Zvyšovanie efektívnosti školských vzdelávacích programov Spojenej školy v Kysuckom Novom Meste





Moderné vzdelávanie pre vedomostnú spoločnosť/Projekt je spolufinancovaný zo zdrojov EU

predmet: **Informačné technológie**

ročník: **tretí**

odbor: **Elektrotechnika –Počítačové systémy**

autor: **Ing. Anton Maslák**





Spojená škola
Nábrežná 1325
024 01 Kysucké Nové Mesto

Zvyšovanie efektívnosti školských vzdelávacích programov Spojenej školy v Kysuckom Novom Meste







Moderné vzdelávanie pre vedomostnú spoločnosť/Projekt je spolufinancovaný zo zdrojov EU

autor: **Ing. Anton Maslák**

grafické spracovanie: **Ing. Anton Maslák**

rok vydania: **2011**

V knihe sú použité názvy programových produktov a firiem, ktoré môžu byť ochrannými známkami alebo registrovanými ochrannými známkami príslušných vlastníkov.

Učebné texty pre žiakov stredných škôl vydané pre potreby školy

OBSAH

PREDHOVOR....5
1. ÚVOD DO DATABÁZ....6
1.1 História databáz....6
1.2 História jazyka SQL....6
1.3 Delenie databáz....6
1.4 Základné pojmy....7
1.5 Vzťahy (relationship) medzi tabuľkami....9
2. NÁVRH RELAČNEJ DATABÁZY PROCESOM NORMALIZÁCIE....10
2.1 1 normálna forma (1NF)....10
2.2 2 normálna forma (2NF)....11
2.3 3 normálna forma (3NF)....11
3. INDEXY....11
3.1 Stlpce, ktoré by mali byť indexované....12
3. 2 Stĺpce ktoré by sa nemali byť indexované....12
4. SERVER....12
4.1 IIS, ASP, MS SQL Server....12
4. 2 LAMP....12
4.3 Linux....13
4.4 Apache....13
4.5 MySQL....14
4.6 Vlastnosti MySQL....14
4.7 PHP....15
Výhody....16
4.8 Perl....17
4.9 Administrátorské prostredie phpMyAdmin....17
5. XAMPP....17
5.1 Vytvorenie MySQL databázy pre potrebné dáta....18
6. DOTAZOVACÍ JAZYK SQL....19
6.1 Komentáre....19
6.2 Dátové typy....19
6.3 Založenie novej databázy a tabuľky....20
6.4 Vkladanie dát....21
6.5 Zrušenie databázy, tabuľky a úprava dát....22
6.6 Výber dát z databázy....23
6.7 Spájanie tabuliek....27
7. VÝPOČTY A FUNKCIE....29
7.1 Agregačné funkcie....29
7.2 Matematické funkcie....29
7.3 Funkcie pre prácu s reťazcami....29
7.4 Funkcie pre prácu s dátumom a časom....30
8. ADMINISTRÁCIA POUŽÍVATEĽSKÝCH PRÁV....30
9. PHP A DATABÁZA MYSQL....31
9.1 Pridanie záznamu do tabuľky....32
9.2 Výber záznamu z tabuľky....32
10. APLIKAČNÉ ROZHRANIE -TENKÝ A TLSTÝ KLIENT....33
11. INTERNET....33

11.1 Architektúra klient/server 34
11.2 Doména 34
11.3 Registrácia domény 34
11.4 Stav domény 35
11.5 Webhosting 35
11.6 Typy webhostingu 36
12. ZÁSADY TVORBY WWW STRÁNOK 36
13. HTML 37
13.1 XHTML 38
13.2 Rozdiely medzi HTML a XHTML 39
13.3 Štruktúra HTML 39
13.4 Zápis textu 41
13.5 Formátovanie textu 41
13.6 Obrázky a pozadie 42
Použitie obrázkov 42
13.7 Odkazy 42
13.8 Zoznamy 43
13.9 Tabuľky 44
13.11 Formuláre 45
13.12 Farby 47
14. WEBOVÉ ŠTANDARDY 47
15. SEO - OPTIMALIZÁCIA PRE VYHĽADÁVAČE 48
15.1 Predpoklady pre správnu optimalizáciu 49
15.2 Vyhľadávače 49
15. 3 Katalógy 50
15.4 Optimalizácia webových stránok 50
15.5 On page faktory 51
15.6 Off page faktory 52
15.7 SEO nástroje 53
16. CSS 54
16.1 Zapisovanie CSS 55
16.2 Triedy, identifikátory a komentáre 55
16.3 Formátovanie písma 57
16.4 Pseudotriedy 57
16.5 Pozadia 57
17. ADOBE FLASH 58
18. JAVASCRIPT 59
18.1 JS Frameworky 59
19. WEBOVÉ ŠABLÓNY 59
19.1 Generátory šablón 60
20. CMS SYSTÉM (REDAKČNÝ SYSTÉM) 60
20.1 Moduly pre CMS Systémy 61
20.2 Moodle 61
20.3 WordPress 62
20.4 Joomla 62
20.5 phpBB 62
20.6 Piwigo 62
20.7 Drupal 62
21. CHARAKTERISTIKA INTERNETOVÉHO OBCHODU 63
21.1 PrestaShop 63
21.2 Porovnanie open source e-shop systémov 63
22. LITERATÚRA 65

Predhovor

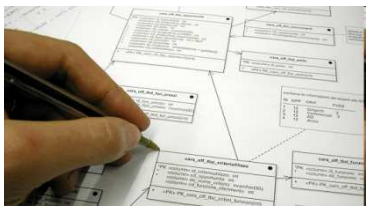
Učebný text pre predmet Informačné technológie, nadväzuje na ďalšie predmety odborného zamerania elektronické počítačové systémy. Je spracovaný v nadväznosti na konkrétne ciele, metódy a formy vzdelávania podľa tematického plánu predmetu. Vzhľadom na veľký zaber predmetu Informačné technológie je problematika, ktorá je obsiahnutá v iných predmetoch spomínaná v učebnom texte len okrajovo. Rovnako tiež, v rámci predpísaného rozsahu pre učebný text, sú jednotlivé kapitoly doplnené o hypertextové odkazy, ktoré obsahujú podrobnejšie spracovanie danej problematiky. Pri vyučovaní sa kladie dôraz na aktívnu prácu žiakov v skupine, vyučovanie je organizované formou 2-hodinových praktických cvičení s počítačmi, so sieťou a s komunikačnou technikou. Trieda sa delí na dve skupiny, ktoré sa po pol roku medzi sebou vymenia. Cieľom (úlohou) pri prvej skupine, je poskytnúť žiakom a dosiahnuť u žiakov, potrebné vedomosti pri práci s databázou a skriptovacím jazykom. Cieľom pri druhej skupine, je oboznámiť žiakov s tvorbou www stránok. Z tohto dôvodu, niektoré kapitoly na seba horšie nadväzujú, zámerne sú členené podľa polrokov pre jednotlivé skupiny. Kapitoly 1 až 10 sú určené pre prvú skupinu a kapitoly 11 až 21 pre skupinu druhú.

Túto formu prezentovania danej problematiky som robil s pocitom, že nájde svojich záujemcov a moje úsilie o ponúknutie nových vedomostí tak bude naplnené. Pri používaní tohto učebného textu vám prajem pozitívne myslenie, pohodu a vlastné presvedčenie, že to bol zmysluplne strávený čas.

Ing. Anton Maslák

1. Úvod do databáz

1.1 História databáz



Prvý databázový riadiaci systém bol vyvinutý v 60. rokoch. Priekopníkom na tomto poli bol *Charles Bachman*. Dovtedy bolo spracovávanie dát založené na dierovacích kartách a magnetických páskach. V rámci konzorcia CODASYL bol vyvinutý sieťový model založený na *Bachmanových* myšlienkach a takmer súčasne bol pri systémovom vývoji v NORTH AMERICAN ROCKWELL použitý hierarchický model, neskôr adoptovaný spoločnosťou IBM ako základný kameň ich produktu IMS. *E.F. Codd* kritizoval existujúci model pre mätúci abstraktný opis informácií. *Relačné databázové systémy* sú postavené na základe definície relačného modelu, ktorý v 1970 zaviedol americký matematik *E.F.Codd* a spolu s relačným systémom riadenia bázy dát sa stal najpoužívanejším *databázovým systémom*. Relačný dátový model vychádzal z nasledovných požiadaviek:

1. zabezpečiť vysoký stupeň dátovej nezávislosti
2. zabezpečiť minimálnu redundanciu dát spolu s konzistenciou podporovanou sémantikou jazyka
3. sprístupnenie databázy pomocou množinovo orientovaného neprocedurálneho jazyka
4. umožnenie jednoduchým spôsobom reštrukturalizáciu (*zmena štruktúry tabuliek*) a rast dátového modelu

Produkty CODASYLu (*IDMS*) a produkty sieťových modelov (*IMS*) boli koncipované ako čiastočné inžinierske riešenia. Prvý komerčný produkt, ORACLE a DB2, sa objavil až v roku 1980. Prvý úspešný databázový produkt pre mikropočítače bol dBASE pre operačné systémy CP/M a PC-DOS/MS-DOS. Počas 80. rokov bola aktivita výskumu zameraná na distribuované databázové systémy a databázové stroje, ale tieto vývojové trendy mali iba malý efekt na trh. V 90. rokoch sa pozornosť upriamila na objektovo orientované databázy. Tieto mali úspech v odvetviach, kde bolo nutné vlastniť komplexnejšie dáta ako mohol relačný systém jednoducho pokryť, napríklad priestorové databázy, inžinierske dáta a multimediálne dáta. Niektoré z týchto myšlienok boli prijaté relačnými obchodníkmi, ktorí začlenili nove možnosti do ich produktov ako výsledok. V 90. rokoch bolo taktiež vidno rozširovanie *open source* databáz ako PostgreSQL a MySQL. V treťom tisícročí je XML database modernou oblasťou pre inováciu. XML databázy majú za cieľ vymazať tradičný rozdiel medzi dokumentmi a dátami.

1.2 História jazyka SQL

Structured Query Language (*SQL*) je počítačový jazyk na manipuláciu a definíciu dát. V súčasnosti je to najpoužívanejší jazyk tohto druhu v relačných systémoch riadenia báz dát. Jazyk SQL sa vyvinul pôvodne z jazyka SEQUEL, čo bol jazyk pre manipuláciu a definíciu dát pre System R od IBM, prvého databázového systému založenom na *Coddovom* relačnom dátovom modeli. Prvý komerčný databázový systém využívajúci SQL vyvinula spoločnosť *Relational Software, Inc.* (*teraz Oracle*) v roku 1979. V roku 1986 inštitúcia ANSI štandardizovala jazyk SQL pod označením SQL-86. O rok neskôr vo viac-menej nezmenenej forme tento štandard ratifikovala aj organizácia ISO (*SQL-87*). V roku 1989 a 1992 boli prijaté revízie označené ako SQL-89, resp. SQL-92 (*aj SQL-2*). Výrazné zmeny prišli v roku 1999, kedy sa do štandardu SQL:1999 (*SQL-3*) dostali napr. rekurzívne dopyty, spúšťače (*triggery*), regulárne výrazy, neskalárne dátové typy a niektoré objektovo orientované vlastnosti. Verzia SQL:2003 priniesla hlavne podporu pre XML, štandardizované postupnosti, stĺpce s automaticky generovanými hodnotami a tzv. oknové funkcie.

1.3 Delenie databáz

Databázy sa dajú rozdeliť (*najmenej*) podľa dvoch významných kritérií.

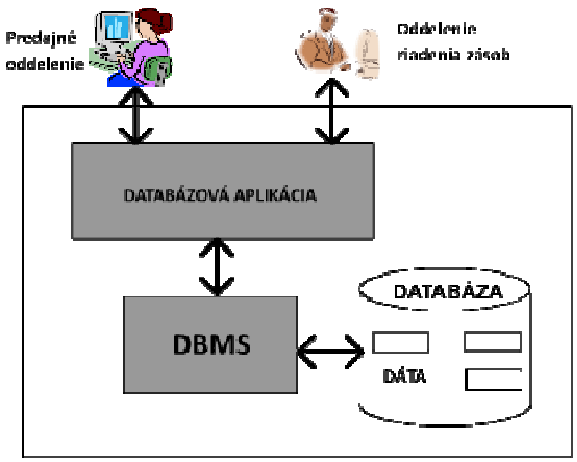
Súborové a systémové

Súborové
Databáza môže byť obsiahnutá v jednom alebo niekoľkých súboroch operačného systému. Potom im hovoríme súborové. Databázu prenesieme na iný stroj jednoduchým prekopírovaním príslušných súborov. Typickým zástupcom súborových databáz sú databázy dbf (*FoxPro*), SQLite alebo trebárs Microsoft Access. Súborové databázy mávajú niekoľko významných obmedzení, hlavne je to ich dostupnosť v rámci siete alebo rôzna úroveň podpory súbežnej práce viac používateľov.

Systémové
Systémové databázy slúžia ako databázové servery. Väčšinou majú veľmi dobrú podporu súbežnej práce viacerých používateľov a bývajú prístupné pomocou nejakého sieťového protokolu (*TCP*). Systémové databázy mávajú vo všeobecnosti zložitejšiu inštaláciu, bývajú sofistikovanejšie a robustnejšie. Medzi typických zástupcov patrí napríklad MySQL, PostgreSQL, Oracle alebo Microsoft SQL server. Všeobecne sa môžete pripojiť k systémovej databáze i v prípade, že na systém, na ktorom beží, nemáte inak prístup.
Poznámka: V poslednej dobe sa významový rozdiel medzi súborovými a systémovými databázami dosť stiera. Existuje totiž niekoľko spôsobov, ako pomocou nejakého softvéru pristupovať k súborovým databázam, ako by to boli databázy systémové. Napríklad pomocou UnixODBC alebo ODBC môžeme pristupovať k databáze dbf z počítačovej siete a podobne.

Objektové a relačné
Objektové
Pomerne podstatné je rozdelenie databáz podľa filozofie uloženia dát. Menej obvyklý spôsob je organizovať dáta v databáze ako objekty. Objektové databázy nie sú príliš rozšírené, zástupcom by mohla byť Caché.
Relačné
Prakticky všetky dnešné databázy sú relačné. Dáta v relačných databázach sú (*alebo aspoň môžu byť*) uložené tak, že súvisiace záznamy môžeme jednoducho vyhľadať. Napríklad v databáze podniku môžeme k pracovníkom rýchlo nájsť šéfov, platové skupiny alebo záznamy o príchodoch na pracovisko.

1.4 Základné pojmy



DÁTA
Surové nespracované fakty, ktoré majú určitú dôležitosť pre jednotlivca alebo organizáciu.

INFORMÁCIE
Dáta ktoré prešli spracovaním alebo dostali štruktúru, ktorá im dáva pre jednotlivca alebo organizáciu význam.

DATABÁZA
Databáza je ucelené úložisko dát, ktoré môžu byť používané súčasne mnohými oddeleniami a používateľmi. Všetky dáta, ktoré používatelia požadujú sú integrované s minimálnym množstvom duplikácií. Dôležité je, že databázu obvykle nevlastní žiadne oddelenie alebo používateľ, ale je zdieľaným zdrojom spoločnosti. Okrem toho, že databáza obsahuje prevádzkové dáta organizácie, obsahuje tiež popis týchto dát. Z tohto dôvodu je databáza definovaná taktiež ako seba popisujúca kolekcia integrovaných záznamov.

DBMS
Database Management System je softvérový systém, ktorý používateľovi umožňuje definovať, vytvárať a udržiavať databázu a poskytuje riadený prístup k tejto databáze.

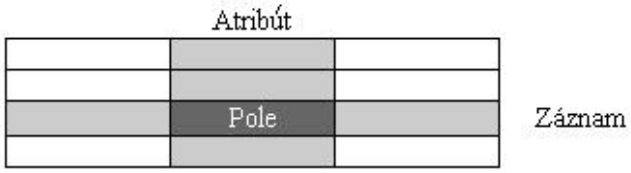
SQL
Structured Query Language je hlavný dotazovací jazyk pre relačné DBMS.

DATABÁZOVÁ APLIKÁCIA
Počítačový program interagujúci s databázou vyvolaním odpovedajúceho požiadavku. (*typicky jedného alebo viacerých príkazov SQL*) pre DBMS.

RELAČNÁ DATABÁZA
Teóriu relačných databáz zjednodušene popisujú tzv. minimálne podmienky relačnosti, kam patrí:

- všetky údaje v databázach sú uložené v tabuľkách
- štruktúra a uloženie fyzických dát je nezávislé na používateľovi
- pre prácu s údajmi v databázy existuje databázový jazyk umožňujúci realizáciu operácií nad týmito dátami

Relačný model definuje predovšetkým štruktúru uložených dát. Ďalej sleduje spôsob ochrany ukladaných dát, tzv. integritu dát, a samozrejme spôsob manipulácie s dátami.
Dáta sú v relačnom modeli ukladané prostredníctvom tabuliek. Riadky predstavujú jednotlivé uložené dátové položky, stĺpce reprezentujú vlastnosti takto uložených položiek. Poľom tabuľky nazývame priesečník riadku a stĺpca tabuľky a to je určené pre uchovávanú hodnotu.



Tabuľky (*objekty*)
Databáza sa skladá z tabuliek (*objektov*). V tabuľkách sú uložené informácie, ktoré spolu nejakým spôsobom súvisia. Napríklad v tabuľke žiak budú informácie o žiakoch.
Stĺpce (*atribút*)
V stĺpcoch sú popísané vlastnosti objektu. Objekt žiak môže mať stĺpce meno, priezvisko, trieda.
Záznamy (*riadky*)
Záznamy sú už konkrétne dáta uložené v databáze. Záznam v tabuľke žiak by mohol vyzerať napr. takto:

Tabuľka ' žiak '

meno	priezvisko	trieda
Fero	Mrkva	3.BE
Peter	Novák	3.AE

Dotazy
Aby sme získali napr. zoznam žiakov zoradených podľa triedy, vytvoríme dotaz.

PRIMÁRNY KĽÚČ (*PRIMARY KEY*)
Primárny kľúč je jednoznačný identifikátor v tabuľke reprezentovaný stĺpcom alebo skupinou stĺpcov a slúži k jednoznačnej identifikácii každého riadku tabuľky. Hodnota poľa primárneho kľúča musí byť v rámci tabuľky jedinečná. Pole primárneho kľúča musí obsahovať konkrétnu hodnotu, inými slovami hodnota primárneho kľúča nesmie byť NULL. Pre tento účel sa často používa jednoduché indexové číslo (*ID záznamu*).

Tabuľka ' žiak '

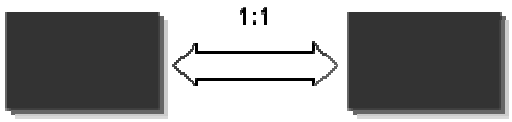
ID_ziaka	meno	priezvisko	trieda
1	Fero	Mrkva	3.BE
2	Peter	Novák	3.AE

CUDZÍ KĽÚČ (*FOREIGN KEY*)
Cudzí kľúč je stĺpec alebo množina stĺpcov dodaná do tabuľky za účelom sprostredkovania väzby s inou tabuľkou databázy. Cudzí kľúč nadobúda hodnoty primárnej kľúča druhej tabuľky a tým jednoznačne identifikuje vzťah medzi jednotlivými riadkami týchto tabuliek.

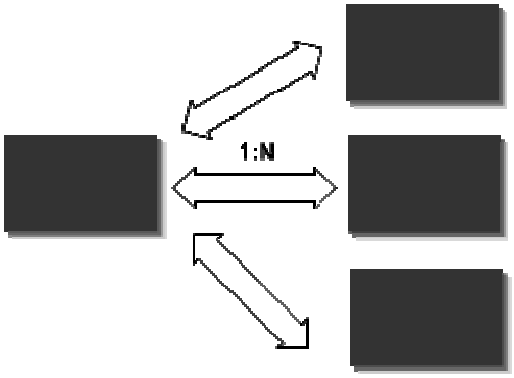
INDEXOVACÍ KĽÚČ (*KEY*)
Väčšina súčasných databáz využíva proces indexovania na zrýchlenie vyhľadávania. Ak chceme rýchlo prehľadávať tabuľku podľa určitého stĺpca označíme tento stĺpec ako indexovací kľúč. Najbežnejším indexom je utriedený obsah indexovaných stĺpcov tabuľky s ukazovateľom na riadok samotný. Index nad stĺpcom alebo stĺpcami tabuľky je samostatným objektom, ktorý si SRBD spravuje, aktualizuje a využíva sám. Používateľ priamo s indexom nepracuje, len ho necháva vytvárať alebo opäť odstrániť.

1.5 Vzťahy (*relationships*) medzi tabuľkami

Relácia 1:1 (*one-to-one*)
Vyjadruje vzťah, keď práve jeden záznam má vzťah k práve jednému jedinému inému unikátnemu záznamu. Napríklad každý jeden unikátny žiak má jedno unikátne rodné číslo. Táto väzba nie je často využívaná pretože sa väčšinou dáta s takouto väzbou umiestňujú v rámci jednej tabuľky.

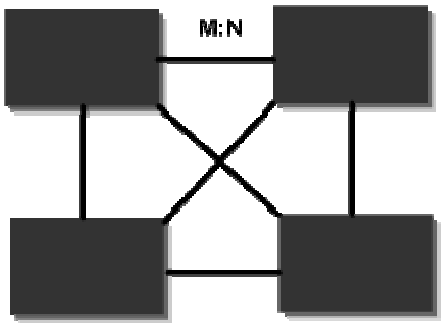


Relácia 1:N (*one-to-many*)
Ide o najčastejšie používanú väzbu. Atribút v tabuľke môže v tomto prípade nadobúdať práve jednu hodnotu z množiny hodnôt definovaných v tabuľke druhej. Napríklad vezmime vzťah žiaka a jeho rodného mesta. Miest je veľa, ale každy sa narodil iba v jednom z nich. Rovnakú väzbu, ale opačne, plní väzba veľa na jedného many-to-one (*N:1*).



Relácia M:N (*many-to-many*)

Pekným príkladom tejto väzby je vzťah čitateľov a novín. Každé noviny majú veľa čitateľov a každý z nich môže byť súčasne čitateľom niekoľkých rôznych novín. V praxi nejde v SQL tento vzťah vyjadriť priamo a používa sa preto "medzi-tabuľka" s väzbami 1:N na požadované tabuľky. Napríklad pre tabuľku noviny a čitateľ vytvoríme tabuľku čitateľ_novín, kde jeden stĺpec bude odkazovať na tabuľku čitateľov a druhý na tabuľku novín. V každom riadku tejto tabuľky teda bude záznam, ktorý bude vyjadrovať, že daný čitateľ číta dané noviny. Jeden čitateľ môže mať v tabuľke viac záznamov (*riadkov*), ale žiadne noviny sa nesmú pre toho istého čitateľa opakovať dvakrát - každý záznam musí byť unikátny. Takto vytvorená tabuľka môže dať odpoveď, kto číta aké noviny, a naopak, kto je čitateľom tých-ktorých novín.



2. Návrh relačnej databázy procesom normalizácie

Normalizácia databázy je dôležitý krok pri návrhu, vývoji a implementovaní nového informačného systému, ako aj spätnej analýzy existujúceho návrhu informačného systému. Pojem "normalizácia" súvisí so snahami databázových teoretikov špecifikovať pravidlá na dosiahnutie efektívnej databázovej schémy tabuliek. Normalizácia databázy sa realizuje kvôli niekoľkým dôvodom:

- Odstránenie tzv. anomálií pri modifikácii dát v tabuľkách
- Zníženie nutnosti redizajnu schémy databázy pri jej rozširovaní
- Sprehľadnenie schémy databázy pre používateľov
- Oddelenie schémy databázy od dopytov, ktoré sa nad ňou budú vykonávať

Správny návrh normalizovaných tabuliek súčasne s dobrým výberom indexovania tabuliek (*vhodne riešeným systémom prístupových klúčov*) vedú k vyšším výkonom aplikácie a jednoduchšej správe databázovej aplikácie.

Existujú 3 bežne používané stupne normalizácie, z ktorých každý vedie k vyššiemu stupňu normalizácie. Proces normalizácie rozdeľuje tabuľku na viaceré tabuľky, čím môže v určitých prípadoch znížiť výkonnosť databázy, preto existuje aj opačný proces nazývaný denormalizácia.

2.1 1. normálna forma (*1NF*)

Relácia (*tabuľka*) je v prvej normálnej forme, pokiaľ každý jej atribút (*stĺpec*) obsahuje len atomické hodnoty, t.j. hodnoty, ktoré z pohľadu databázy už ďalej nemožno deliť na menšie. Táto podmienka nie je splnená, ak napríklad máme meno a priezvisko v jednom stĺpci pričom aplikácia pracuje s týmito položkami ako samostatnými.

Riešenie: Treba vytvoriť nový atribút pre každú zloženú premennú

Osoba	Tel_cislo
Peter Novák	0905151122, 0484212211

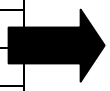
Meno	Priezvisko	mobil	Tel_cislo
Peter	Novák	0905151122	0484212211

2.2 2. normálna forma (*2NF*)

Relácia sa nachádza v druhej normálnej forme, pokiaľ spĺňa podmienky 1NF a každý nekľúčový atribút je úplne funkčne závislý od primárneho kľúča, a to od celého kľúča a nie od ľubovoľnej jeho podmnožiny v prípade zloženého primárneho kľúča.

Riešenie: Rozložte tabuľku a vytvorte novú tabuľku pre každú zložku primárneho kľúča.

Tovar	**Dodavatel**	**Email**	**Cena**
Cement	Pohronsky Ruskov	cement@ruskov.sk	300
Cement	Stavebniny Zilina	stav@zilina.sk	350
Piesok	Liptovsky Mikulas	piesok@lm.sk	95
piesok	Stavebniny Zilina	stav@zilina.sk	100



Tabuľka nespĺňa podmienky 2NF, preto ju musíme rozdeliť na 2 tabuľky

Kod	**nazov**	**Email**
0070	Pohronsky Ruskov	cement@ruskov.sk
0040	Zilina	stav@zilina.sk
0020	Liptovsky Mikulas	piesok@l.sk

Nazov	**Kod_dod**	**Cena**
Cement	0070	300
Cement	0040	350
Piesok	0020	95
Piesok	0040	100

2.3 3. normálna forma (*3NF*)

V tejto forme sa nachádza tabuľka ak spĺňa podmienky 1NF a 2NF a všetky jej nekľúčové atribúty sú vzájomne nezávislé. Tabuľka nemá mať nekľúčový atribút závislý od iného nekľúčového atribútu (*alebo ich kombinácie*).

Riešenie: Napríklad ak množina obsahuje Jednotkovú cenu a množstvo, potom by nemala obsahovať cenu celkom, ktorá sa vypočíta na základe rovnice vloženej do aplikácie: CenaCelkom = JednotkováCena * Množstvo.

Číslo	**Priezvisko**	**Meno**	**Funkcia**	**Plat**
1	Novák	Peter	učiteľ	600
2	Mrkva	Fero	technik	500

Číslo	**Priezvisko**	**Meno**	**Funkcia**
1	Novák	Peter	učiteľ
2	Mrkva	Fero	technik

Funkcia	**Plat**
učiteľ	600
technik	500

Existujú normálne formy vyššie než 3NF, napríklad *Boyce-Coddova* normálna forma (*BCNF*), štvrtá normálna forma (*4NF*) až šiesta normálna forma (*6NF*). Tieto pokročilé normálne formy sa obvykle nepoužívajú, pretože identifikujú a riešia málo frekventované problémy. Kam až normalizovať, záleží na konkrétnej databáze. Normalizácia je tu preto, aby urobila databázu prehľadnejšou, ľahšie rozšíriteľnou a výkonnejšou. Je preto zbytočné normalizovať, keď normalizácia neprinesie zvýšenie prehľadnosti, rozšíriteľnosti alebo výkonu. Zbytočne veľký počet objektov môže naopak urobiť databázu neprehľadnou.

3. Indexy

Index je mechanizmus, ktorý významne urýchľuje vyhľadanie riadka tabuľky s konkrétnou hodnotou daných atribútov. Prirodzene je preto užitočný najmä pre tie stĺpce, na ktoré sa odvoláva klauzula WHERE v dotazovom príkaze SELECT jazyka SQL. Aby index pracoval efektívne, musí byť obnovený zakaždým keď je záznam pridaný, zmazaný, alebo

modifikovaný. Indexy zvyšujú výkon prehľadávania tabuliek a spájania tabuliek medzi sebou. Na druhej strane dokážu zároveň spomaliť čas vkladania, mazania a modifikácie záznamov v tabuľke, preto je veľmi dôležité vytvoriť vhodné indexy. Každý index spomaľuje operácie, ktoré menia obsah indexovaných stlpcov (*napríklad SQL príkazy UPDATE, INSERT, DELETE*). To je dané tým, že databáza se v prípade takejto operácie nad indexovaným stlpcom musí postarať nie len o zmeny v dátach tabuľky, ale i o zmeny v dátach indexov.

3.1 Stlpce, ktoré by mali byť indexované

- Primárne kľúče - všetky stĺpce, ktoré sú súčasťou primárneho kľúča
- Cudzie kľúče - všetky stĺpce ktoré odkazujú na primárne kľúče iných tabuliek
- Stĺpce, ktoré sa používajú často v klauzulách WHERE, a ktoré majú početné hodnoty
- Stĺpce, ktoré sa používajú v klauzulách ORDER BY

3. 2 Stĺpce, ktoré by nemali byť indexované

- Stĺpce, ktoré majú len veľmi málo možných hodnôt
- Stĺpce zriedkavo využívané v klauzulách WHERE alebo ORDER BY

4. Server

4.1 IIS, ASP, MS SQL Server



Microsoft Internet Information Services (***IIS***) je balík vybraných serverových funkcií Microsoft Windows. Vo webových službách je IIS druhým najpoužívanejším serverom, hneď za Apache. Prieskumy uvádzajú, že na IIS beží približne 37.13% všetkých webových stránok. IIS sa v súčasnosti využíva predovšetkým pre jeho podporu **ASP.NET**. ASP je technológia nezávislá na programovacom jazyku (*vyvinutá spoločnosťou Microsoft*), ktorá umožňuje vykonávanie kódu na strane servera a následné odoslanie výsledku používateľovi. **Microsoft SQL Server** je relačný databázový systém (*relational model database server*). Jeho hlavné dotazovacie jazyky sú T-SQL a ANSI SQL. Je využiteľný pre desktop aplikácie, *enterprise* aplikácie aj webovo-založené databázové systémy. Je priamo integrovaný s používateľskými kontami a bezpečnostnými prvkami operačných systémov Windows. Pokrýva celú škálu hardvérových platforiem pre tento operačný systém od clusterových serverov po prenosné a vreckové počítače s operačným systémom Windows Mobile. Najnovšie verzie podporujú funkcie ako: relačný databázový transakčný (*OLTP*) systém, *Data Transformation Services* – nástroj pre extrakciu, transformáciu a loading (*ETL*), OLAP Server – systém pre On-line *Analytical Processing, Data Mining services* – nástroj pre dolovanie dát, nástroje na správu systému, zabezpečenie a správa oprávnení, zálohovanie a obnova, automatizácia správy systému (*služba SQL Server Agent*), monitorovanie výkonu a aktivity, replikácia dát.





4. 2 LAMP



Skratka LAMP znamená softvérové riešenie zvyčajne voľne šíriteľných programov, používaných spolu na beh dynamických web stránok alebo serverov.

LAMP má 3 alternatívy:

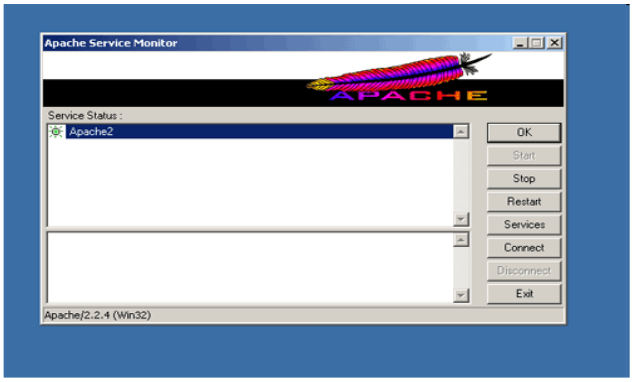
1. Linux + Apache + MySQL + PHP
2. Linux + Apache + MySQL + Perl
3. Linux + Apache + MySQL + Python

4.3 Linux

Linux (*presnejšie GNU/Linux*) je voľne šíriteľný operačný systém. Keďže spadá do kategórie slobodného softvéru, je prístupný aj so zdrojovými kódmi, ktoré môže ktokoľvek modifikovať a ďalej distribuovať. Môžeme ho skopírovať kamarátovi, rodičom alebo do školy. Je to plnohodnotný operačný systém s tisíckami aplikácií, ktoré potrebujeme denne pri práci. Existujú preň kancelársky softvér, programy na prácu s internetom, multimédiá, grafika, zábava, programovanie.

4.4 Apache

Apache HTTP Server je softvérový webový server s *open source* licenciou pre Linux, BSD, Microsoft Windows a iné platformy. V dnešnej dobe je najrozšírenejším na celom svete.



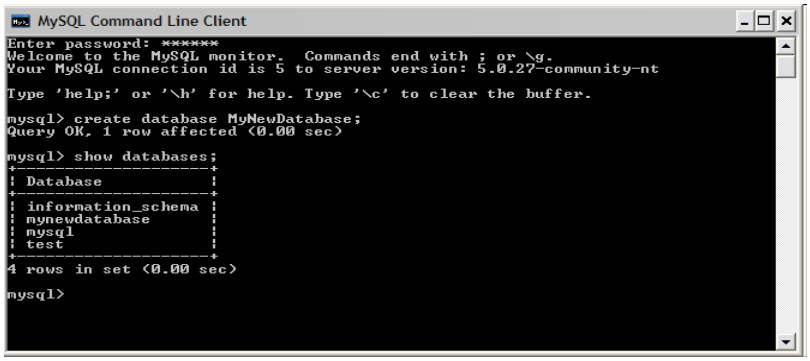
Konfigurácia Apache spočíva hlavne v editovaní súboru httpd.conf. Riadky začínajúce znakom # (mriežkou) sú komentáre.

Hlavné direktívy servera

ServerRoot "C:/Program Files/Apache"
Táto direktíva nám hovorí, kde bude hlavný adresár servera, hľadať sa tam budú konfiguračné súbory a logy (/conf , /logs).
DocumentRoot "C:/Program Files/Apache/htdocs/"
Táto direktíva nám hovorí, kde bude hľadať server súbory (php, html) aby ich vykonal. Čiže po zadaní http://localhost sa zobrazí obsah na ceste C:/ProgramFiles/Apache/htdocs/. DocumentRoot je vhodne zmeniť na C:/www
ServerName 127.0.0.1
Táto direktíva nám uvádza názov hostiteľa pre server. Pre lokálne použitie nám bude stačiť lokálna IP adresa (127.0.0.1) alebo meno localhost.
ScriptAlias /cgi-bin/ "C:/Program Files/Apache/cgi-bin/"
Táto direktíva nám hovorí kde sa budú nachádzať spustiteľné skripty typu CGI, čiže vždy keď sa vyžiada adresa napr. http://localhost/cgi-bin/ bude sa apache snažiť nám predložiť obsah C:/Program Files/Apache/cgi-bin/
Listen 80
Táto direktíva nám hovorí, na akom porte, alebo IP adrese bude Apache čakať požiadavky od klientov.

4.5 MySQL

MySQL je vytvorený švédskou firmou MySQL AB. Jeho hlavnými autormi sú Michael Widenius a David Axmark. Oficiálna stránka je www.mysql.com. Podrobný návod nájdeme na stránke http://dev.mysql.com. Stále si získava väčšiu popularitu vďaka svojej stabilite, rýchlosti, jednoduchosti, nenáročnosti a najmä preto, že takmer pre všetky platformy je zadarmo. Predstavuje slobodný a otvorený viacvláknový, viacužívateľský SQL relačný databázový server. MySQL je slobodný softvér, ale práva k nemu vlastní v súčastnosti firma Oracle. MySQL je podporovaný na viacerých platformách (Linux, Windows či Solaris a podpora preň je implementovaná vo viacerých programovacích jazykoch ako PHP, C++ či Perl. MySQL je relačný systém typu DBMS (*database management system*). MySQL chýbajú niektoré možnosti, ktoré majú iné databázové systémy. MySQL je však nenáročný na zdroje počítača. Je určený najmä pre malé webové databázy.



V prípade inštalácie MySQL do iného adresára ako "C:\MySQL" je potrebné vyhľadať súbor my-example.cnf, premenovať ho na my.cnf a skopírovať do C:\. Taktiež je potrebné v tomto súbore odkomentovať riadky basedir a datadir (*zmazať znak # zo začiatku*) a upraviť ich na basedir=c:/cesta_k_mysql/ a datadir=c:/cesta_k_mysql/data/. Konfigurácia MySQL prebieha cez konfiguračný súbor my.ini *(resp. cez my.cnf*), ktorý je štandardne umiestnený v adresári s nainštalovaným programom MySQL (*nemusí byť vždy pravda*). Znaková sada sa nastavuje v default-character-set, štandardne je nastavená znaková sada latin1, pre použitie diakritiky je potrebné ju zmeniť (na *utf8*).

4.6 Vlastnosti MySQL

Nasledujúci zoznam obsahuje najdôležitejšie vlastnosti MySQL.

Relačný databázový systém: rovnako ako veľká väčšina databázových systémov na trhu je MySQL relačný databázový systém.

Architektúra klient/server: MySQL je systém klient/server. Systém pozostáva z databázového servera (MySQL) a ľubovoľného množstva klientov (*aplikačných programov*), ktorí komunikujú so serverom. To znamená, že sa dotazujú na dáta, ukladajú zmeny a tak ďalej. Klienti môžu bežať na rovnakom počítači ako server, alebo môžu byť na sieti, napríklad kdekoľvek na internete. Skoro všetky známe veľké databázové systémy (*Oracle, PosstgreSQL, Microsoft SQL Server atď.*) sú systémy klient/server. V kontraste s touto architektúrou existujú rovnako systémy so súborovým serverom, napríklad Microsoft Access, dBase, FoxPro. Zásadnou nevýhodou systému so súborovými servermi je malá efektivita práce s rastúcim počtom používateľov na sieti.

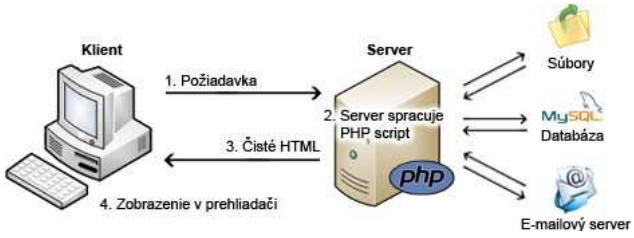
Kompatibilita s SQL: MySQL používa ako svoj databázový jazyk SQL. SQL je štandardný jazyk pre dotazy a aktualizáciu dát, a tak isto sa používa pre správu databázy. Existuje niekoľko dialektov SQL. MySQL dodržuje súčasný štandard SQL (*aktuálne je to SQL:2003*), avšak s významnými obmedzeniami a veľkým množstvom vlastných rozšírení. Na serveri MySQL sa dá pomocou voľby sql-mode nastaviť chovanie kompatibilné s rôznymi databázovými servermi.

Vnorené dotazy: MySQL od verzie 4.1 podporuje spracovanie dotazov vo forme SELECT * FROM tabulka1 WHERE x IN (*SELECT y FROM tabulka2*). (*Existujú ďalšie početné variácie syntaxe vnorených dotazov*).
Pohľady: jednoducho povedané, pohľad je fiktívna tabuľka vytvorená dotazom SQL, umožňuje pohľad na časť databázy. MySQL podporuje pohľady od verzie 5.0.
Uložené procedúry: sú programy v SQL uložené v databáze. Uložené procedúry sa všeobecne používajú pre zjednodušenie určitých krokov, ako napríklad vkladanie a mazanie záznamov. To je výhodné pre programátorov klientov, pretože nie je potreba priamo pracovať s tabuľkami a môžu sa spoľahnúť na uložené procedúry. Rovnako tak ako pohľady, uložené procedúry pomáhajú pri správe veľkých databázových projektov. MySQL podporuje uložené procedúry od verzie 5.0.
Triggery: sú to príkazy SQL, ktoré server automaticky spúšťa pri určitých operáciách (*napr. INSERT, UPDATE, DELETE*). MySQL podporuje triggery v obmedzenej forme od verzie 5.0.
Unicode: MySQL podporuje od verzie 4.1 všetky znakové sady vrátane Latin-1, Latin-2 a Unicode.
Používateľské rozhranie: pre správu MySQL existuje mnoho pohodlných užívateľských rozhraní.
Transakcia: transakcia znamená v kontexte databáz vykonanie niekoľkých databázových operácií nerozdeliteľne (buď sa vykonajú všetky, alebo sa nevykoná ani jedna a databáza zostane nezmenená), konzistentne (dáta v databáze budú správne pred i po vykonaní transakcie), izolovane od ostatných transakcií a s trvale uloženým výsledkom.
Obmedzenie cudzieho kľúča: toto pravidlo zaisťuje, že cudzí kľúč môže ukazovať iba na existujúci záznam v inej tabuľke. MySQL podporuje OCK pre tabuľky InnoDB.
GIS funkcie: MySQL podporuje od verzie 4.1 ukladanie dvojdimenzionálnych geografických dát. To znamená že MySQL je prispôsobená pre GIS (*Geografický Informačný Systém*).
Programovacie jazyky: existuje mnoho API a knižníc určených pre vývoj aplikácií nad MySQL. Pre programovanie klientov môžeme použiť jazyky: C, C++, Java, Perl, PHP, Python, Tcl atď.
Nezávislosť na platforme: Nielen aplikácie klienta môžu bežať v rôznych operačných systémoch. Server MySQL môže byť používaný pod mnohými OS.
Rýchlosť: MySQL je známy ako veľmi rýchly databázový systém. Toto hodnotenie je overené veľkým množstvom porovnávacích testov.
Medzi známe aplikácie založené na PHP patrí napríklad Wikipedia, Google, Facebook.

4.7 PHP



PHP (*PHP: Hypertext Preprocessor, Personal Home Page*) je skriptovací programovací jazyk, určený predovšetkým pre programovanie dynamických internetových stránok. Najčastejšie sa začleňuje priamo do štruktúry jazyka HTML, XHTML či WML, čo sa dá dobre využiť pri tvorbe webových aplikácií. PHP sa dá použiť aj pri tvorbe konzolových a desktopových aplikácií. Pri použití PHP pre dynamické stránky sú skripty vykonávané na strane servera – k používateľovi je prenesený až výsledok ich činnosti. PHP bolo inšpirované jazykmi podporujúcimi štruktúrované programovanie, najviac vlastností prebralo od jazyka C a Perl. PHP je nezávislý na platforme, rozdiely v rôznych operačných systémoch sa obmedzujú na niekoľko funkcií. Hotové skripty v PHP sa väčšinou dajú medzi rôznymi operačnými systémami prenášať bez úprav.



PHP podporuje mnoho knižníc pre rôzne účely, napr. spracovanie textu, grafiky, práca so súbormi, prístup k väčšine databázových systémov (*MySQL, ODBC, Oracle, PostgreSQL,*

MSSQL), podpora celého radu internetových protokolov (*HTTP, SMTP, SNMP, FTP, IMAP, POP3, LDAP, ...).*

PHP je jedným z dvoch najrozšírenejších skriptovacích jazykov pre web. Obľúbeným sa stal predovšetkým vďaka jednoduchosti použitia, bohatou zásobou funkcií a tomu, že kombinuje vlastnosti viacerých programovacích jazykov. V kombinácii s operačným systémom Linux, databázovým systémom (*obvykle MySQL alebo PostgreSQL*) a webovým serverom Apache je často využívaný k tvorbe webových aplikácií. Pre túto kombináciu sa vžila skratka LAMP – teda spojenie Linux, Apache, MySQL a PHP alebo Perl.

Výhody PHP:

- PHP je relatívne jednoduché na pochopenie
- PHP má syntax veľmi podobnú jazyku C a je teda väčšine vývojárov blízky
- PHP podporuje široký rad súvisiacich technológií, formátov a štandardov
- Je to otvorený projekt s rozsiahlou podporou komunity
- Je možné sa nájsť kvantá už hotového kódu k okamžitému použitiu alebo funkčné PHP aplikácie. Podstatná časť z hotového kódu je šírená pod nejakou slobodnou licenciou a dá sa použiť vo vlastných projektoch
- PHP je dobre integrované s webovým serverom Apache
- PHP ľahko komunikuje s databázami, ako je MySQL, PostgreSQL a rada ďalších
- PHP je multiplatformové a je možné ho používať s väčšinou webových serverov a na väčšine dnes existujúcich operačných systémov
- PHP podporuje mnoho existujúcich poskytovateľov webhostingových služieb

Nevýhody PHP:

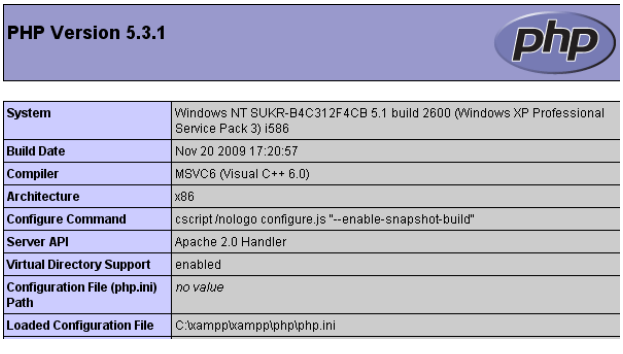
- PHP je interpretovaný, nie kompilovaný jazyk
- Ktokoľvek má priamy prístup k serveru, môže nahliadnuť do vašich PHP skriptov
- Pretože je PHP aktívne vyvíjaný, v budúcich verziách jazyka sa môžu niektoré funkcie zmeniť alebo sa môžu chovať inak ako doposiaľ

Čo sa v PHP dá napísať?

PHP je univerzálny skriptovací jazyk, no od začiatku vznikal pre použitie vo webových aplikáciách. Medzi najčastejšie aplikácie písané v PHP patria napríklad:

- internetové obchody
- podnikové informačné systémy (*nech už intranetové nebo internetové*)
- diskusné fóra
- redakčné systémy
- firemné prezentácie
- dynamické osobné stránky
- vyhľadávače a katalógy
- drobnosti typu počítadlá, ankety a mnoho ďalších

Medzi známe aplikácie založené na PHP patrí napríklad Facebook, phpBB, Drupal, Joomla, WordPress, Moodle a MediaWiki, softvér na ktorom beží Wikipédia.



System	Windows NT SUKR-B4C312F4CB 5.1 build 2600 (Windows XP Professional Service Pack 3) i586
Build Date	Nov 20 2009 17:20:57
Compiler	MSVC6 (Visual C++ 6.0)
Architecture	x86
Configure Command	cscript /nologo configure.js "--enable-snapshot-build"
Server API	Apache 2.0 Handler
Virtual Directory Support	enabled
Configuration File (php.ini) Path	*no value*
Loaded Configuration File	C:\xampp\xampp\php\php.ini

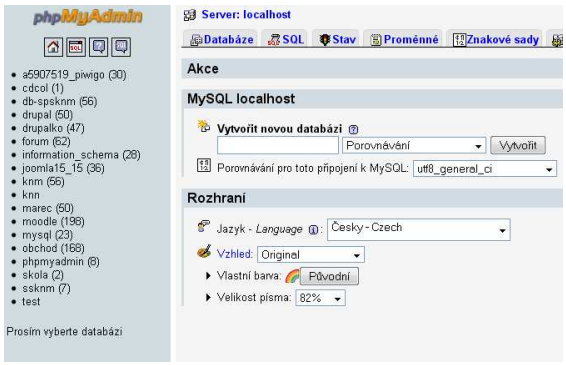
Konfigurácia PHP prebieha cez konfiguračný súbor php.ini, ktorý je umiestnený v adresári s nainštalovaným programom PHP. Pokiaľ však nie sme majiteľom servera k súboru php.ini sa zväčša nedostaneme, niektoré vlastnosti si však môžeme nastaviť, pokiaľ to máme povolené cez .htaccess (*v httpd.conf musí byť povolené*). Žiaľ na väčšine nie len

bezplatných, ale aj platených hostingov máme len minimálnu možnosť ovplyvniť nastavenie php.ini a úpravu .htaccess. Dôsledkom je, že nemôžeme využívať všetky funkcie a možnosti PHP. Napríklad pri spúšťaní starších skriptov môžeme potrebovať v php.ini upraviť register_globals = On/Off. Pri inštalácii niektorého z redakčných systémov môžeme potrebovať upraviť niektoré požadované nastavenia, napríklad pri väčšine CMS systémov je vhodné zvýšiť memory_limit. Na otestovanie je vhodné napísať jednoduchý skript, ktorý preverí naše nastavenia < ?php phpinfo(); ?>. Nazveme ho test.php a umiestnime do adresára pre naše www stránky.

4.8 Perl

Perl (*Practical Extraction and Report Language*) je programovací jazyk primárne určený na prácu s textom. Vyvinutý bol pánom Larrym Wallom. Dnes nachádza využitie v množstve systémov, od systémových skriptov, cez GUI, hry a 3D grafiku pomocou OpenGL, až po internetové aplikácie.

4.9 Administrátorské prostredie phpMyAdmin



PhpMyAdmin je webová aplikácia napísaná v PHP. Je určená pre správu databázy MySQL cez webové rozhranie. S jej pomocou si môžeme pripraviť veľmi rýchlo novú tabuľku, odstrániť tabuľku, vložiť či vymazať riadky, odhaliť chyby v SQL dotazoch a podobne. Toto prostredie je distribuované zadarmo pod GNU licenciou. Aktuálna verzia je k dispozícii už v 57 svetových jazykoch. Samozrejme medzi nimi nechýba slovenčina a podporuje jednak kódovanie win-1250 (typické pre operačné systémy Windows), aj iso-2 (bežné pre OS Linux) a samozrejme aj svetové UTF-8. Slovenský jazyk rozhrania je zvolený automaticky podľa jazykových preferencií v prehliadači a ak nie, dá sa nastaviť. Toto kódovanie by malo byť nastavené v phpMyAdmin rovnako ako na MySQL serveri. V opačnom prípade sa pri zobrazovaní výsledkov dotazov slovenské znaky nezobrazia správne. Automaticky prekódovať diakritiku v MySQL je totiž problematické.

5. XAMPP



XAMPP je dlhodobo vyvíjaný skupinou Apache Friends. Ide o balík programov pre web server, ktorý obsahuje Apache, MySQL, PHP & PEAR, Perl, ProFTPD, phpMyAdmin a niekoľkých ďalších dôležitých a často používaných rozšírení ako sú SSL/TLS (podpora pre šifrovaný prístup k webserveru). Najviac podporuje pridávanie "AddOn" prídavných častí. Jedná sa o kolekciu otvorených softvérových aplikácií pre inštaláciu a používanie webového servera Apache. Existujú rôzne kombinácie aplikácií v závislosti na Xampp verzii. V súčasnosti existujú štyri typy Xampp: Xampp pre Linux (*predtým známy ako LAMPP*), Xampp pre Windows, Xampp pre Mac OS X, Xampp pre Solaris. Stiahnutie a inštalácia XAMPP - webserver balíka (*pre inštaláciu na localhost*). Inštalačný balík nájdeme na stránke www.xampp.en.softonic.com odkiaľ stiahneme jeho verziu pre OS Windows. Zazipovaný súbor uložíme na disku a rozbalíme ho pomocou jedného z archivačných programov (*WinRAR,..www.win-rar.com/download*). Spustíme inštaláciu, ktorá spočíva iba vo vybratí lokalizácie pre program a v niekoľko sekundovej inštalácii. Po nainštalovaní nám vznikne priečinok xampp na mieste, ktoré sme si vybrali, a zástupca na ploche s ikonkou xampp-

u. Po kliknutí na ikonku sa nám otvorí okno ovládacieho panelu XAMPP v ktorom spustíme Apache a MySQL.



Spomedzi priečinkov nás bude zaujímať hlavne *htdocs*, do ktorého budeme kopírovať naše www stránky. Do prehliadača zadáme adresu 127.0.0.1. Zobrazí sa nám úvodná stránka xampp-u. V ľavom dolnom rohu úvodnej stránky si môžeme pozrieť naše nastavenia, či v menu **tools** prejsť na **phpMyAdmin.**

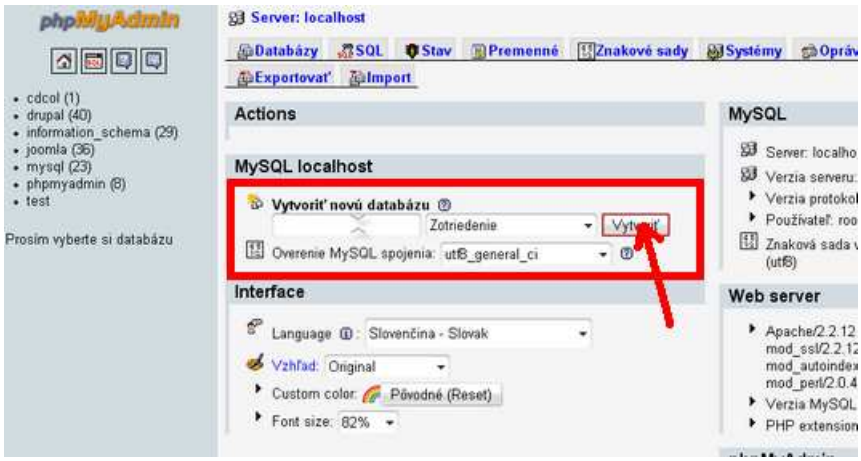


Namiesto balíka XAMPP môžeme použiť webový server Vertrigo (http://vertrigo.sourceforge.net/?lang=sk), či balík WampServer http://www.wampserver.com/en/. Obsahujú tiež všetko potrebné pre jednoduchú inštaláciu serverového prostredia na lokálnom počítači. Inštalácie neponúkajú žiadne špeciálne možnosti a sú rovnako jednoduché.



5.1 Vytvorenie MySQL databázy pre potrebné dáta

Databázu vytvoríme prostredníctvom aplikácie phpMyAdmin, do ktorej sa dostaneme zadaním cesty http://localhost/phpmyadmin/ do svojho prehliadača. V základe máme tri systémové a jednu testovaciu databázu. My si však vytvoríme novú. Vyplníme názov databázy, ktorú budeme používať, a klikneme na tlačidlo Vytvoriť.



6. Dotazovací jazyk SQL

6.1 Komentáre

-- Toto je jednoriadkový komentár

/* Viacriadkový komentár sa používa napríklad pri
komentovaní rozsiahlejších funkčných blokov
*/
MySQL používa pre jednoriadkový komentár namiesto dvoch pomlčiek znak #

6.2 Dátové typy

Pri návrhu tabuľky musíme určiť dátové typy stĺpcov (atribútov). Dátový typ je atribút stĺpca, ktorý určuje, aké hodnoty sa budú môcť do daného stĺpca zapisovať. Každý stĺpec môže teda uchovávať hodnoty iba jedného, dopredu určeného typu.

Hodnoty NULL

Jednotlivé stĺpce rôznych dátových typov môžu okrem konkrétnych hodnôt obsahovať i hodnoty NULL. Táto hodnota vlastne vyjadruje neexistenciu hodnoty premennej. Treba rozlišovať medzi nulou a hodnotou NULL. Nula je konkrétne číslo, ale hodnota NULL znamená, že tento údaj nepoznáme. Naopak NOT NULL nám určuje, že pole musí obsahovať hodnotu.

Textové dátové typy v MySQL

Dátový typ	**Popis**
CHAR (veľkosť)	Pevná dĺžka reťazca (*môže obsahovať písmená, číslice a špeciálne znaky*). Pevná veľkosť je uvedená v zátvorkách. Maximálny počet znakov je 255.
VARCHAR (veľkosť)	Používa sa na uloženie reťazca premenlivej dĺžky. Táto dĺžka je definovaná v zátvorke - *veľkosť*. Maximálny počet znakov je 255. Pri uvedení väčšej hodnoty ako 255 sa automaticky zmení na dátový typ TEXT.
TINYTEXT	Možné uložiť 255 znakov.
TEXT	Možné uložiť 65.535 znakov.
BLOB	Veľký binárny objekt (*BLOB - Binary Large OBjects*). Umožňuje uloženie do 65535 bajtov binárnych dát.
MEDIUMTEXT	Možné uložiť až 16.777.215 znakov.
MEDIUMBLOB	Veľký binárny objekt (*BLOB - Binary Large OBjects*). Umožňuje uloženie do 16 777 215 bajtov (16MB) binárnych dát.
LONGTEXT	Možné uložiť až 4.294.967.295 znakov.

LONGBLOB	Veľký binárny objekt. Umožňuje uloženie do 4 294 967 295 bajtov (4GB) binárnych dát.
Enum (x, y, z, atď)	Vymenovaný typ. Je potrebné určiť zoznam povolených hodnôt. Počet týchto hodnôt môže byť maximálne 65535. Ak sa pokúsime vložiť do tabuľky hodnotu, ktorú sme nedefinovali, vloží sa prázdna hodnota NULL.
SET	Množina. Podobné dátovému typu ENUM s tým rozdielom, že môžeme zadefinovať maximálne 64 položiek a v tabuľke môžeme uložiť viac ako jednu položku.

Číselné dátové typy v MySQL

Dátový typ	**Popis**
TINYINT	Môžeme uložiť čísla -128 až 127 Pri doplnení špecifikátora UNSIGNED je rozsah od 0 do 255. Veľkosť: 1B.
SMALLINT	Celé číslo v intervale od -32768 do 32767. Pri doplnení špecifikátora UNSIGNED je rozsah od 0 do 65535. Veľkosť: 2B.
MEDIUMINT	Rozsah -8388608 až 8388607. Pri doplnení špecifikátora UNSIGNED je rozsah od 0 do 16777215. Veľkosť: 3B.
INT	Celé číslo v intervale od -2147483648 do 2147483647. Pri doplnení špecifikátora UNSIGNED je rozsah od 0 do 4294967295. Veľkosť: 4B.
BIGINT	V rozsahu od -9 223 372 036 854 775 808 po 9 223 372 036 854 775 807. . Pri doplnení špecifikátora UNSIGNED je rozsah dvojnásobný. Veľkosť: 8B.
FLOAT (veľkosť, d)	Číslo s desatinnou časťou. Maximálny počet cifier, pre zobrazenie čísla udáva parameter *Veľkosť*. Maximálny počet číslic vpravo od desatinnej čiarky je uvedené v parametri d. Rozsah typu FLOAT: od -3.402E+38 do -1.1751E-38, od 1.175E-38 do 3.402E+38.
DOUBLE (veľkosť, d)	Číslo s desatinnou časťou - s dvojitou presnosťou. Maximálny počet cifier, pre zobrazenie čísla udáva parameter *Veľkosť*. Maximálny počet číslic vpravo od desatinnej čiarky je uvedené v parametri d. Rozsah typu DOUBLE: od -1.7976E+308 do -2.225E-308; 0; od 2.225E-308 do 1.797E+308.
DECIMAL (veľkosť, d)	Hodnota DOUBLE uložená ako reťazec. Význam parametrov *Veľkosť* a *d* je rovnaký ako v predchádzajúcich príkladoch.

Dátumové dátové typy v MySQL

Dátový typ	**Popis**
DATE	Dátum vo formáte RRRR-MM-DD
DATETIME	Dátum a čas vo formáte RRRR-MM-DD HH: MM: SS
TIMESTAMP	Časové razítko. Hodnota dátového typu TIMESTAMP je počet sekúnd od začiatku Unixovej epochy, teda od 1970-01-01 00:00:00 UTC. Formát pre dátový typ TIMESTAMP je: RRRR-MM-DD HH: MM: SS
TIME	Čas vo formáte HH: MM: SS
YEAR[(2|4)]	YEAR je dvoj /YEAR(2)/ alebo štvor /YEAR(4)/ ciferná reprezentácia roku.

6.3 Založenie novej databázy a tabuľky

Databázy aj tabuľky sa tvoria príkazom CREATE. Novú databázu vytvoríme príkazom CREATE DATABASE nazovdatabazy, novú tabuľku obdobne príkazom CREATE TABLE nazovtabulky. Napríklad vytvorenie databázy s názvom skola by prebehlo zadaním nasledujúceho príkazu:

mysql> CREATE DATABASE skola;

Vytvorili sme databázu skola. Ak zadáme ten istý príkaz opäť, jeho vykonanie zlyhá, pretože rovnomenná databáza už existuje. To môže byť nepríjemné najmä vtedy, ak je tento príkaz súčasťou skriptu vykonávajúceho viacero SQL príkazov – jeho vykonávanie by sa totiž predčasne ukončilo. Tomu však možno zabrániť použitím klauzuly IF NOT EXISTS, ako vidíme nižšie.

mysql> CREATE DATABASE IF NOT EXISTS skola;

Že nová databáza bola skutočne založená, zistíme príkazom SHOW DATABASES, ktorý nám všetky databázy na danom serveri vypíše.

mysql> SHOW DATABASES;

Ak sa budeme chcieť k niektorej z vypísaných databáz na serveri pripojiť, použijeme príkaz USE nazovdatabazy.

mysql> USE skola;

Teraz pomocou spomínaného príkazu CREATE TABLE vytvoríme v databáze skola tabuľku ziak.

```
mysql> CREATE TABLE ziak (
id_uzivatela MEDIUMINT UNSIGNED NOT NULL AUTO_INCREMENT,
nick VARCHAR(15) NOT NULL,
meno VARCHAR(15) NOT NULL,
priezvisko VARCHAR(30) NOT NULL,
trieda VARCHAR(5) NOT NULL,
adresa VARCHAR(40),
email VARCHAR(40),
heslo CHAR(10),
datum_zaregistrovania DATETIME NOT NULL,
PRIMARY KEY (id_uzivatela));
```

Príkazom SHOW TABLES môžeme overiť pre nás vizuálne, či sa vytvorila požadovaná tabuľka

mysql> SHOW TABLES;

Ak chceme vidieť všetky stĺpce tabuľky ziak zadáme príkaz SHOW COLUMNS FROM ziak

mysql> SHOW COLUMNS FROM ziak;

Field	Type	Null	Key	Default	Extra
id_uzivatela	mediumint(8) unsigned	NO	PRI	*NULL*	auto_increment
nick	varchar(15)	NO		*NULL*	
meno	varchar(15)	NO		*NULL*	
priezvisko	varchar(30)	NO		*NULL*	
trieda	varchar(5)	NO		*NULL*	
adresa	varchar(40)	YES		*NULL*	
email	varchar(40)	YES		*NULL*	
heslo	char(10)	YES		*NULL*	
datum_zaregistrovania	datetime	NO		*NULL*	

6.4 Vkladanie dát

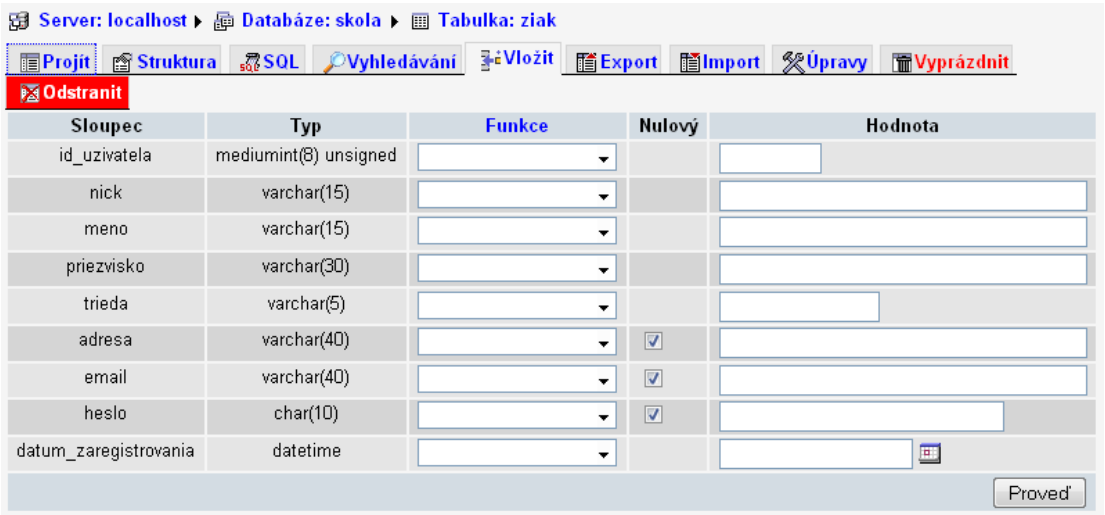
Všetko máme pripravené na naplnenie tabuľky dátami, na vkladanie údajov slúži príkaz INSERT.

```
mysql> INSERT INTO ziak (nick, meno, priezvisko, trieda, adresa, email,
heslo, datum_zaregistrovania)
VALUES ('student1','Jano','Jankovsky', '3.AE','Sturova 1211 KNM','janko@spsknm.sk',
'gfdfggkf',NOW());
```

Týmto spôsobom sme pridali prvé dáta do tabuľky. Vymenovanie jednotlivých stĺpcov môžeme vynechať, ale v tom prípade musíme zadať hodnoty pre všetky stĺpce v takom poradí, ako boli vymenované pri návrhu tabuľky. Všimneme si že musíme nechať prázdne miesto aj pre **id_uzivatela.**

mysql> INSERT INTO ziak VALUES ('', 'student2','Dano','Dankovsky', '3.AE','Javorova 876 CA','dano@spsknm.sk', 'gGFGkf', '2011-05-09 15:59:39');
INSERT INTO ziak VALUES ('', 'student3','Peter','Petrovsky', '3.AE','Smrekova 511 KNM','peto@post.sk', 'WNRE6Ehgjg', '2010-07-07 15:39:09');
INSERT INTO ziak VALUES ('', 'student4','Fero','Cankovsky', '3.BE','Lipova 211 ZA','fero@azet.sk', 'Dghgfhgfh', '2011-05-08 11:50:49');
INSERT INTO ziak VALUES ('', 'student5','Marian','Jakubik', '3.AE',' Lipova 196 CA','mario@pobox.sk', 'poiiiusjc5', '2011-07-09 18:29:30');
INSERT INTO ziak VALUES ('', 'student6','Marek','Novota', '3.BE',' Limbova 1211 KNM','marko@gmail.sk', 'juzsk654hg',NOW());
INSERT INTO ziak VALUES ('', 'student7','Boris','Godunov', '3.BE','Sturova 211 ZA','bobko@knm.sk', '76hghHH',NOW());

Najjednoduchší spôsob, ako vložiť do tabuľky údaje, je použiť nejaký softvér pre správu databáz, napríklad phpMyAdmin. Po prihlásení sa do systému, vybratí databázy a vybratí tabuľky, do ktorej chceme údaje vkladať, môžeme údaje vkladať pomocou jednoduchého formulára. Rovnako tiež v prostredí phpMyAdmin môžeme vytvárať nové databázy, tabuľky a vykonávať množstvo úprav, ktoré by sme inak museli realizovať prostredníctvom SQL kódu. V informačnom bloku SQL dotazu nájdeme tiež možnosti: upraviť dotaz, zobraziť PHP kód, ktorý môžeme použiť pri ďalších úpravách.



6.5 Zrušenie databázy, tabuľky a úprava dát

Rovnako, ako možno databázu založiť, možno ju tiež zrušiť. Slúži k tomu príkaz DROP DATABASE.
mysql> DROP DATABASE skola;

Opakom príkazu CREATE TABLE je príkaz DROP TABLE, ten celú tabuľku odstráni.
mysql> DROP TABLE ziak;
Často budeme potrebovať vložené dáta modifikovať. Žiakom sa môže zmeniť adresa, email, či trieda keď úspešne absolvujú predmet Informačné technológie a postúpia do vyššieho ročníka. Dáta modifikujeme napríklad prostredníctvom príkazu UPDATE.
mysql>UPDATE ziak SET trieda = '4.AE';

Všetkým žiakom by sme nastavili, že navštevujú 4.AE triedu, žiaci 3.BE však postúpia do 4.BE. K tejto úprave použijeme klazolu WHERE a modifikujeme predchádzajúci príklad.
mysql>UPDATE ziak SET trieda = '4.AE' WHERE trieda = '3.AE';

Následne tento kód upravíme a použijeme aj pre triedu 3.BE.

6.6 Výber dát z databázy

Na výber dát používame príkaz SELECT. Jeho základná syntax je, SELECT stĺpce FROM tabuľka.
mysql>SELECT * FROM ziak;

id_uzivatela	nick	meno	priezvisko	trieda	adresa	email	heslo	datum_zaregistrovania
1	student2	Dano	Dankovsky	4.AE	Javorova 876 CA	dano@spsknm.sk	gGFGkf	2011-05-09 15:59:39
2	student3	Peter	Petrovsky	4.AE	Smrekova 511 KNM	peto@post.sk	WNRE6Ehgjg	2010-07-07 15:39:09
3	student4	Fero	Cankovsky	4.BE	Lipova 211 ZA	fero@azet.sk	Dghgfhgfh	2011-05-08 11:50:49
4	student5	Marian	Jakubik	4.AE	Lipova 196 CA	mario@pobox.sk	poiiiusjc5	2011-07-09 18:29:30
5	student6	Marek	Novota	4.BE	Limbova 1211 KNM	marko@gmail.sk	juzsk654hg	2011-06-09 16:52:04
6	student7	Boris	Godunov	4.BE	Sturova 211 ZA	bobko@knm.sk	76hghHH	2011-06-09 16:52:04
7	student1	Jano	Jankovsky	4.AE	Sturova 1211 KNM	janko@spsknm.sk	gfdfggkf	2011-06-09 18:51:34

Zabezpečí vybratie všetkých stĺpcov z tabuľky ziak.

Ak chceme prezerať napríklad len stĺpce meno, priezvisko a email, zadáme:
mysql> SELECT meno, priezvisko, email FROM ziak;

V prípade, že sa nám nepáči pomenovanie stĺpcov v tabuľke, môžeme ich pomocou AS vo výpise premenovať.
SELECT meno AS MENO, priezvisko AS PRIEZVISKO, email AS ŠKOLSKÝ_EMAIL FROM ziak;

	MENO	PRIEZVISKO	ŠKOLSKÝ_EMAIL
	Dano	Dankovsky	dano@spsknm.sk
	Peter	Petrovsky	peto@post.sk
	Fero	Cankovsky	fero@azet.sk
	Marian	Jakubik	mario@pobox.sk
	Marek	Novota	marko@gmail.sk
	Boris	Godunov	bobko@knm.sk
	Jano	Jankovsky	janko@spsknm.sk

Jednoduchý výber dát príkazom SELECT však vracia vo výstupoch stĺpce celej tabuľky. Ak chceme, napríklad zobraziť len informácie o žiakovi Godunov, musíme využiť klauzulu WHERE.
mysql> SELECT * FROM ziak WHERE priezvisko='Godunov';

id_uzivatela	nick	meno	priezvisko	trieda	adresa	email	heslo	datum_zaregistrovania
6	student7	Boris	Godunov	4.BE	Sturova 211 ZA	bobko@knm.sk	76hghHH	2011-06-09 16:52:04

V prípade, že by nás zaujímal len jeho email a trieda, zadáme:
mysql> SELECT email, trieda FROM ziak WHERE priezvisko='Godunov';

	email	trieda
	bobko@knm.sk	4.BE

Na správne zostavenie vhodných podmienok využívame matematické a logické operátory.

Porovnávací operátor	Význam operátora	Príklad
=	rovnosť	WHERE (vek ='18')
<>	ne rovnosť	WHERE (vek <>'18')
>	väčší	WHERE (vek >'18')
<	menší	WHERE (vek <'18')
>=	väčší alebo rovný	WHERE (vek >='18')
<=	menší alebo rovný	WHERE (vek <='18')
BETWEEN	patrí do intervalu	WHERE (vek BETWEEN '18' AND '20')
NOT BETWEEN	nepatrí do intervalu	WHERE(vek NOT BETWEEN '18' AND '20')

NOT – uplatní sa, pokiaľ je podmienka nepravdivá.
AND – uplatní sa, keď sú obe podmienky pravdivé.
OR – uplatní sa, pokiaľ je aspoň jedna podmienka pravdivá.

Môžeme chcieť vypísať mená a priezviská žiakov, ktorý sa zaregistrovali v mesiaci máj roku 2011:
mysql> SELECT meno, priezvisko FROM ziak WHERE (datum_zaregistrovania > '2011-04-30 24:00:00') AND (datum_zaregistrovania < '2011-06-01 00:00:00');

	meno	priezvisko
	Dano	Dankovsky
	Fero	Cankovsky

Pri výbere dát je možné taktiež použiť znaky takzvaného hromadného výberu. Znak _ nahrádza práve jeden akýkoľvek znak a znak % skupinu znakov. Pre ich použitie existujú operátory LIKE a NOT LIKE. Vyberieme mená, priezviská a emaily žiakov, ktorí nemajú emailovú koncovku spsknm.sk, aby sme im ju mohli založiť.
mysql> SELECT meno, priezvisko, email FROM ziak WHERE email NOT LIKE '%@spsknm.sk';

meno	priezvisko	email
Peter	Petrovsky	peto@post.sk
Fero	Cankovsky	fero@azet.sk
Marian	Jakubik	mario@pobox.sk
Marek	Novota	marko@gmail.sk
Boris	Godunov	bobko@knm.sk

Časom môžeme chcieť pridať ďalší stĺpec do tabuľky, v našom prípade pridáme vzdialenosť bydliska od školy.
mysql> ALTER TABLE ziak ADD COLUMN vzdialenost_km INT(7);
Vzdialenosť by sme mohli upraviť pomocou príkazu UPDATE, my však využijeme možnosti phpMyAdmin pre úpravu dát.
Nie vždy sa nám hodí mať vo výpise údaje v takom tvare, ako sú v databázovej tabuľke. Niekedy potrebujeme zverejniť len hodnoty v intervaloch, inokedy priradiť hodnotám iný význam. K tomuto nám slúži príkaz CASE.

mysql> SELECT meno, priezvisko CASE WHEN vzdialenost_km<=2 THEN 'býva blízko' ELSE 'býva ďaleko' END FROM ziak;

Zamedzenie výpisu duplicitných riadkov realizujeme prostredníctvom príkazu DISTINCT. Môže sa stať, že niektorí žiaci sa budú registrovať viackrát. Aby sme zamedzili výpisu opakujúcich sa žiakov, obmedzíme výpis len na rôzny nick, meno a priezvisko.
mysql>SELECT DISTINCT nick, meno, priezvisko FROM ziak;

Výber hodnôt patriacich do vymenovanej množiny realizujeme prostredníctvom príkazu IN. Vyberieme všetkých žiakov, ktorých vzdialenosť od školy je jeden alebo dva kilometre.
mysql>SELECT nick, meno, priezvisko, vzdialenost_km FROM ziak WHERE vzdialenost_km IN('1', '2');

nick	meno	priezvisko	vzdialenost_km
student3	Peter	Petrovsky	1
student6	Marek	Novota	2
student7	Boris	Godunov	1
student1	Jano	Jankovsky	2

Často budeme chcieť zobraziť výsledky nášho dotazu v určitom poradí. K tomu nám slúži klauzula ORDER BY. Nasledujúci SQL dotaz nám zoradí žiakov podľa začiatočného písmena priezviska v abecede.
mysql>SELECT priezvisko, meno, nick, heslo, email, vzdialenost_km FROM ziak ORDER BY priezvisko ASC;
Pokiaľ by sme chceli zaradiť žiakov od konca podľa priezviska abecedy, použijeme nasledujúci dotaz:
mysql>SELECT priezvisko, meno, nick, heslo, email, vzdialenost_km FROM ziak ORDER BY priezvisko DESC;

priezvisko	meno	nick	heslo	email	vzdialenost_km
Petrovsky	Peter	student3	WNRE6Ehgjg	peto@post.sk	1
Novota	Marek	student6	juzsk654hg	marko@gmail.sk	2
Jankovsky	Jano	student1	gfdfggkf	janko@spsknm.sk	2
Jakubik	Marian	student5	poiiiusjc5	mario@pobox.sk	30
Godunov	Boris	student7	76hghHH	bobko@knm.sk	1
Dankovsky	Dano	student2	gGFGkf	dano@spsknm.sk	23
Cankovsky	Fero	student4	Dghgfhgfh	fero@azet.sk	10

Nasledujúci dotaz nám zoradí žiakov podľa dátumu poslednej registrácie:
mysql>SELECT priezvisko, meno, nick, datum_zaregistrovania FROM ziak ORDER BY datum_zaregistrovania DESC;
Radenie podľa abecedy či podľa dátumu registrácie nemusí byť vždy vhodné, niekedy môžeme chcieť zaradiť žiakov podľa vzdialenosti a následne pri rovnakých hodnotách podľa priezviska abecedy:
mysql>SELECT vzdialenost_km, priezvisko, meno FROM ziak ORDER BY vzdialenost_km, priezvisko;

Pridajme ďalší stĺpec do tabuľky, v našom prípade pridáme stĺpec krúžok ktorý žiak navštevuje.
mysql> ALTER TABLE ziak ADD COLUMN kruzok VARCHAR(20);
Krúžok ktorý žiak navštevuje, by sme mohli upraviť pomocou príkazu UPDATE, my však využijeme možnosti phpMyAdmin pre úpravu dát. Niektorí žiaci nenavštevujú žiadny krúžok, v tomto prípade necháme prázdnu hodnotu, čiže hodnotu NULL. Pre výber všetkých žiakov ktorí nenavštevujú žiadny krúžok nemôžeme použiť dotaz:
mysql> SELECT * FROM ziak WHERE kruzok='NULL';

z dôvodu, že porovnanie s hodnotou NULL vracia vždy hodnotu NULL. Riešením sú operátory IS NULL a IS NOT NULL. Operátor IS NULL slúži pre výber údajov, kde testujeme, či niektorý atribút má, alebo nemá hodnotu NULL. Riešením nášho príkladu by bol nasledovný dotaz:
mysql> SELECT priezvisko, meno, nick, kruzok FROM ziak WHERE kruzok IS NULL;

priezvisko	meno	nick	kruzok
Jakubik	Marian	student5	*NULL*
Godunov	Boris	student7	*NULL*

Každému žiakovi v triede budeme chcieť sledovať dochádzku na hodinách informačných technológii. Z tohto dôvodu si založíme novú tabuľku:
mysql> CREATE TABLE dochadzka (id_vymeskania MEDIUMINT UNSIGNED NOT NULL AUTO_INCREMENT, id_uzivatela MEDIUMINT UNSIGNED NOT NULL, cas INT(2), datum_vymeskania DATETIME, PRIMARY KEY (id_vymeskania));

Následne (*podľa triednej knihy*) povkladáme jednotlivým žiakom ich vymeškané hodiny. Tento dotaz vloží žiakovi s identifikačným číslom 1 dve hodiny a aktuálny čas jeho chýbania.

mysql> INSERT INTO dochadzka VALUES ('', 1, 2 ,NOW());
Dotaz ktorý vloží žiakovi s identifikačným číslom 3 jednu hodinu a aktuálny čas jeho chýbania:
mysql> INSERT INTO dochadzka VALUES ('', 3, 1 ,NOW());
Povkladáme údaje o počte vymeškaných hodín, následne zobrazíme prostredníctvom dotazu:
mysql>SELECT * FROM dochadzka;

id_vymeskania	id_uzivatela	cas	datum_vymeskania
1	1	2	2011-06-15 16:00:01
2	1	2	2011-06-15 16:00:07
3	5	1	2011-06-15 16:00:41
4	5	2	2011-06-15 16:00:55
5	4	2	2011-06-15 16:01:06
6	3	1	2011-06-15 16:01:16
7	3	1	2011-06-15 16:34:13

Po vložení dochádzky pre všetkých žiakov, budeme chcieť vypísať súčet vymeškaných hodín. K tomu použijeme klauzulu GROUP BY, ktorá zoskupuje všetky riadky, ktoré majú v stĺpci rovnakú hodnotu. Používa sa v spojení s agregačnými funkciami, napríklad funkciou SUM.
mysql> SELECT id_uzivatela, SUM(cas) AS Celkovy_pocet_hodin FROM dochadzka GROUP BY id_uzivatela;

id_uzivatela	Celkovy_pocet_hodin
1	4
3	2
4	2
5	3

Pokiaľ budeme chcieť obmedziť výpis len na celkový počet vymeškaných hodín väčší ako dve hodiny, použijeme klauzulu HAVING. Tu je potrebné pochopiť rozdiel medzi WHERE a HAWING. Klauzula WHERE slúži pre obmedzenie záznamov, ktoré sú potom zoskupované pomocou klauzuly GROUP BY. Inak povedané, klauzula WHERE pracuje nad vstupnými a nie nad výstupnými údajmi. Pre obmedzenie zoskupených údajov slúži klauzula HAVING.
mysql> SELECT id_uzivatela, SUM(cas) AS Celkovy_pocet_hodin FROM dochadzka GROUP BY id_uzivatela HAVING SUM(cas)>2;

id_uzivatela	Celkovy_pocet_hodin
1	4
5	3

Nie všetky problémy dokážeme riešiť pomocou jednoduchého príkazu SELECT. Napríklad budeme chcieť zistiť, ktorí študenti bývajú ďalej od školy ako žiak Cankovsky, pričom dopredu nevieme, ako ďaleko od školy práve žiak Cankovsky býva. Zistíme to pomocou vnoreného SQL dotazu. Ten sa skladá z dvoch častí: najskôr potrebujeme zistiť, ako ďaleko býva žiak Cankovsky od školy.

mysql> SELECT vzdialenost_km FROM ziak WHERE priezvisko='Cankovsky';

	vzdialenost_km
	10

S výsledkom tohto dotazu potom porovnáme ostatné záznamy. Takto sformulovaný vnorený SQL dotaz musí byť v zátvorkách a keď vracia práve jednu hodnotu, nazývame ho jednoriadkový. Vnorený dotaz môžeme umiestniť do hlavného príkazu SELECT ku klauzulám FROM, WHERE, alebo HAVING.

mysql> SELECT vzdialenost_km, priezvisko, meno FROM ziak WHERE vzdialenost_km>(SELECT vzdialenost_km FROM ziak WHERE priezvisko='Cankovsky');

vzdialenost_km	priezvisko	meno
23	Dankovsky	Dano
30	Jakubik	Marian

Vo vnorenom dotaze môžeme tiež použiť agregačné funkcie a podobne. Keď napríklad budeme chcieť vedieť, ktorí žiaci bývajú ďalej ako je priemerná vzdialenosť od školy, použijeme nasledovný dotaz:

mysql> SELECT vzdialenost_km, priezvisko, meno FROM ziak WHERE vzdialenost_km>(SELECT AVG(vzdialenost_km) FROM ziak);

vzdialenost_km	priezvisko	meno
23	Dankovsky	Dano
10	Cankovsky	Fero
30	Jakubik	Marian

6.7 Spájanie tabuliek

Pokiaľ chceme spojiť tabuľky ziak a dochadzka, môžeme použiť klauzulu WHERE. Ako príklad vypíšeme k absenciám žiakov ich mená:

mysql> SELECT ziak.priezvisko, ziak.meno, dochadzka.cas, dochadzka.datum_vymeskania FROM ziak, dochadzka WHERE ziak.id_uzivatela =dochadzka.id_uzivatela;

priezvisko	meno	cas	datum_vymeskania
Dankovsky	Dano	2	2011-06-15 16:00:01
Dankovsky	Dano	2	2011-06-15 16:00:07
Novota	Marek	1	2011-06-15 16:00:41
Novota	Marek	2	2011-06-15 16:00:55
Jakubik	Marian	2	2011-06-15 16:01:06
Cankovsky	Fero	1	2011-06-15 16:01:16
Cankovsky	Fero	1	2011-06-15 16:34:13
Petrovsky	Peter	1	2011-06-15 18:22:00
Cankovsky	Fero	1	2011-06-15 18:22:20
Godunov	Boris	1	2011-06-15 18:22:44
Jankovsky	Jano	0	2011-06-15 18:23:13

V prípade potreby spočítania jednotlivých hodín pre každého žiaka by sme dotaz upravili nasledovne:

mysql> SELECT ziak.priezvisko, ziak.meno, SUM(dochadzka.cas) AS Celkovy_pocet_hodin, dochadzka.datum_vymeskania FROM ziak, dochadzka WHERE ziak.id_uzivatela = dochadzka.id_uzivatela GROUP BY ziak.id_uzivatela;

Môžeme ešte zoradiť mená žiakov podľa abecedy.

mysql> SELECT ziak.priezvisko, ziak.meno, SUM(dochadzka.cas) AS Celkovy_pocet_hodin, dochadzka.datum_vymeskania FROM ziak, dochadzka WHERE ziak.id_uzivatela = dochadzka.id_uzivatela GROUP BY ziak.id_uzivatela ORDER BY ziak.priezvisko;

priezvisko	meno	Celkovy_pocet_hodin	datum_vymeskania
Cankovsky	Fero	3	2011-06-15 16:01:16
Dankovsky	Dano	4	2011-06-15 16:00:01
Godunov	Boris	1	2011-06-15 18:22:44
Jakubik	Marian	2	2011-06-15 16:01:06
Jankovsky	Jano	0	2011-06-15 18:23:13
Novota	Marek	3	2011-06-15 16:00:41
Petrovsky	Peter	1	2011-06-15 18:22:00

Rovnaký výsledok ako pri klauzule WHERE dostaneme pomocou klauzuly INNER JOIN. Operácia spájania tabuliek sa volá JOIN, a najbežnejšie spájanie tabuliek je INNER JOIN, pri ktorom vo výsledku vyberáme tie riadky, v ktorých je splnené kritérium spájania.

mysql> SELECT ziak.priezvisko, ziak.meno, SUM(dochadzka.cas) AS Celkovy_pocet_hodin, dochadzka.datum_vymeskania FROM ziak INNER JOIN dochadzka ON ziak.id_uzivatela = dochadzka.id_uzivatela GROUP BY ziak.id_uzivatela ORDER BY ziak.priezvisko;

Do triedy pristúpili dvaja noví žiaci, ktorých pridáme do tabuľky žiak prostredníctvom príkazu INSERT.

mysql> INSERT INTO ziak VALUES ('', 'student8','Ernest','Mraz', '4.AE','Studena 35 CA','mraz@spsknm.sk', 'asewGkf', NOW(),25, 'turistický krúžok');
INSERT INTO ziak VALUES ('', 'student9','Jakub','Leto', '4.AE','Jazerova 651 KNM',' letko@post.sk', '2WNREdddg', NOW(),3, 'šachový krúžok');

Poznáme tiež ľavé vonkajšie spojenie LEFT OUTER JOIN. Pri tomto spojení sa vo výsledku objavia všetky riadky ľavej tabuľky z klauzuly ON, aj keď sa v pravej tabuľke nenájde zhodný riadok (v takom prípade však všetky stĺpce vybrané z pravej tabuľky budú niesť hodnotu NULL). LEFT OUTER JOIN môžeme napríklad použiť na vypísanie všetkých žiakov a ich absencií, a to aj tých, ktorí ešte vyučovanie nevymeškali.

mysql> SELECT ziak.priezvisko, ziak.meno, SUM(dochadzka.cas) AS Celkovy_pocet_hodin, dochadzka.datum_vymeskania FROM ziak LEFT OUTER JOIN dochadzka ON ziak.id_uzivatela = dochadzka.id_uzivatela GROUP BY ziak.id_uzivatela ORDER BY ziak.priezvisko;

Rovnako tiež pomocou RIGHT OUTER JOIN, by sme vypísali len údaje ktoré sa nachádzajú v druhej tabuľke. Pri vytváraní tabuľky dochádzka nemôžeme nespomenúť cudzí kľúč FOREIGN KEY, ktorý definuje vzťah k primárnemu kľúču rovnakej alebo inej tabuľky. Našu tabuľku by sme v tomto prípade vytvorili nasledovne:

mysql> CREATE TABLE dochadzka (id_vymeskania MEDIUMINT UNSIGNED NOT NULL AUTO_INCREMENT PRIMARY KEY, id_uzivatela int NOT NULL, cas INT(2), datum_vymeskania DATETIME , FOREIGN KEY fk_id_uzivatela (id_uzivatela) REFERENCES ziak(id_uzivatela))

7. Výpočty a funkcie

7.1 Agregačné funkcie

Agregačné funkcie sú typy dopytov, ktoré spracovávajú hodnoty z celých stĺpcov tabuľky. Zadávame ich do zoznamu výstupných položiek za príkaz SELECT.

AVG(nazov_stĺpca) – vráti aritmetický priemer z množiny hodnôt v stĺpci.
COUNT(DISTINCT nazov_stĺpca) – spočíta počet jedinečných hodnôt v stĺpci.
COUNT(nazov_stĺpca) – spočíta počet hodnôt v stĺpci.
MAX(nazov_stĺpca) – vráti maximálnu hodnotu z množiny údajov.
MIN(nazov_stĺpca) – vráti minimálnu hodnotu z množiny údajov.
SUM(nazov_stĺpca) – vráti súčet hodnôt v množine údajov.
STD(nazov_stĺpca) – vypočíta štandardnú odchýlku nazov_stĺpca.
STDDEV(nazov_stĺpca) – to iste čo STD(nazov_stĺpca).
VARIANCE(nazov_stĺpca) – vypočíta rozptyl nazov_stĺpca.

7.2 Matematické funkcie

Najširšiu množinu funkcií zastupujú číselné funkcie. Ich argumentom býva väčšinou číselná hodnota a tieto funkcie zas vracajú jednu číselnú hodnotu.

ABS(n) – absolútna hodnota čísla n.
ACOS(n) – inverzný kosínus uhla v radiánoch.
ASIN(n) – inverzný sínus uhla v radiánoch.
ATAN(n) – inverzný tangens uhla v radiánoch.
CEILING(n) – zaokrúhli na najbližšie celé číslo, ktoré nie je menšie ako n.
FLOOR(n) – zaokrúhli na najbližšie celé číslo, ktoré nie je väčšie ako n.
DEGREES(n) – prevedie radiány na stupne.
COS(n) – kosínus uhla zadaného v radiánoch.
SIN(n) – sínus uhla zadaného v radiánoch.
TAN(n) – tangens uhla zadaného v radiánoch.
EXP(n) – vráti výsledok e^n.
LOG(n) – prirodzený logaritmus čísla n.
LOG10(n) – dekadický logaritmus čísla n.
POWER(m, n) – vráti výsledok m^n.
ROUND(n, m) – zaokrúhli číslo n na m desatinných miest.
RAND() – vráti náhodné číslo z intervalu 0.0 až 1.0.
PI() – vráti 3.1415927.
SQRT(n) – vráti druhú odmocninu čísla n.
SIGN(n) – vráti 1, ak je n kladné, 0, ak je n rovné 0, -1, ak je n záporné.
TRUNCATE(n) – odstráni desatinnú časť čísla n.
MOD(x,y) – vráti zvyšok po celočíselnom delení čísla x číslom y.

7.3 Funkcie pre prácu s reťazcami

LOWER(reťazec) – konvertuje reťazec na malé písmená. Používa sa napríklad pri vyhľadávaní nezávislom na veľkosti písmen.
UPPER(reťazec) – konvertuje reťazec na veľké písmená.
RIGHT(reťazec, dĺžka) – vráti určený počet znakov z pravej strany (*z konca*) reťazca.
TRIM(reťazec) – odstráni medzery z ľavej a pravej strany reťazca.
LTRIM(reťazec) – odstráni medzery z ľavej strany (*zo začiatku*) reťazca.
RTRIM(reťazec) – odstráni medzery z pravej strany (*z konca*) reťazca.

REPLACE(reťazec1, reťazec2, reťazec3) – každý výskyt reťazca2 v reťazci1 nahradí reťazcom reťazec3.
SUBSTRING(reťazec, začiatok, dĺžka) – vráti podreťazec vstupného reťazca definovanú začiatkom a počtom znakov.
REVERSE(reťazec) – vracia reťazec, v ktorom sú znaky v opačnom poradí.
BIN(n) – vracia hodnotu n v binárnom tvare ako reťazec.
CONCAT(**reťazec1, reťazec2, reťazec3,...**) – spojí reťazce.
MD5(n) – vypočíta kontrolný súčet MD5 pre reťazec n.
REPEAT(s, n) – n krát zopakuje s.

7.4 Funkcie pre prácu s dátumom a časom

Veľmi často potrebujeme v databázach uchovávať aj údaje o čase a dátume. Tieto zaznamenávajú rôzne udalosti v konkrétnom okamihu.

CURDATE() – vráti aktuálny dátum vo formáte RRRR-MM-DD.
CURTIME() – vráti aktuálny čas vo formáte HH:MM:SS.
NOW(), SYSDATE(), CURRENT_TIMESTAMP(), LOCALTIME() – vráti dátum a čas vo formáte RRRR-MM-DD HH:MM:SS.
DATE_ADD(dátum, INTERVAL vyraz typ) – pripočíta k dátumu špecifikovanú hodnotu, synonymom je funkcia ADDATE().
EXTRACT(typ FROM dátum) – vyberie časť dátumu (*sekundy – SECOND, dni – DAY, mesiac – MONTH, ...*).
YEAR(dátum) – vráti rok zo zadaného dátumu.
QUARTER(dátum) – vráti číslo kvartálu v roku (1 – 4).
MONTHNAME(dátum) – vráti názov mesiaca v angličtine.
DAYOFWEEK(dátum) – vráti poradové číslo dňa v týždni (1 – nedeľa, ..., 7 – sobota).
DAYNAME(dátum) – vráti názov dňa v týždni v angličtine.
WEEKDAY(dátum) – vráti poradové číslo dňa v týždni (0 – pondelok, ..., 6 – nedeľa).
WEEK(dátum, 1) – vráti poradové číslo týždňa v roku.
HOUR(čas) – vráti hodnotu hodín zo zadaného času.
MINUTE(čas) – vráti hodnotu minút zo zadaného času.
SECOND(čas) – vráti hodnotu sekúnd zo zadaného času.
SEC_TO_TIME(počet_sekund) – prevedie počet sekúnd od polnoci na čas v tvare HH:MM:SS.
TIME_TO_SEC(čas) – prevedie zadaný čas na počet sekúnd po polnoci.
TO_DAYS(dátum) – vráti počet dní od počiatku nášho letopočtu.
DATE_FORMAT(dátum, formát) – vyjadrí kalendárne dáta alebo časy v inom než obvyklom formáte MySQL.

8. Administrácia používateľských práv

Keď chceme nastavovať prístupové privilégia a práva, musíme sa najskôr prepnúť do databázy MySQL.
mysql> USE mysql;
V tabuľke user sú uložene všetky informácie o používateľoch databázového serveru. Štruktúru a implicitné hodnoty v stĺpcoch privilégií vypíšeme pomocou príkazu:
mysql> DESC user;
Prvé tri stĺpce Host, User a Password určujú oblasť pôsobnosti. V stĺpci Host môže byť buď meno počítača alebo jeho IP adresa. Nového používateľa môžeme vytvoriť napríklad príkazom.
mysql>INSERT INTO user VALUES('localhost', 'elektrikar', PASSWORD('hesloelektro'), 'Y', 'Y', 'Y', 'N', 'Y', 'N', 'N', 'N', 'Y', 'Y', 'Y', 'N', 'N', 'N','N', 'N', 'N', 'N', 'N', 'N', 'N', 'N', 'Y', 'Y', 'Y', 'Y', 'N', 'N', 'N', 'Y', 'N', 'Y', 'N', 'Y', 'N', 'N');

Kde z výpisu DESC user, prideľujeme prostredníctvom Y/N príslušne oprávnenia. (*musíme vyplniť všetky oprávnenia*). V prípade, že by sme chceli povoliť len niektoré z oprávnení (*Reload a Select*) použili by sme nasledovný zápis:
mysql>INSERT INTO user (Host, User, Password, Reload_priv, Select_priv) VALUES ('%', 'student', 'heslostudenta', 'Y', 'Y');
Keď potrebujeme niektorému používateľovi napríklad zmeniť heslo použijeme príkaz v tvare:
mysql>UPDATE user SET PASSWORD= PASSWORD('hesloveslo') WHERE user= 'student';
Pre odobranie práv je určený príkaz REVOKE. Nasledujúci príklad odoberie používateľovi "student" právo "Select".
mysql>REVOKE SELECT ON *.* FROM student;
Okrem priameho prístupu do tabuľky user, môžeme jednotlivé privilégia prideľovať i prostredníctvom príkazu GRANT.
mysql> GRANT select, insert, update, delete, index, alter, create, drop ON *.* to student;
Používateľa vymažeme nasledovne:
mysql>DELETE FROM user WHERE user= 'student';

Podrobnú dokumentáciu k MySQL nájdeme na adrese:
http://dev.mysql.com/doc/refman/5.6/en/index.html.

9. PHP a databáza MySQL

Funkcia mysql_connect inicializuje spojenie s DB MySQL,
mysql_connect("nazov_mysql_servera", "užívateľ", "heslo");
nazov_mysql_servera - je meno MySQL servera poprípade počítača, na ktorom beží MySQL.
užívateľ - je používateľ MySQL servera, pod ktorým sa prihlasujeme.
heslo - je naše heslo k DB v MySQL.
Príklad:
Pokiaľ je na serveri vytvorený používateľ 'student' s heslom 'hesloveslo', potom by prihlásenie na počítač localhost vyzeralo nasledovne:
mysql_connect("localhost", " student", "hesloveslo");

mysql_pconnect otvorí do MySQL trvalé spojenie.
mysql_pconnect("nazov_mysql_servera", "uzivateľ", "heslo");

mysql_select_db sa prepína do zvolenej pracovnej databázy.
mysql_select_db("meno_databázy", "spojenie");
meno_databázy – je meno databázy, s ktorou chceme pracovať.
spojenie – je pripojenie k serveru MySQL (návratová hodnota funkcie mysql_connect).

mysql_query odosiela náš SQL príkaz k spracovaniu - SELECT, INSERT, UPDATE alebo DELETE.
mysql_query("SQL_príkaz", "spojenie");
SQL_ príkaz - je korektný SQL príkaz.
spojenie - je identifikátor aktuálneho spojenia.

mysql_result získava vrátené záznamy z určitého výsledku.
mysql_result(vysledok);

Existujú ďalšie MySQL funkcie napríklad:

mysql_create_db("meno_databázy", "spojenie");
Vytvára databázu.

mysql_drop_db("meno_databázy", "spojenie");
Odstraňuje databázu.
mysql_close("spojenie");
Ukončí aktuálne spojenie s MySQL.
mysql_change_user() - zmení prihlasovaného používateľa.
mysql_data_seek() - presunie výsledkový ukazovateľ.
mysql_fetch_field() - získa pole z výsledku dotazu.
mysql_field_name() - získa názvy polí.
mysql_field_table() - získa názov tabuľky, v ktorej je pole.
mysql_field_type() - získa typ poľa.
mysql_list_dbs() - zoznam databáz dostupných na serveri.
mysql_list_fields() - zoznam polí vo výsledku.
mysql_list_tables() - zoznam tabuliek v databáze.
Podrobnú dokumentáciu nájdeme na adrese http://www.php.net/manual/en/ref.mysql.php.

9.1 Pridanie záznamu do tabuľky

Príklad: insert.php

```
<?php
// Pripojenie k DB serveru
$result = mysql_pconnect("localhost","root","");
//mysql_pconnect("nazov_mysql_servera", "uzivateľ", "heslo");
if (!$result)
{
 echo "Nepodarilo sa pripojiť k DB serveru";
 exit;
}
// Zvolenie DB skola
$result = mysql_select_db("skola");
if (!$result)
{
 echo "Nepodarilo sa vybrať DB skola";
 exit;
}
// Pridanie záznamu do tabuľky ziak
$query = "INSERT INTO ziak VALUES ('', 'student27','Matej','Novak', '4.BE','Letna 821 ZA','novakm@spsknm.sk',
'745w76hH',NOW(),5,'sach')";
$result = mysql_query($query);
if (!$result)
{
 echo "Nepodarilo sa vložiť údaje do tabuľky person";
 exit;
}
echo "Skript prebehol v poriadku<br />";
?>
```

Skript uložíme ako insert.php a vložíme do príslušného priečinka (*priečinok htdocs v programe XAMPP*).

9.2 Výber záznamu z tabuľky

Príklad: select.php

```
<?php
// Pripojenie k DB serveru
$result = mysql_pconnect("localhost","root","");
if (!$result)
{
 echo "Nepodarilo sa pripojiť k DB serveru";
 exit;
}
```

```
// Zvolenie DB skola
$result = mysql_select_db("skola");
if (!$result)
{
  echo "Nepodarilo sa vybrať DB skola";
  exit;
}
// Vybranie záznamu z tabuľky ziak
$query = "select * from ziak";
$result = mysql_query($query);
// Počet nájdených záznamov
$num_results = mysql_num_rows($result);

// Výpis záznamov
for($i=0; $i<$num_results; $i++)
{
  $row = mysql_fetch_array($result);
  echo "ID = ".$row["id_uzivatela"]."<br />";
  echo "Nick = ".$row["nick"]."<br />";
  echo "Meno = ".$row["meno"]."<br />";
  echo "Priezvisko = ".$row["priezvisko"]."<br />";
  echo "Trieda = ".$row["trieda"]."<br />";
 echo "Adresa = ".$row["adresa"]."<br />";
 echo "Mail = ".$row["email"]."<br />";
 echo "Heslo = ".$row["heslo"]."<br />";
 echo "Zaregistrovanie = ".$row["datum_zaregistrovania"]."<br />";
 echo "Vzdialenosť = ".$row["vzdialenost_km"]."<br />";
 echo "Krúžok = ".$row["kruzok"]."<br />";
}
echo "Skript prebehol v poriadku<br />";
?>
```

10. Aplikačné rozhranie -tenký a hrubý klient

Tenký klient (*Thin client*) je klientská časť aplikácie zabezpečujúca prezentáciu a interakciu používateľa s aplikáciou. Tenký klient teda neobsahuje žiadnu aplikačnú logiku, ale pristupuje k aplikačnému serveru, ktorý mu logiku celej aplikácie sprostredkúva. Naopak hrubý klient obsahuje väčšinu aplikačnej logiky. V skutočnosti musí obsahovať nejakú logiku aj tenký klient, avšak ide o zanedbateľnú časť oproti hrubému klientovi. Príklad tenkého klienta môžeme určiť v prípade webových aplikácií. Webová aplikácia využíva webový prehliadač ako tenkého klienta.

Hrubý klient (*Rich client*) je klientská aplikácia, ktorá beží na lokálnom počítači a teda môže naplno využívať výkon jeho hardvéru a možnosti operačného systému. Pre svoju činnosť ale využíva údaje a službu nejakého serveru. Výhodou takéhoto riešenia je, že aplikácia pracuje v oboch režimoch *offline* i *online*, nevýhodou je že pre každú platformu musíme aplikáciu vyvíjať zvlášť.

11. Internet

Hovorovo net alebo sieť je verejne dostupný celosvetový systém vzájomne prepojených počítačových sietí, ktoré prenášajú dáta vo forme paketov za použitia štandardizovaného Internet Protocolu (*IP*) a mnohých ďalších protokolov. Pozostáva z tisícok menších komerčných, akademických, vládnych a vojenských sietí. Slúži ako prenosové médium pre rôzne informácie a služby ako napr. elektronická pošta, chat a systém vzájomne prepojených web stránok a dokumentov World Wide Webu (*WWW*).

11.1 Architektúra klient/server

Hlavnou myšlienkou tejto technológie je, že údaje by sa mali spracovávať tam, kde sú uložené. Ak chce používateľ pracovať s nimi, dochádza k rozdeleniu činnosti medzi základným počítačom – serverom, na ktorom sú uložené údaje, a používateľskou stanicou – klientom. Teda kompetencie sa delia medzi dva procesy – klient je zodpovedný za spracovanie určitej úlohy, pre ktorú vyžaduje služby od druhého procesu – servera.

-Proces klient (*front-end*) –väčšinou prebieha na používateľských počítačoch. Jeho úlohou je komunikovať jednak s používateľom, ale aj so serverom. Nemá prístup k databáze, slúži ako sprostredkovateľ, ktorý prijíma požiadavky klienta a transformuje ich do podoby, ktorá je zrozumiteľná pre server. Server zabezpečuje manipuláciu s údajmi.
- Proces server (*back-end*) – je proces, ktorý obstaráva všetky databázové operácie. Môže bežať na serveri alebo aj na používateľskej stanici, ktorá je v tomto prípade aj serverom. Podstata tejto architektúry je v tom, že údaje sú uložené a spravované na serveri a používateľ pristupuje k nim prostredníctvom klientských staníc.

11.2 Doména

IP adresa je číslo počítača v sieti, podobne ako má telefón svoje telefónne číslo. IP adresa sa prekladá do ľudskej reči na doménové meno. Doména je jedno slovo oddelené bodkami v doménovom mene. Správne zvolená doména je základom úspešnej internetovej prezentácie.

Doména I., II. a III. úrovne
Základným prvkom je doména I. úrovne, čo je najvyššia doména. Inak je nazývaná aj Top Level doménou (*TLD*) alebo národnou doménou. Domény nižších úrovní sa vytvárajú pod TLD smerom sprava do ľava a jednotlivé úrovne sú od predchádzajúcich oddelené bodkou. Máme doménu "spsknm.sk". Jedná sa o doménu II. úrovne, ktorá bola vytvorená v TLD ".SK". Na tejto doméne teraz môžeme vytvoriť ľubovoľné množstvo domén III., IV. a nižších úrovní, napr. "informatika.spsknm.sk", ďalej môže vzniknúť doména "novak.informatika.spsknm.sk " atď.

11.3 Registrácia domény

Registrovať si je možné len také doménové meno, ktoré je voľné a neodporuje registračným pravidlám platným vo zvolenej Top Level doméne (TLD). Registráciou dôjde k vytvoreniu záznamu o doméne v príslušnej národnej databáze a vo väčšine TLD zároveň i k zápisu domény do národných nazvových serverov (DNS) a tým k jej sprevádzkovaniu. V niektorých TLD sú domény funkčné až po úhrade registračného poplatku (ako napr. v doméne .sk alebo .cz). Výška poplatkov za registráciu sa v rôznych TLD líši, ceny stanovujú správcovia príslušných TLD. Doménové meno, ktoré je už registrované, je na vopred zvolenú dobu vyhradené a nemôže si ho zaregistrovať nikto iný. Voľnosť konkrétnej domény si môžeme overiť v databáze príslušného správcu TLD. Vybrať dobrú doménu je veľmi ťažké, keďže väčšinu dobrých názvov už niekto vlastní. Dobrá doména je krátka, už z názvu domény je možné odvodiť, o aký web ide, končí národnou koncovkou (.sk), prípadne inou často používanou (.com, .eu), dá sa ľahko zapamätať, neobsahuje ťažko vysloviteľné časti ako napr. dve rovnaké písmená po sebe. Toto sú však len všeobecné zásady, ktoré neplatia na všetky projekty. Napríklad ak chceme propagovať nejakú značku, aj doména by sa mala volať rovnako. Priebeh registrácie prebieha tak, že občan SR požiada o registráciu domény vlastnými prostriedkami národného registrátora alebo sprostredkovane (*za určitú odmenu*) cez akreditovaného registrátora, napr. www.hosting1.sk. Registrátor ju následne zašle SK-NIC www.sk-nic.sk . Ročný poplatok je 19.90 EUR bez DPH za doménu (*cena nad 19.90 je potom logicky ziskom sprostredkovateľa*). Registrátorov domén je na Slovensku samozrejme množstvo, rozhodujúcim kritériom je nie len cena, ale aj serióznosť "vami zvoleného zástupcu" – registrátora. Ak chceme vedieť, kto je

držiteľom domény, pomôže nám služba WHOIS, napríklad na stránke www.allwhois.com alebo www.webhosting.info. Rovnako sa môžeme stať vlastníkom aj zahraničnej domény, ak sa obrátime na takého registrátora, ktorý ju pre nás za určitý poplatok zaregistruje.

11.4 Stav domény

Pri doménach sa nachádzajú rôzne skratky a značky, takýchto značiek je množstvo, vyberieme aspoň pár s ktorými sa môžeme na Slovensku stretnúť:
DOM_OK -táto skratka znamená, že je všetko v poriadku. Doména existuje, je zaplatená, má registrátora a je plne funkčná.
DOM_EXP -doména síce existuje, ale je nezaplatená. To znamená, že už nie je funkčná.
DOM_DEL -doménový záznam je zrušený a doména nemá registrátora.
Ďalšie stavy domén v systéme SK-NIC nájdeme:
https://www.sk-nic.sk/documents/stavy_domen.html

Na SK-NIC si môžeme všimnúť, že väčšinu dobrých názvov už niekto vlastní a vybrať vhodný názov je veľmi zložité. Doménu si môžeme aj zakúpiť napríklad na stránke: www.predajdomen.sk. Ponúka sa nám však aj ďalšia možnosť a tou sú expirujúce domény.

Expirujúce domény
Každým dňom sa rušia desiatky až stovky .sk domén, za ktoré ich povodní držitelia neuhradili poplatok na ďalší rok. O kvalitné domény je veľký záujem a ich ručná registrácia je prakticky nemožná. Po uvoľnení dobrej domény má záujem o jej registráciu viacero záujemcov. Čím lepšia doména, tým väčší konkurenčný boj a teda menšia pravdepodobnosť ju úspešne získať. (*V súčasnej dobe väčšinu kvalitných domén zaregistrujú roboti.*) Existuje však veľa služieb, ktoré sa prostredníctvom svojho robota pokúsia túto registráciu urobiť za nás. V minulosti bol odchyt expirujúcich domén veľmi dobrou brigádou pre študentov, v dnešnej dobe je ručne odchytiť dobrú doménu skoro nemožné. Stránky zaoberajúce sa problematikou expirujúcich domén:
http://www.backorder.sk/, http://www.kupadomeny.sk/expirovane-domeny-zajtra,
http://www.domainer.sk/zakladne-domenove-pojmy-a-informacie-t441.html
http://www.domainer.sk/uzitocne-tools-a-stranky-k-domenam-t41.html
http://www.i-makings.com/expirujuciskdomeny
http://www.svetdomen.sk/clanky/2008-12-05/expirujuce-domeny-krasny-biznis/
http://zoznamdomen.sk/

Parkovanie domény
Jedná sa o spôsob zarábania na doméne. V prípade, že máme viac domén, na ktorých nechceme alebo zatiaľ nemáme vytvorenú žiadnu stránku, môžeme zvoliť niektorú z parkovacích spoločností. Parkovacia spoločnosť nám vygeneruje stránku podľa tzv. keywords (*kľúčové slovo/á ktoré udáme pri doméne*) a následne podľa záujmu o reklamu generovanú na našej stránke nám prináša zisk. (*Obyčajne sa jedná o malé čiastky, ktoré nám zväčša nepokryjú ročný poplatok za doménu.*)

11.5 Webhosting

Hosting web stránok sa nazýva aj webhosting a je to služba potrebná k tomu, aby naše stránky fungovali na našej doméne. Webhosting označuje službu, ktorá spočíva v prenájme priestoru na serveri, ktorý je potrebný pre funkčnosť internetovej stránky. V jednoduchosti môžeme hosting považovať za službu, pri ktorej si prenajímame miesto na disku a výkon počítača - servera, na ktorom bude stránka uložená. Neoddeliteľnou súčasťou hostingu sú aj ďalšie poskytované služby ako sú emaily, pravidelné zálohovanie údajov, ochrana pre vírusmi a spamom, aktualizácia softvéru a ďalšie.

11.6 Typy webhostingu

Podľa typu zdieľania systémových prostriedkov:
- Zdieľaný hosting - prevádzka viacerých webových stránok na jednom serveri. Zákazníci (*resp. správcovia webových stránok*) zdieľajú systémové prostriedky jedného fyzického servera.
- Vyhradený alebo "dedikovaný" hosting - prevádzka webovej stránky na samostatnom serveri, vyhradenom pre potreby jedného zákazníka (*jednej webstránky, portálu*...). Dedikovaný web hosting je výhodný najmä pre web stránky s vysokou návštevnosťou a značným prenosom dát.

Podľa obchodného modelu:
- Komerčný alebo tiež "profesionálny" webhosting je platená služba. Za komerčný webhosting sa platia mesačné poplatky. Služba sa od freehostingu odlišuje predovšetkým absenciou reklamy a prítomnosťou prídavných služieb, ako je napr. technická podpora. Komerčný webhosting niekedy garantuje dostupnosť webovej stránky zmluvne. Veľmi zjednodušené porovnanie slovenských webhostingov môžeme nájsť na stránke www.porovnaniehostingov.sk. Výsledky z nezávislého merania českých a slovenských webhostingov http://mereni.kyblsoft.cz. Na stránke http://www.jakpsatweb.cz/webhosting.html zas odporúčania pri výbere webhostingu. (*Pri inštalácii niektorého z CMS systémov, treba brať ohľad aj na memory_limit a možností nastavení v .htaccess.*)
- Freehosting je webhosting zadarmo. Prevádzkovateľ freehostingovej služby poskytuje webhosting pre webové stránky zdarma. Spravidla pri zriadení freehostingu, dostávame doménu tretej úrovne tiež zdarma. Obvykle je však webová stránka doplnená prevádzkovateľom o reklamnú plochu, alebo sa reklama nachádza priamo v adrese webovej stránky. Iným modelom býva, že poskytovateľ podmieni využitie freehostingu zakúpením inej komerčnej služby. Zabezpečenie a technická podpora pri freehostingu býva spravidla nízka, až žiadna. Poskytovateľov freehostingu je veľké množstvo, kvalita poskytovaných služieb je však obyčajne dosť nízka. Zo slovenských poskytovateľov, kde stránka nie je doplnená prevádzkovateľom o reklamnú plochu môžeme využiť napríklad www.php5.sk. Lepšie sú na tom zahraničné spoločnosti poskytujúce freehosting, mnohé poskytujú lepšie služby, ako niektorí domáci poskytovatelia plateného webhostingu. Pekný zoznam porovnania zahraničných freehostingov môžeme nájsť na stránke www.free-webhosts.com. Je ťažké odporučiť nejaký konkrétny freehosting, keďže na každú internetovú aplikáciu máme rôzne požiadavky. Z tohto dôvodu sa musia líšiť aj požiadavky na freehosting.

12. Zásady tvorby www stránok

Čo je dôležité pri tvorbe web stránky
-Stránky by mali byť zaujímavé, prehľadné, s jednoduchou navigáciou a „príjemné na pohľad“, teda také, aby sa váš klient chcel zase na ne vrátiť.
-Musíte poznať svojich klientov a prispôsobiť stránky tomu, koho oslovujete.
-Preskúmať stránky konkurencie a získať prehľad o situácii na internetovom trhu.
-Atraktívny dizajn vytvára dojem profesionality.
-Web stránky treba pravidelne aktualizovať.

Zadefinovanie vhodných multimediálnych prvkov
Bežnou súčasťou každého počítača sa stali multimédiá. Ich význam je v tom, že nám dávajú možnosť jednoduchou a prístupnou formou získať informácie, zabaviť sa, uľahčujú prácu s počítačom a dávajú možnosť interaktivity. V súčasnosti je k dispozícii široké spektrum nástrojov určených na tvorbu multimediálnych aplikácií, z ktorých si používateľ môže vybrať.

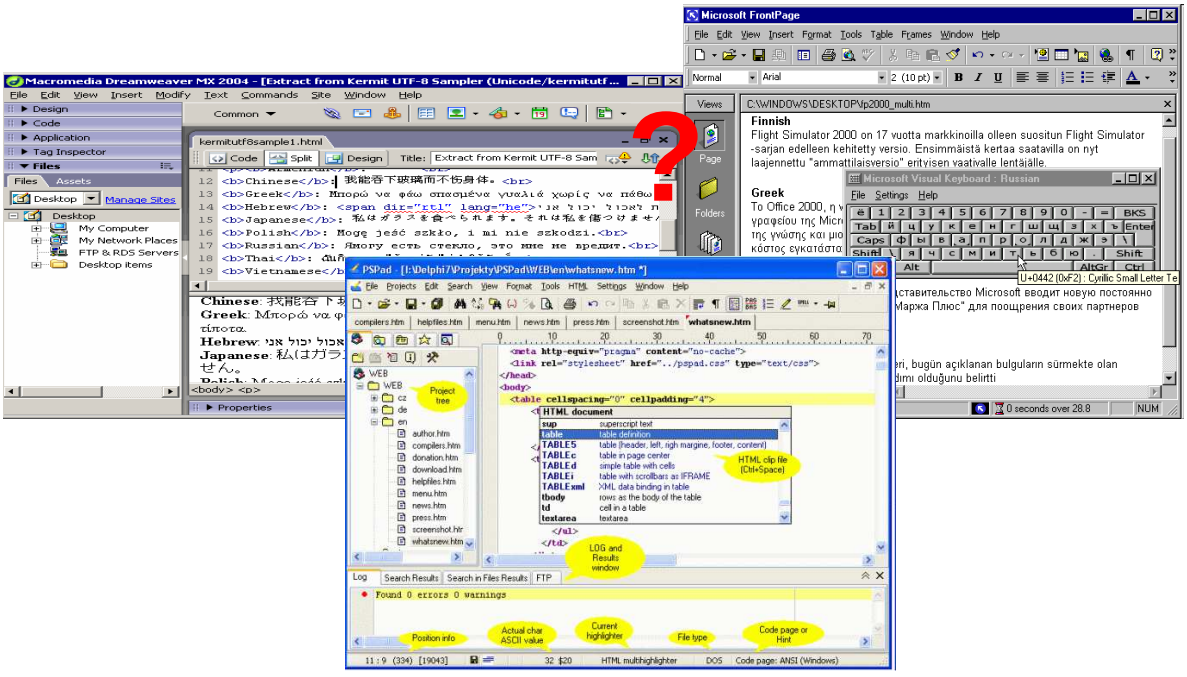
Zvuky a videosekvencie
V mnohých webových stránkach majú veľmi dôležitú úlohu. Môžu poskytovať informáciu iným spôsobom ako text na obrazovke čím sa stáva stránka viac interaktívnou. Zbytočné použitie však pôsobí rušivo, zaberá miesto a spomaľuje načítavanie sa stránky.

Animácie
Predmet, ktorý je v pohybe, najviac upúta pozornosť. Animácia v multimédiách však nepomáha len na upútanie, ale slúži aj ako prostriedok, ktorý nesie informáciu. Dobre navrhnutá animácia dokáže viac osloviť a zaujať ako obrázok či statický text. S animáciami však treba pracovať veľmi opatrne, aby namiesto prinášania informácie nepôsobili skôr rušivo a neodrádzali používateľa. Prínosom animácie je, že dokáže zobraziť aj to, čo by inak obrázkom nebolo možné.

Výber nástrojov potrebných pre tvorbu
Na tvorbu webových stránok je možné použiť veľké množstvo programov určených na tvorbu a publikovanie multimediálneho obsahu v prostredí internetu.

- Editory na tvorbu webových stránok a aplikácií (*FrontPage*, *Dreamweaver*) a samozrejme aj (*Adobe Flash*, *Adobe Director*,...) a v obmedzenej miere aj prezentačný softvér (*Microsoft Powerpoint,..).*

Jednotlivé programy sa líšia spôsobom tvorby multimediálneho obsahu, jeho prezentáciou, možnosťami interaktivity a spätnej väzby výslednej aplikácie, implementovaním štandardov a schopnosťou importu obsahu z rôznych formátov.



13. HTML



(*Hyper Text Markup Language*) je značkovací jazyk, ktorý je základom pre tvorbu internetových stránok. Slúži na presné a stručné vyjadrenie toho, čo chceme zobraziť v programe všeobecne nazvanom prehliadač. Jazyk HTML nie je nikdy kompilovaný do binárnej alebo inej podoby. Súbory, ktoré sú vždy v binárnom formáte, ako sú obrázky alebo zvuky, nie sú umiestnené priamo do súboru HTML, ale sú uvedené len odkazy na tieto súbory. Sémantika a syntax jazyka HTML je definovaná štandardom HTML. Vývoj štandardu

HTML je riadený konzorciom W3C. HTML súbor je obyčajný text obalený značkami nazývanými niekedy tiež tagy, ktoré prehliadač nezobrazuje. Určujú ako bude text vyzerať, jeho formu. Značky sú uzatvorené v ostrých zátvorkách “<>“. Poznáme párové značky, a nepárové značky.

Párové značky (tučný text sa zobrazí: tučný text) – sú tie, ktoré medzi seba uzatvárajú vlastný obsah a priraďujú mu danú hodnotu. Značka na ľavej a pravej strane výrazu je rovnaká s jedným rozdielom, že uzatvárajúca značka má za zátvorkou lomku.

Nepárové značky (<img src=”obr.gif”>) – umiestňujú do stránky jediný element. Je to napríklad pozadie, vodorovná čiara alebo vloženie obrázka. Popis daného elementu je priamo vložený medzi ostré zátvorky.

Prvky majú definované dodatočné vlastnosti nazývané atribúty. Pomocou nich definujeme farbu, písmo atď. Jazyk ignoruje veľké a malé znaky. Na tvorbu HTML stránky sa môžu využívať rôzne Wysiwyg editory (*Dreamwear, Microsoft Expression Web Designer*....), do ktorých sa vpíše text, vloží sa obrázky a editor sám doplní všetky HTML značky (*tagy*). K nevýhodám patrí nekompatibilita s rôznymi verziami internetových prehliadačov, HTML súbor môže byť pri interpretácii v rôznych prehliadačoch (*Internet Explorer, Mozilla Firefox, Opera*...) zobrazený rozdielne. Momentálne je stabilná verzia HTML 4.01. Pôvodne sa predpokladalo, že ďalšie verzie jazyka HTML už nebudú a jeho pokračovateľom bude jazyk XHTML. V roku 2007 však došlo k založeniu pracovnej skupiny, ktorá má za cieľ vytvoriť novú verziu HTML, ktorá ponesie označenie HTML 5 a jej XML variant XHTML 5. HTML5 je nová generácia jazyka HTML, prinášajúca celú sadu nových elementov, ktoré významne zjednodušujú štruktúru stránok. Medzi najvýraznejšie novinky patrí API pre kreslenie grafiky v okne prehliadača, nové elementy <audio> a <video>, prostredníctvom ktorých je jednoduché prehrávať audio vizuálne súbory v prehliadači bez nutnosti inštalácie softvéru tretích strán. (*Oficiálne ešte nebola vydaná finálna špecifikácia HTML 5, počíta sa s tým, že sa bude ešte dlhú dobu ladiť.)* Dokumentáciu k HTML nájdeme napríklad na adrese:
http://www.jakpsatweb.cz/html/, http://programujte.com/?akce=clanek&cl=2006021401-referencni-prirucka-jazyka-html, http://www.emasek.chytrak.cz/prirucka/?id=uvod/, http://www.bajk.sk/02/html/index.html, http://w-web.sweb.cz/, http://fornax.elf.stuba.sk/~singer/netdiakr/vki/html/spec.html



13.1 XHTML



Rozšíriteľný hypertextový značkovací jazyk (*Extensible Hypertext Markup Language*) je značkový jazyk podobný HTML, ale s prísnejšou syntaxou. Zatiaľ čo HTML je aplikáciou SGML, veľmi pružného značkového jazyka, XHTML je aplikáciou XML, obmedzenej podmnožiny SGML. XHTML 1.0 sa stalo odporúčaním W3C 26. januára 2000. Dokumenty XHTML sú súčasne dokumentmi XML, takže môžu byť zobrazované, upravované a validované nielen prehliadačmi a editormi, ale tiež XML nástrojmi. XHTML dokumenty môžu byť vytvorené rovnako ako predtým s existujúcimi editormi s podporou HTML 4 a tiež v nových editoroch, ktoré podporujú jazyk XHTML priamo. Vývojári dokumentov a editorov neustále objavujú nové spôsoby vyjadrenia svojich myšlienok prostredníctvom nových značiek. V XML je pomerne jednoduché pridať nové značky alebo ich atribúty. Rodina XHTML je navrhnutá tak aby prijatie takýchto rozšírení bolo možné prostredníctvom XHTML modulov. Tieto moduly umožnia použitie súčasných i nových vlastností pri vytváraní dokumentov. Stále sa objavujú nové možnosti prístupu k Internetu. Jazyk XHTML je navrhnutý s ohľadom na kompatibilitu všetkých možných editorov a prehliadačov. Vďaka novým mechanizmom profilácie dokumentov sú servery a proxy servery schopné rýchlejšie a lepšie spracovať obsah dokumentu. Vďaka štandardizácii je možné vytvoriť

dokument podľa XHTML, ktorý je použiteľný vo všetkých editoroch a prehliadačoch. Momentálne sa stále najčastejšie používa XHTML verzie 1.0 napriek tomu, že sa už sa objavila aj nová verzia XHTML 1.1. Táto verzia sa veľmi podobá na XHTML 1.0 Strict, vytvoriť stránku validnú podľa XHTML 1.1 je z tohto dôvodu viac prácne. Dokumentáciu k XHTML nájdeme napríklad na adrese: http://www.kirp.chtf.stuba.sk/~cirka/vyuka/xhtml/kap1.php#sec2
http://www.webtvorba.cz/xhtml/

13.2 Rozdiely medzi HTML a XHTML

Na prvý pohľad sa XHTML 1.0 oproti HTML 4.01 nelíši. Oba obsahujú tri typy DTD: Strict, Transitional a Frameset. Oba obsahujú rovnaký počet atribútov aj elementov. Až v XHTML 1.1 boli úplne odstránené niektoré elementy, ktoré sme mohli nájsť v Transitional a Frameset verziách (X)HTML. Ale aby XHTML 1.0 stránka bola validná, musia sa dodržiavať nasledovné pravidlá (*pri verzii XHTML 1.0 strict alebo XHTML 1.1 sú tieto pravidlá prísnejšie*):
-Všetky značky musia byť ukončené, a to vrátane nepárových, ako sú <img>, <link>, <br>, <hr> alebo <meta>. Zápis môže mať viac podôb. Buď použijeme najpoužívanejšie <img />, <img/> alebo klasické <img></img>. Druhý spôsob bez medzery sa neodporúča používať, pretože staršie prehliadače by mohli ignorovať posledný parameter (*ak je zadaný*). Tretí sa nesmie používať, ak zasielame XHTML dokument typom text/html. Všetky značky musia byť správne vnorené a nesmú sa prekrývať.
-V XHTML na rozdiel od HTML musia byť všetky značky a ich atribúty zapísané malými písmenami, a to z toho dôvodu, že sú takto deklarované v odkazovanej DTD a XHTML (a XML) je *case sensitive*, teda záleží na veľkosti písma. Ak by sme si deklarovali vlastnú DTD, môžeme používať aj veľké písmená.
-Všetky hodnoty atribútov musia byť uzavreté do úvodzoviek.
-Dokument musí začínať XML deklaráciou. Jej použitie nie je povinné, ak je dokument kódovaný v UTF-8 (*alebo UTF-16*) alebo ak určujeme kódovanie vyšším protokolom.
-Ak potrebujeme pracovať s rámcami, môžeme pre hlavnú stránku definujúcu rozloženie rámcov deklarovať XTHML 1.0 Frameset a pre jednotlivé stránky XHTML 1.0 Transitional.

13.3 Štruktúra HTML

Základná štruktúra HTML stránky:

```
<!DOCTYPE html>
<html>
<head>
 hlavička stránky
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title>Titul stránky</title>
</head>
<body> obsah stránky
</body>
</html>
```

Prvou značkou je **<html>**, ktorá prehliadaču hovorí, že nasledujúce dáta má spracovávať pomocou syntaktických a štrukturálnych pravidiel platných pre jazyk HTML. Odpovedajúca uzatvárajúca značka **</html>** sa nachádza na konci súboru. Značky **<head>** a **</head>** označujú hranice hlavičkovej oblasti súboru, ktorá obsahuje údaje, ktoré sa na stránke priamo nezobrazujú. Značky **<title>** a **</title>** definujú text, ktorý sa zobrazuje v titulkovom pruhu prehliadača. Telo značky title je dobré vyplniť rozdielne pre každú novú stránku, malo by obsahovať stručný popis obsahu stránky. (*Vyhľadávače kladú na túto značku veľký dôraz.)* Ďalšiu oblasť, ktorú môžeme na výpise vidieť, je uzatvorená medzi značkami **<body>** a**</body>**. Ako sa dá očakávať, tato oblasť predstavuje telo dokumentu a nachádza sa v nej obsah dokumentu - teda to, čo sa zobrazuje na stránke. Náš príklad preto vypíše text: **obsah stránky**.

!doctype

Uvádza špecifikáciu DTD, typ dokumentu, v ktorej verzii jazyka HTML sme vyhotovili našu webstránku. V značke môžeme ešte zvoliť Strict, Transitional, prípadne Frameset (*pri použití rámov*).

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN">
```

(*Odporúčanie W3C pre nové stránky je deklarovať HTML 5 ako aktuálnu najnovšiu verziu HTML: <!DOCTYPE html>*)

Link

Prepojuje HTML stránku s iným súborom, používa sa hlavne pre načítanie externého štýlu CSS.

```
<link rel="stylesheet" type="text/css" href="styl.css" />
```

Meta

Informácie o dokumente, metadata. Tieto meta tágy užívatel nevidí. Umiesňujeme ich do hlavičky webstránok (*po <head>*). META dáta sú informácie o dokumente rôzneho charakteru - autor, popis stránky, kľúčové slová, kódovanie, dátum vypršania platnosti a podobne. Pomocou značky meta zaistíme napríklad správne zobrazenie znakov:

```
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
```

Na Slovensku sa používajú tiež znakové sady **iso-8859-2** a **windows-1250**. (*Pri tvorbe stránky v niektorom z editorov je tiež potrebné označiť celý text a nastaviť kodovanie zvolené v meta značke.*)

Značky meta poskytujú tiež informácie pre vyhľadávače (napr. Google, Yahoo...), naša webstránka sa dá takto lahšie nájsť.

```
<meta name=”description” content=”Sem sa píše krátky text, ktorý popisuje stránku.” />
```

Je to informácia o stránke, ktorú vyhľadávače niekedy používajú ako popis odkazu vo výsledkoch hľadania.

```
<meta name=”keywords” content=”kľúčové slová na web stránke” />
```

Moderné vyhľadávače túto značku ignorujú.

Meta značka autor

```
<meta name=”author” content=”Sem sa píše meno autora stránky.” />
```

V prípade keď treba aktualizovať stránky po určitých časových intervaloch alebo zobrazenú stránku treba vymeniť na inú, používame meta značku refresh. (*Niektoré prehliadače túto značku nepodporujú alebo ju majú zakázanú*.)

```
<meta http-equiv="Refresh" content="8" />
```

Stránka sa každých 8 sekúnd obnoví, ako keby ste stlačili F5. Ak chcete po určitom čase presmerovať používateľa na inú adresu, využijeme rozšírenú možnosť:

```
<meta http-equiv="Refresh"content="5;url=http://www.google.sk" />
```

Za päť sekúnd po načítaní stránky sa používateľ bez vlastného pričinenia dostane na stránku http://www.google.sk.

Script

Pripojí sa ku stránke externý script, zvyčajne JavaScript.

```
<script type="application/javascript" src="skript.js">
```

Ukladanie stránky

Webová stránka v jazyku HTML je uložená ako súbor s koncovkou HTML. Môžeme sa stretnúť aj so staršou príponou HTM. Názov hlavnej stránky (*tá, ktorá sa zobrazí po zadaní adresy v tvare www.xy.....*) musí byť väčšinou index (*index.html, index.php, záleží od nastavenia servera*). Ďalšie stránky sa môžu volať napríklad o-mne.html, kvety.html, domace_zvierata.html. Do názvov súborov sa neodporúča dávať veľké písmená, medzery, špeciálne znaky *(!,=,; a pod.*) a písmená s diakritikou (*á,í,š,č,ž....).*

13.4 Zápis textu

Odstavce a zalomenie

Odstavec označujeme párovou značkou **<p>**. Ako všetky ostatné značky (okrem niektorých), vkladáme ju do tela stránky - medzi značky <body> a </body>.

```
<p>Text </p>
```

Značka <p> môže mať atribút **align**, ktorý určuje zarovnanie textu. Ten môže nadobúdať hodnoty **left** (*vľavo*), **right** (*vpravo*), **center** (*do stredu*) alebo **justify** (*do bloku*).

```
<p align="right">Odstavec zarovnaný doprava.</p>
```

V jazyku HTML sa ručné zalomenie nevkladá obvyklým spôsobom, ale značkou **
**.

```
<p> Druhý text<br /> pokračuje... </p>
```

Nadpisy

Existuje šesť úrovní nadpisov. Označujú sa značkami H1 až H6. Najhlavnejší a najväčší je H1. (*Vyhľadávače kladú na túto značku veľký dôraz.)*

```
<h1>Najväčší nadpis</h1>
<h2>Nadpis 2. úrovne</h2>
<h3>Nadpis 3. úrovne</h3>
<h4>Nadpis 4. úrovne</h4>
<h5>Nadpis 5. úrovne</h5>
<h6>Najmenší nadpis</h6>
```

Špeciálne znaky

Pokiaľ chceme v texte použiť nejaké špeciálne znaky, mali by sme ich nahradiť tzv. **entitami**. Keď chceme zobraziť zdrojový kód stránky a nepoužijeme entity, niektoré znaky sú chápané ako súčasť jazyka HTML a nezobrazili by sa podľa očakávania – dokonca by mohli spôsobiť aj nesprávnu interpretáciu zvyšku HTML dokumentu. Napríklad ostré zátvorky < a > by v texte nebolo možné priamo zobraziť. Znakové entity sa zapisujú ako sekvencia znakov, sekvencia začína znakom & a končí bodkočiarkou. Medzi nimi je krátky alfanumerický zápis daného znaku. Asi najdôležitejšou entitou je pevná medzera ** **. Podrobnejší zoznam číselných entít nájdeme na stránke: http://www.jakpsatweb.cz/html/entity-ciselne-vypis.html.

13.5 Formátovanie textu

Formátovanie textu (*zmena jeho štýlu*):

Tučné písmo a kurzíva

```
<b>Tučné </b> Tučné
<i>Sklonené </i> Sklonené
```

Podčiarknutie a preškrtnutie

```
<u>Podčiarknuté</u>  Podčiarknuté
<s>Preškrtnuté</s>  Preškrtnuté
<strike>Taktiež preškrtnuté</strike>  Taktiež preškrtnuté (HTML5 už túto značku nepodporuje)
```

Veľký a malý text

Najjednoduchšia možnosť, ako zväčšiť písmo, je použiť značku **<big>**, na zmenšenie zas **<small>**.

```
<big>Veľké</big>  Veľké
<small>Malé</small>  Malé
```

Znak	Entita
pevná medzera	
<	<
>	>
&	&
"	"
±	±
×	×
®	®
©	©
€	€
»	»
«	«

13.6 Obrázky a pozadie

K vloženiu obrázku na stránku slúži značka **<img>**. Značka môže obsahovať nasledovné vlastnosti:
src - udáva cestu k obrázku, napr. zápis **src="images/pes.gif"** určuje, že sa má zobraziť obrázok s názvom pes.gif uložený v zložke images (*Pri obrázkoch a odkazoch sa možeme stretnúť tiež s potrebou prejsť o zložku vyššie, vtedy použijeme* **../**)
align - udáva umiestnenie obrázku voči okolitému textu
alt - značka alt udáva text, ktorý sa zobrazuje na mieste obrázka, keď (*kým*) sa obrázok nenačíta, zapisuje sa - **alt="nejaký text"** (*Vyhľadávače kladú na túto značku dôraz*)
border – udáva hrúbku rámika obrázku **border="2"**
width - určuje šírku obrázku, najčastejšie sa zapisuje (*a mala by sa zapisovať*) skutočná šírka obrázku, zápis v tvare **width="150"** pre obrázok široký 150 pixelov
height - určuje výšku obrázku, zápis v tvare **height="100"** pre obrázok vysoký 100px. Kvôli rychlejšiemu a lepšiemu načítaniu stránky sa doporučuje tieto rozmery zadávať.
Obrázok zapíšeme nasledovne:

```
<img src="nazov-obrazku.jpg" alt="Popisný text obrázku" align="right" border="0" height="150" width="200"  />
```

Tento zápis zobrazí obrázok nazov-obrazku.jpg s rozmermi 200x150px, bez rámčeku, zarovnaný vpravo s popisným textom "Popisný text obrázku".

Použitie obrázkov

Na webovej stránke môžeme obrázok použiť hocikde a hocikoľko krát. Treba si uvedomiť, že priveľká veľkosť obrázku môže spôsobiť dlhšie zobrazovanie stránky v prehliadači. (*čo môže u ľudí s pomalším pripojením vyvolať nevôľu*).

Pozadie

Na pozadí webovej stránky sa môže rozprestierať jednofarebné pozadie alebo opakujúci sa obrázok (*treba dať pozor, aby nebol príliš veľký - dlho by sa načítal*). Pri zadávaní farieb je potrebné zadávať farbu pozadia a aj farbu písma, lebo inak riskujeme, že text nebude čitateľný. Jednofarebné pozadie určíme atribútom: **bgcolor** v značke body. Ako hodnotu použijeme červenú farbu.

```
<body bgcolor="#FF0000">
```

Pozadie tvorené obrázkom je podobné, len atribút sa nazýva **background**, jeho hodnotou je adresa obrázku.

```
<body background="logo.gif">
```

13.7 Odkazy



Odkaz definuje oblasť na stránke, kam môže používateľ kliknúť myšou a presunúť sa tak na inú webovú stránku alebo na iné miesto na tej istej stránke. Odkazy sú neoddeliteľnou súčasťou každej webovej stránky a zrejme neexistuje stránka, ktorá by ich neobsahovala. Odkazy sú text, ktorý je zvyčajne podčiarknutý a farebne odlíšený od ostatného textu. Odkazy sa tvoria pomocou značky **<a>** a **** (*z angl. anchor - kotva, odkaz*). Značka má nasledujúce vlastnosti:
href - obsahuje informáciu o odkazovanej adrese, čiže o adrese, na ktorú odkaz vedie
name - definujeme návestie, záložka (*navestie*) je dobrá k tomu, že definuje miesto v dokumente, na ktoré môže ukazovať odkaz, po kliknutí na takýto odkaz sa stránka odroluje na pozíciu záložky. (*HTML5 ho už nepodporuje, namiesto neho treba používať id*)
target - udáva, v akom okne sa odkaz otvorí, hodnoty sú: **_blank** (*otvorenie odkazu v novom okne prehliadača*), **_self** (*v tom istom okne*),**_parent** (*nadradené okno*), **_top** (*okno s najvyššou prioritou*), **meno** (*meno konkrétneho okna*).

title - informuje o obsahu odkazovanej stránky, čiže čo obsahuje stránka, na ktorú odkaz odkazuje.
Nevhodné sú absolútne adresy k lokálnemu súboru na disku pri vytváraní stránky, lebo sa nebude dať preniesť stránka inam. (*túto chybu robia často aj HTML editory*)

Zápis značky A
Medzi úvodzovky atribútu **href** napíšeme adresu, na ktorú sa chceme po kliknutí na odkaz dostať. Adresa môže byť internetová stránka alebo súbor (*napr. obrázok*). Mali by sme ju písať bez diakritiky a medzier. Odkaz do stránky vložíte takto:

```
<a href="http://www.google.sk" target="_blank" title="Google.sk - slovenská verzia vyhľadávača Google">Slovenský Google
</a>
```

Tento zápis značky a zobrazí: Slovenský Google. Po kliknutí na odkaz sa v novom okne prehliadača zobrazí úvodná stránka webu Google.sk. Odkaz môže viesť na 2 druhy internetových adries - **relatívnu adresu** alebo **absolútnu adresu**. V predchádzajúcom príklade sme odkazovali na absolútnu adresu, odkaz s relatívnou adresou má nasledovný tvar:

```
<a href="index.html" target="_blank" title="Články o  HTML"> HTML</a>
```

Ide o to, že hodnotou href nie je adresa v sieti internet (*s http:// na začiatku*), ale relatívna adresa v rámci adresárového systému toho istého webu.

13.8 Zoznamy

Vytváranie zoznamov obsahuje skupinu značiek, ktoré umožňujú vytváranie hierarchických zoznamov. Existujú dva typy zoznamov: **nečíslované** a **číslované**. Položky nečíslovaného zoznamu sú typicky odsadzované odrážkami, položky číslovaného zoznamu typickými číslami. Oba typy je možné kombinovať a vytvárať tak zložitejšie zoznamy.

Nečíslované zoznamy
Ukážka dvoch nečíslovaných zoznamov.

```
<!DOCTYPE html>
<html><head>
        <title>Nečíslovaný Zoznam</title></head><body>
       <ul>Toto je na prvej úrovni.
        <ul>Toto je na druhej úrovni</ul>
        <ul>Toto je na druhej úrovni.
        <ul>Toto je na tretej úrovni.</ul>
        </ul>
        <ul>Toto je na druhej úrovni</ul>
        </ul>  <hr />
        <ul>Toto je hlavička prvej úrovne.
        <li>Toto je prvá položka na prvej úrovni zoznamu. </li>
        <li>Toto je druha položka na prvej úrovni zoznamu. </li>
        <ul>
        <li>Toto je prvá položka na druhej úrovni zoznamu. </li>
        <li>Toto je druhá položka na druhej úrovni zoznamu. </li>
        </ul>
        <li>Toto je tretia položka na prvej úrovni zoznamu. </li></ul>
        </body></html>
```

Číslované zoznamy
Položky číslovaného zoznamu sa zobrazujú s odrážkami, ale i s číslami. Číslovaný zoznam vytvárame značkami ** **.

```
 <ol>
<li>Toto je prvá položka na prvej úrovni zoznamu. </li>
<li>Toto je druha položka na prvej úrovni zoznamu. </li>
<ol>
<li>Toto je prvá položka na druhej úrovni zoznamu. </li>
<li>a toto je druhá položka na druhej úrovni zoznamu. </li>
</ol>
<li>Toto je tretia položka na prvej úrovni zoznamu. </li>
</ol>
```

13.9 Tabuľky

Tabuľky predstavujú jeden z najmocnejších nástrojov pre ohraničenie textu na stránke. Značy používané k vytváraniu tabuľky poskytujú celý rad atribútov, ktoré podporujú čokoľvek od farebných rámčekov a pozadí až po rôzne zarovnávania obsahu buniek vo zvislom i vodorovnom smere.

Jednoduchá tabuľka

Tabuľka je vymedzená značkami **<table>** a **</table>**. V značke <table> sa nachádzajú ďalšie značky, a sú to **<tr>** a **<td>**. Značka <tr> označuje jeden riadok tabuľky a značka <td> označuje jednu bunku v riadku tabuľky. Z toho vyplýva, že sa bude značka <td> zapisovať do značky <tr>. Atribút **border** s hodnotou 1 vytvárajú jeden pixel široký rámček medzi poliami tabuľky i okolo celej tabuľky. Pri zadaní nulovej hodnoty sa orámovanie nevykreslí. Pretože jazyk HTML ignoruje nadbytočné medzery, môžeme pomocou tabuliek s hodnotou border 0 poziciovať rozmiestnenie prvkov na našej web stránke.

```
<table border="1">
<tr><td>R1, D1</td>
<td>Riadok 1, Dáta 2</td>
</tr><tr>
<td>Riadok 2, Dáta 1/td>
<td>Toto je riadok 2, dátový stĺpec 2</td>
</tr> </table>
```

R1, D1	Riadok 1, Dáta 2
Riadok 2, Dáta 1	Toto je riadok 2, dátový stĺpec 2

Jednotlivé atribúty pre **TABLE** sú:
BORDER - šírka čiary v tabuľke v pixeloch
WIDTH - šírka tabuľky (v pixeloch alebo v % stránky)
ALIGN - zarovnanie tabuľky na stránke (LEFT, CENTER, RIGHT)
CELLSPACING - vzdialenosť medzi dvoma bunkami
CELLPADDING - vzdialenosť medzi okrajom bunky a jej obsahom
BGCOLOR - farba pozadia tabuľky
Jednotlivé atribúty pre **TR** sú:
ALIGN - zarovnanie v horizontálnom smere (LEFT, RIGHT, CENTER, JUSTIFY)
VALIGN - zarovnanie vo vertikálnom smere (TOP, MIDDLE, BASELINE, BOTTOM)
BGCOLOR - farba pozadia bunky
Jednotlivé parametre pre **TD** sú:
ROWSPAN - umožňuje spojiť viac buniek ležiacich pod sebou v jednom stĺpci do jednej bunky
COLSPAN - umožňuje spojiť viac buniek ležiacich vedľa seba v jednom riadku do jednej bunky
WIDTH - určuje šírku danej bunky v pixeloch alebo v % (z celej tabuľky)
HEIGHT - určuje výšku danej bunky v pixeloch alebo v % (z celej tabuľky)
ALIGN - zarovnanie v horizontálnom smere (LEFT, RIGHT, CENTER, JUSTIFY)
VALIGN - zarovnanie vo vertikálnom smere (TOP, MIDDLE, BASELINE, BOTTOM)
BGCOLOR - farba pozadia bunky

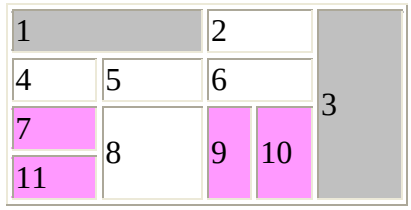
Značka **th** ohraničuje nadpis (*je to v podstate náhrada za TD*)

```
<table border="0"><tr><th>Prvá bunka záhlavia tabuľky</th>
 <th>Druhá bunka záhlavia tabuľky</th>
 </tr><tr><td>Prvá bunka tabuľky</td>
 <td>Druhá bunka tabuľky</td></tr>
</table>
```

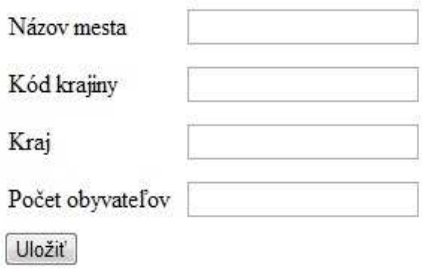
Spájanie buniek

```
<table border="1" width="200"><tr>
  <td width="50%" colspan="2" bgcolor="#c0c0c0">1</td>
  <td width="25%" colspan="2">2</td>
  <td width="25%" rowspan="4" bgcolor="#c0c0c0">3</td>
```

```
</tr> <tr>
  <td width="25%">4</td>
  <td width="25%">5</td>
  <td width="25%" colspan="2">6</td>
</tr> <tr>
  <td width="25%" bgcolor="#ff99ff">7</td>
  <td width="25%" rowspan="2">8</td>
  <td width="13%" rowspan="2" bgcolor="#ff99ff">9</td>
  <td width="12%" rowspan="2" bgcolor="#ff99ff">10</td>
</tr><tr>
  <td width="25%" bgcolor="#ff99ff">11</td>
</tr></table>
```



13.11 Formuláre



Formuláre umožňujú interakciu medzi používateľom a serverom. Základný princíp spočíva v tom, že prehliadač vo webovej stránke zobrazí formulár, v ktorom požaduje zadanie rôznych informácií - napríklad mena, adresy, telefónneho čísla, veku a podobne. Používateľ formulár vyplní a odošle späť na server. Formulár môže obsahovať rad rôznych položiek od klasických textových polí, cez zaškrtávacie voľby až po rôzne zoznamy. Základom je párová značka **form** s povinným atribútom **action**. Ten určuje súbor na serveri, ktorý bude formulárové údaje spracúvať. Používaným atribútom je tiež **method** (*metóda GET zobrazuje formulárové dáta v URL adrese, POST ich prenáša ako samostatnú súčasť HTTP žiadosti*).

<**form**>
action=adresa
method=get alebo post
target=okno
enctype="multipart/form-data"
</ **form**>
<**input**
align=umiestnenie
checked= on alebo off
maxlength= určuje maximálny počet znakov ktoré môže prvok obsahovať
name= je povinné, podľa mena bude tento parameter identifikovaný pri spracovávaní
size= špecifikuje veľkosť (napr. textového poľa), zdáva sa "širka, výška"
src=adresa
type=typ prvku
value=hodnota, zapisuje hodnotu prvku
/>

Typ prvku **TYPE** určuje typ prvku a môže nadobúdať tieto hodnoty:
CHECKBOX - jeho hodnota je typu Boolean (*jedna alebo nula*) teda môže nadobudnúť hodnotu "on" (*zapnutý*) alebo "off" (*vypnutý*). Jednotlivé boxy sa sú na sebe nezávislé.
HIDDEN - tieto informácie nie sú zobrazované, ale slúžia iba ako pre informácie pre spracujúci script. Používateľ nemá možnosť zmeniť ich.
IMAGE - tento typ pracuje ako klikajúca mapa a po spracovaní je odoslaná x-ová a y-ová súradnica závislá na vzdialenosti od počiatku (ľavý horný roh obrázku). Pri použití obrázku sa už nepoužije tlačidlo "submit" (pošli), ale po kliknutí na obrázok bude formulár odoslaný.
PASSWORD - pracuje rovnako ako aj TEXT ibaže vypĺňané hodnoty sú nečitateľné, prepisujú sa, najčastejšie znakom '*'. !!!POZOR hodnota je však posielaná na internet bez akéhokoľvek šifrovania, nie je to bezpečný spôsob prenosu.

RADIO - radio je podobne ako aj CHECKBOX typu boolean. Slúži ako výber jednej z viacerých možností. Vyberá jednu z viac možností. Ak sa použije ako prepínač, jednotlivé prvky musia mať rovnaké meno.
RESET - celý formulár nastaví znova do pôvodného stavu. To znamená, že všetko sa nastaví na hodnotu atribútu value, ktorá je zadaná v značke input, alebo pokiaľ nie je, nastaví sa na nulu.
SUBMIT- odošle všetky hodnoty z dokumentu. Pokiaľ je prvok formulára typu boolean, odošle sa "on" alebo "off", pokiaľ je iného typu, odošle vyplnenú hodnotu.
TEXT - použitie klasického textového poľa. Jeho vlastnosti sa nastavujú pomocou SIZE a MAXLENGTH.
BUTTON - bežne tlačidlo.
IMAGE - obrázkové tlačidlo.
PASSWORD - heslo.
FILE - upload súborov.

```
<!DOCTYPE html>
<html><head>
<title>Formular</title>
</head><body>
<table cellspacing="20"><tr><td>
<form method=post action="mailto:test_form@gmail.com">
<input type="hidden" value="Poslane z ITE" />
Môj ročník:<br /><input type="radio" name="1m" value="I" /> I<br />
<input type="radio" name="1m" value="II" /> II<br />
<input type="radio" checked  name="1m" value="III" /> III <br />
<input type="radio" name="1m" value="IV" /> IV <br />
</td><td>Zameranie v oblasti výpočtovej techniky:<br />
<input type="checkbox" name="SW" /> uživateľ programov, programátor<br />
<input type="checkbox" name="HW" /> technik, údržba PC<br />
<input type="checkbox" name="hry" />bavia ma hry <br />
</td></tr>
<tr><td>adresa na vašu stránku:<br />
vaša e-mail adresa:<br /></td><td>
<input type="text" size="40" name="url" value="http://" /><br />
<input type="text" size="40" name="email" value="@ziak.sk" /><br />
</td></tr>
<tr><td>Na internete mi chýba...
<select NAME="chcem" size="3" multiple>
<option value="Vedecké informácie">vedecké informácie</option>
<option value="história, kultúra">história, kultúra</option>
<option value="šport">šport</option>
<option value="technika">technika</option>
<option value="knihy, literatúra">knihy, literatúra</option>
<option value="...ale veď tam nic nie je">...ale veď tam nič nie je</option>
</select></td>
<td align="right">
Používam<br /><select size="1" name="browser">
<option value="Internet Explorer"> Internet Explorer</option>
<option value="Mozilla Firefox ">Mozilla Firefox </option>
<option value="iné">iné</option>
</select></td></tr><tr><th colspan="2">
Nasledujúce textové pole môžete použiť na vyjadnerie svojich názorov a pripomienok<br />
<textarea cols="70" rows="6" name="pole"></textarea>
</th></tr>
<tr><td colspan="2">
Ak ste ste odpovedali na otázky, pošlite ich stlačením na POŠLI. V prípade nesprávneho
vyplnenia<br />formulár zmažete stlačením na RESET.<p />
<input type="reset" value="RESET" />
<input type="submit" value="POŠLI" />
</td></tr></form>
</table>
</body>
</html>
```

Môj ročník:
I
II
III
IV

Zameranie v oblasti výpočtovej techniky:
užívateľ programov, programátor
technik, údržba PC
bavia ma hry

adresa na vašu stránku: http://
vaša e-mail adresa: @ziak.sk

Na internete mi chýba... vedecké informácie / história, kultúra / šport

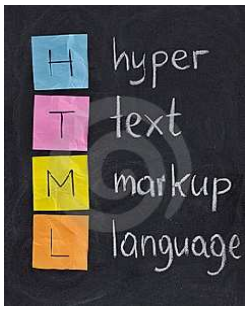
Používam Internet Explorer

Nasledujúce textové pole môžete použiť na vyjadrenie svojich názorov a pripomienok

Ak ste ste odpovedali na otázky, pošlite ich stlačením na POŠLI. V prípade nesprávneho vyplnenia formulár zmažete stlačením na RESET.

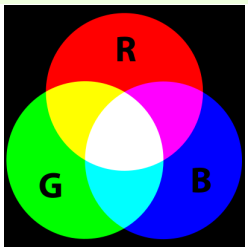
RESET POŠLI

13.12 Farby



Pri zápise farebného kódu v HTML stránkach sa pred príslušný číselný kód farby píše znak #. Pri zápise farieb pomocou slovných kódov (*Nedoporučuje sa z dôvodu, že jednotlivé internetové prehliadače môžu jednotlivé farby implementovať rozdielne.*) ako "yellow", "red" a podobne, sa znak # nepíše. Príklad zápisu pre červený text:

```
<font color="#FF0000">Tento text je červený.</font>
```



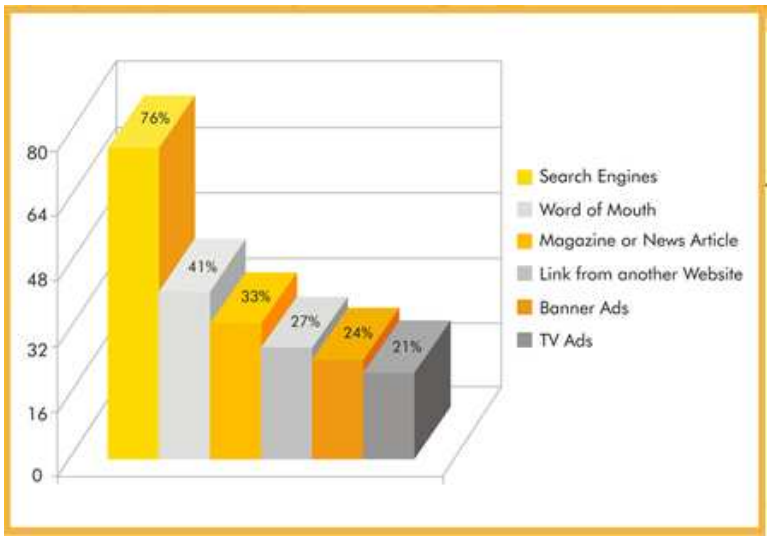
Farby sú zobrazené kombináciou červenej, zelenej a modrej zložky svetla. Hodnoty farieb HTML farby môžu byť definované hexadecimálne, ako zložky RGB svetla (*červené, zelené a modré*). RGB je režim založený na miešaní farebného svetla. Konkrétne miešanie **červeného** (*Red*), **zeleného** (*Green*) a **modrého** (*Blue*) svetla. Ak svietia všetky tri zložky úplnou intenzitou, vzniká biele svetlo. Ak svieti napríklad iba červená a zelená zložka, vzniká žlté svetlo. Tento model umožňuje vytvoriť obrovské spektrum farieb s množstvom odtieňov. Pri zápise farieb môžeme použiť percentuálny zápis, desiatkový zápis alebo šestnástkový zápis. Nasledujúce zápisy červenej farby sú ekvivalentné: red, rgb(100%, 0%, 0%), rgb(255, 0, 0), #FF0000. Jednotlivé kódy farieb nájdeme na stránkach:
http://www.jakpsatweb.cz/archiv/barvy-bezpecne.html, http://media.jex.cz/thema/html-farby, http://sites.google.com/site/jednoduchohtml/html-colors

14. Webové štandardy

Webové štandardy sú odporúčania a pravidlá určené pre tvorbu web stránok navrhované konzorciom W3C (*World Wide Web konzorcium*). Hlavnou úlohou konzorcia W3C je zjednocovať a modernizovať technológie používané na internete. Ak by výrobcovia internetového softvéru a tvorcovia web stránok dodržovali pravidlá organizácie W3C, zobrazovanie web stránok by bolo úplne bezproblémové. Žiaľ, zatiaľ to tak nie je, ale situácia sa neustále zlepšuje. Webové štandardy je potrebné

dodržiavať kvôli rýchlejšiemu zobrazovaniu v internetovom prehliadači, kompatibilite www stránky medzi internetovými prehliadačmi, vyššej pozícii www stránky vo vyhľadávačoch (napr. google, yahoo, MSN atď.). Kontrolu korektnosti (*validitu*) www stránky, teda či je vytvorená podľa štandardov, môžeme vykonať pomocou nástrojov na URL http://validator.w3.org, http://jigsaw.w3.org/css-validator Je možné použiť aj iný štandard ako W3C, ak sa jeho DTD zapíše do DOCTYPE.

15. SEO - optimalizácia pre vyhľadávače



Internetové vyhľadávače sa v poslednej dobe stávajú stále častejším spôsobom hľadania informácií. Takmer každý, kto pracuje s internetom, nejaký vyhľadávač použil. Tým, ako zviditeľniť web stránku vo vyhľadávačoch sa zaoberá internetový marketing, ktorý sa označuje anglickou skratkou SEM (*Search Engine Marketing*). Internetový marketing pokrýva široké spektrum techník zviditeľňovania stránok na internete. Zahŕňa služby ako optimalizácia pre vyhľadávače, sponzorované odkazy *(platená internetová reklamu*), link building (*budovanie spätných odkazov*) a link baiting (*získavanie spätných odkazov netypickými spôsobmi - napr. guerilla alebo viral marketing).* Internet marketing je popri kvalitnom obsahu a dizajne web stránok kľúčovou súčasťou internetovej stratégie. Reklama na internete (*internet marketing*) má na rozdiel od ostatných foriem reklamy tri veľké prednosti: cielenosť, rýchlosť a presnú merateľnosť. Vďaka tomu môže byť internet marketing veľmi efektívny.

Súčasťou internetového marketingu je aj optimalizácia pre vyhľadávače (*SEO*). SEO je skratka anglického *Search Engine Optimization*. Optimalizáciou pre vyhľadávače sa snažíme vytvoriť, upraviť, doplniť a následne propagovať web stránky k lepším umiestneniam v neplatených výsledkoch hľadania vyhľadávačov (*vo fulltextovom vyhľadávaní*) na kľúčové slová, ktoré sú relevantné pre danú stránku. Tým sa zvyšuje návštevnosť stránok a zároveň konverzný pomer, tj. percento návštevníkov, ktorí realizujú na stránke to, čo je jej cieľom (*objednajú si produkt, vyplnia dotazník, kliknú na reklamu, a pod.*). Potreba optimalizácie web stránok pre vyhľadávače (*SEO*) vznikla enormným nárastom webových stránok na internete. V najpoužívanejšom vyhľadávači na svete (*Google*) sa nachádza viac ako 25 miliárd zindexovaných webových stránok. Uspieť a dostať sa na popredné pozície v SERP (*výsledkoch vyhľadávania*) nie je jednoduché. Predovšetkým sa jedná o úpravu textov (*copywriting*), vytváranie spätných odkazov (*linkbuilding*), čisté kódovanie webových stránok, a dodržovanie štandardov W3C. Jedná sa o dlhodobý proces, nakoľko je potrebná trvalá úprava a nepretržité sledovanie priebehu optimalizácie. Z dlhodobého hľadiska je SEO určite menej nákladná a mnohonásobne trvácnejšia ako platená reklama. Skutočným poslaním optimalizácie webových stránok pre vyhľadávače je priniesť vždy najnovšie informácie. Informácie hľadajú návštevníci, vyhľadávače sú len nástrojom k vyhľadaniu aktuálnych informácií z čo najbezpečnejších zdrojov. Práve preto je potrebné vyhľadávače presvedčiť o kvalitách a možnostiach, ktoré prináša webová stránka. Cesta ako uspieť vedie cez kvalitnú optimalizáciu webových stránok (*SEO*) alebo cez platenú reklamu PPC.

Ak si na webe čitatelia majú čo prečítať, je to web, ktorý obsahuje informácie, aké môžu vyhľadávače analyzovať, triediť a následne poskytnúť ako výsledky hľadania. No ak je web postavený na dizajne, na "pekných obrázkoch", či len na technológii Flash, je síce pre

návštevníka efektný, ale vyhľadávač ho nemôže "pochopiť", nemá čo načítať, nemá čo triediť. Nemá z takého webu žiadne dáta, ktoré by mohol predložiť ako výsledky vyhľadávania. Preto pri návrhu webu treba myslieť najskôr na SEO, až potom na webdesign.

15.1 Predpoklady pre správnu optimalizáciu

Web stránky musia spĺňať predpoklady, čiže technicky správne riešenie pre SEO optimalizáciu. Existuje veľa firiem, ktoré ponúkajú technickú analýzu web stránok, ktorá ozrejmí, či vyhľadávač dokáže prečítať vaše web stránky, zistia validnosť HTML kódu, veľkosť stránky, duplicitné URL adresy, neexistujúci obsah atď. Vyhľadávače vidia stránky približne ako http://www.seo-browser.com. Preto môžu JavaScript, Flash a rámce spôsobovať problémy.

Legálne techniky:

white hat – technika, ktorá sa sústredí na správne kódovanie stránok, vyhľadávače preferujú takto vytvorené stránky.

– štruktúra obsahu: title, nadpisy, zvýraznenie textov (pomocou tagu)
– vlastný obsah
– meta tagy: keywords, description
– prístupnosť stránky
– spätné odkazy
– registrácia stánok do vyhľadávačov
– robots.txt

Nelegálne techniky:

black hat – technika, ktorá využíva aj nekorektné alebo neprípustné spôsoby, ako zvýšiť pozíciu danej stránky vo vyhľadávači, ide často o nelegálne alebo neetické spôsoby

– skryté texty a odkazy, nerelevantné texty
– cloaking (*iný obsah pre ľudí a pre vyhľadávače*)
– door-way stránky (*mnoho kľúčových slov na vstupnej stránke*)
– link-farms
– tajné presmerovanie (*prehliadače JavaScript nevidia*)
– automatické programy

15.2 Vyhľadávače



Vyhľadávače sú aktívne, hľadajú web stránky na internete sami pomocou skriptov, tzv. automatických botov ako je napr. Googlebot, Alexa spider, rokudan.jyxo.com a pod. Takýto skript navštívi webstránku, prehľadá ju, zistí, aký má obsah, a indexuje ju do svojej databázy. Podľa toho potom poskytuje výsledky vyhľadávania. Ďalšie web stránky nájde vyhľadávač, resp. jeho spider tak, že jednoducho nasleduje všetky externé odkazy na prehľadávanej web stránke. Tak sa vyhľadávanie neustále rozvetvuje po rozsiahlej sieti internetu. Na zoradenie výsledkov vyhľadávania používajú tieto systémy vždy nejaké pravidlá, ktoré si zväčša dobre strážia. Všeobecne známym hodnotiacim algoritmom je napríklad Google PageRank. Je to prepracovaná technológia na zoraďovanie relevantných výsledkov vyhľadávania v najväčšom vyhľadávači Google. Existujú niektoré zásady, ktoré je nutné dodržať pre úspešné umiestnenie odkazu vo vyhľadávačoch, ale presné kritériá, podľa ktorých vyhľadávače webstránky hodnotia, sú utajované. Známe slovenské vyhľadávače sú napríklad Morpheo (*morfeo.sk*), alebo Zoohoo (*www.zoohoo.sk*), ktorý používa technológiu českého Jyxa. Špecifickým prípadom vyhľadávačov sú tzv. meta-vyhľadávače, zameriavajú sa na vyhľadávanie iba v meta tagoch web stránok. Táto technológia indexovania obsahu je dnes však už na ústupe, pretože meta tagy chýbajú mnohým web stránkam.

15. 3 Katalógy

Katalóg je pasívny - na rozdiel od vyhľadávačov si nevytvára svoj obsah sám, ale závisí na svojich používateľoch, ktorí doň pridávajú záznamy ručne. Keď ho navštívi niekto, kto má svoju vlastnú webstránku (*alebo hocikto iný*), môže pridať do databázy katalógu odkaz na ňu tak, že jednoducho vyplní registračný formulár. Odkaz je do zoznamu pridaný zväčša až po skontrolovaní administrátorom, ale existujú niektoré katalógy, ktoré zaraďujú odkazy okamžite. Vyhľadávanie v katalógu prebieha nalistovaním konkrétnej kategórie vzťahujúcej sa k téme, ktorá nás zaujíma, alebo vyhľadávaním v databáze podľa kľúčového slova. Príkladom kvalitného medzinárodného katalógu je DMOZ (*dmoz.org*). Zoznam katalógov http://www.vyhledavace.net, kde si môžeme zaregistrovať naše www stránky.

15.4 Optimalizácia webových stránok



Realizácia optimalizácie webových stránok:

1. Určenie a vytriedenie kľúčových slov, na ktoré sa bude v ďalšom postupe stránka optimalizovať.
2. Dokonalé sprístupnenie vašej stránky pre všetky vyhľadávače.
3. Vytvorenie textov, ktoré budú tvoriť plnohodnotný obsah ako pre vyhľadávače, tak aj pre návštevníkov webu.
4. Vytvorenie dokonalej a prehľadnej navigácie na stránkach a v odkazoch.
5. Vybudovanie spätných odkazov a registrácia do katalógov.
6. Analýza optimalizovaného webu a sledovanie konkurencie.

Najzávažnejšie nedostatky web stránok:

1. Webová stránka bez titulku.
2. Málo textu na webovej stránke.
3. Text, ktorý vôbec nesúvisí s obsahom alebo bol kopírovaný z inej už existujúcej stránky
4. Žiadne odkazujúce linky.
5. Príliš veľa zvýrazneného textu.
6. Veľa nadpisov (*H1, H2 ...).*

Až 39 % web stránok nemá vyplnený ani titulok, je to ako bezmenná kniha v knižnici. Kto ju nájde?

Ďalšie faktory, ktoré môžu stránke uškodiť:

1. Web stránky neprístupné pre návštevníkov.
2. Zapojenie vašej stránky do rôznych linkových fariem.
3. Web stránky, ktoré sa veľmi dlho načítavajú.

Umiestenie stránok vo výsledkoch vyhľadávania ovplyvňujú:

1. Vzájomná poloha umiestených slov (*u hľadaní viacerých fráz*).
2. Umiestenie a početnosť nájdených slov (*podľa polohy v dokumentoch*).
3. Umiestenie hľadaného slova, alebo slov v titulku stránky, metaznačke Description a v nadpisoch.
4. Podľa váhy stránky v očiach vyhľadávača (*rank*).
5. Na základe počtu odkazov na stránku a kvalita týchto odkazov.

15.5 On page faktory

On page faktormi sa označujú faktory priamo súvisiace s obsahom a kódom stránky, jej umiestnením v rámci celého webu. Starostlivé vyladenie On page faktora môže veľmi priaznivo ovplyvniť hodnotenie stránky vo vyhľadávači a tým aj jej umiestnenie vo výsledkoch vyhľadávania.

Existujú 3 základné hľadiská:
Technické faktory - Technické otázky riešia, ako vytvoriť web stránky tak, aby vyhľadávače mohli indexovať ich obsah a aby ho čo najlepšie pochopili (*sémantika*).
Obsahové faktory - V nich ide najmä o tvorbu kvalitného obsahu a rozmiestnenie zvolených kľúčových slov do prioritných miest na stránke.
Informačná architektúra - Informačná architektúra v SEO je tá časť celkovej optimalizácie, ktorá rieši, ako zahrnúť jednotlivé témy z rozboru kľúčových slov do webového projektu, čiže štruktúru a navigáciu.

Kľúčové slová (*keywords*)
37 expertov na optimalizáciu pre vyhľadávače sa zúčastnilo na prieskume SEOmoz o významnosti vybraných faktorov v algoritme najväčšieho svetového vyhľadávača Google. Spoločne hlasovali o tom, ktoré faktory najväčšou mierou prispievajú k popredným umiestneniam vo výsledkoch vyhľadávania, získané výsledky vo veľkej miery naznačili, ako Google algoritmus funguje. Jednotlivé faktory sú zoradené v poradí podľa ich dôležitosti tak, ako o nich hlasovali SEO experti:
Kľúčové slová v Title Tagu
Priemer: 4,9 bodov z 5, výnimočný význam
Kľúčové slová v texte tela (*body*) stránky
Priemer: 3,7 bodov z 5, veľký význam
Vzťah medzi textom v tele (body) stránky a kľúčovými slovami
Priemer: 3,4 bodov z 5, veľký význam
Kľúčové slová v H1 tagu
Priemer: 3,3 bodov z 5, veľký význam
Kľúčové slová v názve domény
Priemer: 3,0 bodov z 5, stredný význam
Kľúčové slová v URL
Priemer: 2,8 bodov z 5, stredný význam
Kľúčové slová v H2-H6
2,8 bodov z 5, stredný význam
Kľúčové slová v Alt a Title u obrázkov
Priemer: 2,6 bodov z 5, stredný význam
Kľúčové slová v Bold/Strong tagoch
Priemer: 2,3 bodov z 5, stredný význam
Kľúčové slová v meta Description
Priemer: 2,0 bodov z 5, stredný význam
Kľúčové slová v meta Keywords
Priemer: 1,2 bodov z 5, malý význam

Škodlivé slová (*Poison words*)
Ide o slová, ktoré znižujú pozíciu, teda majú negatívny vplyv na hodnotenie stránky vyhľadávačom. Pre používateľa, ktorý nechce obchod, ale stránku s recenziami produktu, je poškodzujúcim slovom napríklad kúpiť, cena, predaj ... (*v angličtine buy, sell, price...*).

15.6 Off page faktory

SEO off-page faktory sú predovšetkým o získavaní spätných odkazov. Vyhľadávač nevie objektívne zhodnotiť kvalitu stránok, túto prácu mu ale pomôžu urobiť sami užívatelia internetu. Hlavnými Off page faktormi sú predovšetkým kvalitné spätné odkazy. Dôležitá je nielen kvantita (*celkový počet*), ale aj ich kvalita (*z akých dôležitých stránok prichádzajú*). Značný význam má aj text odkazov a jeho najbližšie okolie. Medzi Off page faktory patrí aj relevancia odkazujúcej stránky, tematická príbuznosť odkazujúceho webu, *dôležitý* je aj vek odkazov a jeho vznik v reakcii na zmenu odkazovaného dokumentu. Váhu odkazov ovplyvňuje aj jeho umiestnenie na stránke, umiestnenie odkazujúcej stránky v štruktúre webu a veľa ďalších faktorov.

Získavanie spätných odkazov

Kvalitný obsah je najlepším spôsobom, ako získať spätné odkazy. Kvalitný obsah ani nemusíme propagovať, šíri sa sám vysokou rýchlosťou. Ďalším spôsobom, ako získavať odkazy, je registrácia do internetových katalógov (*registrácia do nich je väčšinou zdarma).* Komentáre sú ďalším spôsobom, ako získať spätné odkazy na stránku. Takmer všetky blogy ponúkajú možnosť uviesť v komentároch odkaz na stránku. Niektoré z nich však do odkazov automaticky vkladajú atribút rel="nofollow" (*nofollow vkladá aj Wikipédia*), čo znamená, že vyhľadávač odkaz nezaregistruje. Ďalšou z možností, ako získavať spätné odkazy, je výmena odkazov. Ide o dohodu medzi dvomi stránkami, že umiestnia odkazy na svoje stránky smerujúce na stránky toho druhého. Pri vymieňaní odkazov s inými stránkami by sme však mali myslieť nato, že nám pomôžu len stránky s podobnou tematikou. Vymieňať odkazy s nerelevantnými stránkami je zbytočné. Napríklad ak prevádzkujeme stránku školy, nemá veľký význam vymieňať spätné odkazy so stránkou zaoberajúcou sa SEO.

PageRank, S-Rank, JyxoRank

Každý väčší vyhľadávač má aj svoje vlastné hodnotenie stránok. Google má PageRank, u zoznamu je to S-rank a Jyxo má svoj JyxoRank. Tieto ranky sa dajú zistiť napríklad pomocou rôznych skriptov, resp. ak si nainštalujeme toolbar daného vyhľadávača do svojho prehliadača, ktorý rank zobrazuje.

PageRank

PageRank interpretuje atraktívnosť web stránky číselnou hodnotou od 0 do 10, kde PageRank 0 je najnižšia hodnota, ktorú mávajú nové web stránky, a PageRank 10 je najvyššia hodnota, ktorú dosahuje len málo web stránok na svete.

PageRank je algoritmus pre ohodnotenie relatívnej dôležitosti webových stránok, ktorý tvorí základ vyhľadávača Google. Vytvorili ho zakladatelia Googlu Larry Page a Sergey Brin. V podstate celý systém PageRank sa opiera o prichádzajúce a odchádzajúce odkazy miliárd webových stránok, ktoré tvoria internet. Napríklad ak web A odkazuje na web B, dáva mu vlastne hlas, ktorý zvyšuje jeho dôležitosť. Hodnotenie stránky sa nepočíta z jednoduchého počtu odkazov, ktoré na ňu vedú, ale berie sa do úvahy aj hodnotenie odkazujúcich stránok. Tiež hodnota prichádzajúcich odkazov je rôzna, a to podľa dôležitosti. Až kombinácie PageRanku, obsahu stránky a mnohých ďalších faktorov určujú pozíciu na vyhľadávané slovo. Dôležité je, že PageRank má každá jednotlivá stránka, a nie celý web. Tiež hodnota prichádzajúcich odkazov je rôzna, a to podľa dôležitosti stránky, z ktorej tento odkaz prichádza, a počtu odkazov na tejto stránke. Slovenský google www.google.sk má PR 7, www.azet.sk má PR 6, www.minedu.sk má PR 8 (*vďaka množstvu odkazujúcich školských stránok*), hodnoty namerané 28.5. 2011. Na svete existuje niekoľko stránok s PageRankom 10, nájdeme ich napríklad v zozname: http://www.seocompany.ca/pagerank/pr-10-pages.php.

15.7 SEO nástroje

NAJ.sk [http://www.naj.sk]
O tom, ako sa návštevníci správajú pri návšteve stránky sa môžeme presvedčiť ak ju zaregistrujeme v systéme NAJ.sk na meranie a analýzu návštevnosti stránok.

Google AdWords Keywords Suggestions [https://adwords.google.com/select/tools.html]
Vyhľadávanosť väčšinou rastie s kvalitou výsledkov vyhľadávania. Ak prirodzené vyhľadávanie dáva nerelevantné výsledky, ľudia na ne málo klikajú a nabudúce dané slovo už nehľadajú (*klesá vyhľadávanosť slova*). To isté platí pri platenej reklame: ak je kvalitná vzhľadom k dotazu, viac sa na ňu kliká.

eTarget [http://www.etarget.sk]
eTarget je slovenský reklamný systém, ktorý sa používa aj v Českej republike. eTarget dokáže navrhnúť k vybranému slovu ďalšie alternatívy a tiež zistiť ich dennú návštevnosť.

SEO-SERVICE.CZ Analýza zdrojového kódu stránky [http://seo-servis.cz/source-zdrojovy-kod/]
Prehľadné informácie o zdrojovom kóde www stránky. Analýza odhalí väčšinu závažných chýb, môže pomôcť výrazne zlepšiť kvalitu webu. Analýza preveruje: informácie v hlavičke, chyby v HTML kóde, sémantiku obsahu, prístupnosť.

Analýza kľúčových slov stránky [http://seo-servis.cz/keywords-test-klicovych-slov/]
Zistí informácie o obsahovej stránke webu, vyhodnotenie frekvencie slov a fráz, zistí, ktoré slová a kam by ich bolo lepšie pridať, vyhodnotí "optimalizovanosť" zadaného slova s návrhmi na vylepšenia, podporuje stop-slová, ako sú spojky, predložky atď. bez hlbšieho obsahového významu.

Pozícia vo vyhľadávačoch [http://seo-servis.cz/search-pozice-ve-vyhledavacich/]
Zistenie pozícií webu vo vyhľadávačoch na zadané slovo, hľadanie až do 100 (*Zoznam do 60*) pozície.
Google, Windows Live Search, Seznam.cz, Jyxo, Morfeo, zistí počet spätných odkazov, počet zindexovaných stránok, aktuálny Google Toolbar Pagerank webu.

Analýza sily webovej stránky [http://seo-servis.cz/power-sila-stranky/]
Komplexná analýza OnPage aj OffPage faktorov. Analýza dodá komplexný prehľad o stave, sile a potenciále webu v globálnom meradle. Analýza preveruje: komplexne HTML kód, spätné odkazy a ich potenciál, pozície vo vyhľadávačoch, návštevnosť a potenciál domény.

Linkbuilding - výmena odkazov [http://seo-servis.cz/linkbuilding-zpetne-odkazy/]
Seo servis ponúka možnosť získať nové spätné odkazy, kde si ktokoľvek môže nájsť partnera s tematicky rovnako ladeným webom pre vzájomnú výmenu odkazov. Pre výmenu odkazov môžeme využiť tiež stránku http://webtrh.cz/f102.
Hustota výskytu kľúčových slov
Pre počítanie hustoty slov na webe existuje množstvo nástrojov:
http://www.ranks.nl/tools/spider.html
http://www.webmaster-toolkit.com/keyword-analysis-tool.shtml

Užitočné zdroje a odkazy z oblasti SEO, PPC a internetového marketingu
http://michalkubicek.cz/odkazy, www.seonastroje.cz, www.webtrh.cz, www.ranky.cz, www.ranky.sk, http://websiteprice.net, www.ireklama.sk

16. CSS

```
body {
  background: #FFFFF
  margin: 0;
  font: 9pt verdana;
}
a, a:link, a:visited
  text-decoration: n
  color: #2986d0;
}
a:hover {
  color: #216296;
}
```

Kaskádové štýly CSS (*Cascade Style Sheet*) poskytujú spôsob, ako jednoduchým spôsobom zabezpečiť jednotný vzhľad webových stránok (*písma, pozadia, farieb, atď.*). Priraďujeme jednotlivým značkám jazyka HTML a XHTML ich vzhľadové atribúty, (*spôsob zobrazovania, farba, šírka, výška, typ písma, podklad, pozícia na stránke...*). Pôvodná verzia HTML predpokladala formátovanie textu až pri zobrazení stránky pomocou HTML značiek. V súčasnosti sa preferuje oddelenie významu obsahu stránky od jej vzhľadu, čo veľmi dobre umožňujú priamo kaskádové štýly. Veľmi výhodné je to pri väčšom webe s viacerými stránkami, ak požadujeme ich podobný vzhľad. Definícia štýlu je priamo v hlavičke dokumentu, ale častejšie a efektívnejšie je uloženie štýlu v externom súbore, na ktorý je v dokumente uvádzaný odkaz. Niektoré druhy formátovania sa dajú dosiahnuť len pomocou CSS. Jazyk bol navrhnutý a štandardizovaný organizáciou W3C. Boli vydané zatiaľ dve úrovne špecifikácie CSS1 a CSS2, dokončuje sa revízia CSS 2.1 a pracuje sa na verzii CSS3.

Výhody

Používanie kaskádových štýlov v porovnaní so samotným HTML formátovaním pomocou tagov a ich atribútov v praxi prináša výhody:

-Rozsiahlejšie možnosti formátovania. Napríklad pre formátovanie bloku textu – tj. určenie vzdialenosti od elementu či okraja stránky neponúka HTML nič. CSS má vlastnosti padding a margin. V HTML by bolo potrebné vytvoriť zložitú konštrukciu vnorených tabuliek.

-Jednoduchšia údržba webovej prezentácie. Ak chceme zmeniť nejaký detail, ako napríklad farbu nadpisu, nemusíme zložito prechádzať HTML kód alebo rôzne HTML šablóny, ale môžeme zmeniť iba jednu vlastnosť v CSS súbore, ktorý je vztiahnutý ku všetkým stránkam.

-Oddelenie štruktúry a štýlu. V jednom (*HTML*) dokumente budeme mať iba sémanticky označený obsah, a v druhom (*CSS*) dokumente máme definíciu vzhľadu stránok. Tým dosiahneme ľahšie strojové spracovanie samotného obsahu stránok, do ktorého sa nám nepletú prvky definujúce vzhľad.

-Cachovanie štýlov. Webový prehliadač si môže súbor so štýlmi uložiť do cache pamäte, čím sa môže dosiahnuť zrýchlenie načítania stránky. Na druhú stranu pri použití externého CSS súboru dochádza k ďalšiemu HTTP požiadavku navyše oproti tomu, keď by sme použili buď inline CSS alebo priame formátovanie HTML.

-Dynamické zmeny obsahu. CSS vlastnosti jednotlivých elementov môžeme dynamicky meniť pomocou JavaScriptu.

-Rôzne štýly pre rôzne výstupné zariadenia. Jednoducho tak môžeme definovať rôzny štýl pre tlač, projekciu, mobil, PDA, bežný prehliadač alebo dokonca pre každý prehliadač iný štýl. Špecifikácie CSS nezabúdajú ani na zrakovo postihnutých – je možné napísať štýly pre hlasový syntetizátor alebo hmatovú čítačku Braillovho písma.

-Koncový používateľ si môže napísať svoj vlastný CSS štýl pre ľubovoľnú stránku. Väčšina prehliadačov nejakým spôsobom podporuje používateľské štýly, takže používateľ si môže napríklad nastaviť, aby mal všetky odkazy na všetkých weboch vždy podčiarknuté alebo aby na tomto konkrétnom webe malo písmo vždy čiernu farbu.

Nevýhody

-Hlavnou nevýhodou je nekompatibilita s rôznymi verziami internetových prehliadačov. Prehliadače obsahujú v implementácii CSS chyby a je niekedy ťažké napísať kód tak, aby sa vo všetkých (*resp. v bežných*) prehliadačoch výsledok zobrazil rovnako.



-Problémy s cachovaním. Cachovanie bolo uvedené aj pri výhodách, ale ak máme CSS zapísané v externom súbore, môže vzniknúť aj problém: nenačíta sa vždy aktuálny obsah CSS súboru, pretože ten je uložený v cache pamäti.

16.1 Zapisovanie CSS

Na zapisovanie CSS máme viacej možností:



1. priamy zápis do dokumentu HTML:

```
<!DOCTYPE html>
<html><head>
<style>
  h1 { font-size: 200% }
  </style></head>
  <title>Titulok</title> <body>
  <h1>Nadpis</h1> </body>
 </html>
```

2. čerpanie CSS z externého súboru:

```
<!DOCTYPE html>
<html><head>
  <link rel="stylesheet" href="style.css" type="text/css" media="all"  />
  </head><title>Titulok</title><body>
  <h1>Nadpis</h1> </body>
 </html>
```

3. priamy zápis do elementu:

```
<html>
  <head></head>
  <title>Titulok</title><body>
  <h1 style="font-size: 150%">Nadpis</h1>
  </body>
 </html>
```

Pri prvom spôsobe sa definícia štýlu vzťahuje na celú stránku resp. na všetky nadpisy h1 na stránke. Druhý spôsob je rovnaký ako ten prvý, len s tým rozdielom, že CSS štýl je uložený v samostatnom súbore s ľubovoľným názvom. Pri treťom spôsobe bude štýl využitý len pre daný element. Syntax CSS je trochu odlišná, ak používame tretí spôsob. Vtedy nepoužívame zložené zátvorky, ale iba to, čo by mali obsahovať. Ostatná syntax má nasledujúci tvar:

```
h1 { font-size: 150%; }
```

Ako prvé určíme v CSS body {}, čiže telo stránky.

```
body {
  font-family: "Trebuchet MS", arial, sans-serif;
  font-size: 14px;
  color: #ff0000 }
```

Font-family nám určuje aký typ fontu písma budeme používať na stránke. V našom prípade je primárne písmo Trebuchet MS. Prvé písmo je zapísané v úvodzovkách preto, lebo jeho názov je viacslovný. Ak by však toto písmo náš systém neobsahoval, bude načítané ďalšie písmo, a to arial. Ak by ale toto písmo systém tiež nemal, bude použité písmo sans-serif. Ďalej v body určujeme tiež veľkosť a farbu písma.

16.2 Triedy, identifikátory a komentáre



Štandard CSS podporuje vytváranie tried. Jednoducho povedané, trieda predstavuje variant nejakého existujúceho elementu. Povedzme napríklad, že budeme v texte používať značku , no podľa miesta, kde bude použitá, bude vyžadovať niektoré modifikácie svojho vzhľadu. Náš uvedený výpis demonštruje vytvorenie triedy elementu strong, ktorá rešpektuje požadované variácie vo vzhľade. Trieda potom tejto značke pridáva vlastné formátovanie.

```
<!DOCTYPE html>
<html><head>
<title>Triedy</title>
  <style>
  strong.large {font-size: 24pt}
  strong.small {font-size: 8pt}
  </style>
  </head>
<body>
  <strong>Toto je štandardné strong.</strong>
  <strong class= "small"> Toto je strong triedy small.</strong>
  <strong class= "large">Toto je strong triedy large.</strong>
</body>
</html>
```

Trieda sa definuje uvedením názvu selektora, za ktorým nasleduje bodka a názov triedy (*napríklad strong.large*). Potom sa triede priraďuje hodnota normálnym spôsobom pomocou príkazov uvedených v zložených zátvorkách. U konkrétneho elementu potom triedu aktivujeme pomocou atribútu CLASS, ako to vidíme v značke na predchádzajúcom výpise. Je tiež možné vytvárať globálne triedy, ktoré môžu byť použité akýmkoľvek elementom. Stačí v selektore vynechať názov značky. Globálne triedy predstavujú výkonný nástroj, pretože sa dajú použiť pre ľubovoľnú značku na stránke tak, ako to ukazuje nasledujúci výpis.

```
<!DOCTYPE html>
<html><head>
<title>Globálne triedy</title>
  <style>
  .large {font-size: 24pt}
  .small {font-size: 8pt}
  </style>
  </head> <body> Štandardný text.
  <h2 class= "small">Trieda small v elemente h2 .</h2>
  <span class= "large">Trieda large v elemente span.</span>
  <p class= "small">Trieda small v elemente p .</p>
  <b class= "large">Trieda large v elemente b.</b> </body>
 </html>
```

Aby sme zapísali triedu do CSS, vždy pred jej názvom musí byť bodka. Za bodkou nasleduje názov triedy a hneď po nej zložené zátvorky. Do zložených zátvoriek sa píšu štýly, ktoré chceme prenechať dokumentu.

```
.trieda {
  width: 50px;
  height: 50px;
  background: #000000;
  }
```

Ak sme definovali štýl triedy, ideme na HTML kód. Zapíšeme ho do elementu .

```
<div class="trieda"></div>
```

Na stránku sme vložili čierný štvorec, sa používa najčastejšie na štruktúru stránky resp. tvorí jej vzhľad. Je to univerzálny element. Štýl triedy sme čerpali z CSS a jej štýl sme dostali do div-u pomocou atribútu class vloženého do elementu. Element s touto triedou môžeme na stránke použiť, koľkokrát chceme.

Identifikátory

Zápis do CSS je zhodný s triedami, ale je tu rozdiel v začiatočnom zápise. Namiesto bodky použijeme mriežku(#). CSS zápis bude vyzerať takto.

```
#identifikator {
  width: 50px;
  height: 50px;
  background: #000000;
  }
```

A teraz prejdeme k HTML zápisu.

```
<div id="identifikator"> </div>
```

Tu nastal menší rozdiel tiež. Namiesto class sme použili označenie štýlu pomocou id.

Komentáre
Komentáre do CSS vkladáme podobne ako pri viacriadkových komentároch v PHP, sú uložené medzi znakmi /**/.

```
/* Toto je
viacriadkovy komentar*/
  .css {
  background: red;
  }
```

Vložený komentár nemá žiadny vplyv na CSS kód a môžeme ho použiť kdekoľvek.

16.3 Formátovanie písma

Samotný jazyk HTML nikdy neponúkal príliš rozsiahle možnosti pre formát textu. Pred nástupom štýlov bolo prakticky nemožné zobrazovať formáty, ako je text s väčším riadkovaním, zmeniť medzery medzi slovami alebo nastaviť vlastné písmo. S nástupom CSS sa podobné problémy odstránili. Atribúty štandardu CSS, ktoré sa tykajú formátovaniu písma sú:

Font-family: názov písma
Font-style: štýl písma (*normal* | *italic* | *oblique*)
Font-variant: variant písma (*normal* | *small-caps*)
Font-weight: hrúbka písma (*normal* | *bold* | *bolder* | *lighter*)
Font-size: veľkosť písma
text-indent: odsadenie prvého riadku textu od ľavého okraja
text-align: zarovnanie textu horizontálne (*left* | *right* | *center* | *justify*)
text-decoration: efekty textu (*none* | *underline* | *overline* | *line-through*)
letter-spacing: nastavenie medzier medzi znakmi
word-spacing: nastavenie medzery medzi slovami
text-transform: transformácie textu (*capitalize* | *uppercase* | *lowercase* | *none*)
vertical-align: zarovnanie textu vertikálne
line-height: riadkovanie

16.4 Pseudotriedy

Hypertextové odkazy môžu obsahovať pseudotriedy. Tie sa nedefinujú v (X)HTML kóde, iba v CSS. Môžu byť kombinované s triedami. Oddeľujeme ich dvojbodkou.

- **active** - označuje miesto, kde sa nachádzame
- **hover** - označuje miesto, cez ktoré prechádzame myšou
- **link** - označuje miesto odkazu
- **visited** - označuje miesto, ktoré sme už navštívili
- **first-line** - zmení štýl prvého riadku
- **first-letter** - zmení štýl prvého písma

Prvé štyri sa používajú najmä pre odkazy.

```
 a:link { color: blue; }
a:visited { color: black; }
a:hover { color: red; }
a:active { color: green; }
```

16.5 Pozadia

Použitím selektora background-color v nasledovnom príklade dostaneme čierne pozadie.

```
 background-color: #000000;
```

Je tu ešte aj možnosť nastavenie obrázku ako pozadia. K tomu budeme využívať selektor background-image.

```
 background-image: url('images/obrazok.jpg');
```

URL nastavuje cestu k obrázku. Pre background-image môžeme obmedziť jeho opakovanie. Obmedzenie opakovania je nasledovné.

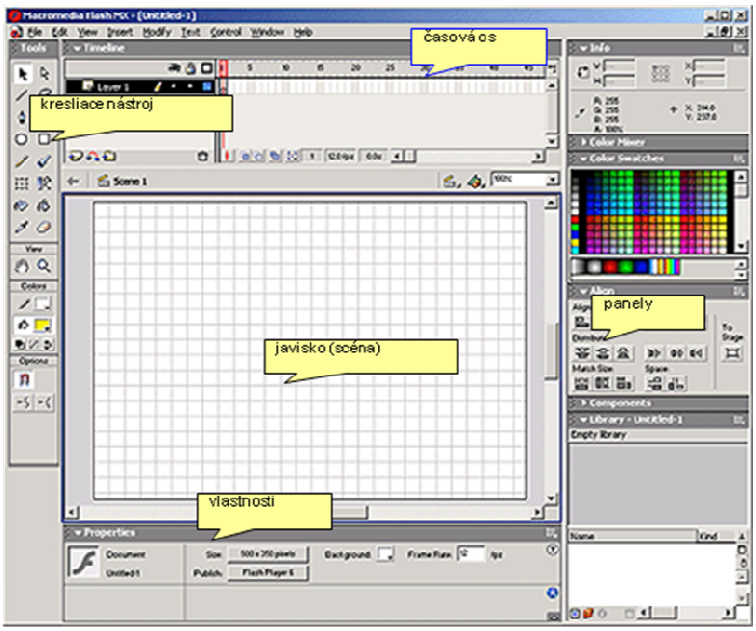
```
background-image: url('images/obrazok.jpg');
background-repeat: no-repeat;
```

Obmedzenie bolo myslené tak, aby sa obrázok neopakoval ako tapeta ani raz. Máme tu ale aj iné možnosti a to obrázok opakovať len vertikálne alebo horizontálne. Na to nám slúži repeat-x a repeat-y.

```
background-image: url('images/obrazok.jpg');
background-repeat: repeat-x;
```

Podrobný popis ohľadom CSS (rozmery, okraje, rámčeky, zobrazenie a pozície elementu, kurzory a pod.) nájdeme na školskom e-learningovom portáli:
http://www.spsknm.sk/moodle/ .
Prípadne na stránkach: http://www.jakpsatweb.cz/css/ , http://www.w3schools.com/css/, http://www.jakpsatweb.cz/css/, http://www.lstudio.szm.com/kniha/obsah.html, http://broe.hyperlink.cz/ucebnice/L7/index.html, http://www.jaknaweb.com/clanky/css/, http://www.pslib.cz/martin.janik/xhtml/?req=kurz20, http://programujte.com/?akce=clanek&cl=2007033102-css-1-uvod, http://www.prirucka.jonweb.cz/css.html,

17. Adobe Flash



Flash je veľmi silný nástroj umožňujúci vytváranie pôsobivých animácií a iných multimediálnych prvkov a ich umiestnenie na webové stránky. Flashové nie sú len internetové prezentácie, ale aj firemné prezentácie na nosičoch CD alebo hry. Základom flashovských aplikácií je vektorová grafika, ale môžu sa v nich skrývať aj importované zvuky, videosekvencie a bitové mapy. Vektorová grafika reprezentuje obrázok ako sústavu grafických objektov (čiary, kružnice atď.). Tieto objekty v sebe nesú informácie o hrúbke čiary, jej farbe a výplni kreslených objektov. Výhodou vektorovej grafiky je, že pri zmene rozmerov, rotácii objektov, úprave jednotlivých častí nedochádza k skresleniu. Informácie sa uchovávajú o objektoch ako o celkoch, nie o každom bode (pixel) zobrazenia. Nevýhodné je použitie vektorovej grafiky na zobrazenie fotorealistických detailov, pri ktorých je výhodnejšie obrázok popísať jednoducho sústavou grafických bodov a ich vlastností (farba, jas). Nástroj Adobe Flash obsahuje plnohodnotný skriptovací jazyk ActionScript. Základom vytváraného programu je časová os, na ktorú sa ukladajú jednotlivé súčasti obsahu. Každý dokument obsahuje knižnicu symbolov. Existujú tri typy symbolov (movie clip, button a graphic), ktoré môžu reagovať na preddefinované udalosti. Výhodou použitia symbolov je úspora miesta a možnosť súčasnej editácie všetkých inštancií určitého symbolu prostredníctvom editácie symbolu v knižnici. Program obsahuje kvalitné možnosti tvorby vektorovej grafiky, nástroje na manipuláciu so zvukmi a videom.

18. JavaScript

```
function LCMCalculat
  function checkInt(
    if (x % 1 != 0)
      throw new Type
      return x;
  }
  this.a = checkInt(
  this.b = checkInt(
}
LCMCalculator.protot
    gcd : function()
        var a = Math
```

Najrozšírenejší skriptovací jazyk na strane klienta je JavaScript. Patrí do kategórie prostriedkov dynamického webu na strane klienta, na strane servera nepredstavuje žiadnu dynamiku. No okrem DHTML (*Dynamic HTML*) sa JavaScript používa na písanie rozšírení pre mnohé aplikácie, napríklad Adobe Acrobat, Mozilla Firefox. Štandardizovaná verzia JavaScriptu je pomenovaná ako ECMAScript a z nej boli odvodené aj ďalšie implementácie, ako je napríklad ActionScript (*využíva sa pri flash animáciách*). Napriek názvu má JavaScript len veľmi málo spoločného s programovacím jazykom JAVA. Výhodou JavaScriptu je, že jeho zdrojový kód je priamo súčasťou stránky HTML a nepotrebujeme ho kompilovať. JavaScript je jazyk interpretovaný a na svoje fungovanie potrebuje prehliadač. Javascript dokáže riadiť správanie sa klientskeho prehliadača: ovplyvňovať obsah dokumentu, pracovať s dialógovými oknami, kontrolovať formuláre a hodnoty v ich poliach, pracovať s cookies, manipulovať s obrázkami. Jeho kód sa zapisuje priamo do HTML kódu medzi špeciálne značky, alebo ho je možné umiestniť do samostatného súboru. Nevýhodou je nekompatibilita s rôznymi verziami internetových prehliadačov. Z dôvodu, že program v JavaScripte sa spúšťa až po stiahnutí www stránky z Internetu, plynú určité bezpečnostné obmedzenia. JavaScript napríklad nemôže pracovať so súbormi, aby tým neohrozil súkromie používateľa.



18.1 JS Frameworky



JavaScript framework-y slúžia na zjednodušenie programovania klientských webových aplikácií. Tým umožňujú vytvárať profesionálne vyzerajúce a správajúce sa webové aplikácie aj programátorom, ktorý nie sú veľmi zdatní v používaní JavaScriptu. Taktiež tieto framework-y často implementujú funkcionalitu pre prácu s Ajax-om. Existuje veľké množstvo JS frameworkov. Najznámejšie sú jQuery, MooTools, Prototype, Dojo, ExtJS, medzi o niečo menej známe frameworky patria napr. Mochikit, GooXdoo, SmartClient, Echo či Rico. Tieto frameworky sa medzi sebou líšia syntaxou, rýchlosťou, kompatibilitou s jednotlivými internetovými prehliadačmi, veľkosťou zdrojového kódu, počtom pluginov a v neposlednom rade podporou a dokumentáciou prostredia. Rovnako ako pre JavaScript existuje veľa framework-ov aj pre iné programovacie jazyky (*PHP, Java, Python, atď.*). Využívanie framework-ov značne šetrí čas a odstraňuje nezanedbateľné množstvo chýb, ktorých sa môžu začínajúci programátori dopustiť. V súčasnej dobe je využívanie framework-ov, pre vývojárov webových aplikácií nevyhnutnosťou. (*pokiaľ nechcú nad rovnakým problémom stráviť ďaleko viac času)* Pekný prehľad dostupných frameworkov nájdeme na wikipédii:
http://en.wikipedia.org/wiki/List_of_web_application_frameworks

19. Webové šablóny

Prvým krokom pri tvorbe webovej stránky je voľba vzhľadu a rozloženia stránky. Ďalším krokom je vytvorenie tejto konštrukcie v HTML editore, ktorého obsluha vyžaduje značnú časovú náročnosť a zručnosť. Posledným krokom je pridanie obsahu. (*skutočný text a obrázky*) Webové šablóny sa starajú o prvé dva kroky za nás. Sú to obrazne povedané "polotovary", ktoré si môžeme ľubovoľne upraviť a dosiahnúť tak profesionálnu web stránku za minimálny čas.

Obava, že pomocou šablóny nezískame unikátnu webovú stránku, neobstojí, existuje viac ako pol miliardy www stránok a nespočetne veľa šablón od rôznych dodávateľov, ktoré sú na internete k dispozícii zadarmo. Navštíviť dve lokality, ktoré používajú rovnakú šablónu, je takmer nemožné. Navyše naša grafika a obsah sa bude prispôsobovať šablóne len do istej miery. Aj s minimálnou znalosťou HTML a CSS je jednoduché prispôsobiť šablónu, zmenou grafiky, farieb alebo písma. Na internete je veľké množstvo free webových šablón. ProWebCreative má širokú ponuku tém pre WordPress, Joomlu a Drupal. Stránka ponúka free témy a premium témy, za ktoré je potrebné zaplatiť, http://prowebcreative.com/. Webová stránka Themebot, http://themebot.com/free-website-templates/ ponúka taktiež veľké množstvo free šablón nielen pre Drupal, ale aj Joomlu a WordPress. Nižšie sú uvedné niektoré ďalšie web stránky na ktorých sa nachádzajú free web šablóny:
Templatesperfect.com, ramblingsoul.com, webtemplates.themebin.com, graformix.com, freecmstemplates.com, zimplit.org, templatesperfect.com, askgraphics.com, askgraphics.com, foxtheme.com, templateworld.com, mantisatemplates.com, breezy.cz, nuviotemplates.com, thestylecontest.com, dcarter.co.uk, designsbydarren.com, mitchinson.net, thenoodleincident.com, freelayouts.com, layouts.ironmyers.com, openwebdesign.org, cssplay.co.uk, oswd.org.
Pri tvorbe www stránok sa často stretneme s potrebou vložiť do stránky nejaký skript. Rovnako aj v tomto prípade nemusíme programovať, ale využiť niektorý z hotových skriptov. Odkazy na niektoré web stránky, kde sa nachádzajú scripty k stiahnutiu zdarma: SourceForge.net, hotscripts.com, php.jonweb.cz, jaknaweb.com, tvorba-webu.zdarek.com, hojko.com, juricek.net, tvorba-webu.cz, forum.freeserver.sk, scriptici.titl.name, crazydog.cz, blog.wwwtvorba.net, dusanko.net, html.over.cz.

19.1 Generátory šablón

Online generátory nám umožňujú tvorbu jednoduchých šablón. Ich ovládanie je pomerne nenáročné a práca s nimi nezaberá veľa času. Každú nami vytvorenú šablónu si môžeme stiahnuť na disk do počítača a následne s ňou pracovať v redakčnom systéme. Príkladom online generátora šablón pre WordPress. http://www.yvoschaap.com/wpthemegen Umožňuje nám upraviť veľkosť stránky, umiestnenie bočného menu buď vpravo alebo vľavo. Taktiež si môžeme zvoliť farebné rozlíšenie písma, odkazov, pozadia, orámovania textu atď. Online generátor Templatr http://templatr.cc/ má ešte pokročilejšie možnosti tvorby. Joomla Template Builder je generátor pre redakčný systém Joomla verzia 1.0 alebo 1.5. http://www.joomlatemplatebuilder.com/ na stránke je možné vyskúšať demo verziu. Po vyplnení daného formulára nám program vygeneruje PHP kód šablóny, ktorý môžeme ďalej použiť podľa vlastnej potreby. Taktiež máme možnosť náhľadu, ako by daná šablóna vyzerala. Cool Template http://cooltemplate.com/ umožňuje tvoriť šablóny pre redakčné systémy WordPress i Joomla a takiež vytvoriť HTML šablónu. Nastavovať tu je možné dizajn, rozloženie, pozadie, rám, tlačidlá, bloky, záhlavie a zápätie, menu. Má viacero funkcií slúžiacich na úpravu dizajnu jednotlivých prvkov stránky. Program dokáže samostatne navrhnúť napr. farebné rozlíšenie, formát článku, atď. Artisteer, jeden z najlepších nástrojov na tvorbu webových šablón, je všestranný a umožňuje nám tvoriť pre rôzne redakčné systémy. Práca v ňom nie je namáhavá, nie je potrebná ani znalosť PHP, CSS alebo HTML. Program má vlastné prvky (*napr. pozadia, efekty, textúry, štýly menu, ...*), no je tu možnosť si pridať do programu aj vlastné obrázky, ikony apod.

20. CMS systém (*redakčný systém*)

Redakčný systém, alebo *Content Management System* je všeobecne softvér na správu obsahu. Ten môže pozostávať z textov, obrázkov a iných mediálnych elektronických súborov. Účelom CMS systému je prehľadne spravovať obsah rôzneho druhu a umožniť viacerým stranám prístup k istým materiálom. To značne uľahčuje komunikáciu, najmä vo firmách.

Jednou z výhod mnohých CMS systémov je automatizácia firemných procesov, tzv. „workflow“. Pomocou CMS systému sa cez rôznych ľudí vo firme môže pohybovať istý materiál, ktorý môžu pripomienkovať, meniť, alebo inak spoločne modifikovať. CMS systém je často krát používaný aj ako nástroj na archiváciu dokumentov. Túto možnosť bežne využívajú mediálne firmy (*televízie, vydavateľstvá atď.*), ktorých predmet podnikania je priamo spojený s poskytovaním obsahu. CMS systémy však používajú aj všetky veľké medzinárodné spoločnosti na správu svojej firemnej komunikácie. Táto funkcia je však čím ďalej tým dôležitejšia aj pre malé a stredné firmy. CMS systémy firmám umožňujú nestratiť sa v dokumentoch a nájsť ich jednoducho a rýchlo na serveri, namiesto klasického archívu.
Väčšina CMS systémov podporuje nasledovné funkcie:

- vytváranie web stránok
- vytváranie položiek menu
- jednoduchá správa obsahu stránok (*texty, obrázky..*) bez znalostí programovania
- editovanie textov online
- WYSIWYG text editor (*What You See Is What You Get*)
- import a následná prezentácia textového a multimediálneho materiálu (*obrázky, videá, animácie...*)
- registrácia a administrácia používateľov
- pridelenie prístupu a práv používateľom prostredníctvom hesla
- e-mailová alebo iná notifikácia vybraných používateľov v prípade zmeny obsahu
- oddelenie obsahu od formy – mnohé CMS systémy majú napríklad preddefinované štýly písma (*veľkosť, typeface, farba*), takže keď pridáte text na stránku, automaticky sa zobrazí v danom štýle

20.1 Moduly pre CMS Systémy

Väčšina CMS systémov ponúka okrem bežných funkcií obsiahnutých v základnej verzii programu tzv. **moduly**. **Moduly** sú časti CMS systému, o ktoré sa dá rozšíriť základná verzia. Medzi moduly, o ktoré sa dá rozšíriť väčšina CMS systémov, patria napríklad: katalóg (*produktov*)**,** kalendár**,** e-shop**,** rezervácie**,** elektronický formulár**,** registrácia**,** mailinglist**,** štatistiky. Moduly, o ktoré sa dajú rozšíriť základné verzie Web-based CMS systémov, sa dajú rozdeliť do niekoľkých kategórií podľa toho, aký účel v rámci CMS systému spĺňajú, a to na business moduly, community moduly a maketingové moduly.

20.2 Moodle



Modular Object-Oriented Dynamic Learning Environment patrí medzi LMS *open source* systémy, ktorý je určený na tvorbu internetových kurzov. Moodle je softvérový balíček pre vytváranie kurzov založených na Internete a web stránkach. Je to projekt, ktorý neustále napreduje, určený na podporu vyučovania pre realizáciu plne dištančnej výučby na Internete. Pomocou jednoduchých modulov je možné vytvárať diskusné fóra, kvízy, prehľady, denníky, publikovať zdroje a oznámenia. Pracovať s LMS Moodle môžu všetci (*študenti, učitelia, administrátori*) prostredníctvom bežných internetových prehliadačov. Výučba v Moodle prebieha v kurzoch, ktorých členenie aj obsah je plne v rukách učiteľa, prípadne učiteľov. Kurzy môžu byť vnútorne rozdelené do menších celkov (*tém alebo týždňov*). Náplňou kurzu je rad učiteľom definovaných činností žiakov. Pre podporu týchto činností sú pre Moodle vyvíjané relatívne nezávislé moduly, z ktorých učiteľ skladá svoj kurz. Učenie sa v prostredí Moodle spočíva v aktívnej účasti študentov na zadaných úlohách, prispievaní do diskusných fór, účasti na hlasovaní, vyplnení testov, vypracovaní písomnej práce a pod. Úkoly je možné obodovať a tým motivovať študenta k aktívnej činnosti v kurze. www.moodle.org

20.3 WordPress



WordPress je slobodný *open source* publikačný systém CMS pre blogy a webstránky. Je vyvíjaný pod licenciou GNU GPL . Má širokú používateľskú a vývojársku komunitu, počet stiahnutí verzie 3.0 samotnej dosahuje takmer 10 miliónov. WordPress je používaný na viac ako dvanástich percentách celo svetového internetu, prekonáva tak *open source* CMS ako Joomla či Drupal, ktoré sa držia pod desiatimi percentami. Aj keď je určený predovšetkým na písanie blogov, vďaka pluginom sme schopní napísať aj zložitejšie stránky. Redakčný systém WordPress sa stal víťazom Packt Open Source Awards 2010 v kategórii "sieň slávy najlepších *open source* CMS". Týmto nechal za sebou také mená ako druhý Drupal či tretiu Joomlu. (*Hlavne kvôli svojej jednoduchosti.*) http://wordpress.org

20.4 Joomla



je redakčný systém určený pre vytváranie web stránok a ďalších online aplikácií. Joomla je *open source* riešenie, voľne dostupné pre každého. Predstavuje jeden z najvýkonnejších redakčných systémov na svete. Profesionálni tvorcovia web stránok používajú redakčný systém Joomla od jednoduchých web stránok až po komplexné firemné aplikácie. Joomla sa vyznačuje jednoduchou správou, spoľahlivosťou a bezpečnosťou. www.joomla.org

20.5 phpBB



Od doby, keď bolo phpBB v roku 2000 vytvorené, sa stalo najpoužívanejším a najznámejším *open source* systémom na svete na tvorbu komunít. Svojimi rozsiahlymi a vysoko prispôsobiteľnými funkciami a vzhľadom phpBB vyhovuje skoro každému na vytvorenie vlastného fóra. S miliónmi inštalácií phpBB na celom svete je phpBB jeden z najznámejších systémov, ako vytvoriť a podporovať komunity. www.phpbb.com

20.6 Piwigo



Piwigo je *open source* webová fotogaléria. Rozšírenia a doplnky robia Piwigo jednoducho prispôsobiteľné. Obsahuje funkcie pre publikovanie a správu fotografií, škálovateľnosť a inteligentné funkcie, ako je prechádzanie podľa kategórií, značiek, alebo chronológie. Piwigo sa dá ľahko upravovať či rozširovať o nové vzhľady a funkcie. Samozrejmosťou je kompletná lokalizácia do slovenčiny a jednoduchá inštalácia. http://piwigo.org/

20.7 Drupal



Drupal je populárny *open source* systém pre správu obsahu (*Content Management Systém – CMS*). Ľahká rozšíriteľnosť pomocou modulov a množstvo možností, ktoré poskytuje, z Drupalu robí jeden z najlepších redakčných systémov. Používateľské rozhranie Drupalu umožňuje pridávanie podstránok, navigácie, obrázkov a súborov a jeho bezpečnostné prvky umožňujú nastavenie rôznych úrovní oprávnení pre používateľov. Drupal je špičkový, uznávaný a najrýchlejšie vyvíjaný CMS systém na svete. Vyznačuje sa jedinečnou univerzálnosťou. Je to platforma na ktorej sa dá vybudovať akýkoľvek projekt, od firemného webu po multiužívateľský portál. Existuje cca 3000 hotových modulov na doplnenie funkcionality. Popri použití hotových modulov sa tvoria nové špecifické moduly. Drupal je odporúčaný ako jeden z najlepších webových frameworkov a jeho použitie vysoko rastie, sú pre neho typické tieto vlastnosti:

Modularita – Vhodný pre vytvorenie blogov E-shopov fóra. To všetko Drupal umožňuje vďaka svojmu modulárnemu systému: Malé, ale stabilné a rýchle jadro s dobrým rozhraním a modulmi, na ktorých stavia. Každý môže vytvoriť vlastný modul, zoznam modulov je udržiavaný na domovskej stránke Drupalu.
Kvalita – Do jadra Drupalu sa nedostávajú neoverené patche, jadro má tiež veľmi dobre navrhnutú štruktúru. To z neho robí bezpečný a stabilný systém.
Open Source – GNU/GPL license, PHP programovací jazyk, podpora pre MySQL a PostgreSQL, pripravovaná podpora pre MS SQL a Oracle. www.drupal.org **Podrobný popis inštalácie a správy Drupalu, nájdeme na školskom e-learningovom portáli** http://www.spsknm.sk/moodle/.

21. Charakteristika internetového obchodu

Internetový obchod alebo e-shop (*e-obchod*, *online obchod*) je obchod na internete. Opakom internetového obchodu je klasický, v tejto súvislosti prezývaný "kamenný" obchod. Výhodou internetového obchodu sú spravidla nižšie náklady než v prípade kamenných obchodov, a tým aj nižšie ceny. Na slovenskom trhu sa nachádza dosť široké spektrum dodávateľov internetových obchodov, pričom nimi ponúkané produkty a služby sa samozrejme v mnohých ohľadoch líšia. Popri komerčných riešeniach existuje veľa *open source* e-shopov, vyvíjaných dobrovoľníkmi a ponúkaných ako slobodný softvér, takže založenie a prevádzka e-shopu je dnes už záležitosť prístupná aj bežnému používateľovi internetu. Podobne ako pri tvorbe vlastnej webstránky môžeme aj v tomto prípade využiť jeden z množstva redakčných systémov ponúkaných na trhu. Medzi tie najpoužívanejšie, ktoré sú dostupné vrátane slovenskej lokalizácie patria: PrestaShop, VirtueMart, Magento, OpenCart, osCommerce a ZenCart.

21.1 PrestaShop

PrestaShop je plnohodnotný *open source* elektronický obchod s možnosťami automatického výpočtu poštovného a dopravného pre rôzne možnosti dopravy, rôzne zľavy, akcie, ponuka jednoduchého výberu farebných a veľkostných variantov produktu s rôznou cenou, ako aj mnoho iných vlastností. Viťaz *Open-Source Award* 2010 v kategórii *E-Commerce Application.* Presta Shop je jedna z najpoužívanejších alternatív v zahraničí, ale udomácňuje sa aj na Slovensku. Komunita používateľov vo svete presahuje 75 000 stránok. PrestaShop je rýchly, nenáročný internetový obchod s intuitívnym ovládaním, ktorý poskytuje množstvo spôsobov administrácie v základnom balíku. Základný balík obsahuje možnosti pridávania produktov, štatistiky, spravovanie zákazníkov, platieb a aj zamestnancov. V rozšíreniach e-shopu si môžeme zadarmo pridať napríklad preklad gTranslate, jednoduché maily, správanie najčastejších zákazníkov, nové šablóny a pod. http://www.prestashop.com/

21.2 Porovnanie open source e-shop systémov

Zákaznícka podpora	Oxid e-shop	OpenCart	Oscomm	ZenCart	VirtueMart	Magento	PrestaShop	Ubercart
Zákaznícke fórum	✓	✓	✓	✓	✓	✓	✓	✓
Technická podpora zadarmo								
Technická podpora spoplatnená	✓	Cez partnera	⊘	⊘	⊘	✓	⊘	⊘

Vysvetlivky:
✓ áno, obsahuje ⊘ nie, neobsahuje
No Limit neobmedzený počet dodatočný modul podpora komunity

Funkcie e-shopu	Oxid e-shop	OpenCart	Oscomm	ZenCart	VirtueMart	Magento	PrestaShop	Ubercart
Kategórie	No Limit	No Limit	No Limit	No Limit	No Limit	No Limit	No Limit	No Limit
Tovar	No Limit	No Limit	No Limit	No Limit	No Limit	No Limit	No Limit	No Limit
Parametre tovaru	✓	✓	✓	✓	✓	✓	✓	✓
Obrázková galéria	✓	✓	✓	✓	✓	✓	✓	✓
Vyhľadávanie	✓	✓	✓	✓	✓	✓	✓	✓
Import dát	Import modul	CSV Raw Import	MySQL / modul	MySQL / CSV / ext.modul	CSV	XML / XLS / Tool	CSV	XML
XML export dát	✓	⊘	⊘	⊘	✓	⊘	⊘	✓
Skladová evidencia	✓	✓	✓	✓	✓	✓	✓	✓
Darčekové poukážky	✓	⊘	✓	✓	✓	✓	✓	🔌
Vernostný program	🔌	⊘	✓	⊘	⊘	✓	✓	🔌
Zľavové kupóny	✓	✓	✓	✓	✓	✓	✓	🔌
Newsletter (novinky na e-mail)	✓	✓	✓	✓	✓	✓	✓	✓
Registrácia zákazníkov	✓	✓	✓	✓	✓	✓	✓	✓
História objednávok	✓	✓	✓	✓	✓	✓	✓	✓

Doprava a platba	Oxid e-shop	OpenCart	Oscomm	ZenCart	VirtueMart	Magento	PrestaShop	Ubercart
Konfigurovateľné spôsoby platby	✓	✓	✓	✓	✓	✓	✓	✓
Konfigurovateľné druhy dopravy	✓	✓	✓	✓	✓	✓	✓	✓
Doplnkové nástroje	**Oxid e-shop**	**OpenCart**	**Oscomm**	**ZenCart**	**VirtueMart**	**Magento**	**PrestaShop**	**Ubercart**
Sprievodca začiatočným nastavením	✓	⊘	✓	✓	⊘	⊘	✓	⊘
Jazykové mutácie	✓	✓	✓	✓	✓	✓	✓	✓
SEO (search engine optimization)	✓	✓	✓	✓	✓	✓	✓	✓
Mapa stránky (sitemap)	⊘	🔌	🔌	🔌	🔌	🔌	🔌	🔌
Technické riešenie	**Oxid e-shop**	**OpenCart**	**Oscomm**	**ZenCart**	**VirtueMart**	**Magento**	**PrestaShop**	**Ubercart**
Podporované operačné systémy	Nezávisí	Windows, Linux, Unix	Windows, Linux, Unix, OSX	Windows, Linux, Unix	Linux	Windows, Linux, Unix	Nezávisí	Linux
Programovací jazyk	php	php	php	php	php	php	php	php
Použitá databáza	MySQL	MySQL	MySQL	MySQL	MySQL	MySQL	MySQL	MySQL
Integrovaný CMS	✓	✓	✓	✓	✓	✓	✓	✓

22. Literatúra

Cody Lindley: jQuery - Kuchařka programátora, Computer Press,2010, ISBN: 9788025131527
Connolly T., Begg C., Holowczak R.: Mistrovství – Databáze - Profesionální průvodce tvorbou efektivních databází, Computer Press,2009, ISBN:978-80-251-2328-7
Dan Rubin, Ian Lloyd, Jeff Croft: Mistrovství v CSS Computer Press, 2007, ISBN: 978-80-251-1705-7
Darie C., Sirovich J.: SEO v PHP-Programujeme profesionálně, Computer Press,2008, ISBN: 978–80–251–2083–5
DUBOIS, P.: MySQL profesionálně. Praha: Mobil Media, 2003, ISBN: 80-865-9341-X
GUTMANS, A.: Mistrovství v PHP 5. Praha: Computer Press,2005, ISBN: 80-251-0799-X
Hernandez, M.J., Viescas, J.L.: Myslíme v jazyku SQL - tvorba dotazu. Praha: GRADA Publishing, 2004. ISBN 80-247-0899-X
KOFLER, M.: Mistrovství v MySQL. Praha: Computer Press, 2007, ISBN 978-80-251-1502-2
KOSEK, J.: PHP Tvorba interaktivních internetových aplikací. Praha: Grada, 1999, ISBN: 80-716-9373-1
Lacko, L.: WEB a databáze. Programujeme internetové aplikace. Praha: Computer Press, 2001. ISBN 80-7226-555-5
LACKO, Ľ.: PHP a MySQL Hotová řešení. Praha: Computer Press,2007, ISBN 80-251-1695-1
Marc Delisle: phpMyAdmin – efektivní správa MySQL. Zoner Press, 2004, ISBN 80–86815–09–9
Mohammed J. Kabir: Apache Server 2 Kompletní příručka administrátora, Computer Press, 2004, ISBN: 80-251-0319-6
Molinaro A.: SQL Kuchařka programátora, Computer Press,2009, ISBN:978-80-251-2617-2
MySQL Group. MySQL Documentation [online]. Dostupný z WWW: http://dev.mysql.com/doc/#refman
Nicholas Z. Zakas: JavaScript pro webové vývojáře, Computer Press, 2009, ISBN 978-80-251-2509-0
PANKRÁC, M.: PHP a MySQL bez předchozích znalostí. Praha: Computer Press,2007, ISBN: 80-251-1758-3
Polzer J.: 333 tipů a triků pro Drupal, Computer Press, 2010, ISBN: 9788025129425
Rebenschied S.: Macromedia Flash 8 Professional, Computer Press, 2007, ISBN: 80-251-1696-4
Ryan K. Stephens, Ronald R. Plew: Naučte sa SQL za 28 dní, Computer Press, 2010, ISBN:978-80-251-2700-1
Wempen F.: HTML a CSS, Computer Press, 2007, ISBN: 978-80-251-1505-3

Internet–web: odkazy umiestnené na jednotlivých stranách učebného textu