

Databázový systém pro evidenci přístrojů

Database system for tools evidence

Václav Dorazil

Bakalářská práce
2011



Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky
akademický rok: 2010/2011

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení: **Václav DORAZIL**
Osobní číslo: **A08682**
Studijní program: **B 3902 Inženýrská informatika**
Studijní obor: **Informační a řídicí technologie**

Téma práce: **Databázový systém pro evidenci přístrojů**

Zásady pro vypracování:

1. Analyzujte problematiku a vypracujte literární rešerši na dané téma.
2. Navrhněte strukturu databáze IS.
3. Vytvořte databázi a webovou aplikaci včetně popisu řešení.
4. Popište způsob implementace IS.
5. Zajistěte správu databáze a její zabezpečení.

Rozsah bakalářské práce:

Rozsah příloh:

Forma zpracování bakalářské práce: **tištěná/elektronická**

Seznam odborné literatury:

1. PROKOPOVÁ, Z.: Databázové systémy MySQL+PHP. FAI UTB Zlín, s. 126, 2006, Vysokoškolská skripta. ISBN 80-7318-486-9.
2. LACKO, L. PHP a MySQL ? hotová řešení. Computer Press, 2006, ISBN: 80-251-1249-7.
3. RIORDAN, Rebecca M. Vytváříme relační databázové aplikace. Praha : Computer Press, 2000. 280 s. ISBN 80-7226-360-9.
4. SCHNEIDER, R.,D. MySQL – Oficiální průvodce tvorbou, správou a laděním databází. Grada, ISBN: 80-247-1516-3.
5. ULLMAN, L. PHP a MySQL, Computer Press, Brno, 2004,534 s. ISBN 80-251-0063-4.
6. WELLING, L., THOMSON, L. MySQL Průvodce základy databázového systému. Computer Press, Brno, 2005, ISBN: 80-251-0671-3.
7. MILOSLAV PONKRÁC, PHP a MySQL bez předchozích znalostí, Computer Press, 2007, ISBN: 978-80-251-1758-3.

Vedoucí bakalářské práce:

doc. Ing. Zdenka Prokopová, CSc.

Ústav počítačových a komunikačních systémů

Datum zadání bakalářské práce:

25. února 2011

Termín odevzdání bakalářské práce:

7. června 2011

Ve Zlíně dne 25. února 2011

prof. Ing. Vladimír Vašek, CSc.
děkan



prof. Ing. Vladimír Vašek, CSc.
ředitel ústavu

ABSTRAKT

Hlavním cílem bakalářské práce bylo vytvořit databázový systém pro evidenci různých typů přístrojů na Fakultě aplikované informatiky UTB ve Zlíně.

Bakalářská práce je zaměřena na propojení databázového systému MySQL se skriptovacím jazykem PHP.

V teoretické části jsou uvedeny základní pojmy a definice užívané v databázových systémech. Je zde popsáno propojení skriptovacího jazyka PHP s MySQL a zabezpečení databázových aplikací.

Praktická část obsahuje návrh a zpracování databáze pro evidenci přístrojů. Vytvořený databázový systém umožňuje archivaci, vkládání, mazání a vypůjčování přístrojů.

Klíčová slova: SQL, PHP, databázové systémy, formuláře pro zpracování dat.

ABSTRACT

This thesis is focused on connection between MYSQL database system and PHP scripting language.

In teoretical part of the thesis are described basic terms and definitions which are used in database systems. In teoretical part is also connection between PHP scripting language with MySQL and safety mechanisms for database applications.

Practical part of the thesis contains structure and solution of database system for tools evidention. Database system allows tools archivation, insertion, deletion and borrows of existing records.

Keywords: SQL, PHP, database systems, forms for data processing.

Poděkování:

Děkuji všem, kteří mě po celou dobu studia a vytváření této práce podporovali. Speciálně děkuji paní doc. Ing. Zdence Prokopové CSc. za její odborné připomínky a osobní přístup.

Prohlašuji, že

- beru na vědomí, že odevzdáním bakalářské práce souhlasím se zveřejněním své práce podle zákona č. 111/1998 Sb. o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších právních předpisů, bez ohledu na výsledek obhajoby;
- beru na vědomí, že bakalářská práce bude uložena v elektronické podobě v univerzitním informačním systému dostupná k prezenčnímu nahlédnutí, že jeden výtisk bakalářské práce bude uložen v příruční knihovně Fakulty aplikované informatiky Univerzity Tomáše Bati ve Zlíně a jeden výtisk bude uložen u vedoucího práce;
- byl/a jsem seznámen/a s tím, že na moji bakalářskou práci se plně vztahuje zákon č. 121/2000 Sb. o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) ve znění pozdějších právních předpisů, zejm. § 35 odst. 3;
- beru na vědomí, že podle § 60 odst. 1 autorského zákona má UTB ve Zlíně právo na uzavření licenční smlouvy o užití školního díla v rozsahu § 12 odst. 4 autorského zákona;
- beru na vědomí, že podle § 60 odst. 2 a 3 autorského zákona mohu užít své dílo – bakalářskou práci nebo poskytnout licenci k jejímu využití jen s předchozím písemným souhlasem Univerzity Tomáše Bati ve Zlíně, která je oprávněna v takovém případě ode mne požadovat přiměřený příspěvek na úhradu nákladů, které byly Univerzitou Tomáše Bati ve Zlíně na vytvoření díla vynaloženy (až do jejich skutečné výše);
- beru na vědomí, že pokud bylo k vypracování bakalářské práce využito softwaru poskytnutého Univerzitou Tomáše Bati ve Zlíně nebo jinými subjekty pouze ke studijním a výzkumným účelům (tedy pouze k nekomerčnímu využití), nelze výsledky bakalářské práce využít ke komerčním účelům;
- beru na vědomí, že pokud je výstupem bakalářské práce jakýkoliv softwarový produkt, považují se za součást práce rovněž i zdrojové kódy, popř. soubory, ze kterých se projekt skládá. Neodevzdání této součásti může být důvodem k neobhájení práce.

Prohlašuji,

- že jsem na bakalářské práci pracoval samostatně a použitou literaturu jsem citoval. V případě publikace výsledků budu uveden jako spoluautor.
- že odevzdaná verze bakalářské práce a verze elektronická nahraná do IS/STAG jsou totožné.

Ve Zlíně

.....

podpis diplomanta

OBSAH

ÚVOD	9
I TEORETICKÁ ČÁST	10
1 ZÁKLADNÍ TERMINOLOGIE DATABÁZOVÉHO SYSTÉMU	11
1.1 DATABÁZE	11
1.2 SYSTÉM ŘÍZENÍ BÁZE DAT.....	11
1.3 DATABÁZOVÝ SYSTÉM	11
1.4 RELAČNÍ DATABÁZE	12
1.5 TYPY TABULEK.....	13
1.5.1 Tabulky typu MyISAM	13
1.5.2 Tabulky typu InnoDB.....	13
1.5.3 Tabulky typu HEAP	14
1.5.4 Dočasné tabulky	14
1.6 INTEGRITNÍ OMEZENÍ	15
1.6.1 Primární klíč.....	15
1.6.2 Cizí klíč	16
1.7 NORMÁLNÍ FORMY	16
1.7.1 1. normální forma – 1NF.....	17
1.7.2 2. normální forma – 2NF.....	17
1.7.3 3. normální forma – 3NF.....	17
1.7.4 Boyce-Coddova normální forma – BCNF.....	17
1.8 VZTAHY MEZI TABULKAMI	17
1.8.1 Vztah jedna k jedné (1 : 1, one-to-one).....	18
1.8.2 Vztah jeden k více (1 : N, one-to-many).....	18
1.8.3 Vztah jeden k více (1 : N, one-to-many).....	18
1.9 DATOVÉ TYPY	18
1.9.1 Celá čísla	18
1.9.2 Čísla s pevnou a s plovoucí desetinou čárkou.....	19
1.9.3 Datum a čas	19
1.9.4 Řetězce znaků.....	20
1.9.5 Další datové typy.....	21
2 PROPOJENÍ MYSQL A SKRIPTOVACÍCH JAZYKŮ	22
2.1 PŘÍSTUP K DATABÁZÍM Z INTERNETU.....	22
2.2 PHP.....	22
2.3 PROČ POUŽÍVAT PHP A JAK FUNGUJE?	23
2.4 PROPOJENÍ PHP A MYSQL	23
3 ZABEZPEČENÍ	25

3.1	ZABEZPEČENÍ WEBOVÉ APLIKACE.....	25
3.2	ZABEZPEČENÍ DATABÁZE MYSQL	25
3.2.1	Uživatelská oprávnění	25
3.2.2	Tvorba uživatelů a jejich práv	25
II	PRAKTICKÁ ČÁST	27
4	ANALÝZA PROBLEMATIKY	28
4.1	SPECIFIKA PRO PŘÍSTROJE.....	28
4.2	JEDNOZNAČNÁ LOKALIZACE PŘÍSTROJE	28
4.3	MOŽNOST VYPŮJČENÍ PŘÍSTROJE	29
5	DATABÁZE PRO EVIDENCI PŘÍSTROJŮ	30
5.1	VYTVORENÍ DATABÁZE	31
5.2	VYTVÁŘENÍ TABULEK V DATABÁZI	31
5.2.1	Vytvoření tabulky tool.....	32
5.2.2	Vytvoření tabulky institute.....	33
5.2.3	Vytvoření tabulky room	34
5.2.4	Vytvoření tabulky borrow	35
5.2.5	Vytvoření tabulky person	36
5.3	STRUKTURA DATABÁZE TOOLS_DATABASE	37
5.3.1	ERA diagram databáze tools_database	38
5.3.2	Primární klíče v databázi tools_database	39
5.3.3	Cizí klíče v databázi tools_database	39
5.4	IMPORT A EXPORT DATABÁZE TOOLS_DATABASE	40
6	WEBOVÁ APLIKACE.....	42
6.1	STRUKTURA WEBOVÉ APLIKACE	43
6.2	VKLÁDÁNÍ ZÁZNAMŮ	44
6.2.1	Vložení nové zodpovědné osoby.....	44
6.2.2	Vložení nového přístroje	45
6.3	KONTROLA HODNOT PŘI VKLÁDÁNÍ ZÁZNAMŮ	48
6.4	ZOBRAZOVÁNÍ ZADANÝCH ZÁZNAMŮ VE WEBOVÉ APLIKACI.....	48
6.5	VYHLEDÁVÁNÍ A MAZÁNÍ ZÁZNAMŮ VE WEBOVÉ APLIKACI.....	49
7	ZABEZPEČENÍ DATABÁZOVÉHO SYSTÉMU.....	50
	ZÁVĚR	52
	ZÁVĚR V ANGLIČTINĚ	53
	SEZNAM POUŽITÉ LITERATURY	54
	SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK	55
	SEZNAM OBRÁZKŮ	56
	SEZNAM TABULEK.....	57
	SEZNAM PŘÍLOH.....	58

ÚVOD

Ve světě informačních technologií je základem informace. Databázové systémy slouží právě pro operace s informacemi, a proto je můžeme považovat za jeden ze základů moderních informačních technologií.

Databázové systémy zaručují archivaci a distribuci informací nejen interně v rámci subjektu, ale zprostředkovávají i tok informací ze společnosti ven. Pro každou společnost je životně důležité, aby její databázový systém pracoval efektivně, spolehlivě, přesně a zaručoval maximální bezpečnost zpracovávaných dat.

V moderním a rychle se vyvíjejícím světě e-commerce získaly databázové systémy ještě více na důležitosti. Pro internetové obchody, aukční portály jakož i všechny velké internetové společnosti jsou systémy, které zpracovávají data jejich uživatelů, základním stavebním kamenem.

Rovněž pro efektivní řízení společností jsou databázové systémy nepostradatelné. Přesné evidence přístrojového vybavení, zaměstnanců, seznamy prodejů a nákupů vyžadují databázový systém, který bude do detailů splňovat požadavky konkrétní společnosti.

V teoretické části bakalářské práce jsou popsány základy potřebné pro vytváření databází pomocí jazyků SQL a PHP. Praktická část obsahuje vlastní řešení databázového systému pro evidenci přístrojů. Jednou z hlavních rolí tohoto systému je přispět k efektivnímu řízení společnosti a optimalizovat využití již pořízených přístrojů, nebo minimalizovat nákup nadbytečného vybavení.

I. TEORETICKÁ ČÁST

1 ZÁKLADNÍ TERMINOLOGIE DATABÁZOVÉHO SYSTÉMU

1.1 Databáze

Databázi chápeme jako úložiště údajů, které jsou uloženy, zpracovávány nezávisle na aplikačních programech. Databáze zapouzdřují jednak vlastní údaje, ale i relační vztahy mezi jednotlivými prvky a objekty v databázi, schémata popisující struktury údajů a integritní omezení. [1]

1.2 Systém řízení báze dat

Speciální software pro přístup k údajům v databázi nazýváme česky systém řízení báze dat (SŘBD), nebo anglicky Database Management System (DBMS). Uživatel, nebo aplikace, potom nemusí vědět, kde přesně se data nachází, protože k údajům v databázi přistupují přes systém řízení báze dat. Komunikace klienta nebo aplikace SŘBD probíhá pomocí jazyka SQL.

1.3 Databázový systém

Pod databázovým systémem si představujeme údaje, které jsou uloženy a spravovány v databázi, a také software pro přístup k těmto údajům. Všechny nástroje a postupy, které můžeme využít při práci s databázemi, nazýváme databázová technologie. Databázový systém je tvoří systémem řízení báze dat (SŘBD) a databáze. Databázové systémy můžeme rozčlenit na:

- **Hierarchické a síťové:** u hierarchických a síťových databázových systémů jsou aplikační programy závislé na databázi, z čehož vyplívá například problematika údržba a podobně[1].
- **Relační:** pro relační databáze je typická neprocedurální manipulace s daty, ukládání jednoduchých dat s pevnou strukturou, tedy v tabulkové formě[1].
- **Objektové:** databázové systémy používají složité datové struktury a složitá pravidla založená na obchodní logice (Business rules) [1].

1.4 Relační databáze

Nejpopulárnějším modelem datového úložiště je relační databáze, která se zrodila z klíčové studie s názvem „A Related Model of Data for Large Shared Data Banks“ (relační model dat pro rozsáhlé sdílené banky dat), kterou napsal Dr. E. F. Codd v roce 1970. Jazyk SQL se vyvinul tak, aby sloužil principům relačního modelu databáze. Dr. Codd definoval pro relační model 13 pravidel, kterým se říká 12 pravidel Dr. Codd:

0. Relační databázový systém musí být schopen spravovat databáze jen s využitím svých relačních schopností.
1. Informace: Všechny informace v relační databázi (včetně názvů tabulek a sloupců) jsou reprezentovány explicitně jako hodnoty v tabulkovém formátu.
2. Zaručený přístup: U každé hodnoty v relační databázi je zaručeno, že bude přístupná prostřednictvím kombinace názvu tabulky, hodnoty primárního klíče a názvu sloupce.
3. Systematická podpora nulitních hodnot: Databázový systém poskytuje systematickou podporu pro práci s nulitními hodnotami (neznámé či nepoužitelné data), které jsou odlišné od výchozích hodnot a nezávislé na jakékoli doměně.
4. Aktivní relační katalog dostupný online: Popis databáze a jejího obsahu je reprezentován na logické úrovni v tabulkové formě, a proto lze nad ním spouštět dotazy pomocí databázového jazyka.
5. Ucelený datový podjazyk: Alespoň jeden podporovaný jazyk musí mít dobře definovanou syntaxi a musí být ucelený. Musí podporovat definici dat, manipulaci s daty, integritní pravidla, autorizaci a transakce.
6. Aktualizování pohledů: Veškeré pohledy, které lze teoreticky aktualizovat, mohou být aktualizovány pomocí systému.
7. Vkládání, aktualizace a mazání na úrovni množin: Databázový systém podporuje nejen získávání dat na úrovni množin, ale také vkládání, aktualizace a mazání na úrovni množin.
8. Fyzická datová nezávislost: Změna fyzických přístupových metod nebo úložných struktur nemá z hlediska logického pohledu na aplikační a ad-hoc programy žádný vliv.

9. Logická datová nezávislost: Změna tabulkových struktur nemá z logického hlediska na aplikační a ad-hoc programy v maximální možné míře vliv.
10. Integritní nezávislost: Databázový jazyk musí být schopen definovat integritní pravidla. Tato pravidla musejí být uložena v katalogu dostupném online a nesmí existovat množina pro jejich obejití.
11. Nezávislost distribuce: První distribuce nebo redistribuce dat nemá na aplikační a ad-hoc požadavky z logického hlediska vliv.
12. Nenarušitelnost: Nesmí existovat možnost umožňující obejití integritních pravidel definovaných v databázovém jazyku pomocí jazyků nižší úrovně. [3]

1.5 Typy tabulek

Pokud budujeme databázi v MySQL, tak při vytváření tabulek vždy musíme zvolit typ tabulky z nabízeného seznamu. V drtivé většině využíváme tabulku InnoDB, která je také přednastavená. Níže popisují 3 nejvyužívanější druhy tabulek.

1.5.1 Tabulky typu MyISAM

MyISAM představuje v MySQL standardní typ tabulek. Jedná se o stabilní, vyspělý a jednoduše spravovatelný typ tabulek. Pokud nemáte žádný zvláštní důvod, proč používat jiný typ tabulek, pak použijte právě tento. [4]

1.5.2 Tabulky typu InnoDB

Typ InnoDB je v porovnání s předchozím typem tabulky nový. Podporuje všechny vlastnosti typu MyISAM, navíc se pyšní dvěma odlišnostmi:

- Databázové operace v tabulkách InnoDB se dají pouštět jako transakce. Transakce jsou v mnoha případech bezpečnější a občas jsou i rychlejší, pokud se můžete vyhnout provádění tzv. operací Locking.
- Tabulky InnoDB podporují používání pravidel integrity.

Bohužel existují i důvody, které hovoří proti používání tabulek typu InnoDB:

- InnoDB ještě nedosáhl vysoké stability tabulek typu MyISAM.
- U tabulek InnoDB nemůžete vytvořit fulltextový index.

- Správa tabulek typu InnoDB je o něco složitější.
- Komerční licence MySQL s podporou InnoDB je dvojnásobně dražší, než verze bez ní. (tento důvod nás však samozřejmě bude zajímat pouze tehdy, pokud budeme vyvíjet projekt ke komerčnímu využití). U aplikací šířených pod hlavičkou Open Source, u projektů typu Indole a samozřejmě u obyčejných internetových stránek vystačíme se zdarma dostupnou verzí MySQL. [4]

1.5.3 Tabulky typu HEAP

Tabulky HEAP jsou zajímavé tím, že se vytváří pouze v operační paměti RAM (neukládají se tedy vůbec na pevný disk) a že používají tzv. index typu hash, který umožňuje obzvláště rychlý přístup k záznamům. Oproti tabulkám MyISAM a InnoDB mají tabulky typu HEAP řadu funkčních omezení. Mezi ta nejpodstatnější patří:

- Nesmíme používat žádné datové typy xxxTEXT nebo xxxBLOB.
- Záznamy můžeme prohlížet pouze pomocí znaků = nebo <=> (nelze tedy použít znaky <, >, <=, =>).
- Není podporován AUTO_INCREMENT.
- Indexy lze tvořit pouze pro sloupce NOT NULL.

Tabulky typu HEAP bysme pak měli používat pouze tehdy, pokud chceme maximální rychlostí spravovat malá množství dat. Vzhledem k tomu, že se tento typ tabulek ukládá pouze do paměti RAM, tabulky se po ukončení MySQL odstraní.

Je také možné použít uvnitř databáze různé typy tabulek. Můžeme tak třeba navzájem kombinovat tři tabulky typu MyISAM a dvě tabulky typu InnoDB. [4]

1.5.4 Dočasné tabulky

U většiny z výše zmíněných typů tabulek je možné vytvořit tabulky pouze dočasně. Tyto tabulky se automaticky odstraní, jakmile skript PHP ukončí připojení k MySQL. Dočasné tabulky jsou pro ostatní skripty PHP či pro jiné klienty MySQL neviditelné, takže dva skripty PHP mohou používat vedle sebe dvě dočasné tabulky se stejným jménem, aniž by došlo k nějakým konfliktům.

Dočasné tabulky nepředstavují žádný vlastní typ tabulek, ale víceméně se jedná o variantu již zmíněných typů tabulek. Dočasné tabulky občas vytváří i samo SQL, pokud potřebuje zpracovat složité dotazy obsahující příkaz SELECT.

Dočasné tabulky se neukládají do stejné složky jako jiné tabulky MySQL, ale vytváří si vlastní dočasnou složku. [4]

1.6 Integritní omezení

Integritní omezení hrají při fungování databáze a hlavně při jejím praktickém fungování zásadní roli. Pokud při vytváření tabulek nastavíme omezení špatně, nebo je nenastavíme vůbec, může se následně stát, že databáze nebude fungovat a ukládaná data nebudou odpovídat představám zadavatele. Při vytváření tabulek a definování sloupců využíváme nejvíce integritní omezení default a not null.

Omezení default určuje hodnotu, která je do sloupce zadána v případě, že uživatel při vyplňování údajů nezadá svou vlastní hodnotu. Default pomáhá předcházet situacím, kdy uživatel zapomene vyplnit některý z důležitých údajů. Jako defaultní hodnota se často zadává údaj, který je statisticky do sloupce zadáván nejčastěji, nebo který se jeví jako nejvhodnější.

Pravděpodobně nejvíce využívaným integritním omezením je not null. Při zadání parametru not null garantujeme, že nový záznam musí mít v daném sloupci povinně zadanou hodnotu. U klíčových nebo určujících parametrů je integritní omezení not null zadáván téměř automaticky. Pokud bychom například u názvu přístroje nezadali hodnotu not null znamenalo by to, že záznam o novém přístroji může být do databáze zadán, aniž by muselo být uvedeno jeho jméno.

Nesprávné ošetření integritních omezení se dá upravit i ve zdrojovém kódu webové aplikace, ale toto řešení již nemusí být plně efektivní.

1.6.1 Primární klíč

Úkolem primárního klíče je co možná nejrychlejší nalezení určitého záznamu v tabulce. Tato operace se musí provést pokaždé, když potřebujeme získat data z více tabulek – tedy velmi často!

V MySQL můžeme použít rovněž primární klíč, který se skládá z více políček tabulky. Na druhou stranu v každém případě musí mít primární klíč tyto vlastnosti:

Za prvé se musí pro pole, které obsahuje primární klíč, nastavit index (primární index), aby vyhledávání záznamů probíhalo co možná nejrychleji. Udržování indexu je tím snadnější, čím celistvější je pole pro primární klíč. Proto se jako pole pro primární klíč hodí daleko více to pole, které obsahuje celé číslo, než pole, které obsahuje řetězec s proměnnou délkou.

V případě MySQL se pro políčko s primárním klíčem zpravidla používá pole s datovým typem INT nebo BIGINT a s atributy NOT NULL AUTO_INCREMENT PRIMARY KEY. MySQL pak pro pole automaticky vytvoří čísla ve vzestupném pořadí. [4]

1.6.2 Cizí klíč

Úkolem cizího klíče je ukazovat na záznam v podřízené tabulce. Tento odkaz vzniká až v okamžiku formulování dotazu na databázi.

Při deklaraci tabulky nehraje cizí klíč žádnou významnou roli. Pro MySQL je pole označené jako cizí klíč stejné jako kterékoliv ostatní pole. Nejsou potřeba ani žádná zvláštní klíčová slova. Neměli bychme ale zapomenout na to, aby bylo pole s cizím klíčem stejného datového typu jako pole s primárním klíčem, protože jinak bude odkazování velmi pomalé. [4]

1.7 Normální formy

Normální formy představují souhrn základních pravidel, která by měla platit při navrhování databází. S normálními formami, je třeba se seznámit předtím, než začneme strukturu databáze vytvářet. Pokud by navrhovaná struktura databáze byla v rozporu s normálními formami, může to mít pro její funkčnost při dlouhodobém používání. Při plném využívání databáze a zadání například deseti tisíc záznamech o přístrojích se jeví oprava struktury tabulek a jejich atributů jako téměř neuskutečnitelná.

1.7.1 1. normální forma – 1NF

První, nejjednodušší, normální forma (1NF) říká, že všechny atributy jsou atomické již dále nedělitelné. Obecně bychom se měli snažit, aby obsahem jedné databázové položky byla právě jedna hodnota. [5]

1.7.2 2. normální forma – 2NF

Tabulka splňuje 2NF, právě když splňuje 1NF a navíc každý atribut, který není primárním klíčem a je na primárním klíči úplně závislý. To znamená, že se nesmí v řádku tabulky objevit položka, která by byla závislá jen na části primárního klíče. Z definice vyplývá, že problém 2NF se týká jenom tabulek, kde volíme za primární klíč více položek než jednu. Jinými slovy, pokud má tabulka jako primární klíč jenom jeden sloupec, pak 2NF je splněna triviálně. [5]

1.7.3 3. normální forma – 3NF

Relační tabulky splňují třetí normální formu (3NF), jestliže splňují 2NF a žádný atribut, který není primárním klíčem, není tranzitně závislý na žádném klíči. [5]

1.7.4 Boyce-Coddova normální forma – BCNF

Tabulka splňuje BCNF, jestliže pro každou netriviální funkční závislost $X \rightarrow Y$ je X superklíčem. Ve většině případů, pokud splňují tabulky 1NF až 3NF, je splněna i BCNF.

Dekompozice nezachovává závislosti v rámci relací (relace jsou nezávislé). Přejít od nižší normální formy k vyšší provádíme pomocí operace projekce. Přejít od vyšší normální formy k nižší provádíme pomocí operace spojení – JOIN. Pro vkládání, aktualizaci a odstraňování dat je vhodné mít tabulky ve třetí normální formě. [5]

1.8 Vztahy mezi tabulkami

Vztahy mezi tabulkami definují v podstatě možnosti kolik záznamů z první tabulky může být navázáno na záznamy ze druhé tabulky. V případě naší databáze určují vztahy mezi tabulkami tool a room, na kolik místností může být evidován jeden přístroj. Jelikož závisí na pořadí tak druhou významem je kolik přístrojů může být evidováno v jedné místnosti.

1.8.1 Vztah jedna k jedné (1 : 1, one-to-one)

V tomto vztahu první entitě, tedy záznamu v databázové tabulce, odpovídá maximálně jedna druhá entita, záznam z jiné databázové tabulky. Tedy každý řádek primární tabulky je možné svázat právě s jedním řádkem sekundární tabulky. [1]

1.8.2 Vztah jeden k více (1 : N, one-to-many)

Pro vztah jeden k více je důležitý směr neboli pořadí tabulek. Každý řádek primární tabulky je možné svázat s jedním nebo více řádky sekundární tabulky. [1]

1.8.3 Vztah jeden k více (1 : N, one-to-many)

Více řádků primární tabulky může být svázáno s více řádky sekundární tabulky. Protože většina databázových systémů nedokáže přímo pracovat se vztahy typu N : M. V praxi se používá dekompozice, což znamená, že tento vztah implementujeme pomocí spojovací tabulky. To znamená, že vztah N : M, můžeme rozložit na dva vztahy typu N : 1. [1]

1.9 Datové typy

Při vytváření tabulek je nutné zadat vždy datový typ, který definuje jaké hodnoty je možné při vkládání nového záznamu zadávat. Níže popisují nejčastěji využívané datové typy.

1.9.1 Celá čísla

Standardně se pro všechna celá čísla v MySQL používá znaménko. Do sloupce TINYINT můžeme například zadat čísla z intervalu -127 až +128. Pokud chceme používat výlučně kladná čísla, musíme u definice sloupce zadat atribut UNSIGNED. [4]

Tab. 1. Datové typy pro celá čísla

Klíčové slovo v MySQL	Význam
TINYINT (m)	8-Bit-Integer (1 Byte); nepovinný parametr m udává šířku sloupce ve výsledcích při použití příkazu SELECT (maximum display width). Nijak neovlivňuje povolený rozsah čísl.
SMALLINT (m)	16-Bit-Integer (2 Byte)
MEDIUMINT (m)	24-Bit-Integer (3 Byte)
INT (m), INTEGER (m)	32-Bit-Integer (4 Byte)

BIGINT (m)	64-Bit-Integer (8 Byte)
SERIÁL	Synonymum pro BIGINT AUTO_INCREMENT NOT NULL PRIMARY KEY

1.9.2 Čísla s pevnou a s plovoucí desetinnou čárkou

Rozdíl mezi číslem s pevnou a s pohyblivou desetinnou čárkou spočívá v jeho vnitřní interpretaci. Čísla s plovoucí desetinnou čárkou se v MySQL ukládají v binárním formátu, zatímco čísla s pevnou desetinnou čárkou se ukládají jako řetězce.

U čísel s pevnou desetinnou čárkou jsou naprosto vyloučeny chyby způsobené zaokrouhlováním, což je výhoda zejména při ukládání finančních obnosů. Tuto výhodu vyvažují následující nevýhody: požadavky na paměť jsou daleko vyšší a také je omezen obor čísel, která se dají použít. Také zpracování čísel s pevnou desetinnou čárkou je pomalejší. [4]

Tab. 2. Datové typy pro čísla s pevnou a s plovoucí desetinnou čárkou

Klíčové slovo v MySQL	Význam
FLOAT (m, d)	Číslo s plovoucí desetinnou čárkou s přesností na 8 desetinných míst (4 bajty); nepovinné parametry m a d udávají počet cifer před a za desetinnou čárkou ve výsledcích, kdy se použilo v dotazu klíčové slovo SELECT; nijak neovlivňuje povolený rozsah číslic
DOUBLE (m, d)	Číslo s plovoucí desetinnou čárkou s přesností na 16 desetinných míst (8 bajtů).
REAL (m, d)	Synonymum pro DOUBLE
DECIMAL (p, s)	Číslo s pevnou desetinnou čárkou. Parametr p udává celkový počet cifer (maximálně 64), parametr s počet míst za desetinnou čárkou (maximálně 30); Jako standard je nastaveno DECIMAL (10,0).
NUMERIC, DEC	Synonymum pro DECIMAL

1.9.3 Datum a čas

Z níže uvedených formátů data a času se nejčastěji používá formát DATETIME. Pole s tímto formátem se hodí k ukládání libovolných časových údajů. Datový typ TIMESTAMP hraje svoji roli jinde: Sloupce, v němž použijeme tento formát, spravuje MySQL automaticky. Při každé změně záznamu automaticky ukládá MySQL do tohoto sloupce čas změny (pokud při ukládání sami nezadáme do sloupce TIMESTAMP nějaký

jiný časový údaj). V každé tabulce funguje tato automatická aktualizace časových údajů pouze pro první sloupec s nastaveným datovým typem `TIMESTAMP`. (Pravdou je, že u většiny tabulek nemá smysl nastavovat více než jeden sloupec `TIMESTAMP`.)

Sloupce `TIMESTAMP` má smysl používat pouze tehdy, pokud si přejeme automaticky zaznamenávat časy posledních změn. Nejsou tedy v žádném případě určeny k ukládání libovolných časových údajů. K tomuto účelu se daleko více hodí sloupec s datovým typem `DATETIME`.

Přejeme-li si, aby se při změně záznamu hodnota `TIMESTAMP` nezměnila, pak musíme v příkazu `update` použít příkaz `SET ts=st` (kde `ts` je název sloupce s datovým typem `TIMESTAMP`). [4]

Tab. 3. Datové typy pro datum a čas

Klíčové slovo v MySQL	Význam
<code>DATE</code>	Datum ve formátu '2006-12-31' (3 bajty). Hodnoty z intervalu od 1000-01-01 do 9999-12-31.
<code>TIME</code>	Čas ve formátu '23:59:59' (3 bajty). Hodnoty z intervalu +/- 838:59:59.
<code>DATETIME</code>	Kombinace formátů <code>DATE</code> a <code>TIME</code> ve formátu '2006-12-31 23:59:59' (8 bajtů).
<code>YEAR</code>	Letopočet 1900-2155 (1 bajt).
<code>TIMESTAMP</code>	Datum a čas ve formátu '2006-12-31 23:59:59' pro časové údaje z intervalu 1970 až 2038 (4 bajty). Obsah sloupce <code>TIMESTAMP</code> se při každé změně v MySQL automaticky aktualizuje

1.9.4 Řetězce znaků

Pro ukládání krátkých řetězců se zpravidla používají sloupce `VARCHAR (n)`, kde písmenko `n` určuje maximální počet znaků v řetězci (255 znaků u MySQL 4.1, 65535 znaků u MySQL 5.0). Pro ukládání delších řetězců jsou určeny datové typy `MEDIUM` a `LONGTEXT`.

Podobně jako datový typ `VARCHAR` je na tom datový typ `CHAR(n)`, v tomto případě je zde nezávisle na délce řetězce rezervováno pouze `n` znaků. Je to jasné plýtvání místem, zvyšují se tím nároky na kapacitu disku pro umístění tabulky, a proto jsou sloupce

s datovým typem CHAR jen zřídka účinnější než sloupce s datovým typem VARCHAR. To všechno jsou důvody k tomu, že se v praxi datový typ CHAR používá čím dál řidčeji. [4]

Tab. 4. Datové typy pro řetězce znaků

Klíčové slovo v MySQL	Význam
CHAR (n)	Znakové řetězce s předem zadanou délkou (maximálně 255 znaků)
VARCHAR (n)	Znakové řetězce s předem zadanou délkou (maximálně 255 znaků u MySQL 4.1, popřípadě 65535 znaků ve verzi MySQL 5.0)
TINYTEXT	Znakové řetězce s proměnnou délkou (maximálně 255 znaků)
TEXT	Znakové řetězce s proměnnou délkou (maximálně $(2^{16})-1$ znaků)
MEDIUMTEXT	Znakové řetězce s proměnnou délkou (maximálně $(2^{24})-1$ znaků)
LONGTEXT	Znakové řetězce s proměnnou délkou (maximálně $(2^{32})-1$ znaků)

1.9.5 Další datové typy

Mezi zvláštnosti MySQL patří datové typy ENUM a SET. Umožňují zvláště efektivní možnost správy řetězců čísel, popřípadě jejich kombinaci na serveru MySQL.

Pomocí ENUM můžeme spravovat seznam až 65 535 řetězců znaků, kterým jsou po řadě přiřazena čísla. V poli se pak může vybrat libovolný z těchto řetězců.

Datový typ SET využívá podobné myšlenky, je zde však povolena libovolná kombinace. Uvnitř systému jsou řetězcům přiřazena čísla odpovídající mocnině dvou (1, 2, 4, 8 atd.), takže je možnost vytvořit libovolnou kombinaci vycházející z jednotlivých bitů. Jsou zde ovšem daleko vyšší nároky na paměť (každý řetězec obsadí jeden bit). Maximálně lze kombinovat 64 řetězců. (V paměti se pak obsadí 8 bajtů). [4]

2 PROPOJENÍ MYSQL A SKRIPTOVACÍCH JAZYKŮ

2.1 Přístup k databázím z internetu

Internet je v dnešní době místem neomezených možností. Snad jediným heslem, které platí stále více a více je, že dynamika je naprostá nutnost.

Hlavním úkolem internetové aplikace při propojení k databázi, je nabídnou uživatelům co nejpříjemnější a nejefektivnější prostředí, ve kterém budou moci s databází pracovat. Internetové aplikace mohou do značné míry také pomoci při zadávání nových záznamů do databází. Ošetření správně zadaných nových záznamů umožňují využití například javascriptu.

K tvorbě webových aplikací se v dnešní době využívá více druhů technologií (skriptovacích jazyků) avšak asi nejvíce využívanější je skriptovací jazyk PHP.

2.2 PHP

Zkratka PHP se původně interpretovala jako Personal Home Page (tedy osobní domovská stránka) a označovala jazyk používaný především k realizaci formulářů používaných na webových stránkách. Plná verze je označována jako PHP/FI (tedy Personal Home Page Forms Interpreter). V průběhu času se ustálil pojem Hypertext Procesor.

Nastupuje nová verze PHP 5. Pátá verze tohoto oblíbeného skriptovacího jazyka je nepochybně známkou vyzrálosti a stability produktu a v neposlední řadě (i když ne přímo) rovněž známkou jeho úspěšnosti. Nová verze je přizpůsobena současným trendům a technologiím. Kdybychom měli vybrat nejvýznamnější novinky a rozšíření, určitě bychom zmínili podporu objektivě orientovaného programování a jazyka XML. Ve svých prvních verzích oslovil jazyk PHP příznivce hlavně jednoduchostí a množstvím zabudovaných funkcí, jejichž počet se s každou další verzí utěšeně rozšiřoval. V nejnovějších verzích byl do kvantity funkcí vnesen určitý řád – především kvůli podpoře objektivě orientovaného programování, jež umožňuje jednotlivé funkce uspořádat do tříd. [2]

2.3 Proč používat PHP a jak funguje?

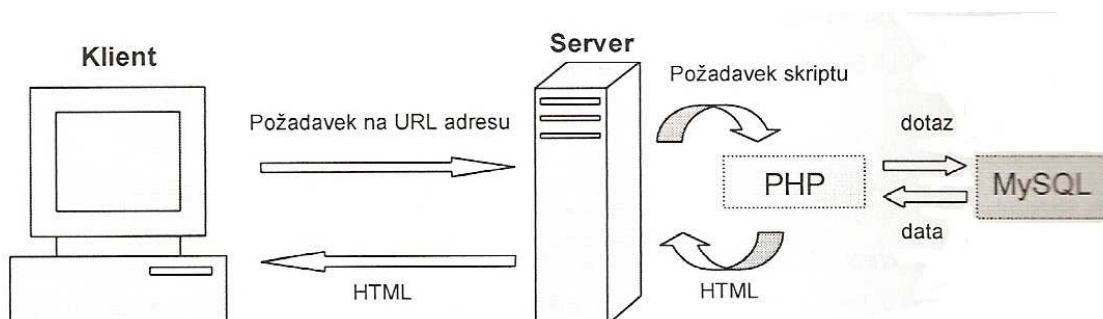
Jazyk PHP je jedním z nejlepších kompromisů mezi tím, jak snadno jej můžou používat začátečníci a jak širokou škálu možností využití nabízí zkušeným programátorům. K přednostem technologie PHP patří výkon, stabilita, přenositelnost, těsná integrace s většinou databázových systémů a téměř neomezené možnosti rozšiřování – to vše prakticky zdarma (PHP je dodáván s veřejným zdrojovým kódem).

PHP je jazykem určeným pro skriptování na straně serveru- tzn. kód, který je napsán v jazyce PHP, je uložen na počítači nabízejícím webové stránky návštěvníkům.

Navštíví-li uživatel webovou prezentaci napsanou v jazyce PHP a zpracuje ji podle zadaných parametrů, server pak data odešle webovému prohlížeči, který pak s daty pracuje jako se standardními stránkami HTML. Tím se liší dynamická prezentace od statické, v níž na požádání server pouze odešle data HTML do webového prohlížeče. [5]

2.4 Propojení PHP a MySQL

Začleněním databáze do webové aplikace lze určitá data generovaná PHP načítat z databáze MySQL. Na server lze pohlížet ze dvou úhlů. Server je vybraný stroj, na kterém je nainstalovaný databázový systém, na jehož discích jsou uložena naše data. Na druhou stranu je server proces (program) který běží na zvoleném počítači a který obsahuje jednotlivé požadavky od klientů. Klientem pak může být konkrétní databázová aplikace, nebo také řádkový terminál, ve kterém můžeme SQL dotazy zadávat přímo. [5]



Obr. 1. Komunikace serveru PHP a MySQL

Pro připojení k serveru MySQL se používá příkaz `mysql_connect`, kterému předáme tři údaje: název počítače (`hostname`) – serveru MySQL, uživatelské jméno a nakonec heslo pro přihlášení k MySQL. Pokud MySQL běží na stejném počítači jako skript PHP, použije se jako název počítače `localhost`.

Jako nepovinný čtvrtý parametr můžeme zadat, zda se při vícenásobném spuštění příkazu `mysql_connect` se stejnými přihlašovacími parametry má vrátet odkaz na již existující připojení (`false`, standardní nastavení), nebo zda se má vytvořit připojení nové (`true`). Způsob zápisu `mysql_connect($host, $name, $pw, true)` je tedy nutno použít, pokud potřebujeme vytvořit několik samostatných připojení.

Pátý parametr, rovněž nepovinný, nastavuje příznak (flag) klienta. Lze u něj použít například konstantu `MYSQL_CLIENT_COMPRESS` (pokud se mají vyměňovat data v komprimované podobě).

Podrobné informace o aktuálním připojení mohou poskytnout funkce `mysql_get_client_info`, `mysql_host_info`, `mysql_proto_info`, `mysql_server_info`, `mysql_client_encoding`, `mysql_stat` a `mysql_thread_id`. Jedná se o funkce, které zjistí verzi knihovny klienta, způsob připojení k serveru, verzi protokolu, verzi serveru, nastavenou znakovou sadu, aktuální stav serveru a číslo vlákna v procesu serveru MySQL. [4]

```
<?php
$link = mysql_connect("127.0.0.1", "admin", "admin")
        or die("Nelze se připojit: " . mysql_error());
mysql_query("SET CHARACTER SET UTF8") or die(mysql_error());
mysql_query("SET collation_connection=utf8_czech_ci") or die(mysql_error());
mysql_select_db("tools_database") or die("<BR>Databazé neexistuje<BR>");
mysql_close();
?>
```

Obr. 2. Zdrojový kód v PHP, který realizuje připojení k databázi MySQL

3 ZABEZPEČENÍ

Ve světě informačních technologií jsou způsoby zabezpečení dat stále silnějším tématem jak pro vývojáře a programátory tak pro koncové uživatele. Zabezpečení údajů o uživatelských databázích je i vzhledem k legislativě povinností všech databázových systémů.

3.1 Zabezpečení webové aplikace

Při práci s webovou aplikací je potřeba zvážit následující aspekty zabezpečení:

- Způsob uložení dat na serveru – je nutné chránit především samotnou databázi MySQL, jednak nastavením vhodných uživatelských oprávnění, jednak ochranou adresáře, ve kterém jsou ukládána a používána data.
- Ochrana citlivých informací – vytvořením neveřejné části aplikace, ke které má přístup pouze správce.
- Zabezpečení dat během přenosu – je nutné použít zabezpečení transakcí. K tomu se používá na serveru protokol SSL (Secure Socket Layer), ke kterému se připájí platný certifikát serveru. [5]

3.2 Zabezpečení databáze MySQL

3.2.1 Uživatelská oprávnění

Webová aplikace by měla s ohledem na zabezpečení používat specifického uživatele databáze MySQL a server MySQL by neměla používat jako uživatele ROOT, ani jako anonymní uživatel. Proto je potřebné vytvořit na serveru MySQL nového uživatele, jemuž nastavíme heslo a přístup pouze na vybranou databázi. Tento uživatel by měl mít oprávnění pouze ke spuštění přidávacích, aktualizčních a výběrových dotazů. Tím se minimalizuje riziko zneužití přístupových informací. [5]

3.2.2 Tvorba uživatelů a jejich práv

System nastavení uživatelských práv v databázovém systému MySQL byl navržen takovým způsobem, aby zajistil vhodná oprávnění k vykonávání určitých příkazů ve vybraných databázích.

Každý uživatel v systému MySQL může mít specifická oprávnění ve vybraných databázích, podle toho z jakého počítače se k databázi připojuje. Uživatel ROOT má nejvyšší pravomoci – superuživatel a používá se k tvorbě dalších uživatelů. Informace o uživateli a jejich oprávněních jsou uloženy v databázi MySQL v tabulkách DB, HOST, USER, FUNC, TABLES_PRIV, COLUMNS_PRIV. Uživatele a jeho oprávnění je možné definovat různými způsoby. Jako oprávnění lze nastavit libovolné privilegium z následující tabulky. [5]

Tab. 5. Uživatelská práva v MySQL

Oprávnění	Umožňuje
SELECT	Prohlížení a výběr z tabulek
INSERT	Přidávání dat do tabulek
UPDATE	Úprava existujících dat v tabulkách
DELETE	Odstraňování existujících dat z tabulek
INDEX	Vytvářet a odstraňovat indexy
ALTER	Upravovat strukturu tabulek
CREATE	Vytvářet nové tabulky nebo databáze
DROP	Odstraňovat tabulky nebo databáze
RELOAD	Opětovně načíst tabulku oprávnění
SHUTDOWN	Zastavit server MySQL
PROCESS	Prohlížet, zastavit existující procesy v MySQL
FILE	Importovat data z textových souborů
GRANT	Vytvářet nové uživatele
REVOKE	Odebrat uživatelům oprávnění

Abychom mohli instalaci MySQL považovat za bezpečnou, musíme splňovat následující podmínky:

- Nesmí existovat žádný uživatel, který nemá heslo.
- Nesmí existovat žádný uživatel bez jména.
- Nesmí existovat žádný uživatel, který má ve sloupci Host znak %.
- K databázím má přístup co možná nejméně uživatelů (v ideálním případě pouze uživatel root), který jako jediný má ničím neomezená oprávnění. [4]

II. PRAKTICKÁ ČÁST

4 ANALÝZA PROBLEMATIKY

Prvotní myšlenkou při zadávání bakalářské práce bylo vytvořit databázový systém, který by mohla Fakulta aplikované informatiky využít pro evidenci záznamů o nakoupených přístrojích. Cílem bylo zefektivnit evidenci, nákup a využívání přístrojů, které jsou na Fakultě aplikované informatiky k dispozici jak pedagogům, tak i studentům.

I když se v průběhu konzultací přišlo na skutečnost, že evidence přístrojů již na Fakultě aplikované informatiky existuje, od původního záměru jsme zcela neupustili a rozhodli se zaměření práce neměnit. I když momentální přímá potřeba pro nový databázový systém na evidenci přístrojů není, je možné bakalářskou práci v budoucnu rozšiřovat a upravovat podle aktuálních potřeb tak, aby byla reálně využívána.

4.1 Specifika pro přístroje

Základem pro vytváření databáze je zjištění specifikací, která budou definovat jednotlivé přístroje. V našem databázovém systému bude tabulka pro evidenci přístrojů pravděpodobně tou nejdůležitější. Každý přístroj je jednoznačně určen následujícími hodnotami:

- název přístroje,
- pořizovací cena přístroje,
- rok výroby přístroje,
- popis přístroje,
- ústav, na který byl přístroj pořízen,
- místnost, ve které se přístroj nachází.

4.2 Jednoznačná lokalizace přístroje

Fakulta aplikované informatiky je z hlediska organizačního členění rozdělena do několika ústavů. Každý ústav má své vlastní přístroje a při vkládání záznamu o novém přístroji je nutné vždy přiřadit právě jeden ústav z nabídky.

Ústavy na Fakultě aplikované informatiky mohou být rozděleny na několika lokalitách a ne všechny místnosti jednoho ústavu se musejí nacházet v jedné konkrétní budově. Poloha

přístroje tedy není jednoznačně určena pouze přiřazeným ústavem, ale je nutné ji specifikovat přímo místností, do které byl přístroj pořízen. Jak uvidíme dále, ústav a budova jsou v databázi reprezentovány vlastními tabulkami a při vkládání záznamu o novém přístroji je nutnost zadat ústav a místnost ošetřena jak na straně MySQL, tak i ve webové aplikaci vytvoření skriptovacím jazykem PHP.

4.3 Možnost vypůjčení přístroje

Možnost vypůjčení přístroje není pro samotnou evidenci přístrojů přímou nutností. Databáze pro evidenci přístrojů umožňuje uživatelům si přístroj pouze vypůjčit, pokud je již k dispozici a nevzniká tak nutnost nákupu dalšího přístroje, který by se v budoucnu mohl ukázat jako nadbytečný.

Vypůjčení přístroje lze definovat několika možnými způsoby. Základním údajem pro vypůjčení je čas. Je důležité, aby byla výpůjčka pevně časově ohraničena začátkem a koncem. Druhým, neméně důležitým aspektem je osoba, která si přístroj půjčuje. Osoba je pevně definována jménem a příjmením, ale je možné dodat do databáze další určující údaje, které pomohou při efektivnější práci s databázovým systémem. Jako doplňující údaje se nabízejí nejvíce údaje kontaktní jako například, telefonní číslo anebo emailová adresa. Pomocí nich bude možné osobu rychle vyhledat.

Právě možnost vypůjčovat si dostupné přístroje z praktického pohledu vede k využívání databáze a přispívá k efektivnímu vedení organizace. Minimalizace nákladů na nákup nových přístrojů je z dlouhodobého hlediska pro velké organizace nutností a může vést k maximální efektivitě nakládání s finančními prostředky.

5 DATABÁZE PRO EVIDENCI PŘÍSTROJŮ

Databáze je postavena na systému řízení báze dat MYSQL. Samotná databáze je ovládaná pomocí rozhraní PhpMyAdmin. Pro instalaci byl využit balíček EasyPHP 5.3.6.0, který má tu výhodu, že obsahuje Apache, PHP, MySQL a PHPMyAdmin se snadnou instalací ve Windows.



The screenshot displays system information organized into three sections:

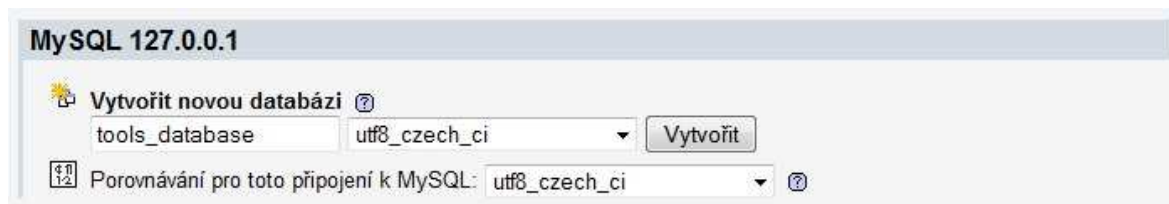
- MySQL**
 - Server: 127.0.0.1 via TCP/IP
 - Verze MySQL: 5.5.10-log
 - Verze protokolu: 10
 - Uživatel: root@localhost
 - Znaková sada v MySQL: UTF-8 Unicode (utf8)
- Webservice**
 - Apache/2.2.17 (Win32) PHP/5.3.6
 - Verze MySQL klienta: mysqlnd 5.0.8-dev - 20102224 - \$Revision: 308673 \$
 - Rozšíření PHP: mysql
- phpMyAdmin**
 - Informace o verzi: 3.3.9.2
 - [Dokumentace](#)
 - [Wiki](#)
 - [Oficiální stránka phpMyAdmina](#)
 - [\[ChangeLog\]](#) [\[Git\]](#) [\[Lists\]](#)

The phpMyAdmin logo is visible in the bottom right corner of the screenshot.

Obr. 3. Systémové informace z PhpMyAdmin

5.1 Vytvoření databáze

Při vytváření databáze je nutné zadat jméno databáze vybrat dva údaje. Prvním je typ, jakým budou údaje kódovány, a druhým je typ kódování, který bude zvolen při porovnávání dat. V našem případě jsou obě dvě hodnoty zadány jako utf8_czech_ci.



Obr. 4. Vytvoření databáze

5.2 Vytváření tabulek v databázi

Program PhpMyAdmin zobrazí po zadání názvu tabulky a počtu řádků rozsáhlý formulář, pro nějž do každého sloupce tabulky zadáte následující data:

- Sloupec: Název pole.
- Typ: Datový typ, například INT nebo VARCHAR.
- Délka/množina: tato položka je u většiny datových typů nepovinná.
- Porovnání: v tomto poli se zadává požadovaná znaková sada a požadovaný způsob řazení dat.
- Vlastnosti: pokud si u celočíselných typů přejete zobrazovat čísla bez znaménka, můžete zde nastavit položku unsigned.
- Nulový: tento parametr udává, zda smí sloupce obsahovat hodnotu NULL.
- Výchozí: zde můžete zadat výchozí hodnotu pro parametr.
- Extra: zde je k dispozici pouze jediný atribut a to AUTO_INCREMENT. Ten má smysl použít pouze pro políčko obsahující primární klíč.
- Nastavení indexů: můžete vybrat hodnotu indexu z možností primary, unique, index a full text. [4]

5.2.1 Vytvoření tabulky tool

Tabulka tool je pravděpodobně tou nejdůležitější v databázi. Obsahuje 7 sloupců, které jsou všechny při vkládání záznamu o novém přístroji povinné. Ani jeden tedy nemá nastavenou možnost, že může obsahovat po vložení znak NULL, což je vidět na obrázku Obr. 5.. Jako primární klíč je zvoleno toolID a rovněž je aktivní možnost AUTO_INCREMENT. V praxi to pro nás bude znamenat, že při vložení nového záznamu bude hodnota toolID o jedničku vyšší.

Sloupec	Typ	Porovnávání	Vlastnosti	Nulový	Výchozí	Extra	Akce
<input type="checkbox"/> toolID	int(10)			Ne	Žádná	AUTO_INCREMENT	<input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>
<input type="checkbox"/> tool_name	varchar(30)	utf8_czech_ci		Ne	Žádná		<input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>
<input type="checkbox"/> tool_price	int(10)			Ne	Žádná		<input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>
<input type="checkbox"/> tool_year	int(4)			Ne	Žádná		<input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>
<input type="checkbox"/> tool_specification	text	utf8_czech_ci		Ne	Žádná		<input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>
<input type="checkbox"/> instituteID	int(10)			Ne	Žádná		<input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>
<input type="checkbox"/> roomID	int(10)			Ne	Žádná		<input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>

Obr. 5. Struktura tabulky tool

Důležité jsou sloupce instituteID a roomID. U obou sloupců je zvolný index a obě jako cizí klíče odkazují na další tabulky. Tabulka institute obsahuje všechny zadané ústavy a je pomocí relace 1:N připojena k tabulce tool. Tabulka room obsahuje údaje o místnostech a je stejně jako tabulka institute, pomocí vazby 1:N připojena k tabulce tool.

Sloupce tool_name a tool_specification mají sadu znaků nastavenou na VARCHAR a TEXT. U obou sloupců bylo pro porovnávání znakových sad nastaveno utf8_czech_ci, které podporuje porovnávání znaků specifických pro český jazyk.

	toolID	tool_name	tool_price	tool_year	tool_specification	instituteID	roomID
<input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>	1	HYUNDAI WSC 2032B	2499	2011	Součástí balení je senzor na měření teploty Způsob...	4	8
<input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>	2	multimetr	200	2005	Digitální multimetr s měřením teploty FK 8400	4	4
<input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>	3	Zkoušečka UTP	1925	2010	Zkoušečka UTP a koaxiálních kabelů s měřením délky...	5	3
<input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>	4	SanDisk Micro SDHC	159	2010	Spolehlivá paměťová karta s kapacitou 4 GB	1	5
<input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>	5	CISCO SD205T-EU	697	2009	Konektivita: Ethernet 10/100Base-T(X)	2	8
<input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>	6	IBM x3200M3 Tower	25957	2008	Procesor: Model: Intel Xeon X3440 Počet jader: 4...	7	9
<input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>	7	BenQ W1000+	21479	2009	DLP projektor nat. Full HD 1080p (1920x1080), max....	7	2
<input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>	8	Xerox Documate 252	20422	2004	Rychlost skenování (A4, 150 dpi, barevné): Jednos...	3	7
<input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>	9	Microsoft Visual Studio 2010	22586	2010	Tvorba velmi výkonných aplikací pro Windows 7 nati...	1	10
<input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>	10	Evolve detektor kouře	458	2007	Citlivost detekce: odpovídá standardu UL217	5	9

Obr. 6. Záznamy v tabulce tool

Tabulka tool je pro praktickou ukázkou naplněna deseti hodnotami, jak je vidět z obrázku Obr. 6., tak všechny položky mají vyplněny všechny sloupce a je tedy splněna podmínka, že žádný sloupec nemá dovoleno obsahovat hodnoty NULL. Pokud neznáme bližší specifikaci přístroje a nevím jakou hodnotu při vkládání nového záznamu uvést, doporučovaná standardní hodnota je neznám/ neznámá.

5.2.2 Vytvoření tabulky institute

Tabulka institute je určena pro uchovávání záznamů o ústavech na Fakultě aplikované informatiky. Jako primární klíč je zvolen sloupec instituteID a rovněž je u něho aktivní možnost AUTO_INCREMENT, která zaručuje automatické zvyšování instituteID pro každý další záznam v tabulce.

Sloupec	Typ	Porovnávání	Vlastnosti	Nulový	Výchozí	Extra	Akce
<input type="checkbox"/> instituteID	int(10)			Ne	Žádná	AUTO_INCREMENT	
<input type="checkbox"/> institute_name	varchar(60)	utf8_czech_ci		Ne	Žádná		

Obr. 7. Struktura tabulky institute

Sloupec institute_name je omezen na 60 znaků datového typu VARCHAR a pro porovnávání je nastaveno utf8_czech_ci. Názvy ústavů tedy můžeme zadávat jak v anglickém tak českém jazyce.

I když by se pro ústavy daly najít některé další atributy, tak pro naše účely plně stačí název ústavu. Naopak by se při povinném zadávání parametrů mohlo stát, že by zadané údaje byly nadbytečné a nevyužívané což by vedlo ke zbytečnému zabírání místa na disku.

Jak je vidět na obrázku Obr. 8., jsou v tabulce vloženy záznamy o 7 aktuálních ústavech do kterých je rozčleněna Fakulta aplikované informatiky. Na obrázku Obr. 6. jsou viditelné hodnoty institute ID v tabulce tool, které jsou pro jednotlivé přístroje spojeny se záznamy v tabulce institute. Vhodným využitím příkazu SELECT se nám tak jako výstup může zobrazit ne číslo, ale název institutu, na který je přístroj zaznamenán.

	instituteID	institute_name
<input type="checkbox"/>	1	Ústav informatiky a umělé inteligence
<input type="checkbox"/>	2	Ústav počítačových a komunikačních systémů
<input type="checkbox"/>	3	Ústav automatizace a řídicí techniky
<input type="checkbox"/>	4	Ústav elektroniky a měření
<input type="checkbox"/>	5	Ústav bezpečnostního inženýrství
<input type="checkbox"/>	6	Ústav matematiky
<input type="checkbox"/>	7	Ústav řízení procesů

Obr. 8. Záznamy v tabulce institute

5.2.3 Vytvoření tabulky room

Tabulka room je velice podobná tabulce institute a je určena pro uchovávání záznamů o místnostech na Fakultě aplikované informatiky. Jako primární klíč je zvolen roomID, které stejně jako v předchozích případech díky aktivnímu AUTO_INCREMENT zaručuje jedinečnou hodnotu pro každý záznam.






























Sloupec	Typ	Porovnávání	Vlastnosti	Nulový	Výchozí	Extra	Akce
roomID	int(10)			Ne	Žádná	AUTO_INCREMENT	<input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>
room_name	varchar(30)	utf8_czech_ci		Ne	Žádná		<input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>




Obr. 9. struktura tabulky room

Na hodnotu roomID se z tabulky tool odkazuje cizí klíč roomID a je tedy nutné aby měli oba sloupce stejné vlastnosti. Po porovnání obrázků Obr. 5. a Obr. 9. je patrné, že oba obsahují datový typ INT(10) a pro odkazování je tak zaručena maximální efektivita a výkonnost.

Tabulka je naplněna záznamy o deseti reálných místnostech. Skrze webovou aplikaci se dají vkládat další záznamy, ale pro zachování jednotného způsobu zapisování je nutné zadávat názvy místností ve tvaru UXX-YYY. Toto omezení zaručí jednotnost zapisovaných hodnot a vede k lepší organizaci dat.

Tabulka person je naplněna 13 záznamy o zodpovědných osobách, jak můžeme vidět na obrázku Obr. 14.

  	personID	person_name	person_surname	person_phone	person_email
<input type="checkbox"/>  	1	Milan	Adámek	576035251	adamek@fai.utb.cz
<input type="checkbox"/>  	2	Ján	Ivanka	576035247	ivanka@fai.utb.cz
<input type="checkbox"/>  	3	Bronislav	Chramcov	576035186	chramcov@fai.utb.cz
<input type="checkbox"/>  	4	Roman	Šenkeřík	576035189	senkerik@fai.utb.cz
<input type="checkbox"/>  	5	Zdenka	Prokopová	576035011	prokopova@fai.utb.cz
<input type="checkbox"/>  	6	Martin	Sysel	576035180	sysel@fai.utb.cz
<input type="checkbox"/>  	7	Jana	Řezníčková	576035107	reznickova@fai.utb.cz
<input type="checkbox"/>  	8	Tomáš	Dulík	576035187	dulik@fai.utb.cz
<input type="checkbox"/>  	9	Pavel	Pokorný	576035181	pokorny@fai.utb.cz
<input type="checkbox"/>  	10	Milan	Navrátil	576035283	navratil@fai.utb.cz
<input type="checkbox"/>  	11	Libor	Pekař	576035261	pekar@fai.utb.cz
<input type="checkbox"/>  	12	Michal	Heczko	576033008	heczko@fai.utb.cz
<input type="checkbox"/>  	13	Radek	Vala	576035133	vala@fai.utb.cz

↑ Zaškrtnout vše / Odškrtnout vše Zaškrtnuté:   

Obr. 14. Záznamy v tabulce person

5.3 Struktura databáze tools_database

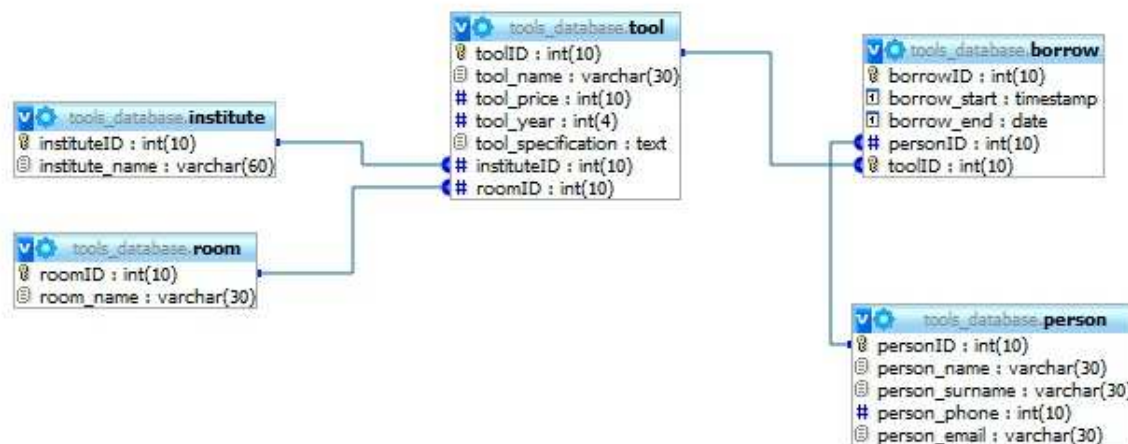
Nejvhodnějším způsobem jak představit strukturu databáze je znázornění ERA diagramu.

ERA diagram má následující pravidla:

- Zobrazujeme pouze data a jejich vztahy.
- Každý atribut vždy zobrazujeme pouze jednou.
- Zobrazujeme pouze trvalé objekty.
- Entity by měly být normalizované.
- Zobrazujeme pouze nezbytně nutné vztahy.
- Nezobrazujeme redundantní vztahy a odvozené vztahy.
- Nezobrazujeme procesy.

5.3.1 ERA diagram databáze tools_database

ERA diagram je možné zobrazit v PhpMyAdmin po výběru databáze a kliknutí na návrhář. V PhpMyAdminu je možné upravovat relace mezi tabulkami a přímo v zobrazení ERA diagramu je také možné vytvořit novou nebo upravovat již existující tabulky.



Obr. 15. ERA diagram databáze tools_database

Jak vidíme na obrázku Obr. 15, centrální tabulkou v digramu je tabulka tool, která má sama dva cizí klíče, odkazující na tabulky institute a room. Vztahy mezi tabulkami institute, room a mezi tabulkou tool jsou 1:N. Logicky tak vyplývá, že na jednom ústavu, nebo na jedné místnosti může být evidováno více přístrojů, ale naopak jeden přístroj může být vždy spojen pouze s jednou místností nebo ústavem.

V tabulce borrow jsou dva cizí klíče personID, který odkazuje na tabulku person, a toolID, který odkazuje na tabulku tool. Sloupec toolID musí mít unikátní hodnotu, jelikož logicky je možné si jeden přístroj z ústavu vypůjčit pouze jednou a další výpůjčka může následovat, až když ta první skončí a přístroj je vrácen. Vztah mezi tabulkami tool a borrow je tedy 1:1, sloupec personID však unikátní hodnotu mít nemusí, jelikož jedna osoba může mít půjčených více přístrojů. Vztah mezi tabulkami person a borrow je tedy definován jako 1:N.

Vzhledem k integritě dat není možné mazat záznamy, které jsou provázány a tudíž využívány dalšími tabulkami. Ve webové aplikaci není možné smazat osobu s ID=8 (Tomáše Dulíka), jelikož má momentálně vypůjčen přístroj s názvem Microsoft Visual Studio 2010.

5.3.2 Primární klíče v databázi tools_database

Ve vytvořené databázi tools_database je pět tabulek. Každá tabulka musí být definována nejméně jedním primárním klíčem. V našem případě toto pravidlo platí a každá tabulka je definována právě jedním primárním klíčem. Pro maximální efektivitu při zpracovávání dat je za primární klíč zvolen vždy sloupec, který záznam určuje nejjednoznačněji. V našem případě nejjednoznačnějším údajem ve všech tabulkách je ID záznamu.

Tab. 6. Primární klíče v databázi tools_database

Primární klíč	Datový typ	AUTO_INCREMENT
tool.toolID	INT(10)	ano
institute.instituteID	INT(10)	ano
room.roomID	INT(10)	ano
borrow.borrowID	INT(10)	ano
person.personID	INT(10)	ano

Jak je vidět v tabulce Tab. 6., tak je dodržena základní rada pro unifikovaný systém názvů sloupců v tabulkách. Primární klíče vždy obsahují název tabulky a za ním přiřazené ID. Díky tomuto systému unifikovaného značení jde v tabulce ihned jednoznačně určit, který sloupec je zvolen za primární klíč.

Nastavení automatického číslování a hodnoty NOT NULL, zaručují plnění datové jedinečnosti. Z tabulky Tab. 6. je patrné že všechny primární klíče mají shodně zvolený datový typ a to INT (10).

5.3.3 Cizí klíče v databázi tools_database

Databáze tools_database obsahuje tabulky pouze typu InnoDB. Všechny relace mezi tabulkami jsou tedy chráněny pravidly referenční integrity. Při každé změně v záznamech databáze kontroluje, jestli není měněný záznam využíván jinou tabulkou a jestli změna její obsah neovlivní.

Nejvíce se pravidla referenční integrity projevují při mazání záznamů. Pokud například zadáme požadavek na smazání ústavu nebo místnosti, na které je momentálně evidován přístroj, nebude kvůli referenční integritě tento požadavek proveden. V tabulce tool musí být povinně zadané hodnoty pro všechny sloupce a není povolena hodnota NULL. Není

tedy možné, aby byl z tabulek institute a room odstraněn záznam, jehož ID je v tabulce tool využíváno. Stejná pravidla platí pro záznamy v tabulkách person a borrow.

Tab. 7. Cizí klíče v databázi tools_database

Cizí klíč	Odkazovaný klíč
tool.instituteID	Institute.instituteID
tool.roomID	room.roomID
borrow.personID	person.personID
borrow.toolID	tool.toolID

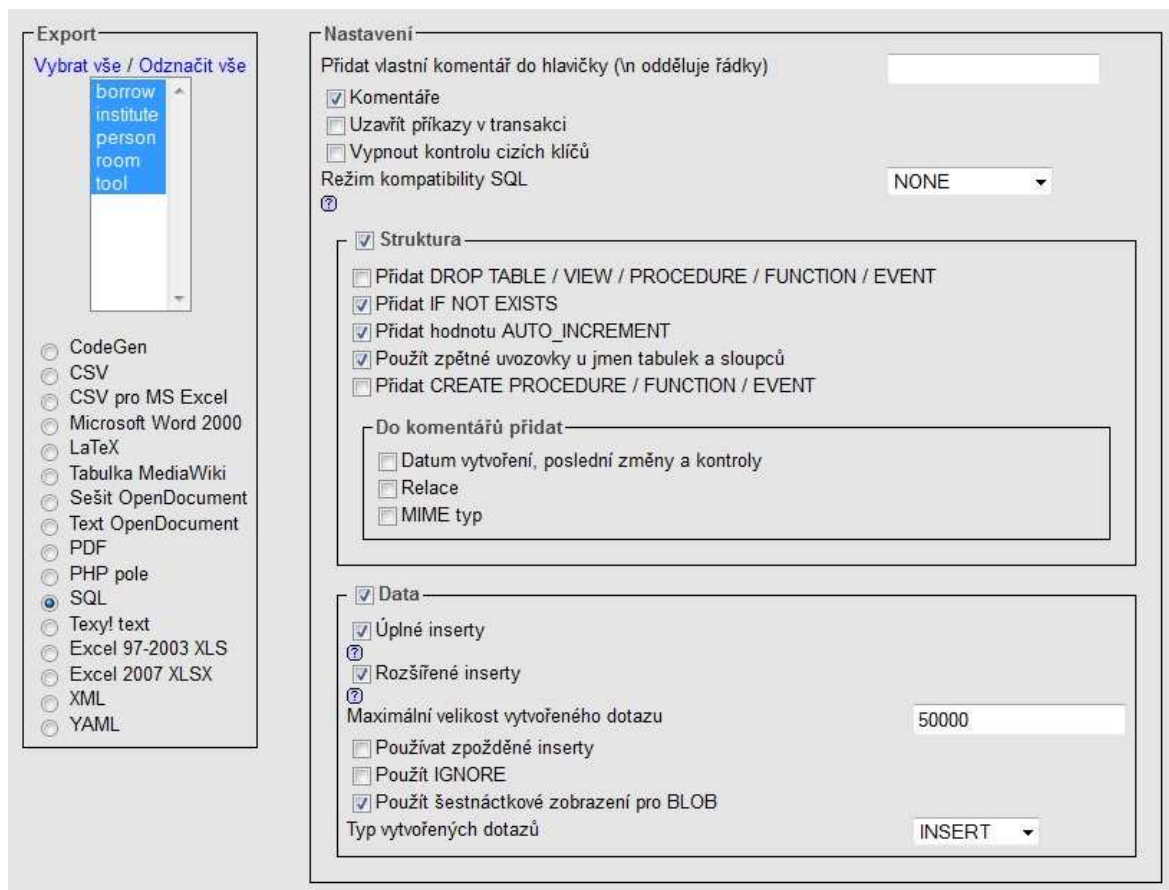
Jak vidíme z tabulky Tab. 7., tak jsou cizí klíče pouze v tabulkách tool a borrow. Tabulky tool a borrow tedy můžeme považovat za pravděpodobně nejdůležitější tabulky naší databáze. Rozhodně jsou to tabulky, se kterými přijde uživatel databázového systému nejvíce do kontaktu ve webové aplikaci. Tabulky person, institute a room slouží pro zaznamenávání pomocných údajů, jejichž výčet je omezen. Například už teď je v tabulce institute uvedeno všech sedm ústavů, která Fakulta aplikované informatiky má. Při důkladném vložení záznamů do těchto tří tabulek nemusí být do budoucna ve velké míře využívány.

5.4 Import a export databáze tools_database

Po zvolení databáze ji lze v prostředí PhpMyAdmin importovat a exportovat, což velice usnadňuje práci s databázemi a jejich implementaci.

Obr. 16. Import databáze

Po zvolení záložky import je třeba zadat cestu k souboru, který chceme importovat. Pro importované soubory je podporována i jejich komprimace a to ve formátech gzip, bzip2 a zip. Pro formát importovaného souboru je standardně nastaven SQL, ale je možné zvolit i formáty CSV, XLS, XLSX, XML, OPEN DOCUMENT.



Obr. 17. Export databáze

Možnosti pro export databáze jsou značné. Lze vybrat přímo tabulky, které chceme exportovat a jestli chceme exportovat strukturu nebo i data. Uživatelsky příjemná je i možnost vybrat si z dostatečně široké nabídky formátů souborů, do kterých budeme databázi exportovat. Pro přenos je nejvýhodnější použít export do souboru s SQL, který má také nejširší možnosti nastavení jak je z obrázku Obr. 17. patrné.

6 WEBOVÁ APLIKACE

Pro vytváření webové aplikace byl využit program Netbeans IDE 6.9.1. a balíček EasyPHP 5.3.6.0, obsahující Apache, PHP, MySQL a PHPMyAdmin. Zdrojový kód webové aplikace obsahuje prvky skriptovacího jazyka PHP, javascriptu (kontrola zadávaných hodnot) a HTML, ve kterém jsou vytvářeny tabulky a formuláře. Při navrhování stránky byly použity kaskádové styly CSS a stránka byla také rozdělena na tři základní rámy.

PHP Version 5.3.6	
System	Windows NT WENCESLAS-PC 6.1 build 7600 (Unknow Windows version Home Premium Edition) i586
Build Date	Mar 17 2011 10:34:15
Compiler	MSVC9 (Visual C++ 2008)
Architecture	x86
Configure Command	cscript /nologo configure.js "--enable-snapshot-build" "--disable-isapi" "--enable-debug-pack" "--disable-isapi" "--without-mssql" "--without-pdo-mssql" "--without-pi3web" "--with-pdo-oci=D:\php-sdk\oracle\instantclient10\sdk,shared" "--with-oci8=D:\php-sdk\oracle\instantclient10\sdk,shared" "--with-oci8-11g=D:\php-sdk\oracle\instantclient11\sdk,shared" "--enable-object-out-dir=.\obj/" "--enable-com-dotnet" "--with-mcrypt=static"
Server API	Apache 2.0 Handler
Virtual Directory Support	enabled
Configuration File (php.ini) Path	C:\windows
Loaded Configuration File	C:\Program Files\EasyPHP-5.3.6.0\apache\php.ini
Scan this dir for additional .ini files	(none)
Additional .ini files parsed	(none)
PHP API	20090626
PHP Extension	20090626
Zend Extension	220090626
Zend Extension Build	API220090626,TS,VC9
PHP Extension Build	API20090626,TS,VC9

Obr. 18. Konfigurace PHP

6.1 Struktura webové aplikace

Při realizaci struktury webové aplikace byly využity rámy (frames), které umožňují rozdělovat stránku na 3 základní oblasti, se kterými se následně uživatel pracuje.

V prvním zobrazovaném rámu pojmenovaném head, se zobrazuje pouze obrázek s logem a názvem univerzity a fakulty. Po celou dobu využívání webové aplikace je obsah rámu head stejný. Rám head rozděluje strukturu stránky a zabírá standardně 12 procent z výšky stránky.

Druhý rám, který je pojmenovaný menu, rozděluje strukturu stránky horizontálně a zabírá 20 procent jejího obsahu od levého okraje. Rám obsahuje navigační menu pro práci s webovou aplikací a jeho obsah je také po celou dobu užívání neměnný. Rám menu obsahuje 7 hypertextových odkazů, které směřují uživatele na hlavní rám naší webové aplikace. Všechny odkazy jsou zarovnané na střed rámu menu, což je realizováno příkazem `align="center"`, který je uveden u každého jednotlivě. Navigační menu je při používání webové aplikace stále viditelné a to zaručuje snadnou orientaci v aplikaci a možnost rychlého přepnutí mezi jednotlivými seznamy a možnostmi pro vkládání záznamů.



Obr. 19. Navigační menu

Zobrazování a samotná práce s webovou aplikací probíhá ve třetím rámu text, kam je uživatel přesměrován kliknutím na jednotlivé odkazy v navigačním menu. Přesměrování do rámu text realizuje příkaz `<base target="text">`.

Rám text je co do procentuálního využití největší oblastí, která se uživateli databáze zobrazuje. Kvůli předpokládanému velkému obsahu dat, která se budou v tomto rámu zobrazovat, je zde aktivní možnost rolování, kterou realizuje příkaz `scrolling="yes"`.

6.2 Vkládání záznamů

Vkládání záznamů do databáze je ve webové aplikaci realizováno pomocí formulářů, které jsou vytvořeny prostřednictvím jazyka HTML.

```
<form name="room" action="insert_room.php" onsubmit="return check_room()" method="post">
<table align="center">
  <tr>
    <td><label for="room">Název místnosti: </label></td>
    <td><input type="text" name="room"
      value="<?php if (isset($_POST['room'])) echo $_POST['room'];?>"</td>
  </tr>
  <tr>
    <td><input type="reset" value="Vymazat údaje"></td>
  </tr>
  <tr>
    <td><input type="submit" value="Uložit" name="ulozit"></td>
  </tr>
</table>
</form>
```


Obr. 20. Zdrojový kód formuláře pro vkládání nových místností

Jednotlivé položky formuláře jsou zobrazovány jako kolonky tabulky, což zaručuje jejich přehlednost. Jméno formuláře pro vkládání nových místností je `room`. Příkaz `action` udává název skriptu, který se bude po odeslání formuláře vykonávat. Kontrola údajů, které jsou do formuláře zadávány je realizována pomocí příkazu `onsubmit="return check_room()"`. Tento příkaz doslova říká, po potvrzení údajů zavolej funkci `check_room()`. Kontrola údajů je realizována javascriptem a dostaneme se k její praktické ukázce v kapitole 6.3.

Pro přenos dat je zvolena metoda `post`. Mimo tlačítka pro uložení údajů ve formuláři máme k dispozici ještě tlačítko pro vymazání údajů, které jsou vyplněny, avšak ještě nebyly odeslány k uložení.

6.2.1 Vložení nové zodpovědné osoby

Vložení nové zodpovědné osoby je realizováno pomocí webového formuláře, jeho zobrazení si můžeme prohlédnout na obrázku Obr. 21.



Vložte novou zodpovědnou osobu

Jméno:

Příjmení:

Telefon:

E-mail:

Vymazat údaje

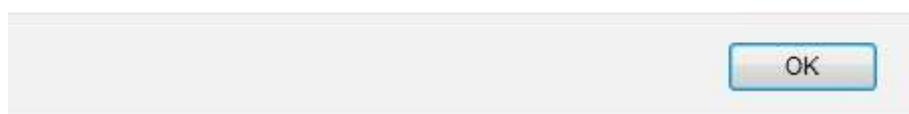
Uložit

Obr. 21. Vložení nové zodpovědné osoby

Pro vložení záznamu o nové zodpovědné osobě je nutné vyplnit hodnoty jméno, příjmení, telefon, e-mail. Záznam se ukládá do tabulky person v databázi tools_database. Pokud není jakýkoliv údaj zadán a uživatel zmáčkne tlačítko uložit, zobrazí se chybové hlášení upozorňující na nutnost zadat příslušnou hodnotu. Speciální kontrola probíhá při zadávání hodnoty telefon a email. Telefonní číslo je nutné zadávat jako 9 číselných hodnot.

Zadávání položky email je specifikováno na tvar jmeno@fai.utb.cz, což je standardním formátem emailové adresy, která je přidělována zaměstnancům Fakulty aplikované informatiky a je veřejná ve výpisu zaměstnanců u jednotlivých ústavů.

CHYBA: Položku email musíte zadavat v nasledujícím tvaru: jmeno@fai.utb.cz!

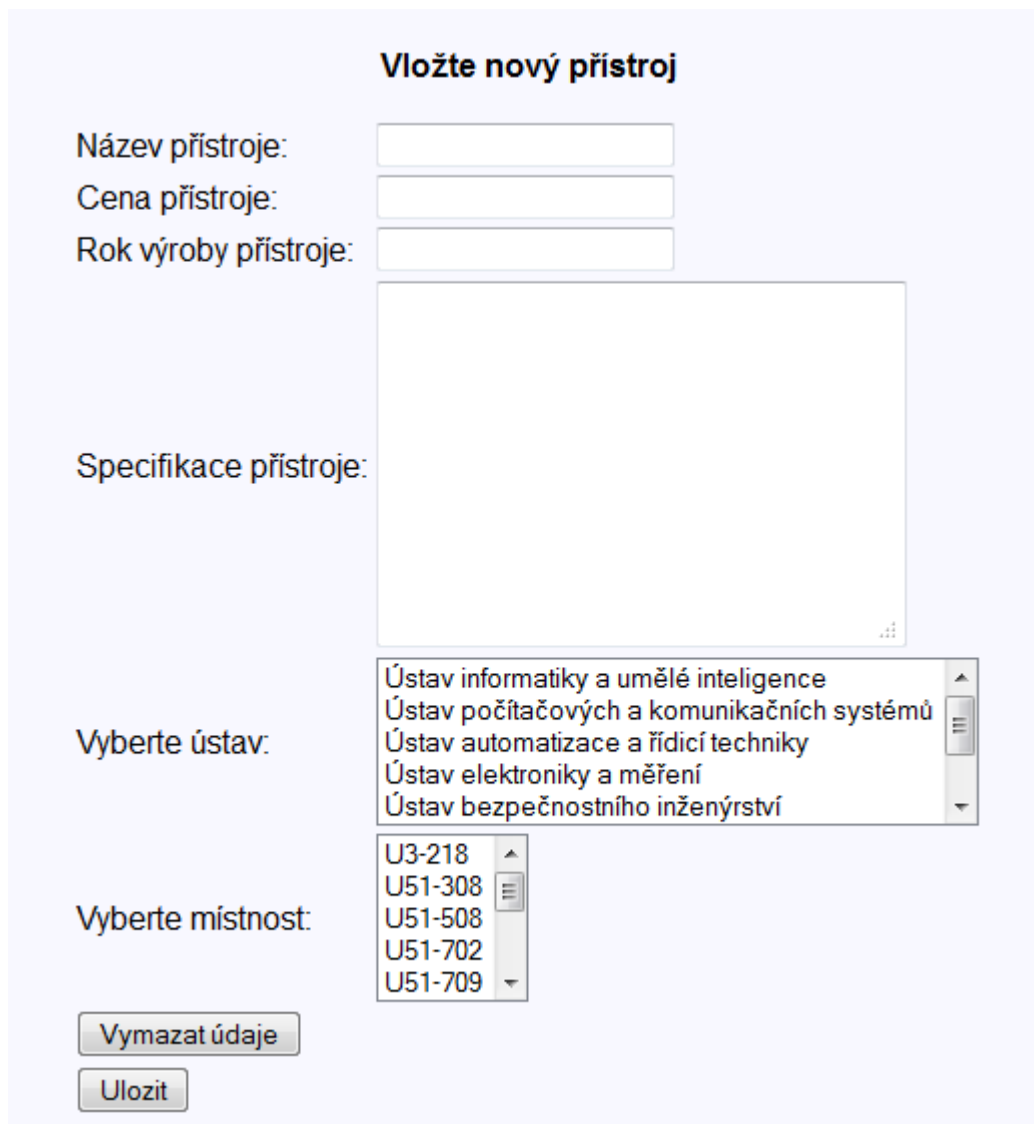


Obr. 22. Chybové hlášení pro hodnotu email

6.2.2 Vložení nového přístroje

Formulář pro vkládání nového přístroje je pravděpodobně nejdůležitějším formulářem ve webové aplikaci. Při zadávání záznamu o novém přístroji bylo cílem nabídnout uživateli

databáze co nejefektivnější způsob pro zadání jednotlivých hodnot. Údaje cena přístroje a rok výroby jsou zadávány jako čísla, ostatní údaje se zadávají jako text.



Vložte nový přístroj

Název přístroje:

Cena přístroje:

Rok výroby přístroje:

Specifikace přístroje:

Vyberte ústav:

- Ústav informatiky a umělé inteligence
- Ústav počítačových a komunikačních systémů
- Ústav automatizace a řídicí techniky
- Ústav elektroniky a měření
- Ústav bezpečnostního inženýrství

Vyberte místnost:

- U3-218
- U51-308
- U51-508
- U51-702
- U51-709

Vymazat údaje

Uložit

Obr. 23. Vložení nového přístroje

Jak vidíme na obrázku Obr. 23., tak při zadávání hodnot ústav a místnost má uživatel na výběr z již vložených záznamů o místnostech ústavů. Díky této možnosti je vkládání intuitivní, maximálně efektivní a jsou minimalizovány překlepy a omyly při zadávání záznamů o nových přístrojích. I tyto údaje je nutné zadat. Pokud při zadávání nebudou vybrány, zobrazí se standardní chybové hlášení o nutnosti zvolit příslušnou hodnotu.

```

<tr>
  <td><label for="ustav">Vyberte ústav: </label></td>
  <td>
    <SELECT name="ustav" size="5">
      <?php while ($zaznam_ustavu = mysql_fetch_array($zaznam1)):?>
        <OPTION value="<?php echo $zaznam_ustavu["instituteID"];?>">
          <?php echo $zaznam_ustavu["institute_name"];?>
        </OPTION>
      <?php endwhile;?>
    </SELECT>
  </td>
</tr>

```

Obr. 24. Zdrojový kód pro výběr ústavů

Ve zdrojovém kódu je možnost výběru z již zadaných ústavů realizována pomocí formulářového prvku select. Příkaz `size="5"` udává počet záznamů, ze kterých uživatel vybírá, a jsou zobrazeny najednou bez nutnosti rolování. Jak vidíme ze zdrojového kódu na obrázku Obr. 24., tak do hodnoty `ustav` je ukládána hodnota `instituteID`, ale zobrazena je hodnota `institute_name`. Pokud by měl uživatel na výběr z ID ústavu, příliš by mu to při zadávání nepomohlo. Je rozhodně uživatelsky příjemnější, pokud můžeme vybírat z názvu ústavů a ne z jejich ID.

```

mysql_query("INSERT INTO tool (tool_name, tool_price, tool_year,
  tool_specification, instituteID, roomID)
VALUES ('${_REQUEST['nazev']}', '${_REQUEST['cena']}',
  '${_REQUEST['rok']}', '${_REQUEST['specifikace']}',
  '${_REQUEST['ustav']}', '${_REQUEST['mistnost']}')");

```

Obr. 25. Zdrojový kód příkazu pro vložení záznamu do tabulky `tool`

Po odeslání údajů z formuláře je zavolán skript `insert_tool.php`, který má na starosti vložení nového záznamu a zadaných hodnot do tabulky `tool` v databázi MySQL. První činností skriptu je přihlášení do databáze, které je viditelné na obrázku Obr. 2. Samotný příkaz pro vložení hodnot z formuláře realizuje příkaz viditelný na obrázku Obr. 25. Pokud je nastavena ve formuláři metoda pro přenos dat na post, jsou hodnoty ukládány a dají se vyvolat příkazem `$_REQUEST['nazev']`.

6.3 Kontrola hodnot při vkládání záznamů

Jak jsme již zmínili v předchozí kapitole, kontrola zadaných údajů je realizována ve formuláři pro zadávání dat příkazem `onsubmit=""`, po kterém je zavolána příslušná funkce.

```
function check_room()
{
    if (!check_room_name(self.document.forms.room.room.value))
    {
        alert("CHYBA: Polozku nazev musite zadavat ve tvaru UXX-YYY");
        return false;
    }

    function check_room_name(mistnost)
    {
        km = /^U+[0-9]{1,2}-[0-9]{1,3}$/;
        return mistnost.search(km) == 0;
    }
}
```

Obr. 26. Zdrojový kód funkce `check_room`

Jak vidíme na obrázku Obr. 26., tak při zadávání názvu místnosti kontroluje funkce, jestli je název zadán ve tvaru U, jedno nebo dvě celá čísla, znak – a jedno až tři celá čísla. Pokud není tento formát dodržen, zobrazí se chybové hlášení „CHYBA: Polozku nazev musite zadavat ve tvaru UXX-YYY“.

6.4 Zobrazování zadaných záznamů ve webové aplikaci

Ve webové aplikaci je možné zobrazit vložené záznamy o místnostech, ústavech, osobách, přístrojích a jejich výpůjčkách. Záznamy jsou vypisovány do tabulek, které mají pro přehlednost stejný formát a jde je tedy jednoduše odlišit. Pro zvýraznění je nastaveno u tabulek jednotné pozadí příkazem `bgcolor="cadetblue"`. Všechny tabulky jsou pomocí příkazu `align="center"`, zobrazovány uprostřed rámu.

Vhodné využití příkazu `select` a efektivní zobrazení si nejlépe ukážeme na výpisu seznamu evidovaných přístrojů. Pro maximální efektivitu a uživatelskou příjemnost spojuje výpis přístrojů údaje ze čtyř tabulek (`tool`, `room`, `borrow` a `person`). V seznamu přístrojů se tak přímo zobrazí název místnosti, ve které se přístroj nachází, a údaje o případné výpůjčce. U výpůjčky vidíme, kdy začala a komu byl přístroj vypůjčen. Pokud u přístroje není evidovaná výpůjčka, jsou tyto sloupce prázdné.

ID	Název přístroje	Cena přístroje	Místnost	Specifikace	Datum vypůjčení	Vypůjčeno
1	HYUNDAI WSC 2032B	2499	U51-718	Součástí balení je senzor na měření teploty Způsob přenosu informací: bezdrátový, Dosah senzoru v otevřeném prostoru: 30 m, Max. počet externích senzorů: 3, Napájení: 1,5 AAA baterie		
2	multimetr	200	U51-308	Digitální multimetr s měřením teploty FK 8400		
3	Zkoušečka UTP	1925	U51-709	Zkoušečka UTP a koaxiálních kabelů s měřením délky UTP R222	2011-06-01 18:25:25	Navrátil
4	SanDisk Micro SDHC	159	U51-508	Spolehlivá paměťová karta s kapacitou 4 GB		
5	CISCO SD205T-EU	697	U51-718	Konektivita: Ethernet 10/100Base-T(X)		
6	IBM x3200M3 Tower	25957	U3-218	Procesor: Model: Intel Xeon X3440 Počet jader: 4 Počet threadů: 8 Frekvence: 2,53 GHz Smart Cache: 8 MB Výrobní proces: 45 nm Max. spotřeba: 95 W		
7	BenQ W1000+	21479	U51-702	DLP projektor nat. Full HD 1080p (1920x1080), max. WUXGA (1920x1200), 2000 ANSI Lm, kontrast 3500:1, 2x HDMI		
8	Xerox Documate 252	20422	U51-810	Rychlost skenování (A4, 150 dpi, barevně): Jednostranně: 24 stran/min Oboustranně: 40 stran/min	2011-06-01 18:25:59	Šenkeřík
9	Microsoft Visual Studio 2010	22586	U54-308	Tvorba velmi výkonných aplikací pro Windows 7 nativně v C++	2011-06-01 18:21:16	Dulík
10	Evolve detektor kouře	458	U3-218	Citlivost detekce: odpovídá standardu UL217		

Obr. 27. Seznam evidovaných přístrojů

6.5 Vyhledávání a mazání záznamů ve webové aplikaci

Ve webové aplikaci je možné mazat všechny vložené záznamy. Na každé stránce s výpisem záznamů je formulář, který smaže záznam po zadání jeho ID, které uživatel zjistí z výpisu jednotlivých záznamů, které jsou na stránce zobrazeny. V seznamu přístrojů je možné vyhledávat a to podle dvou parametrů specifikace a názvu. Do formuláře uživatel vloží znaky, které chce ve specifikaci nebo názvu vyhledat a odešle požadavek. Výsledek vyhledávání se zobrazí v tabulce, které má stejná specifika jako standardní výpis přístrojů.

Vyhledávání podle specifikace přístroje

Zadejte znaky specifikace, které chcete vyhledat:

Smazání přístrojů

Zadejte ID přístroje, který chcete smazat:

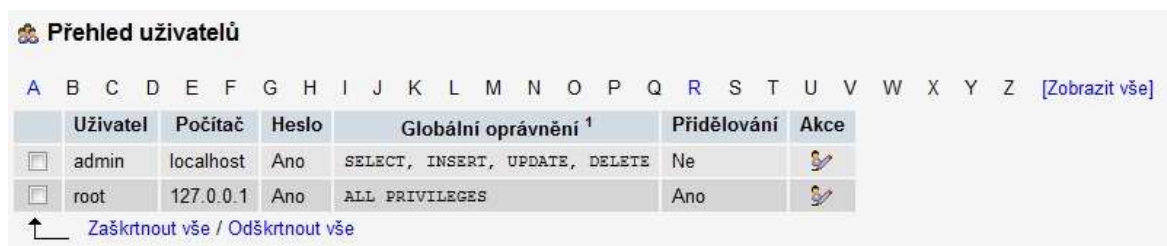
Obr. 28. Vyhledávání a mazání záznamů ve webové aplikaci

7 ZABEZPEČENÍ DATABÁZOVÉHO SYSTÉMU

Zajištění bezpečnosti databáze se nevztahuje pouze na informace obsažené v databázi. Pokud je porušena informační bezpečnost celkově, tak to může mít zpětně vliv i na databázový systém. Při zabezpečování databáze je nutné zahrnout i zabezpečení hardwaru, softwaru a zaměstnanců.

Prvním úkolem, který si informační bezpečnost klade a který vede k minimalizaci bezpečnostních rizik, je omezení přístupu k informacím pouze pro ty uživatele, pro které je práce z danými informacemi naprosto nezbytná. Při návrhu zabezpečení je prvním krokem zajistit, aby k databázovému systému měli přístup pouze uživatelé lokální sítě, čímž minimalizujeme další rizika, která pro bezpečnost databázového systému hrozí. Základem je tedy fyzická nebo administrační kontrola osob, které budou mít do databáze přístup pouze na zaměstnance fakulty aplikované informatiky.

Jednou z dalších pouček pro zabezpečení databáze je skutečnost, že bezpečnost databáze je pouze tak dobrá, jako je bezpečnost operačního systému. Do procesu ověřování je tedy nutné zahrnout i přidělování práv jednotlivým uživatelům v systému. Některé databázové systémy evidují seznam uživatelů a přístupových údajů, které mohou být jiné, než přístupové údaje k operačnímu systému. Je možné také porovnávat přihlašovací údaje uživatele databáze s přihlašovacími údaji do operačního systému. Touto kontrolou je možné zamezit přístupu do databáze s jinými údaji, než se kterými se uživatel přihlásil do operačního systému.



Přehled uživatelů

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z [\[Zobrazit vše\]](#)

	Uživatel	Počítač	Heslo	Globální oprávnění ¹	Přidělování	Akce
<input type="checkbox"/>	admin	localhost	Ano	SELECT, INSERT, UPDATE, DELETE	Ne	
<input type="checkbox"/>	root	127.0.0.1	Ano	ALL PRIVILEGES	Ano	

[↑](#) [Zaškrtnout vše](#) / [Odškrtnout vše](#)

Obr. 29. Seznam uživatelů z PhpMyAdmin

Základním stupněm je zabezpečení přímo MySQL. Z obrázku Obr. 29. vidíme, že oba uživatelé, kteří jsou v MySQL zadáni, mají zadané heslo, čímž je splněna základní podmínka pro bezpečnost databáze MySQL. Neexistuje uživatel, který nemá heslo. Neexistuje uživatel bez jména a k databázi tools_database má přístup minimum uživatelů.

K databázi tools_database má uživatel admin, jehož přihlašovací údaje jsou zadány ve webové aplikaci při přihlašování, práva pouze SELECT, INSERT, UPDATE, DELETE. Uživatel admin má tedy i po přihlášení možnost do databáze vkládat, mazat, měnit a vyhledávat údaje. Nemá možnost měnit strukturu databáze a pracovat s nastavením tabulek nebo relací.



Uživatelé mající přístup k „tools_database“					
Uživatel	Počítač	Typ	Oprávnění	Přidělování	Akce
admin	localhost	globální	SELECT, INSERT, UPDATE, DELETE	Ne	
root	127.0.0.1	globální	ALL PRIVILEGES	Ano	

Obr. 30. Uživatelé mající přístup k databázi tools_database

Speciálním rizikem pro zabezpečení databáze je využití SQL injection. Díky SQL injection je útočník schopný přes nezabezpečený vstup dat pozměnit příkaz a provést místo něj svůj vlastní, kterým například odstraní údaje nebo celou tabulku. Mazání záznamu v tabulce institute je realizováno příkazem `$dotaz="DELETE FROM institute WHERE instituteID='{$_REQUEST['ID']}'";`.

Navrhovaná úprava zabezpečení proti SQL injection je následující:

```
$deleteid=intval($_REQUEST['ID']);
```

```
$dotaz="DELETE FROM institute WHERE instituteID=$deleteid";
```

Momentálně není tímto způsobem proti napadení SQL injection databázový systém chráněn. Pokud bude splněna podmínka omezení přístupu k informacím pouze pro zaměstnance Fakulty aplikované informatiky, nevníkne potřeba bránit se proti napadení SQL injection.

ZÁVĚR

Teoretická část bakalářské práce obsahuje definice a základní pojmy potřebné pro vytváření databázových systémů a jejich propojení se skriptovacím jazykem PHP. Jsou zde rovněž uvedena základní pravidla pro zabezpečení databází.

Na základě zjištěných specifik a analýze potřeb byla navržena struktura databáze, která by sloužila k evidenci přístrojů pořízených na Fakultu aplikované informatiky. Navržená databáze obsahuje pět tabulek, které umožňují evidovat záznamy o přístrojích, určovat přiřazení přístroje k místnosti a ústavu a půjčovat pořízené přístroje osobám v určitém časovém úseku.

Databáze byla naplněna skutečnými hodnotami o místnostech, ústavech a osobách. Byly také vytvořeny záznamy o deseti fiktivních přístrojích, z nichž tři byly zapůjčeny.

Při vytváření webové aplikace byl kladen důraz na uživatelsky příjemný systém a prevenci chyb při vkládání nových záznamů. Webová aplikace umožňuje také mazat a vyhledávat záznamy. Na výpisu seznamu předmětů bylo prakticky předvedeno propojení 4 tabulek za účelem získání kompletních informací o evidovaném přístroji.

I když nebyl databázový systém přímo implementován a využíván Fakultou aplikované informatiky, byl navržen způsob zabezpečení, včetně návrhu ochrany proti SQL injection.

ZÁVĚR V ANGLIČTINĚ

Theoretical part of the thesis contains definitions and basic terms, which are needed for creating of database systems and their connection with scripting language PHP. In theoretical part of the thesis are also described basic security rules.

Structure of database for tools evidential on Faculty of Applied Informatics was created after analysis and describing the specific demands. Proposed structure contains 5 tables, which allows evidence of tools and their connection with institute and room. Database allows also to make borrow of tool for one person and specific period of time.

Records of rooms, institutes and persons were inserted into database. Also records of 10 tools and 3 borrows were inserted.

Web application was developed with focus on user's friendly system and prevention of mistakes in the process of inserting records. Web application allows us to delete and search in the records. Connection of 4 tables was shown on the table, which contains records of tools. Reason for creating specific table was to give complete information about records of tools.

Security mechanism, which contains protection against SQL injection was proposed, even though database system was not implemented on Faculty of Applied Informatics.

SEZNAM POUŽITÉ LITERATURY

- [1] Lacko, Luboslav. SQL Hotová řešení. Computer Press, 2007, Dotisk prvního vydání, ISBN: 80-7226-975-5
- [2] Lacko, Luboslav. PHP a SQL Hotová řešení. CP Books, 2005, Vydání první, ISBN: 80-251-0397-8
- [3] Stephens, Ryan, Plew, Ron, Jones, Arie D. Naučte se SQL za 28 dní. Computer Press, 2010, Vydání první, ISBN: 978-80-251-2700-1
- [4] Kofler, Michael, Öggl, Bernd. PHP 5 a MYSQL 5 Průvodce webového programátora. Computer Press, 2007, Vydání první, ISBN: 978-80-251-1813-9
- [5] Prokopová, Zdenka, Databázové systémy MYSQL + PHP. Univerzita Tomáše Bati ve Zlíně, 2006, Vydání první, ISBN: 80-7318-486-9

SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK

SQL	Strukturovaný dotazovací jazyk
PHP	Osobní domácí stránky.
ad-hoc	Dočasné síťové spojení mezi dvěma rovnocennými prvky.
tzn	To znamená.
1NF	1. normální forma.
2NF	2. normální forma.
3NF	3. normální forma.
BCNF	Boyce-Coddova normální forma.
HTML	HyperText Markup Language.
www	World Wide Web.
SSL	Secure Sockets Layer.
CSS	Cascading Style Sheets.

SEZNAM OBRÁZKŮ

<i>Obr. 1. Komunikace serveru PHP a MySQL</i>	23
<i>Obr. 2. Zdrojový kód v PHP, který realizuje připojení k databázi MySQL</i>	24
<i>Obr. 3. Systémové informace z PhpMyAdmin</i>	30
<i>Obr. 4. Vytvoření databáze</i>	31
<i>Obr. 5. Struktura tabulky tool</i>	32
<i>Obr. 6. Záznamy v tabulce tool</i>	32
<i>Obr. 7. Struktura tabulky institute</i>	33
<i>Obr. 8. Záznamy v tabulce institute</i>	34
<i>Obr. 9. struktura tabulky room</i>	34
<i>Obr. 10. Záznamy v tabulce room</i>	35
<i>Obr. 11. Struktura tabulky borrow</i>	35
<i>Obr. 12. Záznamy v tabulce borrow</i>	36
<i>Obr. 13. Struktura tabulky person</i>	36
<i>Obr. 14. Záznamy v tabulce person</i>	37
<i>Obr. 15. ERA diagram databáze tools_database</i>	38
<i>Obr. 16. Import databáze</i>	40
<i>Obr. 17. Export databáze</i>	41
<i>Obr. 18. Konfigurace PHP</i>	42
<i>Obr. 19. Navigační menu</i>	43
<i>Obr. 20. Zdrojový kód formuláře pro vkládání nových místností</i>	44
<i>Obr. 21. Vložení nové zodpovědné osoby</i>	45
<i>Obr. 22. Chybové hlášení pro hodnotu email</i>	45
<i>Obr. 23. Vložení nového přístroje</i>	46
<i>Obr. 24. Zdrojový kód pro výběr ústavů</i>	47
<i>Obr. 25. Zdrojový kód příkazu pro vložení záznamu do tabulky tool</i>	47
<i>Obr. 26. Zdrojový kód funkce check_room</i>	48
<i>Obr. 27. Seznam evidovaných přístrojů</i>	49
<i>Obr. 28. Vyhledávání a mazání záznamů ve webové aplikaci</i>	49
<i>Obr. 29. Seznam uživatelů z PhpMyAmin</i>	50
<i>Obr. 30. Uživatelé mající přístup k databázi tools_database</i>	51

SEZNAM TABULEK

<i>Tab. 1. Datové typy pro celá čísla</i>	<i>18</i>
<i>Tab. 2. Datové typy pro čísla s pevnou a s plovoucí desetinnou čárkou</i>	<i>19</i>
<i>Tab. 3. Datové typy pro datum a čas</i>	<i>20</i>
<i>Tab. 4. Datové typy pro řetězce znaků</i>	<i>21</i>
<i>Tab. 5. Uživatelská práva v MySQL</i>	<i>26</i>
<i>Tab. 6. Primární klíče v databázi tools_database</i>	<i>39</i>
<i>Tab. 7. Cizí klíče v databázi tools_database</i>	<i>40</i>

SEZNAM PŘÍLOH

CD obsahující aplikační část.

PŘÍLOHA P I: CD