

Spletne tehnologije, UL, FRI (VSP) PHP, MySQL



doc.dr. Mira Trebar

Vsebina

- ▶ MySQL
- ▶ PHP in MySQL
- ▶ XML
- ▶ AJAX
- ▶ JSON

- ▶ Literatura:
 - ▶ Meloni, Julie C., PHP, MySQL and Apache (All in One), USA 2012
 - ▶ <http://www.w3schools.com/php/default.asp>

- ▶ Upravljanje podatkovnih baz
 - ▶ Knjižnica PDO - PHP Data Objects (Vaje)
 - ▶ <http://php.net/manual/en/intro.pdo.php>

Podatkovne baze

► Informacije:

- dejstva o predmetih, osebah, dogodkih, situacijah, ...
- dobimo jih z obdelavo podatkov.

► Podatki:

- Na določen način zapisana dejstva: številke, besede, slike, ...

► Podatkovna baza:

- urejena zbirka vsebinsko povezanih podatkov in njihovih opisov.
- model okolja, ki lahko služi kot osnova za nadzor, odločanje in izvajanje akcij.

► Primera:

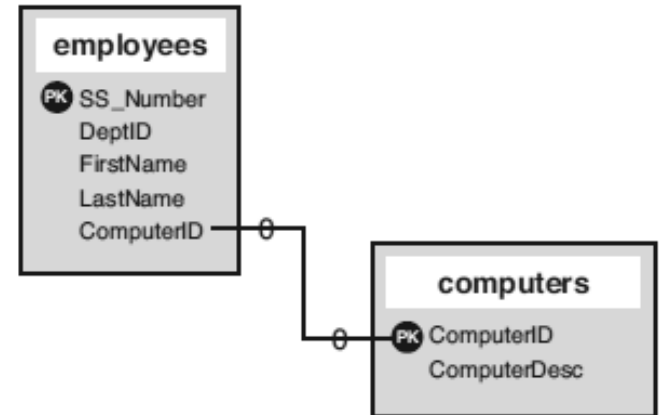
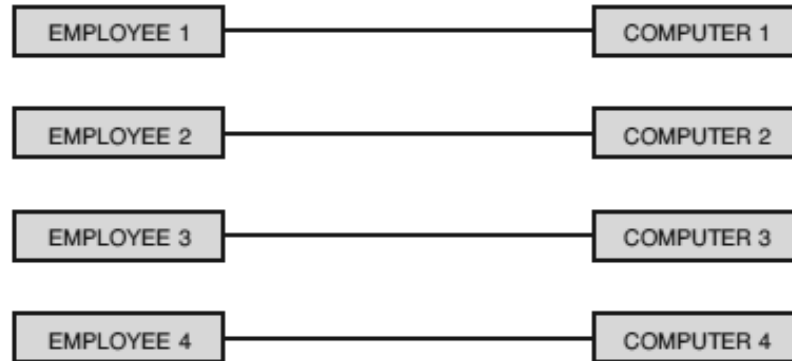
- ✿ Informacijski sistem: Google
- ✿ Podatkovna baza: Sistem zdravstvenega zavarovanja

▶ Sistemi za upravljanje podatkovnih baz

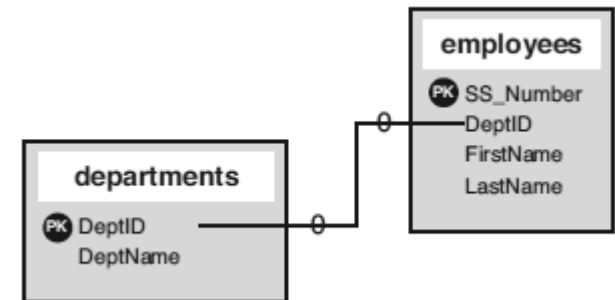
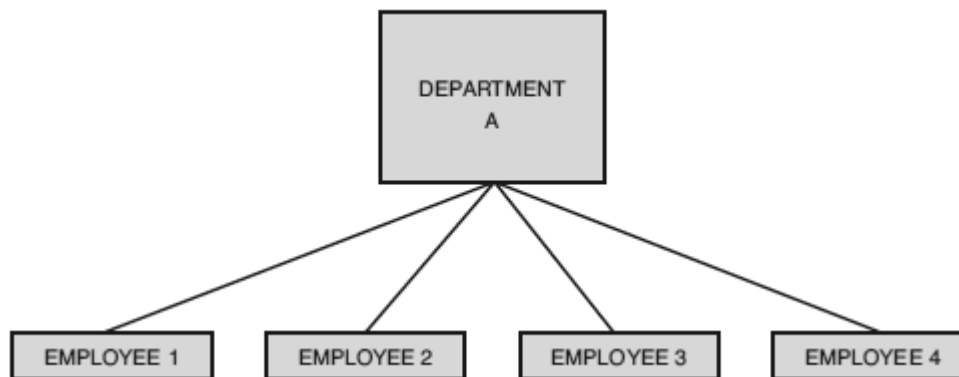
- ▶ orodja za izdelavo, vzdrževanje, nadzor podatkovnih baz
- ▶ omogočajo upravljanje s podatki
 - ▶ shranjevanje, organizacijo, pridobivanje, brisanje
- ▶ Naloge sistemov
 - ▶ zagotavljanje razpoložljivosti podatkov
 - ▶ nadzor nad uporabo podatkov
 - skrb za celovitost podatkov,
 - zagotavljanje uporabnosti podatkov

Relacijske podatkovne baze

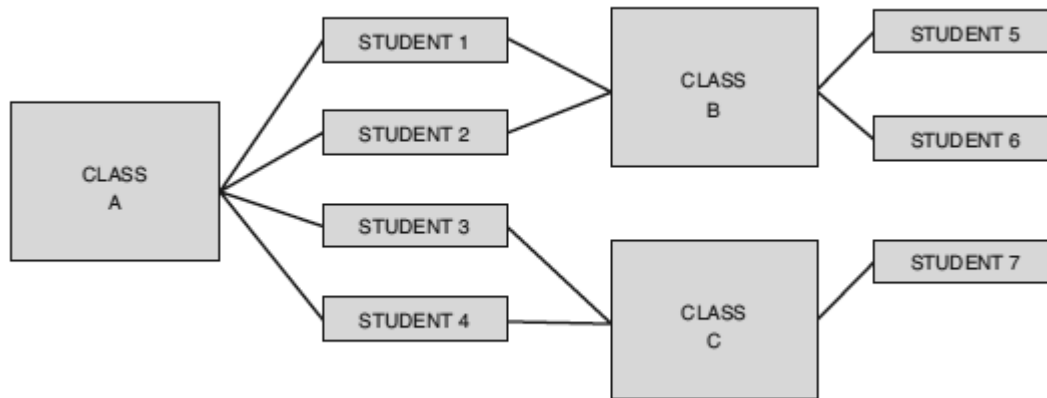
► One-to-one



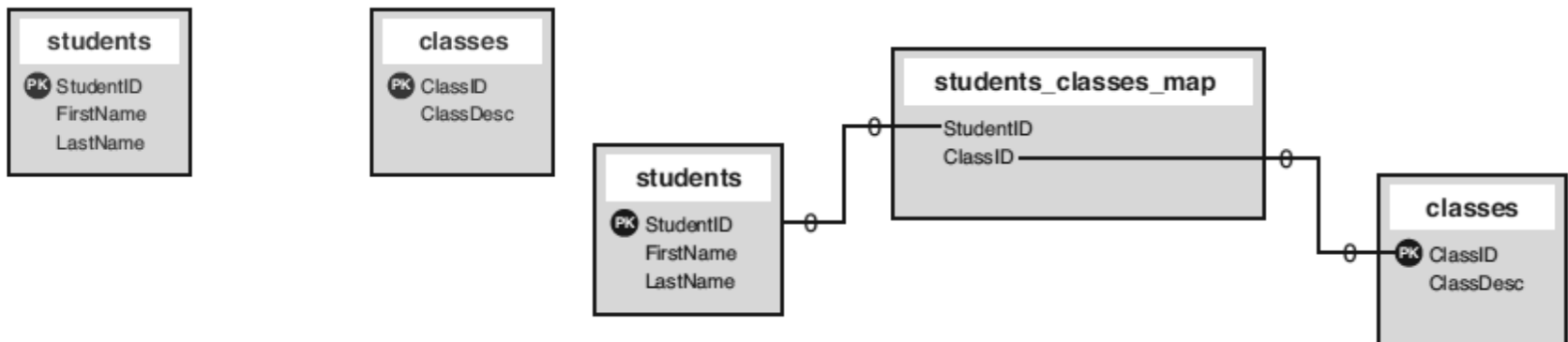
► One-to-many



► Many-to-many

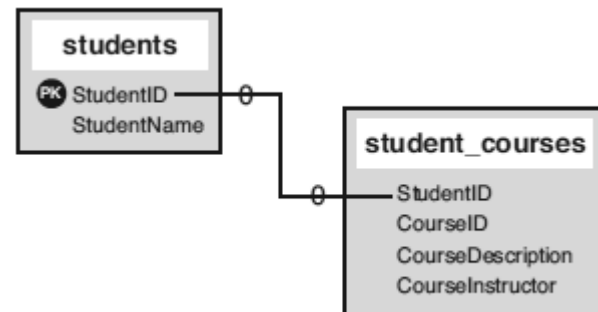


STUDENTID	CLASSID
STUDENT 1	CLASS A
STUDENT 2	CLASS A
STUDENT 3	CLASS A
STUDENT 4	CLASS A
STUDENT 5	CLASS B
STUDENT 6	CLASS B
STUDENT 7	CLASS C
STUDENT 1	CLASS B
STUDENT 2	CLASS B
STUDENT 3	CLASS C
STUDENT 4	CLASS C

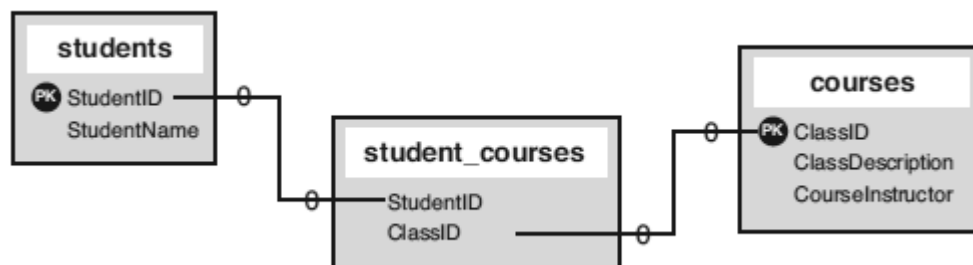


Normalizacija

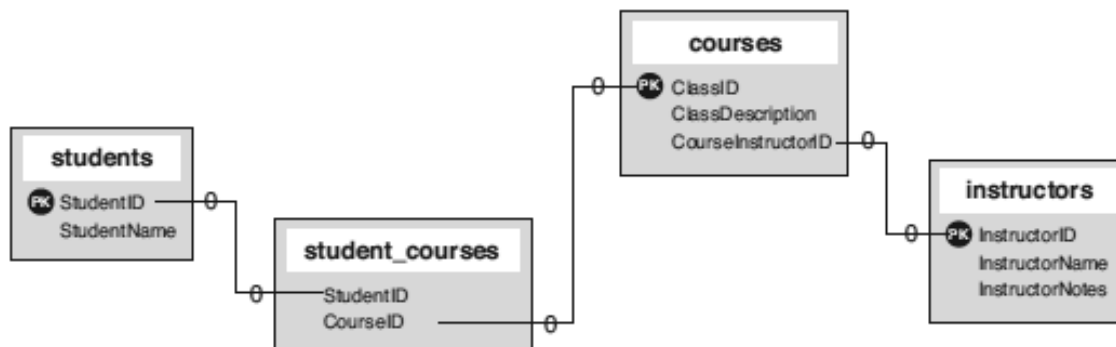
► Prva normalna oblika



► Druga normalna oblika



► Tretja normalna oblika



Osnove in MySQL

- ▶ The basic MySQL data types
- ▶ How to use the CREATE TABLE command to create a table
- ▶ How to use the INSERT command to enter records
- ▶ How to use the SELECT command to retrieve records
- ▶ How to use basic functions, the WHERE clause, and the GROUP BY clause in SELECT expressions
- ▶ How to select from multiple tables, using JOIN or subselects
- ▶ How to use the UPDATE and REPLACE commands to modify existing records
- ▶ How to use the DELETE command to remove records
- ▶ How to use string functions built in to MySQL
- ▶ How to use date and time functions built in to MySQL

Podatkovni tipi MySQL

Numerični

- ▶ INT,
- ▶ TINYINT,
- ▶ SMALLINT, MEDIUMINT, BIGINT
- ▶ FLOAT(M,D)
- ▶ DOUBLE(M,D)
- ▶ DECIMAL(M,D)

Nizi

- ▶ CHAR(M)
- ▶ VARCHAR(M)
- ▶ BLOB ali TEXT
- ▶ TINYBLOB ali TINYTEXT
- ▶ MEDIUMBLOB ali MEDIUMTEXT
- ▶ LONGBLOB ali LONGTEXT
- ▶ ENUM

Datum (YYYY-MM-DD) in Čas (HH:MM:SS)

- ▶ DATE
- ▶ DATETIME
- ▶ TIMESTAMP
- ▶ TIME
- ▶ YEAR(M)

Kreiranje tabele (table-creation)

CREATE TABLE table_name (column_name column_type);

Primer:

```
CREATE TABLE grocery_inventory (  
    id INT NOT NULL PRIMARY KEY AUTO_INCREMENT,  
    item_name VARCHAR (50) NOT NULL,  
    item_desc TEXT,  
    item_price FLOAT NOT NULL,  
    curr_qty INT NOT NULL  
);
```

Ukazi:

- ▶ INSERT
- ▶ SELECT
- ▶ WHERE
- ▶ UPDATE
- ▶ REPLACE
- ▶ DELETE

Funkcije z nizi v MySQL

- ▶ Dolžina: `SELECT LENGTH('Tekst')`
- ▶ Spajanje: `SELECT CONCAT('Tekst','in',števila')`
- ▶ Dodajanje in odvzemanje znakov – vključno s presledkom
`SELECT RTRIM('Tekst ')`
`SELECT LTRIM(' Tekst')`
- ▶ Lokacija: `SELECT LOCATE('števila','Tekst in števila')`
- ▶ Podniz: `SELECT SUBSTRING('Tekst' , 3, 2)`
`SELECT LEFT ('Tekst' , 2)`
`SELECT RIGHT ('Tekst' , 3)`
- ▶ Pretvorba: `SELECT LCASE(),`
`SELECT UCASE()`

Uporaba funkcij Datum in čas

- ▶ DAYOFWEEK(), WEEKDAY()
- ▶ DAYOFMONTH()
- ▶ DAYOFYEAR(), DAYNAME()
- ▶ MONTH(), MONTHNAME()
- ▶ WEEK()

- ▶ HOUR()
- ▶ MINUTE()
- ▶ SECOND()

- ▶ DATE_FORMAT(date, format)
- ▶ DATE_ADD(date, INTERVAL value type)
- ▶ CURDATE(), CURTIME(), NOW(),

Transakcije v MySQL

- ▶ Transakcija – niz poizvedb, ki se mora izvesti (vsa morajo biti uspešna)
- ▶ Sintaksa:
 - ▶ COMMIT – na koncu niza poizvedb v transakciji
 - ▶ ROLLBACK – uporabljena v primeru, da je ena ali več poizvedb neuspešnih, postavi vsebine tabel na začetno stanje
- ▶ Primer: Nakup v spletni trgovini
- ▶ Dodatno gradivo:
 - ▶ `mysqli_autocommit()`
 - ▶ `mysqli_commit()`
 - ▶ `mysqli_rollback()`

PHP in MySQL

- ▶ Funkcije MySQL – starejša verzija
- ▶ Funkcije MySQLi – novejša komunikacijske metode (v primerih)
- ▶ Povezava MySQL in PHP
 - ▶ Mesto izvajanja - lokalno
 - ▶ Uporabnik: username, password
 - ▶ Podatkovna baza - testDB
- ▶ Sintaksa

```
$mysqli = mysqli_connect("hostname", "username", "password", "database");
```

- ▶ Primer:

```
$mysqli = mysqli_connect("localhost", "root", "", "testDB");  
if (mysqli_connect_errno()) {  
    printf("Connect failed: %s\n", mysqli_connect_error());  
    exit();  
} else {  
    printf("Host information: %s\n", mysqli_get_host_info($mysqli));  
}
```

```
?>
```

Kreiranje tabele z imenom testTable

```
<?php
$mysqli = mysqli_connect("localhost", "root", "", "testDB");

if (mysqli_connect_errno()) {
    printf("Connect failed: %s\n", mysqli_connect_error());
    exit();
} else {
    $sql = "CREATE TABLE testTable
            (id INT NOT NULL PRIMARY KEY AUTO_INCREMENT,
             testField VARCHAR (75))";
    $res = mysqli_query($mysqli, $sql);

    if ($res === TRUE) {
        echo "Table testTable successfully created.";
    } else {
        printf("Could not create table: %s\n", mysqli_error($mysqli));
    }

    mysqli_close($mysqli);
}
?>
```

Vnos podatkov v bazo

```
<?php
    $mysqli = mysqli_connect("localhost", "root", "", "testDB");

    if (mysqli_connect_errno()) {
        printf("Connect failed: %s\n", mysqli_connect_error());
        exit();
    } else {
        $sql = "INSERT INTO testTable (testField) VALUES ('some value')";
        $res = mysqli_query($mysqli, $sql);

        if ($res === TRUE) {
            echo "A record has been inserted.";
        } else {
            printf("Could not insert record: %s\n", mysqli_error($mysqli));
        }

        mysqli_close($mysqli);
    }
?>
```


Obrazec – vnos podatka v bazo

```
<!DOCTYPE html>
<html>
<head>
<title>Record Insertion Form</title>
</head>
<body>
<form action="insert.php" method="POST">
<p><label for="testfield">Text to Add:</label><br/>
<input type="text" id="testfield" name="testfield" size="30" /></p>
<button type="submit" name="submit" value="insert">Insert Record</button>
</form>
</body>
</html>
```

Text to Add:

```
<?php
$mysqli = mysqli_connect("localhost", "root", "", "testDB");

if (mysqli_connect_errno()) {
    printf("Connect failed: %s\n", mysqli_connect_error());
    exit();
} else {
    $sql = "INSERT INTO testTable (testField) VALUES ('some value')";
    $res = mysqli_query($mysqli, $sql);

    if ($res === TRUE) {
        echo "A record has been inserted.";
    } else {
        printf("Could not insert record: %s\n", mysqli_error($mysqli));
    }

    mysqli_close($mysqli);
}
?>
```

Število vrstic

```
<?php
$mysqli = mysqli_connect("localhost", "root", "", "testDB");

if (mysqli_connect_errno()) {
    printf("Connect failed: %s\n", mysqli_connect_error());
    exit();
} else {
    $sql = "SELECT * FROM testTable";
    $res = mysqli_query($mysqli, $sql);

    if ($res) {
        $number_of_rows = mysqli_num_rows($res);
        printf("Result set has %d rows.\n", $number_of_rows);
    } else {
        printf("Could not retrieve records: %s\n", mysqli_error($mysqli));
    }

    mysqli_free_result($res);
    mysqli_close($mysqli);
}
?>
```

Izpis: Result set has 2 rows.

Pridobitev podatkov

```
<?php
$mysqli = mysqli_connect("localhost", "root", "", "testDB");

if (mysqli_connect_errno()) {
    printf("Connect failed: %s\n", mysqli_connect_error());
    exit();
} else {
    $sql = "SELECT * FROM testTable";
    $res = mysqli_query($mysqli, $sql);

    if ($res) {
        while ($newArray = mysqli_fetch_array($res, MYSQLI_ASSOC)) {
            $id = $newArray['id'];
            $testField = $newArray['testField'];
            echo "The ID is ".$id." and the text is: ".$testField."<br/>";
        }
    } else {
        printf("Could not retrieve records: %s\n", mysqli_error($mysqli));
    }

    mysqli_free_result($res);
    mysqli_close($mysqli);
}
?>
```

Izpis: The ID is 1 and the text is: Mira
 The ID is 2 and the text is: Jaka

Primer 1: Simple Mailing list

- ▶ Kreiranje tabele: id, email
- ▶ Datoteka skupnih funkcij:
 - ▶ Kreiranje in povezava s podatkovno bazo: funkcija doDB()
 - ▶ Preverjanje e-mail naslova v tabeli: funkcija emailChecker(\$email)
- ▶ Kreiranje obrazca za prijavo/odjavo:
- ▶ Mehanizem za sporočanje:

Field	Type	Null	Key	Default	Extra
id	int(11)	NO	PRI	NULL	auto_increment
email	varchar(150)	NO	UNI	NULL	

-
- ▶ Prijava
 - ▶ Odjava

Subscribe/Unsubscribe to a Mailing List

Your Email Address:

Action:

☒ subscribe
☐ unsubscribe

- ▶ Pošiljanje sporočila

Send a Newsletter

Subject:

Mail Body:

Primer 2: Online Address Book

- ▶ Podatkovne tabele
- ▶ Obrazci za dodajanje in brisanje zapisov
- ▶ Skripte za pregled zapisov

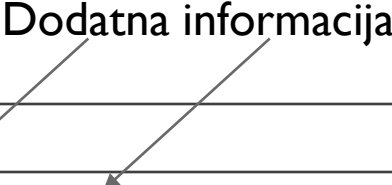


Table Name	Field Names
master_name	id, date added , date modified , f_name, l_name
address	id, master_id, date_added, date_modified, address, city, state, zipcode, type
telephone	id, master_id, date_added, date_modified, tel_number, type
fax	id, master_id, date_added, date_modified, fax_number, type
email	id, master_id, date_added, date_modified, email, type
personal_notes	id, master_id, date_added, date_modified, note

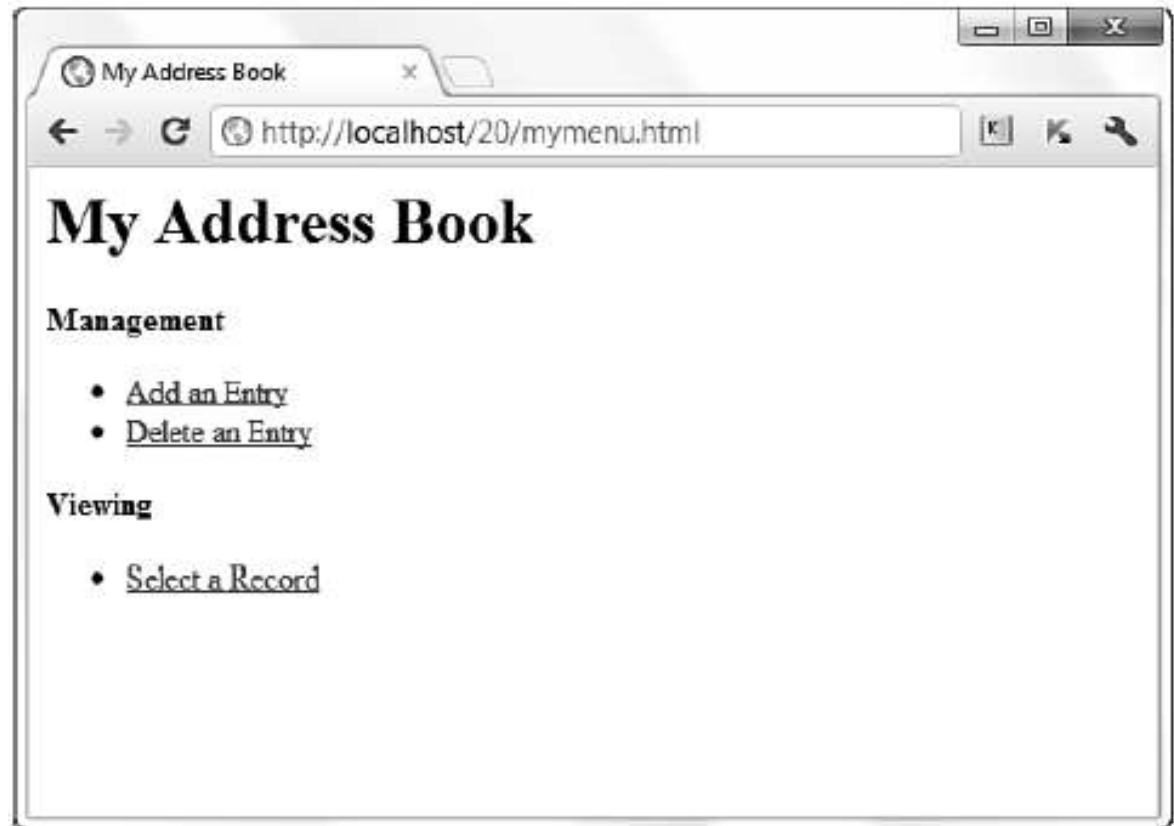
► Tabele:

- master_name,
- address,
- telephone,
- fax,
- email,
- personal_notes

```
CREATE TABLE master_name (  
    id INT NOT NULL PRIMARY KEY AUTO_INCREMENT,  
    date_added DATETIME,  
    date_modified DATETIME,  
    f_name VARCHAR (75),  
    l_name VARCHAR (75)  
);  
CREATE TABLE address (  
    id INT NOT NULL PRIMARY KEY AUTO_INCREMENT,  
    master_id INT NOT NULL,  
    date_added DATETIME,  
    date_modified DATETIME,  
    address VARCHAR (255),  
    city VARCHAR (30),  
    state CHAR (2),  
    zipcode VARCHAR (10),  
    type ENUM ('home', 'work', 'other')  
);
```

```
CREATE TABLE telephone (  
    id INT NOT NULL PRIMARY KEY AUTO_INCREMENT,  
    master_id INT NOT NULL,  
    date_added DATETIME,  
    date_modified DATETIME,  
    tel_number VARCHAR (25),  
    type ENUM ('home', 'work', 'other')  
);  
CREATE TABLE fax (  
    id INT NOT NULL PRIMARY KEY AUTO_INCREMENT,  
    master_id INT NOT NULL,  
    date_added DATETIME,  
    date_modified DATETIME,  
    fax_number VARCHAR (25),  
    type ENUM ('home', 'work', 'other')  
);  
CREATE TABLE email (  
    id INT NOT NULL PRIMARY KEY AUTO_INCREMENT,  
    master_id INT NOT NULL,  
    date_added DATETIME,  
    date_modified DATETIME,  
    email VARCHAR (150),  
    type ENUM ('home', 'work', 'other')  
);  
CREATE TABLE personal_notes (  
    id int NOT NULL PRIMARY KEY AUTO_INCREMENT,  
    master_id INT NOT NULL UNIQUE,  
    date_added DATETIME,  
    date_modified DATETIME,  
    note TEXT  
);
```

► Obrazci za dodajanje in brisanje zapisov



Add an Entry

First/Last Names:

Street Address:

City/State/Zip:

Address Type:
☒ home ☐ work ☐ other

Telephone Number:
 ☒ home ☐ work ☐ other

Fax Number:
 ☒ home ☐ work ☐ other

Email Address:
 ☒ home ☐ work ☐ other

Personal Note:



► Pregled zapisov



XML - eXtensible Markup Language

- ▶ Označevalni jezik za prenos in shranjevanje podatkov
- ▶ Navaden tekst podan z XML značkami

```
<text>
```

```
Pozdrav!
```

```
</text>
```

- ▶ Uporaba XML:
 - Ločuje podatke od HTML - razlike
 - Poenostavlja
 - izmenjavo podatkov
 - prenos podatkov
 - zamenjavo platforme
 - Naredi podatke uporabne
 - Zapis konfiguracijskih datotek
 - Uporabljen za kreiranje novih internetnih jezikov: WSDL, WAP, WML

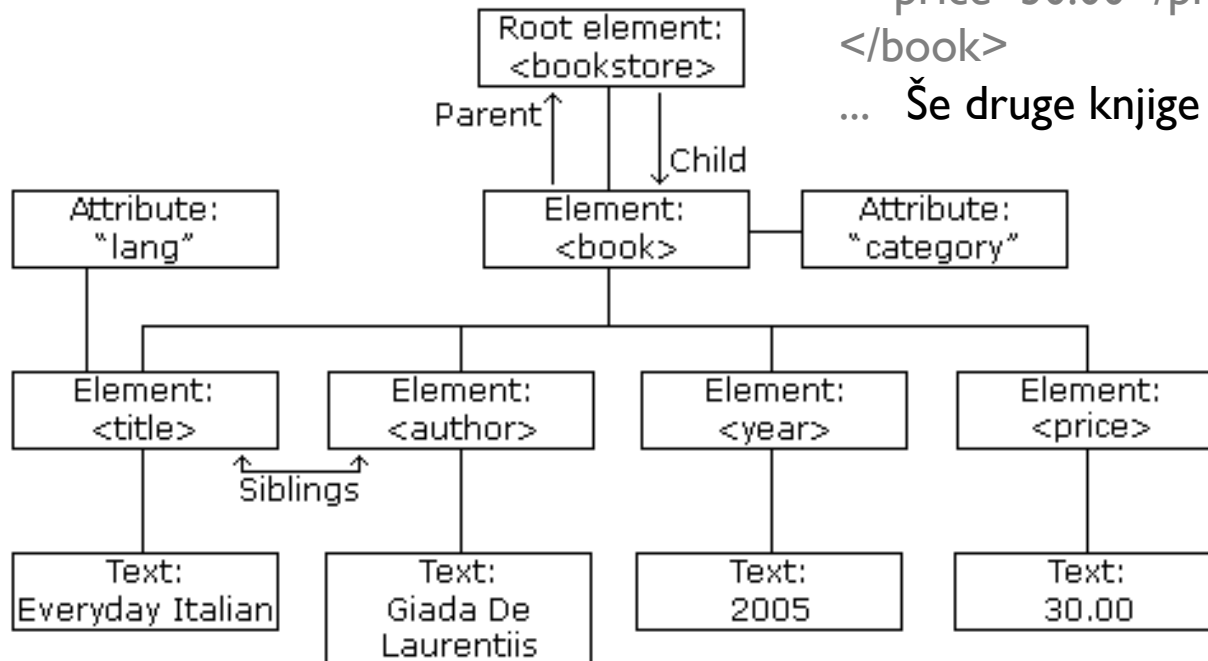
Dokument XML tvori drevesno strukturo

```
<root>
  <child>
    <subchild>....</subchild>
  </child>
</root>
```

Ena knjiga v XML

```
<bookstore>
  <book category="COOKING">
    <title lang="en">Everyday Italian</title>
    <author>Giada De Laurentiis</author>
    <year>2005</year>
    <price>30.00</price>
  </book>
```

... Še druge knjige



Sintaksa dokumenta XML

- ▶ Enostavna, logična pravila
- ▶ Enostaven za učenje in uporabo
- ▶ Dokumenti morajo imeti element root
- ▶ Vsi elementi morajo vedno imeti tudi končno značko
- ▶ Vse značke so 'case sensitive' (<firstname> je različno od <Firstname>)
- ▶ Vsi elementi morajo biti pravilno gnezdeni
- ▶ Entitete referenc: določeni znaki: <, >, &, ', "
- <message>če je znesek < 1000 potem</message>
 - <message>če je znesek < 1000 potem</message>
- ▶ Komentarji:
 - <!-- Tekst -->
- ▶ Presledek – več presledkov se prevede na enega
- ▶ Shranitev nove vrstice kot LF (Line Feed)

Elementi in lastnosti

- ▶ Dokument XML vsebuje element XML
- ▶ Element vključuje (vključno z začetno značko, vsebino, končno značko):
 - druge elemente,
 - tekst,
 - attribute,
 - mešano
- ▶ Prazen element: `<Ime></Ime>` ali `<Ime />`
- ▶ Lastnost: ime = “vrednost”
ime = ‘vrednost’
(enojne ali dvojne navednice)
- ▶ Izogibajte se lastnosti pri zapisu podatkov

```
<person>  
  <sex>female</sex>  
  <firstname>Ana</firstname>  
  <lastname>Skala</lastname>  
</person>
```

```
<person sex="female">  
  <firstname>Ana</firstname>  
  <lastname>Skala</lastname>  
</person>
```

XML Razno

- ▶ XML Namespaces – metoda za preprečitev konflikta imen
 - Potrebna zaradi uporabe različnih datotek XML v aplikacijah
 - `<h:table>` in `<f:table>`
- ▶ XML Encoding – internacionalni znaki
 - `<?xml version="1.0" encoding="UTF-8"?>`
 - UTF-8 je vnaprej izbran, če ni podan v elementu `<xml>`
- ▶ XML Viewing - Prikaz datotek v brskalniku
 - obarvane značke (+ in – označujeta razširitev ali zožitev prikaza elementov)
 - Prikaz ni enak kot pri HTML
- ▶ XML CSS – prikaz elementov v povezavi z oblikovanjem v CSS, ni običajno
- ▶ Validacija XML
 - XML Doctypes – pravilna sintaksa omogoča dobro oblikovan dokument XML
 - XML Validator – preverjanje sintakse, napake zaustavijo delovanje aplikacije
 - XML DTD – definira strukturo za veljavnost dokumenta XML
 - XML Schema – definira strukturo dokumenta XML (alternativa DTD)

XML JS - objekt XMLHttpRequest (1)

- ▶ Brskalniki imajo vgrajen objekt XMLHttpRequest
- ▶ Uporabljen je za izmenjavo podatkov s strežnikom
 - Ažurira spletno stran
 - Zahteva podatke od strežnika, ko je stran naložena
 - Sprejme podatke od strežnika, ko je stran naložena
 - V ozadju pošilja podatke strežniku
- ▶ Kreiranje objekta

```
var xmlhttp=new XMLHttpRequest();
```
- ▶ Pošiljanje zahteve strežniku- dve metodi objekta
 - ▶ `open(method,url,async)`

<i>method</i>	tip zahteve: GET ali POST
<i>url</i>	lokacija datoteke na strežniku
<i>async</i>	true (asinhrono) or false (sinhrono)
 - ▶ `send(string)` – pošlje zahtevo (*string*: uporabljen pri zahtevi POST)
- ▶ Odgovor strežnika – lastnosti objekta:
 - `responseText` – odgovor so podatki kot niz, string
 - `responseXML` - odgovor so podatki XML

XML JS- objekt XMLHttpRequest (2)

► Lastnosti objekta:

- onreadystatechange – shrani funkcijo ali ime funkcije, ki se avtomatsko kliče ob vsaki spremembi lastnosti readyState
- readyState – status objekta:
 - 0: request not initialized
 - 1: server connection established
 - 2: request received
 - 3: processing request
 - 4: request finished and response is ready
- status:
 - 200: "OK"
 - 404: Page not found

► Primer: Ko sta readyState enak 4 in status enak 200, je odgovor pripravljen.

XML JS

► XML Parser

- pretvorba dokumenta XML v objekt XML DOM - uporaba z JavaScript
- pretvorba stringa XML v objekt XML DOM - uporaba z JavaScript

► XML DOM

- Definira standarden način za dostop in uporabo dokumentov XML
- Dokument je viden kot drevesna struktura

► XML v HTML

- Prikaz podatkov XML na spletni strani (v HTML)

► XML application- aplikacija za prikaz podatkov v HTML elementu <div>

```
<?xml version="1.0" encoding="iso-8859-1" ?>
<!-- Edited by XMLspyO
-->
<CATALOG>
<CD>
<TITLE>Empire Burlesque</TITLE>
<ARTIST>Bob Dylan</ARTIST>
<COUNTRY>USA</COUNTRY>
<COMPANY>Columbia</COMPANY>
<PRICE>10.90</PRICE>
<YEAR>1985</YEAR>
</CD>
<CD>
<TITLE>Hide your heart</TITLE>
<ARTIST>Bonnie Tyler</ARTIST>
```

```
xmlDoc.load("cd_catalog.xml");
document.write("<table border='1'>");
var x=xmlDoc.getElementsByTagName("cd");
for (var i=0; i<x.length; i++)
{
    document.write("<tr>");
    document.write("<td>");
    document.write(x[i].getElementsByTagName("artist") [0].childNodes[0].nodeValue);
    document.write("</td>");
```

Bob Dylan	Empire Burlesque
Bonnie Tyler	Hide your heart
Dolly Parton	Greatest Hits
Garv Moore	Still got the blues

Sintaksa dokumenta XML

► Primer 5: knjige_catalog.xml, knjige_catalog.html

```
<!-- Edited by XMLSpy@
-->
▼<bookstore>
  ▼<book category="CHILDREN">
    <title>Harry Potter</title>
    <author>J K. Rowling</author>
    <year>2005</year>
    <price>29.99</price>
  </book>
  ▼<book category="WEB">
    <title>Learning XML</title>
    <author>Erik T. Ray</author>
    <year>2003</year>
    <price>39.95</price>
  </book>
  ▼<book category="Cooking">
    <title>15 minutes Meals</title>
    <author>Jamie Oliver</author>
    <year>2012</year>
    <price>20.90</price>
  </book>
</bookstore>
```

J K. Rowling	Harry Potter
Erik T. Ray	Learning XML
Jamie Oliver	15 minutes Meals

AJAX – Asynchronous JavaScript in XML

- ▶ Je oblika izmenjave podatkov s strežnikom
- ▶ Posodobitev dela spletne strani brez ponovnega nalaganja cele strani.
- ▶ Zasnovan je na obstoječih standardih in uporablja:
 - Objekt XMLHttpRequest (asinhronska izmenjava podatkov s strežnikom)
 - JavaScript/DOM (za prikaz in interakcijo z informacijo)
 - CSS (za oblikovanje podatkov)
 - XML (format za prenos podatkov)



AJAX = Asynchronous JavaScript and XML.

AJAX is not a new programming language, but a new way to use existing standards.

AJAX is the art of exchanging data with a server, and updating parts of a web page - without reloading the whole page.

Brskalnik

Pojavi se dogodek
Ustvari se objekt XMLHttpRequest
Pošlje se zahteva HttpRequest

Internet

Strežnik

Obdelava zahteve
HttpRequest

Ustvari se odgovor in
podatki se pošljejo
brskalniku

Internet

Brskalnik

Obdelava vrnjenih podatkov z
JavaScript
Posodobitev vsebine strani

JSON - JavaScript Object Notation

- ▶ Sintaksa za shranjevanje in izmenjavo tekstovne informacije.
- ▶ Uporablja sintakso JavaScript za opis podatkovnih objektov
- ▶ Obstajajo JSON razčlenjevalniki in knjižnice
- ▶ Navaden tekst, enostavno berljiv
- ▶ Lahko je razčlenjen z JavaScript
- ▶ Se lahko prenaša z uporabo AJAX
- ▶ Je manjši kot XML, hitrejši in enostavnejši za razčlenjevanje.
- ▶ XML
 - Dostava dokumenta XML
 - Uporaba XML DOM za pregled dokumenta
 - Izpis vrednosti in shranitev v spremenljivke
- ▶ JSON
 - Dostava stringa JSON
 - Funkcija eval() za izvajanje podatkov JSON

Sintaksa JSON

- ▶ Podatki so v paru Ime/Vrednost, ločeni so z vejico

- ▶ Primer:

JSON: "Ime":"Janez"

(v JavaScript: ime="Janez")

- ▶ Vrednost JSON je:

- Število (integer ali floating point)
- Niz (v dvojnih navednicah)
- Boolova vrednost (true or false)
- Polje (v oglatih oklepajih)
- Objekt (v zavutih oklepajih)
- null

- ▶ Objekt – zaviti oklepaji : { "Ime":"Janez", "Priimek":"Novak" }

- ▶ Polje – oglati oklepaji (objekt zaposleni je polje dveh objektov)

```
{"zaposleni": [  
  { "firstName":"Jaka" , "lastName":"Duh" },  
  { "firstName":"Ana" , "lastName":",Prah" },  
]}
```

-
- ▶ Pridobivanje podatkov (iz prejšnjega primera: Jaka Duh)
`zaposleni[0].firstName + " " + zaposleni[0].lastName;`
 - ▶ Spreminjanje podatka:
`zaposleni[0].firstName = "Miha";`

- ▶ Primer objekta:

Kreiranje objekta JSON v JavaScript

Ime in priimek: Jaka Prah
Starost: 33
Naslov: Ljubljanska 16
Tel: 01 1234567

Kreiranje objekta JavaScript

Ime: Jaka
Priimek: Novak
Starost: 55

- ▶ Datoteke JSON
 - ▶ Tip datoteke JSON je ".json"
 - ▶ Tip MIME za JSON tekst je "application/json"

Pretvorba teksta JSON v objekt JavaScript

- ▶ String JavaScript vsebuje sintakso JSON

```
var txt = '{ "zaposleni" : [' +  
  '{ "firstName":"Jaka" , "lastName":"Duh" },' +  
  '{ "firstName":"Ana" , "lastName":"Prah" } ]}';
```

- ▶ funkcija eval() uporabi JavaScript prevajalnik – razčlenitev teksta JSON v objekt JavaScript

```
var obj = eval ("(" + txt + ")");
```

- ▶ Primer:

Kreiranje objekta iz stringa JSON

Ime: Jaka
Priimek: Duh

- ▶ Razčlenjevalnik JSON
 - Podpora v brskalnikih (firefox 3.5, IE 8, Chrome, ..)
 - Prepozna samo tekst JSON
 - Hitrejši kot uporaba funkcije eval()