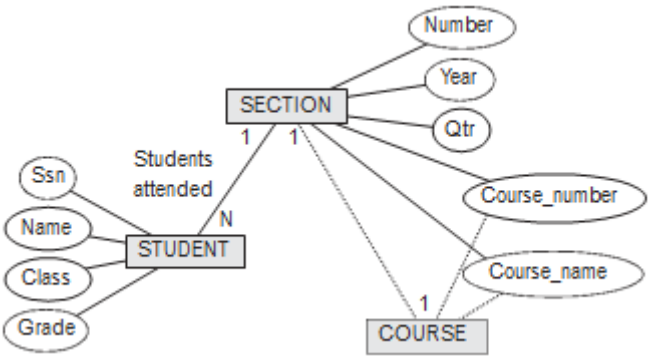
Drugi način je odabir STUDENTA kao root (slika 8). Svaki student ima sekciju kao podelement, i svaka sekcija je povezana sa jednim course kao podelementom jer je relacija između SECTION i COURSE N:1. Međutim, možemo rastaviti COURSE i SECTION elemente (kao na slici). Usput, GRADE bazu podataka možemo prebaciti u element SECTION. Tako bi kombinacija COURSE/SECTION bila zamijenjena unutar svakog studenta što bi zaokružilo sekciju (primjer 24)

```
<xsd:element name="root">
<xsd:sequence>
<xsd:element name="student" minOccurs="0" maxOccurs="unbounded">
    <xsd:sequence>
        <xsd:element name="ssn" type="xsd:string" />
        <xsd:element name="sname" type="xsd:string" />
        <xsd:element name="class" type="xsd:string" />
        <xsd:element name="section" minOccurs="0" maxOccurs="unbounded">
            <xsd:sequence>
                <xsd:element name="secnumber" type="xsd:unsignedInt" />
                <xsd:element name="year" type="xsd:string" />
                <xsd:element name="quarter" type="xsd:string" />
                <xsd:element name="cnumber" type="xsd:unsignedInt" />
                <xsd:element name="cname" type="xsd:string" />
                <xsd:element name="grade" type="xsd:string" />
            </xsd:sequence>
        </xsd:element>
    </xsd:sequence>
</xsd:element>
</xsd:sequence>
</xsd:element>
```

Primjer 24. COURSE/SECTION (STUDENT kao root)

Izvor: Elmasri R., Navathe S. B., *Foundamentals of Database System*, poglavlje 12 (2011.)

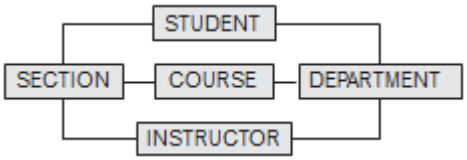
Treći mogući način je SECTION kao root (primjer 25). Slično kao u drugom načinu, pogled, COURSE informacije mogu se rastaviti na element SECTION. GRADE baza podataka atributa mogla bi se prebaciti u element STUDENTA. Kao što vidimo u ovom jednostavnom primjeru, može biti više načina hijerarhijskih pogleda (view), svaki odgovara različitom root-u i različitoj XML strukturi.



Primjer 25. SECTION kao root

Izvor: Elmasri R., Navathe S. B., *Foundamentals of Database System*, poglavlje 12 (2011.)

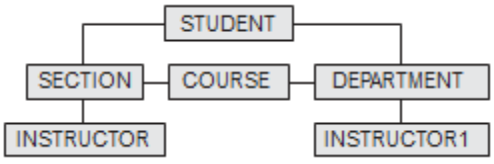
Uzimajući u obzir informacije svih entiteta i relacija koje nam trebaju, u ovom poglavlju je opisano kako je moguće napraviti hijerarhijski model. Prvo, imamo STUDENT kao root (a).



Slika 17. a

Izvor: Elmasri R., Navathe S. B., *Foundamentals of Database System*, poglavlje 12 (2011.)

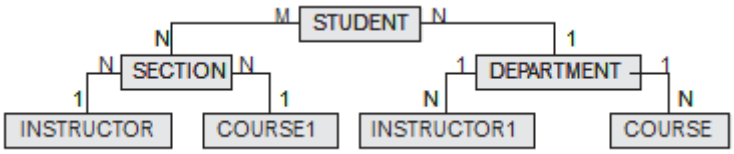
Nije hijerarhijska struktura zbog kruga. Prvi korak je „slomiti“ krug prebacivanjem entiteta iz kruga. Za početak prebacujemo INSTRUCTOR (b) i nazivamo ga INSTRUCTOR1. INSTRUCTOR lijepo predstavlja vezu između instruktora i sekcije učenja, gdje INSTRUCTOR1 prebačen na desnu stranu predstavlja vezu između instruktora i odjela u kojem radi.



Slika 18. b

Izvor: Elmasri R., Navathe S. B., *Foundamentals of Database System*, poglavlje 12 (2011.)

Nakon ovoga, i dalje imamo u krugu COURSE pa ćemo njega isto prebaciti (c). COURSE1 lijevo predstavlja vezu između courses i sekcija, dok na desnoj strani predstavlja vezu između courses i odjela koji nudi courses. Slika c je konvertirana u inicijalni graf hijerarhije.



Slika 19. c

Izvor: Elmasri R., Navathe S. B., *Foundamentals of Database System*, poglavlje 12 (2011.)

6. PREDNOSTI, NEDOSTATCI I INTEROPERABILNOST

Prednosti XML su:

- jednostavno je čitljiv i čovjeku u običnom tekstualnom editoru i računalu
- XML dokument je obična tekstualna datoteka čitljiva na svakoj platformi koja može čitati tekstualne podatke.
- podržava Unicode i omogućuje prikaz teksta na svim danas poznatim jezicima
- format je samodokumentirajući. Oznake opisuju sadržaj koji se nalazu unutar njih.
- ima stroga sintaksna pravila tako da je jednostavno kontrolirati ispravnost nastalog dokumenta.
- međunarodno prihvaćen standard.
- hijerarhijska struktura je pogodna za opisivanje mnogih sadržaja (ali ne i svih!)
- kompatibilan je sa SGMLom koji se koristi od 80ih godina 20. stoljeća, a za SGML postoji dosta računalnih programa koji ga mogu obrađivati.

Nedostatci XML su:

- sintaksa je redundantna i opširna što može zamarati i zbunjivati osobu koja čita XML dokument.
- da bi dokument bio dovoljno dobro "samoopisan" nazivi oznaka moraju biti dovoljno precizni što dovodi do dugih "kobasičastih" naziva Nije dovoljno osloniti se na "samoopisivanje" sadržaja
- redundancija i velika količina podataka stvaraju velike zahtjeve za propusnosti mreže
- programi koji obrađuju XML podatke su dosta složeni jer moraju obrađivati velike količine ugniježđenih podataka u više razina
- nedostatak formalno propisanih formata za podatke može stvarati probleme ako sudionici u razmjeni nisu dobro opisali
- pohrana XML podataka u relacijske baze podataka nije prirodan način i to dovodi do smanjenja performansi sustava koji koriste takav način pohrane. S druge strane XML baze podataka koje su razvijene za pohranu XML podataka još su u fazi razvoja

INTEROPERABILNOST

Interoperabilnost je sposobnost informacijskih i komunikacijskih sustava i poslovnih procesa da podrže protok podataka i omoguće razmjenu informacija i znanja. Mora se osigurati na tehničkoj (norme i standardi za povezivanje računalnih sustava i servisa), semantičkoj (značenje podataka) i procesnoj razini (definiranje poslovnih ciljeva, modeliranje poslovnih procesa i ostvarivanje suradnje između različitih upravnih jedinica).

Interoperabilnost se može ostvarivati primjenom nacionalnih i međunarodnih normi. Okvir interoperabilnosti je skup normi, standarda i preporuka koji opisuju postignuti ili željeni dogovor nezainteresiranih strana o načinu međupovezivanja. Taj isti okvir je promjenjivi dokument koji mora pratiti normativne, poslovne i tehnološke promjene.

Posebna vrsta interoperabilnosti je ona koja ukazuje na pojam unutar nekog poduzeća odnosno organizacije, tzv. *inter-operativnost*.

Sam pojam interoperabilnosti informacijskih sistema može se gledati na tri načina:

1. Aplikacijska domena
2. Konceptualno projektiranje
3. Implementacija softverskih sistema, tj. tehnologija

Tri sloja interoperabilnosti:

1. Tehnički sloj – poruke su razmijenjene sigurno i pouzdano od pošiljatelja ka primatelju. Primateljska infrastruktura je odgovorna za isporuku podataka aplikacije na njenoj strani. Tehnička interoperabilnost brine o tehnologiji, standardima i politici koje se koriste za povezivanje računalskih sistema.
2. Semantički sloj – aplikcije znaju za poslovni kontekst kojem podaci pripadaju. Aplikaciji su podaci validni te ih ona procesuira. Ovaj sloj ignorira tehničku interoperabilnost i fokusira se sa na sadržaj prenesenih informacija.
3. Organizacijski sloj – aplikacija obavještava one korisnike koji su odgovorni za verifikaciju i odobravanje poslovnih odluka, kao i za praćenje rokova izvršenja. Organizacijska interoperabilnost brine o operacijama i organizacijskoj strukturi među sistemima.

Softverska arhitektura koja podržava koncept interoperabilnosti je SOA (servisno-orijentirana arhitektura). Njen najvažniji faktor je velika fleksibilnost koja se može postići ovom

arhitekturom. Poslovni procesi mogu se modificirati u trenutku izvođenja te se usluge mogu mijenjati, razmjenjivati ili uvećavati.
Struktura SOA-e je nešto drugo. To je tehnološki nezavisna međuaplikcijska integracijska arhitektura. SOA sadrži standarde koji ne zavise o pojedinim platformama, odnosno standardi internetske usluge (WSDL, SOAP) imaju važnu ali ne isključivu ulogu.

Poslovna logika je procesno strukturirano izvođenje poslovnih zahtjeva zajedno sa ograničenjima, zavisnostima i vanjskim utjecajima na te zahtjeve.
Aplikacijska logika automatizira rješenja poslovnih procesa kroz različite tehnologije. Sadrži rješenja potrebna za uspješno savladavanje tokova poslovnih procesa te se obično realizira kroz aplikacije zasnovane na pogodnim tehnološkim platformama. SOA zato uvodi treći sloj između poslovne i aplikacijske logike, tj. sloj interface usluga čime se postiže mogućnost enkapsulacije i aplikacijske i poslovne logike.

Web servisi su važan korak u razvoju Web-baziranih tehnologija i predstavljaju distribuirane serverske komponente. Oni predstavljaju metodu za pružanje servisa kojima se preko mrežne komunikacije može pristupiti programskim putem. Te komponente, za razliku od Web site-ova, nemaju korisnički interface, već su dizajnirane tako da budu korištene od strane drugih aplikacija, kao što su klijentske aplikacije, Web-bazirane aplikacije i drugi Web servisi. Web servisi mogu biti korišteni interno (od strane jedne aplikacije) ili eksterno preko Interneta za korištenje od strane većeg broja aplikacija. Web servisi koriste XML bazirane poruke kao glavni način prenošenja podataka, kako bi spojili sisteme koji koriste nekompatibilne modele komponenti, operativne sisteme i programske jezike. Tako su Web servisi izgrađeni na standardima kao što su WSDL (*Web Servise Description Language*, jezik za opis Web servisa), SOAP (protokol za poziv udaljenih metoda) i UDDI (*Universal Description, Discavery and Integration*).

Osnovna karakteristika Web servisa je visoki nivo apstrakcije između implementacije servisa i korištenja servisa. Upotrebom XML zasnovanih poruka kao mehanizma za kreiranje i pristup servisu, klijent Web servisa i sam Web servis oslobođeni su potrebe da znaju bilo što osim ulaza, izlaza i lokacije. Zaključujemo da je Web servis autonomna, modularna aplikacija koja se može opisati, objaviti, locirati i pozvati preko mreže.

Standardiziran način prezentacije podataka je XML, dok za opis servisa koristi se WSDL, protokol opisan na XML jeziku kao skup krajnjih mrežnih točaka i povezanih metoda koji se pružaju klijentu na korištenje. Kao i WSDL, SOAP specificira standardni format, čiji su djelovi podaci koji su enkodirani u XML dokumentu (pogledati poglavlje 1.2. Dijelovi: elementi i atributi).

7. RAZLIKE IZMEĐU XML I RELACIJSKOG MODELA

Kao što je navedeno ranije, bitna razlika između XML-a i relacijskog modela je u strukturi, odnosno, kod XML-a imamo hijerarhijsku strukturu gdje čvorovi imaju elemente i/ili atribute koji mogu biti ugniježđeni te imati definirati redoslijed dok je schema opcionalna.

Za razliku od XML-a, u relacijskom modelu podaci su smješteni u više tablica koje imaju jedinstvenu vrijednost, a što se tiče redoslijeda, ne postoji dok je schema u ovome slučaju obavezna.

Native XML (skraćeno NXD) se ne razlikuje od XML-a, jer može biti i XML dokument i XML vrsta podataka. Ono što XML smješta u hijerarhijsku strukturu, to u pregledniku pokazuje native XML.

Uz navedene razlike XML, relacijskog modela i Native XML-a, moramo spomenuti i razlike interoperabilnosti, odnosno interoperabilnost kroz tri načina pojednostavljuje razmjenu podataka između nekompatibilnih sustava (objašnjeno u prethodnom poglavlju 6).

ZAKLJUČAK

XML tehnologija ima prednost nad ostalim tehnologijama baš zbog svoje neovisnosti, otvorenosti i prenosivosti. Omogućuje poprilično jednostavno definiranje formata dokumenata i podataka koji su jednostavni i lako čitljivi i čovjeku u običnom tekstualnom editoru i računalu.

XML je prvenstveno namijenjen strojnoj i programskoj obradi, programska obrada XML-a je jednostavna iz razloga što postoji veliki broj različitih biblioteka programa koji ubrzavaju razvoj aplikacijskog programa. Trenutno postoje dvije verzije XML-a verzija 1.0 i verzija 1.1, iako se verzija 1.1 koristi isključivo zbog svojih određenih karakteristika i za razliku od 1.0 nije toliko rasprostranjena ni implementirana.

XML ima najviše kritika u vezi njegove složenosti i preopširnosti unatoč tome našao je primjenu u izuzetno puno područja, a upravo zbog njegove jednostavnosti i svojstvu da je prilagodljiv korisničkim potrebama mu daje jamstvo da će biti i ostati jedan od najraširenijih i korištenijih jezika za označavanje u budućnosti.

LITERATURA

Knjige:

1. Bourret R., Coates A. B., Harvey B., i ostali, *Advanced XML Applications* (2007.)
2. Elmasri R., Navathe S. B., *Foundamentals of Database System*, 6th edition, poglavlje 12 (2011.)
3. Powell G., *Beginning XML Databases*, Wiley Publiching, Inc., Indiana (2007.)

Internet stranice:

4. Di Brita D., *Google servisi za prostorne podatke i WMS*, https://bib.irb.hr/datoteka/375788.DiBrita.pdf, Zagreb 2008., 14. rujna 2015
5. Interoperabilnost, http://autopoiesis.foi.hr/wiki.php?name=KM+-+Tim+12&parent=NULL&page=interoperabilnost, 8. rujna 2015
6. Jeremić M., *Pojam baza*, https://racunarstvoiigimkg.files.wordpress.com/2013/09/pojam-baza-e28093-skripta-kg-miljan-g-jeremic487.doc, 16. kolovoz 2015
7. Kirasić D., *XML tehnologija i primjena u sustavima procesne informatike*, http://www.fer.unizg.hr/_download/repository/mipro_xml_tekst.pdf, 15. kolovoz 2015
8. Pilipović D., *Interoperabilnost na primjeru aplikacije o studentima*, 2010., 14. uujna 2015
9. Sigurnjak M., Stažić A., Savanović M., *XML*, www.mathos.unios.hr/~msigurnj/XML.ppt, 17. kolovoz 2015
10. Tomašević J., *XML baze podataka*, http://download.tutoriali.org/Tutorials/XML/XML_baze_podataka.pdf, 17. kolovoz 2015
11. Povijest baze podataka, http://www.seminarski-diplomski.co.rs/INFORMATIKA/BazePodataka.html, 17. kolovoz 2015
12. Unicode, http://www.unicode.org/, 17. kolovoz 2015
13. Uvod u XML i XML tehnologije, http://www.viser.edu.rs/download/uploads/6531.pdf, 25. kolovoz 2015
14. W3C, http://www.w3.org, 17. kolovoz 2015
15. XML logo, http://akraya.com/2015/08/tbt-where-does-xml-fit-in-with-big-data/, 10. kolovoz 2015

16. XML tekst editori, http://akshayam.in/wp-content/uploads/2014/06/xml_conversion_services.jpg, 10. kolovoz 2015
17. XML uvjeti, http://www.w3schools.com/xml/xml_syntax.asp, 16. kolovoz 2015

POPIS SLIKA

POPIS TABLICA

POPIS PRIMJERA