

# Databázová aplikace pro evidenci nebezpečných látek v laboratoři

Database application for keeping  
evidence of dangerous  
substances in laboratory

Miroslav Pilát

Bakalářská práce  
2005



Univerzita Tomáše Bati ve Zlíně  
Fakulta Technologická

***Vložit oficiální zadání diplomové/bakalářské práce***

Děkuji vedoucí práce paní doc. Ing. Dagmar Janáčové, CSc. za odborné vedení a připomínky, které mi při vypracování bakalářské práce poskytla.

Děkuji rodičům a manželce za trpělivost.

Prohlašuji, že jsem na celé diplomové práci pracoval samostatně a použitou literaturu jsem citoval.

Ve Zlíně, 26. 05. 2005

.....

jméno

## **Abstrakt**

Cílem této bakalářské práce je vytvoření databázové aplikace pro evidenci nebezpečných látek v laboratoři v prostředí programu MS Access 2002. Tato práce je vytvářena s ohledem na konkrétní potřeby chemické laboratoře Fakulty Technologické UTB ve Zlíně.

Navržená databázová struktura umožňuje snadno spravovat a udržovat konzistentní data a jednoduše aplikaci rozšiřovat o další tabulky a vazby mezi nimi. Celá aplikace je chráněna uživatelskou úrovní zabezpečení, která umožňuje vytvořit uživatele a jejich skupiny, a tím omezit přístup k jednotlivým objektům databáze.

## **Abstract**

The key objective of this thesis is creation of database application in MS Access 2002 for keeping evidence of dangerous substances in laboratory. Application is based on specific needs of chemical laboratory of the Faculty of Technology in Zlín.

Database design allows easily administer and keep consistent data and can be extended by adding new tables and relations between them. Application is secured by user-level security which allows creation of users and user groups and so forth control user access to database objects.

## OBSAH

|                                                             |           |
|-------------------------------------------------------------|-----------|
| <b>ÚVOD .....</b>                                           | <b>6</b>  |
| <b>1 DATABÁZOVÉ SYSTÉMY .....</b>                           | <b>8</b>  |
| 1.1 DATABÁZE .....                                          | 8         |
| 1.2 SYSTÉM ŘÍZENÍ BÁZÍ DAT .....                            | 9         |
| 1.3 NÁVRH DATABÁZOVÉHO SYSTÉMU .....                        | 10        |
| 1.3.1 Konceptuální model .....                              | 10        |
| 1.3.2 Databázový model .....                                | 14        |
| 1.4 JAZYK SQL .....                                         | 17        |
| 1.5 DATABÁZOVÉ TECHNOLOGIE .....                            | 18        |
| <b>2 PROGRAM MICROSOFT ACCESS 2002 .....</b>                | <b>22</b> |
| 2.1 POUŽITÍ APLIKACE MICROSOFT ACCESS .....                 | 22        |
| 2.2 OBJEKTY MICROSOFT ACCESS .....                          | 22        |
| 2.2.1 Tabulky .....                                         | 23        |
| 2.2.2 Dotazy .....                                          | 23        |
| 2.2.3 Formuláře .....                                       | 24        |
| 2.2.4 Sestavy .....                                         | 24        |
| 2.2.5 Makra .....                                           | 24        |
| 2.2.6 Moduly .....                                          | 25        |
| 2.2.7 Datové stránky .....                                  | 25        |
| 2.2.8 Relace .....                                          | 25        |
| 2.2.9 VBA .....                                             | 26        |
| 2.3 ZABEZPEČENÍ .....                                       | 26        |
| <b>3 DATABÁZE NEBEZPEČNÝCH LÁTEK V LABORATOŘI .....</b>     | <b>28</b> |
| 3.1 EVIDENCE NEBEZPEČNÝCH CHEMIKÁLIÍ V LABORATOŘI .....     | 28        |
| 3.2 NÁVRH A VYTVOŘENÍ OBJEKTŮ DATABÁZE .....                | 28        |
| 3.2.1 Tabulky .....                                         | 29        |
| 3.2.2 Relace .....                                          | 31        |
| 3.2.3 Dotazy .....                                          | 31        |
| 3.2.4 Formuláře .....                                       | 32        |
| 3.2.5 Sestavy .....                                         | 35        |
| 3.2.6 Makra .....                                           | 36        |
| 3.2.7 Moduly .....                                          | 36        |
| 3.2.8 Rozdělení aplikace na část front-end a back-end ..... | 38        |
| 3.2.9 Zabezpečení aplikace .....                            | 38        |
| <b>4 IMPORT DAT .....</b>                                   | <b>40</b> |
| 4.1 IMPORT DAT ZE SEŠITU APLIKACE EXCEL .....               | 40        |
| 4.2 IMPORT HODNOTY CAS .....                                | 40        |
| 4.3 R-VĚTY A S-VĚTY .....                                   | 41        |
| 4.4 KLASIFIKAČNÍ SYMBOLY .....                              | 42        |
| <b>ZÁVĚR .....</b>                                          | <b>43</b> |
| <b>SEZNAM POUŽITÝCH ZDROJŮ .....</b>                        | <b>44</b> |
| <b>SEZNAM OBRÁZKŮ .....</b>                                 | <b>45</b> |
| <b>SEZNAM TABULEK .....</b>                                 | <b>46</b> |
| <b>SEZNAM PŘÍLOH .....</b>                                  | <b>47</b> |

## ÚVOD

Tématem této bakalářské práce je Databázová aplikace pro evidenci nebezpečných látek v laboratoři. Tato práce je vytvářena s ohledem na konkrétní potřeby chemické laboratoře Fakulty Technologické UTB ve Zlíně.

Evidence nebezpečných látek je nedílnou součástí práce v chemické laboratoři. Každá láhev s nebezpečnou látkou musí být popsána pomocí R-vět a S-vět (standardní věty označující specifickou rizikovost resp. standardní pokyny pro bezpečné nakládání), musí na ní být vyznačeny klasifikační symboly nebezpečnosti, její název a případně další informace zjednodušující její identifikaci.

V laboratoři s nebezpečnými látkami pracuje více pracovníků, jejichž znalosti výpočetní techniky se různí, proto bylo potřeba vytvořit uživatelsky příjemnou aplikaci, která by umožnila jednoduše spravovat záznamy o nebezpečných chemikáliích a tisknout štítky na lahve. Cílem této bakalářské práce je vytvoření takovéto aplikace v prostředí programu Microsoft Access, který je většině pracovníků dobře známý.

Největší část správy se odehrává ve formuláři *frmChemikálie*, jehož prostřednictvím uživatel edituje jednotlivé záznamy v tabulce *tblChemikálie*. Z tohoto formuláře je také možné otevřít formuláře pro přidání R- a S-vět a klasifikačních symbolů. Ostatní formuláře slouží k editaci ostatních tabulek databáze a jsou vytvořeny pomocí průvodců aplikace MS Access.

Navržená databázová struktura umožňuje snadno spravovat a udržovat konzistentní data a jednoduše aplikaci rozšiřovat přidáním dalších tabulek a vazeb mezi nimi. Aplikace umožňuje snadnou správu ostatních informací, které se uvádějí u jednotlivých chemikálií (jako např. R a S věty, klasifikační symboly).

Pro zobrazení dat aplikace využívá formulářů aplikace MS Access, které se uživateli jeví jako standardní okna operačního systému MS Windows. Tímto je ušetřen od přímé editace dat v tabulkách. Používání aplikace by tedy mělo být jednoduché a intuitivní podobně jako ostatní aplikace balíku MS Office.

Aby nedošlo k poškození dat jejich špatným zadáním, je celá aplikace zabezpečena uživatelskou úrovní zabezpečení, která umožňuje vytvořit uživatele a jejich skupiny, a tím omezit přístup k jednotlivým objektům databáze. Protože se aplikace bude používat ve více-uživatelském prostředí, kdy se každý uživatel může nacházet na jiném místě, je

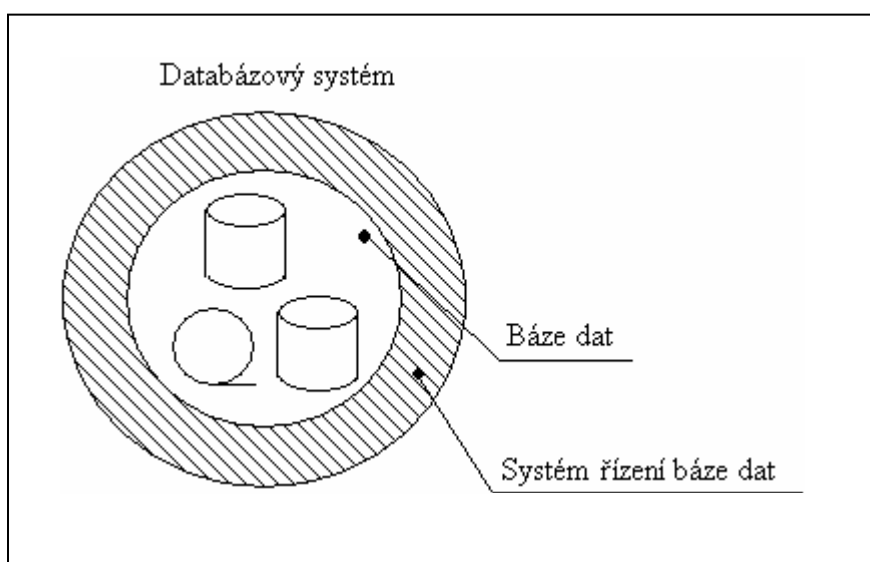
aplikace rozdělena na část front-end (formuláře, makra, dotazy, atd.) a na část back-end (tabulky), která může být umístěna na sdíleném disku.

Práce v první kapitole obecně popisuje databázové systémy a jejich vývojové trendy. V další kapitole se pak věnuje samotnému databázovému systému Microsoft Access a popisu jeho součástí. V třetí kapitole je popsána výsledná aplikace pro správu nebezpečných chemikálií v laboratoři. Zde jsou podrobně popsány všechny vytvořené objekty databáze MS Access. Čtvrtá kapitola je zaměřena na popis importu stávajících dat do databáze. V závěru jsou stručně shrnuty dosažené výsledky a zkušenosti získané při tvorbě aplikace.

## 1 DATABÁZOVÉ SYSTÉMY

Pod pojmem databáze dnes chápeme systém, který nám umožňuje efektivním způsobem pracovat s velkým množstvím dat. Databázové systémy se poprvé objevují v 60. letech minulého století. V té době existovaly převážně účelově vyvinuté programy, pro které je příznačná úzká provázanost fyzických a logických datových struktur a odtud plynoucí nesnadnost získat údaje v jiné podobě, než pro jakou byla úloha vytvořena.

Snahy odstranit nevýhody tohoto přístupu vedly k oddělení dat od programu. Data jsou uložena samostatně v **bázi dat** a programy si vybírají potřebné informace. Na tomto principu pracují **databázové systémy**. Databázový systém je tedy spojení **systému řízení báze dat** (SŘBD) a vlastní **báze dat** (FARANA, R., 1995).



Obr. 1 Databázové systémy

### 1.1 Databáze

Databáze může být specifikována jako souhrn systematicky uspořádaných dat uložených a zpracovávaných odděleně od aplikačních programů. Souhrn postupů, nástrojů a technik využívaných v souvislosti s databázemi je označován jako databázová technologie. V širším smyslu zahrnuje tento pojem i datové modelování, tj. techniky návrhu databáze. Databázové technologie doznaly za 30 let svého vývoje velký pokrok iniciovaný zejména intenzivním rozvojem v oblasti výpočetních prostředků. Databáze, které dnes představují

základní součást moderních informačních systémů, obsahují obecně čtyři komponenty (POKORNÝ, J., 1998):

- datové prvky - vlastní data
- vztahy mezi datovými prvky - jsou uloženy ve složitějších strukturách
- integritní omezení - podmínky kladené na data
- schéma - uživatelský popis struktury dat

## 1.2 Systém řízení bází dat

Již v 70. letech se začaly nahrazovat souborové aplikace postupy vedoucími k oddělení aplikačních programů od vlastních dat. Aplikační programátor tak není nucen znát fyzický způsob uložení dat a metody přístupu k nim. Problémy spojené s organizací dat přebírá Systém řízení bází dat (SŘBD). Vedle rozvoje SŘBD je dnes předmětem velkého vývoje i oblast návrhu databází.

V souvislosti se SŘBD se rozlišují dva typy jazyků:

- jazyk pro definici dat (JDD) - definuje uživatelská data (ang. DDL - Data Definition Language)
- jazyk pro manipulaci dat (JMD) - aktualizuje data a provádí výběry dle zadaných kritérií (ang. DML - Data Manipulation Language)

Jako základní jazyk SŘBD je dnes velmi rozšířen jazyk SQL (Structured Query Language) - strukturovaný (neprocedurální, deklarativní) dotazovací jazyk (kap. 1.5). Obsahuje nejen JDD a JMD, ale i další příkazy, např. pro udělení práv uživatelům.

Jiným typem databázových jazyků jsou 4GL jazyky (jazyky čtvrté generace nebo také formulářové jazyky) určené pro snadnou tvorbu uživatelských rozhraní aplikací. Pro tyto jazyky neexistuje žádný standard - jednotlivé databázové systémy používají vlastní jazyk.

Charakteristickými vlastnostmi současných SŘBD jsou:

- transakční zpracování - každé zpracování dat je realizováno jako posloupnost operací (transakce) převádějící databázi z jednoho konzistentního stavu do druhého
- zotavení z chyb - pokud se nepodaří transakci úspěšně ukončit, musí být všechny její efekty odstraněny
- řízení souběžného přístupu více uživatelů

- ochrana dat - před neoprávněným přístupem, změnou nebo poškozením formou speciálních uživatelských práv zpravidla chráněných heslem (autorizace)
- variabilní architektura dat
  - centralizované uložení dat - data jsou fyzicky uložena na jednom počítači
  - distribuované uložení dat - data jsou fyzicky uložena na více počítačích
- zpracování v reálném čase - umožňuje ovlivňovat procesy běžící v reálném čase (lze definovat čas odezvy)
- práce s multimediálními daty - text, zvuk, obraz, video, apod.

### 1.3 Návrh databázového systému

Návrh databázového systému je obvykle spojen s návrhem informačního systému (IS), jehož je součástí. Při návrhu IS se zjišťuje existence dat a analyzují se funkce, které s daty pracují. Postupně se vyvinuly techniky, postupy a metody návrhu IS obecně nazývané metodologie. Jsou základem řady programových systémů označovaných zkratkou CASE (Computer Aided System Engineering). Při návrhu IS se provádí souběžně funkční a datová analýza.

Datová analýza uplatňující metodu shora - dolů postupně upřesňuje datovou strukturu databázového systému formou vytváření hierarchie popisů schémat lišících se mírou abstrakce:

- konceptuální schéma - vyjádření konceptuálního modelu
- databázové schéma - vyjádření databázového modelu
- realizace databázového schématu - pomocí konkrétního SŘBD
- fyzická implementace databázového schématu - např. indexy pro rychlejší přístup k datům, atd.

#### 1.3.1 Konceptuální model

Konceptuální model popisuje data v databázi na abstraktní úrovni nezávisle na jejich fyzickém uložení. Proces tvorby konceptuálního modelu se nazývá konceptuální modelování. Jeho výsledkem je konceptuální model znázorněný jako konceptuální schéma, které má co nejvýstižněji zachycovat pohled člověka na danou část reálného světa.

Konceptuální schéma se v další etapě návrhu databázového systému konkretizuje do podoby databázového schématu.

Konceptuální schéma obvykle obsahuje tyto komponenty (POKORNÝ, J., 1998):

- struktura dat - data a vazby
- manipulace s daty - řešena v rámci funkční analýzy
- integritní omezení dat - přípustnost datových hodnot

Tvorba konceptuálních modelů vychází ze čtyř principů:

- orientace na objekty - modelovaná realita je vnímána objektově
- podstatou vztahů jsou funkce - vztahy mezi objekty lze chápat jako funkce
- ISA hierarchie - pro objekty mohou existovat nadtypy a podtypy (z ang. IS A)
- hierarchický mechanismus - objekty mohou být konstruovány z jiných objektů

### ***E-R konceptuální model***

Nejznámějším konceptuálním modelem je E-R model (Entity Relationship model), který dnes tvoří v podstatě světový standard.

E-R model popisuje uživatelskou aplikaci na konceptuální úrovni abstrakce za účelem návrhu struktury databáze. Obsahuje především jednotlivé typy entit (Entity types) a typy vztahů (Relationship types) - odtud název modelu. Typ entity (entitní typ) je jednoznačně profilovaný a nezávisle existující objekt reálného světa. Typ vztahu (vztahový typ) je vazba mezi dvěma nebo více typy entit. Samotný pojem entita či vztah označuje výskyt (instanci) typu entity či typu vztahu. Entitám a vztahům lze přiřazovat určitou vlastnost označovanou jako atribut. Podle různých hledisek lze rozlišit různé typy atributů:

- identifikační (klíčové) - jsou součástí identifikačního klíče entity
- popisné - nejsou součástí identifikačního klíče entity
- atomické (jednoduché) - přiřazují entitnímu nebo vztahovému typu nejvýše jednu, dále již nedělitelnou hodnotu
- skupinové (složené) - tvořeny několika složkami (např. adresa), umožňují odkazy na celý atribut nebo jeho jednotlivé složky

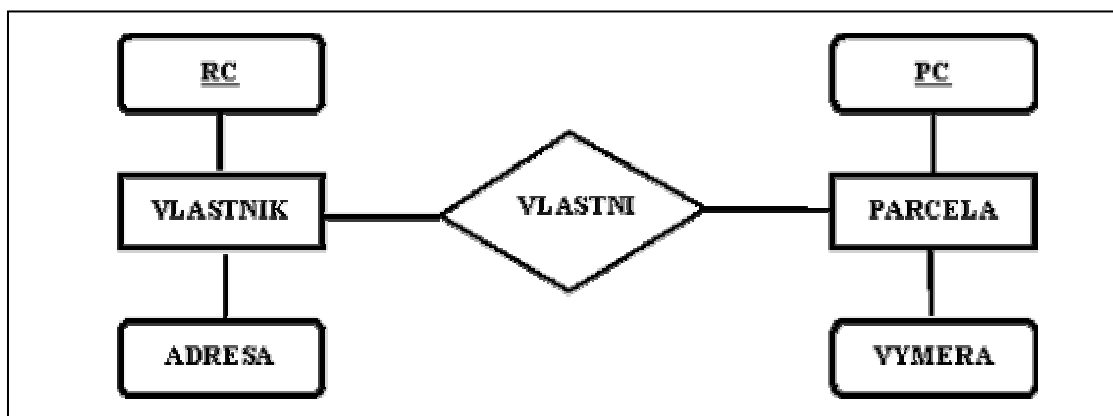
- vícehodnotové - mohou obsahovat více rovnocenných hodnot

Přípustná množina hodnot atributu se nazývá doména atributu. Typy entit identifikovatelné pouze svými atributy jsou silné typy entit (regulární). Ostatní typy entit jsou slabé typy entit (pro jednoznačnou identifikaci musí být použit i atribut jiného typu entity - identifikačního vlastníka).

Postup vytváření E-R modelu je obvykle následující:

- identifikace typů entit
- identifikace typů vztahů
- definice atributů
- definice integritních omezení

Schéma E-R modelu lze znázorňovat buď textovým zápisem či graficky formou diagramu.



Obr. 2 Ukázka diagramu typů entit a vztahů v E-R konceptuálním modelu

E-R diagram spolu s tabulkami vlastností atributů entit tvoří konceptuální schéma vytvořené v E-R modelu. Volba entit a jejich atributů je nejzávažnější činností při tvorbě konceptuálního modelu.

Vedle E-R konceptuálního modelu existuje celá řada dalších modelů. Navzájem se odlišují tím, které typy vztahů rozlišují.

### ***Integritní omezení pro vztahy***

Vztah mezi entitami charakterizuje kardinalita vztahu (poměru). Vztah dvou entit (binární vztah) může nabývat hodnoty:

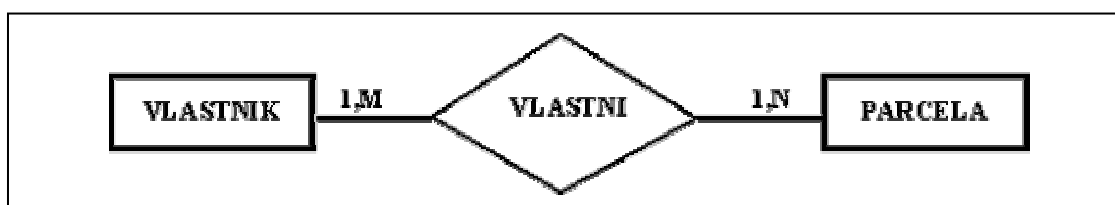
1 : 1 - každé entitě odpovídá maximálně jedna druhá entita

1 : N - první entitě odpovídá více než jedna druhá entita, druhé entitě odpovídá maximálně jedna první entita

M : N - první entitě odpovídá více než jedna druhá entita, druhé entitě odpovídá více než jedna první entita

Typ entity, který je zapojen do vztahu, se nazývá člen vztahu. Jednotlivé entity členu vztahu musí nebo nemusí vstupovat do vztahu. Podle toho se rozlišuje povinné a nepovinné členství ve vztahu. Entitní typ s povinným členstvím ve vztahu je existenčně závislý na druhém entitním typu.

Entita jednoho typu, která jednoznačně určuje entitu druhého typu je determinant entity druhého typu. Tentýž vztah bývá často označován jako funkční závislost druhého entitního typu na prvním entitním typu. Počet výskytů entity ve vztahu se často uvádí číselně minimem a maximem.



Obr. 3 Diagram binárního vztahu s uvedením min. a max. počtu entit

### ***Dekompozice vztahu M:N***

Často bývá výhodné rozložit komplikovaný vztah M:N na dva jednodušší vztahy 1:N. Řešením je zavedení nové entity (průnikový entitní typ) na pozici původního vztahu a vyjádření jejích nových vztahů k původním entitám.

### ***Integritní omezení pro entity***

Každý typ entity a typ vztahu může obsahovat atributy. Atributy se pro každý entitní typ sestavují do tabulky s uvedením integritních omezení:

- typ atributu - pro atomický atribut se definuje obor přípustných hodnot (doména) a množina přípustných operací
- rozlišení klíčového a popisného atributu
- přípustnost prázdné, nedefinované hodnoty atributu
- unikátnost hodnoty atributu

V případě relačního databázového modelu (RDM) se používají ještě též tato integritní omezení:

- klíč relace - pro každou relaci musí existovat určitá minimální množina jejích atributů, která bude jednoznačně identifikovat každou n-tici relace
- referenční integrita - při propojení dvou relací zabezpečí existenci klíče použitého k propojení
- Další integritní omezení mohou být uvedena z důvodu neformalizovatelnosti odlišným způsobem. Volba entit a jejich atributů je nejzávažnější činností při tvorbě konceptuálního modelu.

(SOUKUP, P., 2001)

### **1.3.2 Databázový model**

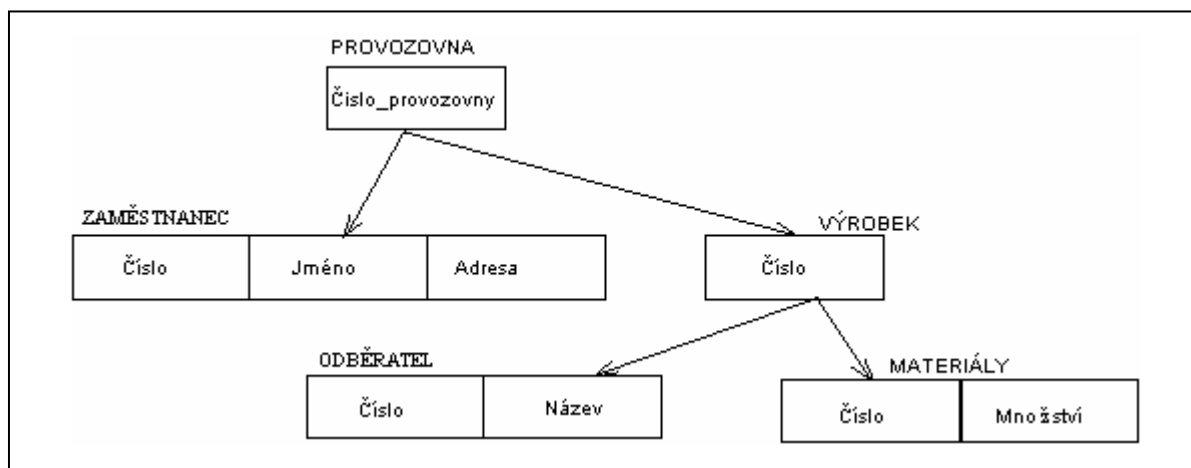
**Model** je souhrn pravidel pro reprezentaci logické organizace dat v databázi (FARANA, R., 1995). V současnosti rozeznáváme hierarchický, síťový, relační a nejnověji i objektový model. Nejpoužívanějším modelem však stále zůstává relační model.

#### ***Hierarchický model***

Data jsou organizována do stromové struktury. Každý záznam představuje uzel ve stromové struktuře, vzájemný vztah mezi záznamy je typu rodič/potomek.

Použití hierarchického modelu je vhodné tam, kde i zájmová realita má hierarchickou strukturu. Mezi nevýhody hierarchického modelu patří (FARANA, R., 1995):

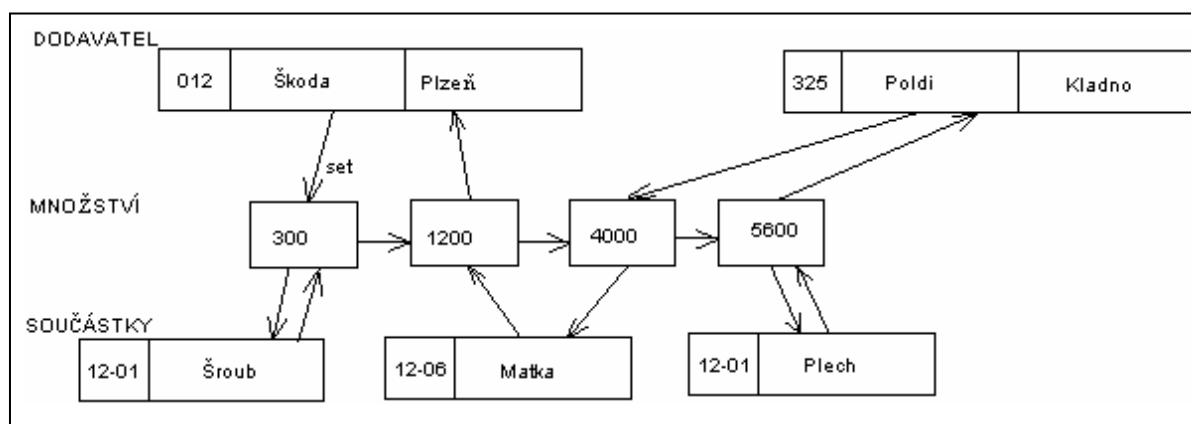
- v některých případech nepřírozená organizace dat (zejména obtížné znázornění vztahu M:N, který se řeší např. pomocí virtuálních záznamů),
- složité operace vkládání a rušení záznamů.



Obr. 4 Hierarchický model dat

### Síťový model

Síťový model dat je v podstatě zobecněním hierarchického modelu dat, který doplňuje o mnohonásobné vztahy. Tyto vztahy jsou označovány jako C-množiny. Tyto množiny propojují záznamy různého či stejného typu, přičemž spojení může být realizováno na jeden nebo více záznamů. Nevýhodou síťové databáze je zejména nepružnost a obtížná změna její struktury (FARANA, R., 1995).



Obr. 5 Síťový model dat

### ***Relační model***

Relační model dat definoval v roce 1970 E. F. Codd.

Základní a jedinou strukturou pro ukládání dat v relačním modelu je tabulka (relace), kde každá řádka (záznam) odpovídá jednomu prvku a každý sloupec (pole) odpovídá jednomu atributu. Zápis polí do jednoho záznamu současně definuje jejich vzájemný vztah. Mezi jednotlivými poli záznamu není žádná hierarchie, každé pole lze použít pro přístup k datům vlastní tabulky či jiné tabulky.

Operace nad relační databází vychází z matematické teorie relační algebry, která např. vyvozuje, že tabulka nesmí obsahovat totožné řádky. Jedno pole nebo kombinace více polí záznamu, která slouží pro jednoznačnou identifikaci záznamů tabulky se nazývá klíč. Spojení záznamů, které jsou uloženy ve dvou různých tabulkách, ale mají stejnou hodnotu klíčů, se nazývá relační propojení. Relační propojení umožňuje snadno a operativně kombinovat hodnoty záznamů různých tabulek a tak získávat odpovědi na nejrozličnější dotazy (SOUKUP, P., 2001).

### ***Objektový model***

Alternativou k relačním databázovým systémům jsou objektové databáze. Místo vytváření tabulek složených z jednotlivých polí je vytvářena značně volnější struktura pomocí jednoznačně identifikovatelných jednotek informace neboli objektů, které obsahují atributy, jež mohou buď uchovávat data, nebo odkazovat na další objekty.

Technologie objektových databází byla vytvořena především pro permanentní uložení datových objektů vytvořených objektově orientovanými softwarovými aplikacemi. Tato technologie je však vhodná i pro uložení dokumentů XML, neboť dokáže snadno popsat sekvenční a hierarchické vztahy.

Výhody uvedeného přístupu jsou však potlačeny snížením výkonu, a tak je tato technologie stále považována za nevhodnou pro masivní průmyslové použití. Na novou konkurenci reagovali dodavatelé relačních databází objektově-relačními databázemi, což je v zásadě objektová vrstva nad existující relační databází, nabízející kompromis mezi výkonem a možnostmi reprezentace objektů (KORENEK, J., 2003).

## 1.4 Jazyk SQL

Jazyk SQL (Structured Query Language - strukturovaný dotazovací jazyk) vyvinula firma IBM počátkem sedmdesátých let zejména pro formulaci dotazů na data v databázi. Od té doby jazyk SQL zaznamenal značný vývoj a dodnes zůstává základním nástrojem databázového světa. Jeho popis i praktické ukázky využití jsou dostupné v řadě publikací (např. POKORNÝ, J., HALAŠKA, I., 1998). Jazyk SQL je neprocedurální jazyk, tj. obsahuje příkazy popisující co se má provést a nikoliv jak se to má provést. V roce 1986 byl jazyk SQL standardizován do podoby SQL86 a v roce 1992 byl rozšířen na verzi SQL92. Vedle příkazů pro formulaci dotazů obsahuje příkazy pro definici a aktualizaci dat, pro definování přístupových práv k tabulkám, pro řízení transakcí, pro definování indexů umožňujících rychlejší přístup k řádkům tabulky, pro definování pohledů umožňujících vytvářet virtuální tabulky z jedné nebo více reálných tabulek, atd.

Databázové modelování světa v SQL má tyto rysy (POKORNÝ, J., 1998):

- data jsou uložena ve formě tabulek
- uživatel nemusí registrovat fyzickou strukturu a uložení dat
- umístění tabulek v databázi není významné - tabulky jsou identifikovány jménem
- umístění sloupců v tabulkách není významné - sloupce jsou identifikovány jménem
- umístění řádků v tabulkách není významné - řádky jsou identifikovány hodnotami ve sloupcích
- data jsou vždy prezentována jako tabulky

Jazyk SQL může být použit jako:

- dotazovací jazyk pro relační databáze
- část hostitelského jazyka pro vývoj databázových aplikací
- prostředek komunikace mezi různými zdroji dat

Základním a nejpoužívanějším příkazem jazyka SQL je příkaz SELECT, který slouží k zápisu požadavků na výběr dat z databáze. Umožňuje zapsat operace relační algebry -

kartézský součin, sjednocení (UNION), rozdíl (EXCEPT), projekci relace na množinu atributů a selekci podle logické podmínky (WHERE). Jazyk SQL je tedy relačně úplný.

Složitější dotaz v jazyku SQL lze obvykle vyjádřit více způsoby v závislosti na způsobu uvažování uživatele jazyka a případně na snaze o optimalizaci dotazu. Cílem optimalizace dotazu je obvykle co nejefektivnější vyhodnocení požadované operace z hlediska času a potřebného diskového prostoru. Optimalizace dotazů může být založena na relativní selektivitě relačních operátorů (eventuelně zpřesněné na základě statistických dat o relacích) nebo na indexech a velikostech relací. V některých SŘBD hraje značnou roli z hlediska optimalizace i vlastní způsob zápisu dotazu. V takovém případě se jedná o optimalizaci řízenou syntaxí, což sice odporuje základní myšlence neprocedurálnosti jazyka SQL, ale často může znatelně zefektivnit vyhodnocení dotazu.

Moderní databázové systémy jsou vybaveny možností automaticky vyvolat SQL příkaz nebo celou proceduru při vzniku určité události (trigger - spoušť). Takovou aktivující událostí může být přidání řádky do tabulky, vymazání řádky z tabulky, změna hodnot ve sloupci, atd. (SOUKUP, P., 2001).

## 1.5 Databázové technologie

Za databázové lze v dnešní době považovat většinu existujících aplikací – od jednoduchých evidencí přes účetní programy či internetové portály až po podnikové informační systémy. V některých případech přitom uložení dat do databáze a jejich následné zpracování ani neočekáváme, například při vstupu do obchodu s nad vchodem umístěným počítačem.

### *Finanční stránka*

Všeobecná představa o vyspělých databázových produktech je taková, že si je mohou dovolit jen velké společnosti a že jsou drahé. Pravda je ale do značné míry opačná. Současné produkty z oblasti databázových technologií jsou cenově srovnatelné s ostatními stavebními kameny podnikové informační architektury. Navíc v celé řadě případů lze využít výhod různých licenčních programů, a získat tak levně kvalitní databázové prostředí. Navíc je zde celá řada velmi kvalitních platform k dispozici zcela zdarma v rámci různých open source projektů. Mezi nejznámější platformy této kategorie patří PostgreSQL či Firebird, případně MySQL, které ale již nemusejí být v některých případech k dispozici zcela zdarma.

### ***Současné trendy***

Mezi současné trendy v databázových technologiích mimo jiné patří snaha o zajišťování vysoké dostupnosti a výkonu, možnosti rozložení zátěže, snižování celkových nákladů na vlastnictví, zjednodušování administrace, zajištění přístupu k datům uloženým v heterogenních zdrojích či o automatickou správu a ladění. Stranou zájmu nezůstávají ani snahy o stále větší zabezpečení.

Moderní databázové technologie nejsou určeny pouze pro pouhé zpracování dat ve smyslu různých evidencí, ale podporují také oblasti analytického zpracování (OLAP, dolování dat, datové sklady), k dispozici jsou deduktivní databáze usnadňující vyhledávání ukrytých závislostí a znalostí. Současné databázové platformy ve větší či menší míře podporují jazykové zpracování textů, nejčastěji pak v podobě fulltextového vyhledávání doplněného o některé sémantické prvky. Samozřejmostí se stává také zpracování dalších nestrukturovaných dat, a to včetně multimediálních.

Stále větší oblibě se těší také podpora technologie XML – ať již v podobě mapované na některý současný model nebo i v podobě nativních XML databází. Stranou zájmu už dlouhá léta nezůstávají ani prostorová a geografická data používaná například pro provoz geografických informačních systémů, stejně jako podpora třívrstvé architektury.

Uživateli a zájemci o databázové technologie pochvalovaným trendem je také široká podpora různých operačních systémů – v mnoha případech jsou jádra databázových platform vyvíjena rozdílně pro odlišené operační systémy. Existuje jen málo databázových platform, které by nebyly k dispozici pro více operačních systémů, jednou z nich je MS SQL Server, který možná zbytečně ztrácí u příznivců unixových platform. Na druhou stranu platí, že někdy je provázanost s daným operačním systémem natolik velká, že různé varianty pro další systémy – jsou-li vůbec možné – nemusejí být příliš povedené.

Rostoucí oblibě se těší také ucelená databázová řešení založená na vzájemné spolupráci dodavatelů databázových platform s předními dodavateli hardwaru, mnohdy navíc doplněná o dodávky aplikačních serverů. Výhodu z toho mají nejen společnosti, které dokáží vytvářet silné aliance, ale také koncoví zákazníci a uživatelé – ti díky tomu dostávají prověřená komplexní prostředí.

***Klient a server***

Obecně je známo, že pro architekturu klient/server platí zpracování rozdělené mezi klienta a server, případně mezi tři vrstvy (klient, aplikační server, databázový server). Již menší povědomí je o tom, že ve skutečnosti existuje celkem pět tříd klient/server, které se liší podle toho, jaké procesy jsou zpracovávány na jaké úrovni. Patří sem plně distribuované zpracování, úplný klient/server, klient/server s mosty, omezený klient/server a minimální klient/server.

Například v případě plně distribuovaného zpracování má server zcela na starosti aplikační logiku a databázi, částečně zajišťuje také správu a funkčnost komponent klientské aplikace (uživatelského rozhraní). Podstatnou podmínkou je přihlašování k jedinému databázovému/aplikačnímu serveru, ovšem z logického pohledu – fyzicky může serverů existovat více. Klient/server s mosty provádí mapování požadavků klienta na vybrané servery pomocí tzv. mostů. Strana serveru zajišťuje databázové pozadí, aplikační logika je rozdělena mezi server a klienta, klient se zcela stará o uživatelské rozhraní.

***Grid***

Základní myšlenka Gridu vychází z využití volné výpočetní kapacity vzájemně trvale či dočasně propojených počítačů tak, aby byl v maximální možné míře využit dostupný výpočetní výkon. V případě databázového zpracování ve gridu nás nemusí zajímat, kde jsou data uložena, a ani to, který počítač či počítače naše požadavky zpracují. Kdykoli si vzpomeneme, položíme dotaz a získáme odpověď. Nebo zadáme příkaz a on se provede. Ve chvíli, kdy to potřebujeme, nikoli ve chvíli, kdy nám dostupný počítač má volný výpočetní výkon. Ne nadarmo je zpracování ve gridu přirovnáváno k elektrické síti — ani zde nás nezajímá, která elektrárna proud do sítě dodává, důležité pro nás je, že vůbec proud máme k dispozici.

Platforma Oracle Database 10g přináší možnost využití zpracování ve gridu všem databázovým a ve své podstatě i nedatabázovým aplikacím. Tíha zajištění výhod gridu leží na provozních platformách, nikoli na aplikacích jako takových. Grid tak otevírá novou generaci databázových technologií, navíc bude díky Oracle Database 10g dostupný i pro běžné komerční účely.

***Jakou databázovou platformu?***

Která databázová platforma je nejlepší? Vždy záleží na úhlu pohledu a vždy je nutné se ptát především na to, k čemu chceme danou databázovou platformu používat.

Z technologického pohledu lze databázové produkty srovnávat především na základě nabízené funkčnosti, podpory progresivních technologií a souladu s průmyslovými standardy. Z pohledu výkonu lze vycházet především ze standardizovaných testů, ale nejpoužívanější výkonnostní testy, mezi které patří testy TPC, totiž jen málokdy odpovídají reálnému provozu, reálnému nasazení v praxi. Obvykle jsou tyto testy prováděny ve velmi omezeném časovém úseku, a zjišťují tak pouze okamžitý výkon. K dispozici pak většinou nejsou informace o nárůstu zátěže, výpadcích, problémech se škálovatelností apod.

Jinými slovy – pro velmi jednoduchou aplikaci může být kvalitnější MS FoxPro 6.0 než IBM DB2 Universal Database. Pod kvalitou se totiž skrývá především vhodnost daného produktu ke splnění požadavků kladených na řešení. (KOCAN, M., 2003)

## 2 PROGRAM MICROSOFT ACCESS 2002

Microsoft Access 2002 je databázový program, který je součástí profesionální verze balíku Microsoft Office XP. Umožňuje uživateli velmi jednoduchým a uživatelsky přívětivým způsobem vytvářet databázové aplikace malého až středně velkého rozsahu. MS Access poskytuje obrovské množství výkonných nástrojů pro vytváření tabulek k uchovávání dat, vytváření formulářů určených k zadávání a upravování dat v tabulkách, a také tvorbu sestav pro prezentování dat uložených v databázi (FEDDEMA, H., 2002).

### 2.1 Použití aplikace Microsoft Access

V programu Microsoft Access můžeme ukládat data buď vnitřně (do souboru s aplikací) nebo se připojit k datům z vnějších zdrojů (např. k MS SQL databázi někde na vzdáleném serveru). Uložená data pak můžeme použít buď docela prostě, nebo velmi propracovaně. Aplikaci Access můžeme použít pro vytvoření adresářů či pozvánek nebo je možné si vytvořit databázi, kam si uspořádáme svou sbírku CD. S určitou znalostí programování můžeme vytvořit aplikace pro uživatele v malých firmách (například s propojením k programům Microsoft Outlook, Microsoft Word nebo Microsoft Excel) nebo vytvoříme propracovanou databázi klient-server pro více uživatelů, která bude fungovat na síti intranet či Internet (s propojením k datům uloženým v databázích na serveru Microsoft SQL Server).

Aplikace Access poskytuje skvělé nástroje pro vytváření formulářů určených k zadávání a upravování dat a nástroje pro vytváření sestav, které můžeme použít pro shromažďování a prezentování daných dat. Formuláře a sestavy mohou být připojeny k místním tabulkám, propojeny s vnějšími zdroji dat nebo převedeny na datové stránky pro zobrazování informací na síti WWW nebo na firemní síti intranet (FEDDEMA, H., 2002).

### 2.2 Objekty Microsoft Access

Součástí databázového systému Microsoft Access jsou **objekty**, které slouží pro ukládání či manipulaci s daty. Mezi tyto objekty patří **tabulky, dotazy, formuláře, sestavy, makra, moduly** a **datové stránky** (FEDDEMA, H., 2002).

### 2.2.1 Tabulky

Tabulky jsou základním prvkem relační databáze a jsou místem, kam Access ukládá všechna data. Tabulky databází se skládají z řádků a polí. *Řádek* (nebo záznam) je sada dat, týkajících se určitého předmětu; *pole* je kategorie informace.

Například tabulka Chemikálie obsahuje pro každou nebezpečnou chemikálii v laboratoři jeden řádek (záznam) a informace ukládá do polí jako například Název, Vzorec, Skupina atd. Různá pole mohou ukládat různé druhy dat, jako jsou čísla, text a hodnoty Ano/Ne. Tím se zajistí správný tvar a zobrazení zadaných dat. To je důležité pro řazení dat a provádění výpočtů. K identifikaci jednotlivých záznamů v tabulkách je možné využít tzv. primárního klíče. Ten se používá nejen při propojování tabulek relační databáze, ale hraje důležitou roli i při optimalizaci výkonu databáze, např. při vyhledávání v databázi. Primární pole jednoznačně identifikují záznamy v tabulce. Propojením polí mezi tabulkami, základ relační databáze, můžeme vytvořit různé zobrazení dat z více než jedné tabulky (STŘEDULINSKÝ, J., 2005).

### 2.2.2 Dotazy

Dotazy se používají pro uspořádání, filtrování, přidávání, odstraňování a upravování dat v databázích. Uživatel může data v dotazech aktualizovat, vkládat, mazat nebo počítat nové hodnoty (např. sumy, průměry) (FEDDEMA, H., 2002).

Dotaz je v databázi trvale uložen a jeho výsledek je zobrazen ve formě tabulky. Při každém spuštění dotazu jsou data vybírána ze současného obsahu tabulek tak, aby informace byla vždy aktuální. Takto uložené dotazy se pak použijí v dalších objektech databáze, jako jsou formuláře nebo sestavy.

Existuje několik typů dotazů:

- **Výběrové dotazy** – Tyto dotazy načítají data vyhovující určitým podmínkám, seskupují záznamy pro zobrazování souhrnných dat a zobrazují kalkulace prováděné v datových polích.
- **Akční dotazy** – Upravují existující data určitým způsobem. Akční dotazy můžeme použít k odstraňování, aktualizování, přidávání dat nebo k vytvoření nové tabulky.
- **Parametrické dotazy** – Tyto dotazy nás žádají o zadání kritérií před vlastním spuštěním dotazu.

- **Křížové dotazy** – Shrnují data a poté seskupují souhrnné hodnoty do kategorií.
- **Dotazy SQL** – Patří mezi ně sjednocovací, předávací a definiční dotazy a vyžadují určité příkazy SQL, které je třeba sestavit v zobrazení SQL (FEDDEMA, H., 2002).

### 2.2.3 Formuláře

Databáze i projekty používají formuláře pro vkládání, upravování a editování dat v místních nebo propojených tabulkách. Formuláře (stejně jako sestavy) používají *ovládací prvky*, které jsou objekty rozhraní zobrazující data z tabulek nebo dekorační prvky (FEDDEMA, H., 2002). Pomocí těchto ovládacích prvků pak uživatelé zadávají, mění nebo jen prohlíží data v databázi. Formuláře tak usnadňují a zpřjemňují práci s celou databázovou aplikací. Bez nich by bylo nutné měnit data přímo v tabulkách a to by bylo pro uživatele poměrně nesnadné.

Lze také vytvořit *hlavní formulář*, který slouží ke komunikaci s celou aplikací a z něhož je možné spouštět všechny ostatní formuláře (JAKUBEC, Z., 2004).

### 2.2.4 Sestavy

Sestavy se používají pro zobrazení nebo tisk dat, nikoli pro jejich přímou úpravu, takže interaktivní ovládací prvky, jako například pole se seznamem, zde nejsou použitelné. Sestavy však mají několik zvláštních funkcí, které nelze použít pro formuláře, jako například schopnost sečíst číselná pole pro skupiny dat a pro celou sestavu, což je užitečné při přípravě finanční sestavy, která obsahuje rozpisy podle měsíce nebo čtvrtletí. Dále je v sestavách mnoho možností pro zobrazování dat, včetně sloučených sestav, sloupcových sestav (datový list), grafů, diagramů a kontingenčních tabulek a grafů (FEDDEMA, H., 2002).

### 2.2.5 Makra

*Makra* jsou kolekce akcí, které umožňují provádět úlohy v databázi pomocí klávesnice nebo myši. Postupy činností se sestavují jako skládanka (FEDDEMA, H., 2002). Tím jsou makra vhodná i pro běžného uživatele, který není programátorem. S jejich pomocí lze otvírat a zavírat objekty MS Access. Lze do nich psát různé podmínky, které určí, kdy se mají činnosti provést nebo přeskočit (JAKUBEC, Z., 2004).

### 2.2.6 Moduly

*Moduly* jsou kolekce procedur VBA (Visual Basic for Applications), které se používají pro provádění akcí v databázi. Kód VBA umožňuje přeskočit omezení maker a propracovanou databázovou aplikaci plně automatizovat. Procedury můžeme zaznamenávat v samostatných modulech (tj. standardní moduly) nebo v modulech připojených k formulářům a sestavám (tj. kód za formuláři) a můžeme také určit nové objekty databáze v modulech tříd. Některé programovací činnosti, jako například ošetření chyb, vyžadují kód VBA a nemohou být zpracovány makrem.

### 2.2.7 Datové stránky

Datové stránky se používají pro práci s daty přenášenými přes síť Internet či intranet nebo pro práci s daty v databázi. Datové stránky mohou zobrazit data v bezmála takové podobě jako formuláře. Nabízejí také režim pro práci se sloučenými daty, které se více podobají sloučené sestavě než tradičnímu formuláři aplikace Access. Datové stránky nabízejí oproti sestavám výhody v případě, že potřebujeme rozšiřovat data prostřednictvím sítě Internet nebo pomocí e-mailu, kdy uživatel vidí aktuální data při otevření zprávy, oproti stacionárním datům zobrazeným ve snímku sestavy (FEDDEMA, H., 2002).

### 2.2.8 Relace

V relační databázi můžeme mezi tabulkami vytvořit relace. To znamená, že dvě nebo více tabulek může sdílet informace. Relace jsou navázány propojením jedinečného pole (*primárního klíče*) v první tabulce s příslušným polem (*nevlastním klíčem*) v druhé tabulce.

V relaci 1:1 může mít záznam v jedné tabulce pouze jeden odpovídající záznam v druhé tabulce.

V relaci 1:N může mít záznam v jedné tabulce více odpovídajících záznamů v jiné tabulce.

V relaci N:M může mít mnoho záznamů v jedné tabulce více odpovídajících záznamů v jiné tabulce. (FEDDEMA, H., 2002)

### 2.2.9 VBA

Visual Basic for Application (VBA) je implementace jazyka Visual Basic (VB) do prostředí MS Office. Oproti VB je VBA obohacen o objektovou strukturu jednotlivých aplikací balíku MS Office.

Kód VBA lze psát celkem ve třech různých místech: v modulech tříd, v modulech uvnitř formulářů a sestav a v samostatných modulech tříd. Veškerý kód, který operuje s určitým formulářem nebo sestavou patří do modulu tohoto formuláře či sestavy. Nezávislé moduly umožňují vytvářet vlastní objekty, obsahující své vlastnosti a metody.

#### *Kdy je nutné použít kód*

- **Řešení chybových situací.** Pokud používáte makra, a v hotové aplikaci dojde k chybě, objeví se jedno či více dialogových oken s hlášením o vzniku chyby a musíme začít od začátku. V kódu VBA je však možné vytvářet chybové rutiny, zachytávající tyto situace a ošetřující vzniklé chyby.
- **Technologie Automation,** používaná pro výměnu informací s jinými aplikacemi.
- **Transakční zpracování.** Díky němu mají vývojáři plnou kontrolu nad ukončením úlohy či jejím odvolání.
- **Deklarace a volání rutin Windows API.**
- **Provádění úloh spojených s replikacemi.**

(BARKER, F., S., 2002)

### 2.3 Zabezpečení

Jestliže vyvíjíme aplikaci, která bude používána více uživateli, potřebujeme často omezit přístup určitých uživatelů k jednotlivým objektům aplikace. MS Access poskytuje různé způsoby omezení přístupu uživatelů k součástem databáze. Mezi tyto způsoby patří:

- Heslo pro databázi
- Šifrování databáze
- Ochrana kódu VBA heslem
- Uložení databáze v podobě souboru MDE – zkompiluje všechny moduly, odstraní všechny upravitelné kódy a přiřadí databázi k souboru MDE.

- Uživatelská úroveň zabezpečení – umožňuje přidělit různé úrovně přístupu k určitým objektům databáze pro rozdílné skupiny uživatelů.

### 3 DATABÁZE NEBEZPEČNÝCH LÁTEK V LABORATOŘI

#### 3.1 Evidence nebezpečných chemikálií v laboratoři

Součástí práce v chemické laboratoři je evidence nebezpečných látek. Na základě této evidence se pak tisknou štítky na lahve, ve kterých jsou chemikálie uchovávány. Doposud byla evidence vedena pomocí programu MS Excel. Protože bylo potřeba zautomatizovat tisk štítků na lahve a zároveň zefektivnit udržování evidence bylo nutné vytvořit databázovou aplikaci, která by tuto úlohu zajistila.

Jako nástroj pro zpracování této aplikace byla vybrána databáze MS Access, protože je uživatelsky příjemná, její prostředí je většině uživatelů dobře známo již z jiných aplikací balíku MS Office a na fakultě je nainstalována téměř na každém počítači.

Aby aplikace správně fungovala, je doporučeno použít MS Access alespoň ve verzi 2000 a operační systém MS Windows 98 a vyšší. Ke správnému zobrazení všech formulářů je doporučeno použít rozlišení obrazovky 1024x768 bodů a vyšší.

#### 3.2 Návrh a vytvoření objektů databáze

V databázi jsou používány tzv. *konzistentní názvy* pro databázové objekty, což usnadňuje práci s databází, zabýváme-li se tabulkami, dotazy a ostatními databázovými objekty v rozhraní a také zapisujeme-li kód VBA (FEDDEMA, H., 2002).

V následujícím seznamu jsou předpony pro označení šesti hlavních typů databázových objektů:

- Formulář – frm
- Makro – mcr
- Modul – mod
- Dotaz – qry
- Sestava – rpt
- Tabulka – tbl

Aplikaci jsem se snažil navrhnout tak, aby byla pro uživatele příjemná a jednoduchá na ovládání. Všechny hlavní formuláře obsahují ve spodní části sadu tlačítek pro pohyb mezi

záznamy, tlačítko pro přidání nového záznamu a smazání záznamu a zavření formuláře. Tím je zajištěna konzistentnost ovládání, která usnadňuje a zrychluje sžití s aplikací.

### 3.2.1 Tabulky

Při návrhu struktury tabulek jsem vycházel z dat, která mi poskytla vedoucí práce. Tato data byla ve formě sešitu aplikace MS Excel. Hned při prvním pohledu bylo jasné, že bude nutné tuto tabulku rozdělit na několik dílčích tabulek, aby se předešlo duplicitám a zajistila se konzistentnost dat.

Tab. 1 Vybrané sloupce vstupních dat ze sešitu aplikace MS Excel.

| Název                  | Symbol nebezp. |               | S-věta            |
|------------------------|----------------|---------------|-------------------|
|                        | Klasifikace    | R-věta        |                   |
| Amoniak                | C,N            | 34-50         | 26-36/37/39-45-61 |
| Bromičnan draselný     | O,T            | 45-9-25       | 53-45             |
| Bromičnan sodný        |                |               |                   |
| Bromid draselný        |                | 36/37/38      | 26-36             |
| Bromid hořečnatý čistý |                |               |                   |
| Bromopyrogallol        |                |               |                   |
| Calcium formicium      |                |               |                   |
| Cetylpyridinium bromid |                |               |                   |
| Dihydrofosforečnan     |                |               |                   |
| Dimetylsulfoxid        | Xn             | 7-22-31       | (2)-7/8-26-28-43  |
| DL - Methyonin čistý   |                |               |                   |
| Dusičnan barnatý       |                | 8-22-36/37/38 | 17-26-36          |
| Dusičnan draselný      | O,Xi           | 8-36/38       | 17-26-36          |
| Dusičnan hlinitý       | O,Xi           | 8-36/38       | 17-26-36          |
| Dusičnan hořečnatý     | O,Xi           | 8-36/37/38    | 17-26-36          |

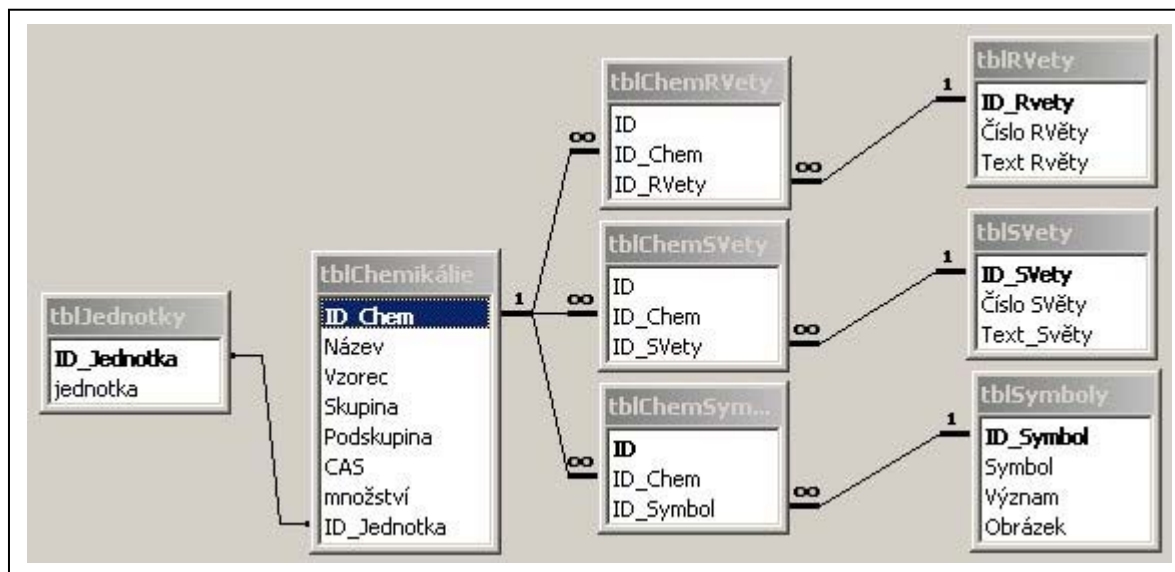
Především se jednalo o oddělení dat ve sloupci „R-věta“, „S-věta“ a „Klasifikace“. Tato data se objevují v každém řádku tabulky jako seznam, jehož jednotlivé prvky jsou od sebe odděleny pomocí znaku „ , “ nebo „-“. Pokud by se změnilo označení některého z prvků musela by se tato změna provést na každém řádku tabulky, kde se prvek objevil. Takováto data by se časem stala neudržovatelná a používání výsledné aplikace by bylo srovnatelné s používáním programu MS Excel.

Výsledná aplikace proto obsahuje 8 tabulek, které jsou navzájem propojeny. Popis těchto vztahů je uveden v následující kapitole.

Každá tabulka je uvozena předponou *tbl*. V následujícím seznamu je jejich stručný popis a jednotlivé atributy.

- **tblChemikálie** – v této tabulce jsou uchovávány informace o jednotlivých nebezpečných látkách evidovaných v laboratoři. Obsahuje tyto atributy: ID\_Chem (primární klíč) , Název, Vzorec, Skupina, Podskupina, CAS, množství a ID\_Jednotka (což je cizí klíč do tabulky *tblJednotky*).
- **tblSymboly** – tato tabulka obsahuje seznam symbolů nebezpečnosti chemických látek. U každého symbolu je uvedeno: ID\_Symbol (primární klíč), Symbol (textové vyjádření), Význam (význam textového vyjádření), Obrázek.
- **tblRVety** – obsahuje seznam R vět. Atributy: ID\_RVety, Číslo Rvěty, Text Rvěty.
- **tblSVety** – obsahuje seznam S vět. Atributy: ID\_Svety, Číslo Světy, Text Světy.
- **tblJednotky** – obsahuje seznam používaných jednotek soustavy SI.
- **tblChemSymboly** – je tabulka zprostředkující vazbu N:M mezi tabulkami *tblChemikálie* a *tblSymboly*.
- **tblChemRVety** – je tabulka zprostředkující vazbu N:M mezi tabulkami *tblChemikálie* a *tblRVety*
- **tblChemSVety** – je tabulka zprostředkující vazbu N:M mezi tabulkami *tblChemikálie* a *tblSVety*

### 3.2.2 Relace



Obr. 6 Relace mezi tabulkami v databázi

Na Obr.6. jsou relace obsažené v řešené databázové aplikaci. Mezi tabulkami *tblChemikálie*, *tblRVety*, *tblSVety* a *tblSymboly* je vždy relace typu N:M, protože každá chemikálie může mít 0..N R-vět (S-vět, Symbolů) a zároveň se každá R-věta (S-věta, Symbol) může objevit u 0..M záznamů v tabulce *tblChemikálie*. Protože aplikace Access neposkytuje konkrétní typ relace N:M, je tato relace zavedena nepřímou pomocí propojovacích tabulek *tblChemRVety*, *tblChemSVety* a *tblChemSymboly*.

Mezi tabulkou *tblChemikálie* a *tblJednotky* je vazba 1:N, protože každá chemikálie musí obsahovat záznam o jednotce, ve které je evidováno množství látky v laboratoři.

### 3.2.3 Dotazy

- **qryAnorganicke** – vrátí seznam *ID\_Chem* těch chemikálií, které mají atribut Skupina roven „A“. Je použit v makru *mcrTiskAnorganicke* jako filtr.
- **qryOrganicke** – vrátí seznam *ID\_Chem* těch chemikálií, které mají atribut Skupina roven „O“. Je použit v makru *mcrTiskOrganicke* jako filtr.
- **qryChemRVety** – vybírá R-věty ke každé chemikálii. Je použit v modulu formuláře *frmChemikálie* v proceduře *UpdateChemRVety* a *UpdateRVetyList*.
- **qryChemSVety** – vybírá S-věty ke každé chemikálii. Je použit v modulu formuláře *frmChemikálie* v proceduře *UpdateChemSVety* a *UpdateSVetyList*.

- **qryChemSymboly** - vybírá symboly ke každé chemikálii. Je použit v modulu formuláře *frmChemikálie* v proceduře *UpdateChemSymboly* a *UpdateSymbolyList* a funkci *Symboly*.

### 3.2.4 Formuláře

Jednotlivé formuláře v aplikaci jsou různě složité. Mezi ty nejjednodušší patří *frmJednotky*, *frmRVety*, *frmSVety* a *frmSymboly*, které jsou vytvořeny pomocí průvodce a slouží čistě k editaci nevázaných dat v tabulkách *tblJednotky*, *tblRVety*, *tblSVety* a *tblSymboly*.

#### ***frmChemikálie***

je formulář pro správu chemikálií. Tento formulář také obsahuje největší počet procedur a funkcí jazyka VBA, které zajišťují správné plnění ovládacích prvků na samotném formuláři, formulářích *frmPřidejRVety*, *frmPřidejSVety* a *frmPřidejSymboly* a tabulek databáze. Samozřejmostí je přidávání, mazání a pohyb mezi jednotlivými záznamy. Využil jsem též možnosti přidat formulář pro hledání poskytovaný aplikací Access. Z formuláře je také možné přímo vytisknout štítek na lahev. Všechny akce se provádějí pomocí tlačítek ve spodní části formuláře.

**Chemikálie**

**1,2,3 Benztriazol**

Najdi:

Název:

Vzorec:

Organická/Anorganická:

Podskupina:

CAS:

Množství:

**Klasifikace:** F, Xi

vysoce hořlavý  
dráždivý

**R Věta:**

|    |                                                               |
|----|---------------------------------------------------------------|
| 8  | dotek s hořlavým materiálem může způsobit požár               |
| 18 | při používání může vytvářet hořlavé nebo vysoce hořlavé směsi |
| 45 | může vyvolat rakovinu                                         |
| 48 | při dlouhodobé expozici nebezpečí vážného poškození zdraví    |

**S Věta:**

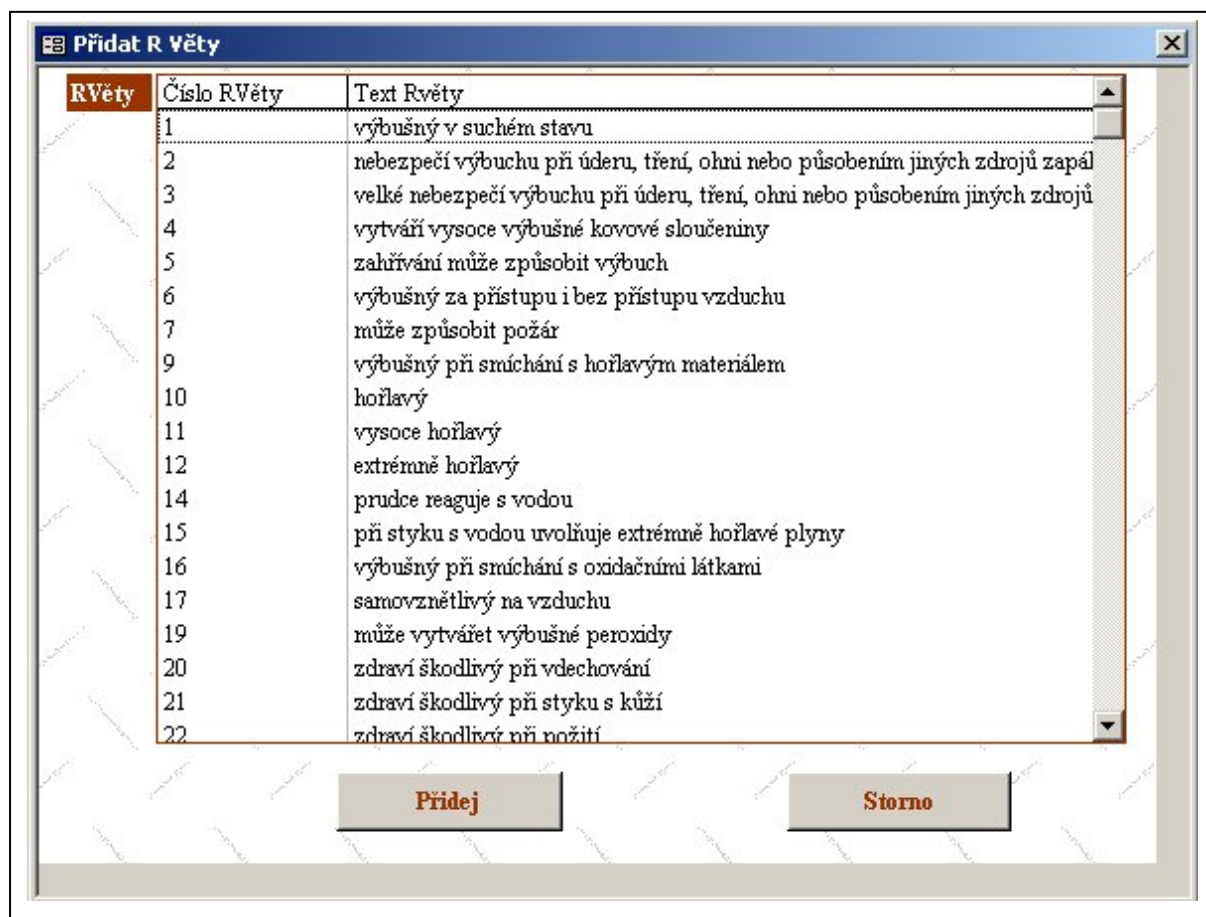
|    |                                                               |
|----|---------------------------------------------------------------|
| 1  | uchovávejte pod uzamčením                                     |
| 5  | uchovávejte pod ... (příslušnou kapalinu specifikuje výrobce) |
| 20 | nejezte a nepijte při používání                               |
| 23 | nevdechujte plyn/dým/páry/aerosoly (příslušný výrobce)        |

**Buttons:** Přidej Symboly, Odeber Symboly, Přidej R Věty, Odeber R Věty, Přidej S Věty, Odeber S Věty

**Navigation:** [Icons for back, forward, search, etc.]

Obr. 7 Formulář Chemikálie pro editaci záznamů tabulky *tblChemikálie*

Při přidávání symbolů, R-vět, S-vět nebo klasifikačních symbolů se vždy otevře nové modální okno (*frmPřidejRVěty*, *frmPřidejSVěty*, *frmPřidejSymboly*) se seznamem, ze kterého si uživatel může vybrat, co chce přidat. Aby uživatel nemohl přidat dva stejné záznamy (symboly, R-věty, S-věty), je seznam naplněn tak, že neobsahuje již zadané položky. Díky tomu pak nemusím testovat, zda-li se již některý z volených záznamů nevyskytuje ve vazebních tabulkách *tblChemRVety*, *tblChemSVety*, *tblChemSymboly*. Je to tedy jednoduchý prostředek, jak zabránit uživateli, aby udělal chybu.



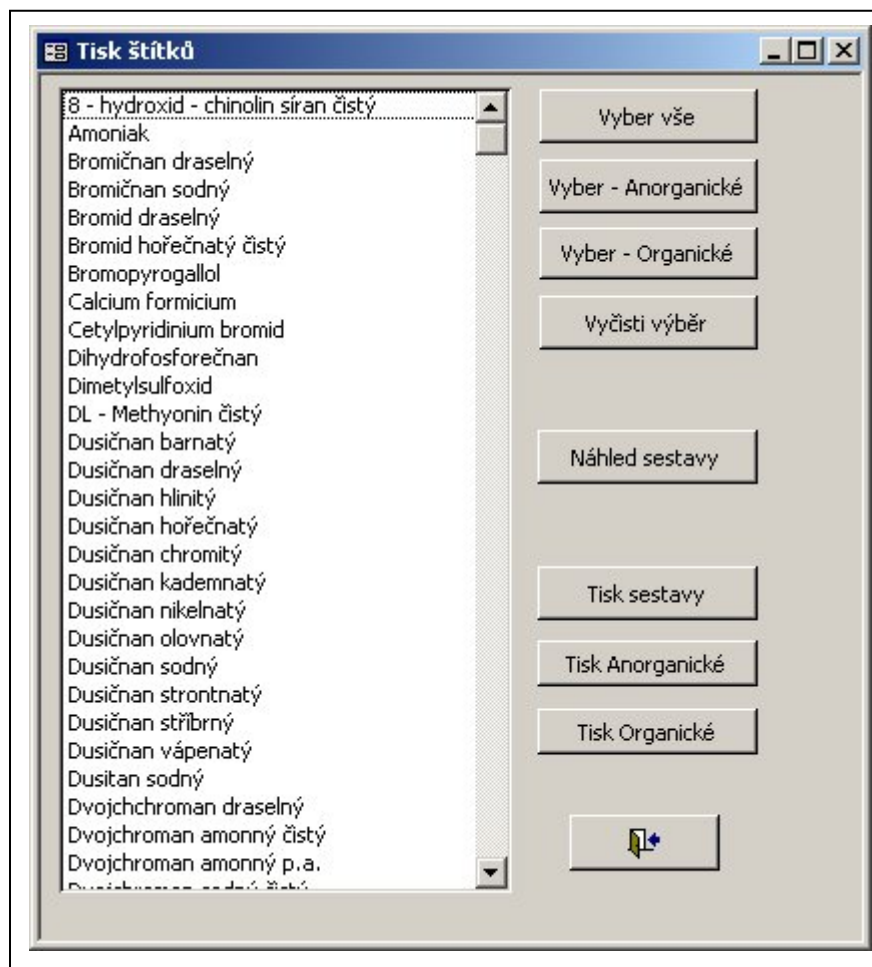
| RVěty | Číslo RVěty | Text RVěty                                                                  |
|-------|-------------|-----------------------------------------------------------------------------|
|       | 1           | výbušný v suchém stavu                                                      |
|       | 2           | nebezpečí výbuchu při úderu, tření, ohni nebo působením jiných zdrojů zapál |
|       | 3           | velké nebezpečí výbuchu při úderu, tření, ohni nebo působením jiných zdrojů |
|       | 4           | vytváří vysoce výbušné kovové sloučeniny                                    |
|       | 5           | zahřívání může způsobit výbuch                                              |
|       | 6           | výbušný za přístupu i bez přístupu vzduchu                                  |
|       | 7           | může způsobit požár                                                         |
|       | 9           | výbušný při smíchání s hořlavým materiálem                                  |
|       | 10          | hořlavý                                                                     |
|       | 11          | vysoce hořlavý                                                              |
|       | 12          | extrémně hořlavý                                                            |
|       | 14          | prudce reaguje s vodou                                                      |
|       | 15          | při styku s vodou uvolňuje extrémně hořlavé plyny                           |
|       | 16          | výbušný při smíchání s oxidačními látkami                                   |
|       | 17          | samovznětlivý na vzduchu                                                    |
|       | 19          | může vytvářet výbušné peroxidy                                              |
|       | 20          | zdraví škodlivý při vdechování                                              |
|       | 21          | zdraví škodlivý při styku s kůží                                            |
|       | 22          | zdraví škodlivý při požití                                                  |

Přidej
Storno

Obr. 8 Formulář frmPřidatRVěty pro přidání R-vět k aktuální chemikálii

***frmTiskSestav***

Tento formulář slouží k volbě chemikálií, pro které chce uživatel vytisknout štítky na lahve.



Obr. 9 Formulář pro výběr chemikálií k tisku štítků

Formulář obsahuje seznam chemikálií v databázi, několik tlačítek pro rychlý výběr, tlačítko pro náhled sestavy a pro její tisk. Výběr chemikálií provádí uživatel buď ručně nebo pomocí tlačítek, jejichž funkci zajišťuje kód VBA na pozadí.

### 3.2.5 Sestavy

Součástí zadání bylo vytvoření tiskové sestavy určené pro tisk štítků na lahve s chemikáliemi. Protože data jsou uložena v navzájem provázaných tabulkách obsahuje hlavní sestava *rptChemikálie* tři podsestavy *rptChemRVěty*, *rptChemSVěty* a *rptChemSymboly*.

Jak u hlavní sestavy, tak u všech podsestav využívám možnosti MS Access zobrazit vybraná data do sloupců. Takto je možné přehledně zobrazit všechny symboly, R- a S-věty na jednom štítku a zároveň zobrazit více štítků na stránku tak, aby nedocházelo k plýtvání papírem.

Pro výběr chemikálií k tisku je určen formulář *frmTiskSestav*, který popisují v předchozí kapitole.

Náhled výsledné sestavy je příloze P 1.

### 3.2.6 Makra

Většina vytvořených maker slouží k otevírání jednotlivých formulářů v režimu pouze pro čtení.

- **mcrOtevřifrmRVetyproČtení** – otevře formulář *frmRvety* jen pro čtení
- **mcrOtevřifrmSVetyproČtení** – otevře formulář *frmSvety* jen pro čtení
- **mcrOtevřifrmSymbolyproČtení** – otevře formulář *frmSymboly* jen pro čtení
- **mcrOtevřifrmChemikálieproČtení** – otevře formulář *frmChemikálie* jen pro čtení
- **mcrOtevřifrmJednotkyproČtení** – otevře formulář *frmJednotky* jen pro čtení
- **mcrTiskAnorganicke** – spustí sestavu *rptChemikálie* s filtrem *qryAnorganicke*
- **mcrTiskOrganicke** – spustí sestavu *rptChemikálie* s filtrem *qryOrganicke*

### 3.2.7 Moduly

V aplikaci jsou použity jen moduly připojené k jednotlivým formulářům.

#### ***Modul FORM\_frmChemikálie***

je nejobsáhlejší modul připojený k formuláři, protože *frmChemikálie* je ústředním formulářem celé aplikace. Většina procedur slouží k vykonání akce po kliknutí na tlačítko ve formuláři. Další skupinou jsou podprogramy, které aktualizují obsah jednotlivých seznamů R-vět, S-vět a Symbolů.

V modulu jsou tři funkce, konkrétně *Symboly*, *Rvety*, *Svety*, které vrátí seznam ID symbolů, ID R-vět a ID S-vět z aktuálního záznamu jako řetězec. Jednotlivé ID jsou odděleny čárkami. Tyto funkce jsou veřejné, aby je bylo možné volat i z jiných modulů. Tato vlastnost umožnila naplnit seznamy ve formulářích *frmPřidejRVěty*, *frmPřidejSVěty*, *frmPřidejSymboly*, pouze záznamy, které se neobjevují v seznamech v hlavním formuláři

*frmChemikálie*. Díky tomu není možné přiřadit jednomu záznamu o chemikálii dva stejné symboly, R-věty nebo S-věty.

Další skupinu tvoří další veřejné podprogramy *UpdateChemRVety*, *UpdateChemSVety*, *UpdateChemSymboly*, které přidávají záznamy do vazebních tabulek po přidání nových Rvět, S-vět nebo Symbolů k záznamu o chemikálii.

### ***FORM\_frmMain***

je modul, který obsahuje pouze procedury k otevření ostatních formulářů aplikace po kliknutí na tlačítko.

### ***FORM\_frmTiskSestav***

obsahuje procedury pro akce, které se provedou po stisku jednotlivých tlačítek ve formuláři.

Při výběru anorganických nebo organických chemikálií do tiskové sestavy jsou použity dva velmi neefektivní podprogramy. Jejich úkolem je označit buď anorganické nebo organické chemikálie v seznamu. Při jejich provádění je nutné pro každou chemikálii zvolené skupiny projít seznam znova od začátku, takže tyto funkce mají kvadratickou časovou náročnost.

Tento nedostatek je možné obejít použitím tlačítek s popisky „Tisk Anorganické“, „Tisk Organické“, která využívají výše zmíněná makra *mcrTiskAnorganicke* a *mcrTiskOrganicke* a provedou přímý tisk zvolené skupiny.

### ***Moduly FORM\_PřidejRVěty, FORM\_PřidejRVěty, FORM\_PřidejSymboly***

Tyto moduly jsou funkčně podobné, a proto je popíšu v jedné kapitole. Procedury v nich obsažené naplní seznam při otevření formuláře, při stisku tlačítka „Přidej“ zavolají příslušnou veřejnou proceduru modulu *FORM\_frmChemikálie*, která přidá vybrané položky seznamu do vazebních tabulek a při stisku tlačítka „Storno“ zavřou formulář.

### 3.2.8 Rozdělení aplikace na část front-end a back-end

Protože se aplikace bude používat ve více-uživatelském prostředí, kdy se každý uživatel může nacházet na jiném místě, je aplikace rozdělena na část front-end (formuláře, makra, dotazy, atd.) a na část back-end (tabulky), která může být umístěna na sdíleném disku ve fakultní počítačové síti.

Při prvním spuštění aplikace je tedy nutné se připojit k *propojeným tabulkám*. Toto lze provést pomocí „*Správce propojených tabulek*“ nebo v menu „*Soubor-> Načíst externí data -> Propojené tabulky*“.

Toto rozdělení jsem provedl pomocí nástroje „*Rozdělení databáze*“.

### 3.2.9 Zabezpečení aplikace

Součástí zadání je navržení aplikace tak, aby bylo možné různým uživatelům omezit přístup k jednotlivým objektům databáze. Proto jsem využil možnosti *Průvodce uživatelskou úrovní zabezpečení* a definoval jsem tyto skupiny uživatelů:

- Správci – mají veškerá práva k aplikaci (editaci, zobrazování, tisk položek a úprava návrhu databáze),
- Úplní uživatelé dat – mohou editovat, zobrazovat a tisknout data,
- Uživatelé pro čtení – mohou zobrazovat a tisknout data,
- Uživatelé – základní skupina, nemá přiřazena žádná práva

Dále jsem nadefinoval tyto uživatele:

- miro – je správcem a vlastníkem všech objektů databáze.
- janacova - je správcem databáze.
- úplný-uživatel – patří do skupiny *úplní uživatelé dat* a může tedy editovat, zobrazovat a tisknout data. Jeho heslo je: chemlab-admin
- uživatel - patří do skupiny *uživatelé pro čtení* a může tedy pouze zobrazovat a tisknout data. Jeho heslo je: chemlab

Hesla pro uživatele miro a janacova nejsou uvedena, aby se zabránilo změnám struktury databáze neoprávněnými osobami.

## 4 IMPORT DAT

Data jsem importoval ze třech druhů zdrojů:

- Sešit aplikace MS Excel
- HTML stránek
- Jiné databáze MS Access

Větší část této úlohy jsem zpracoval s využitím programovacího jazyka PHP a databáze MySQL. Často jsem využíval možnosti importovat data ze souboru CSV a vyměňovat data mezi databázemi MS Access a MySQL pomocí ODBC ovladačů.

### 4.1 Import dat ze sešitu aplikace Excel

Zdrojovým souborem byl sešit aplikace MS Excel. Tento soubor jsem importoval do prázdné databáze MS Access a následně exportoval do MySQL pomocí ODBC ovladačů. Tím jsem vytvořil zdrojovou tabulku.

V databázi MySQL jsem vytvořil stejnou databázovou strukturu jako v databázi Access a napsal jsem PHP skript, který projde všechny záznamy zdrojové tabulky a vytvoří adekvátní záznamy ve výsledné databázové struktuře. Následně je spuštěno doplnění hodnoty CAS, které popisují v následující kapitole. Výsledná databázová struktura s daty je zpětně importována do databáze nebezpečných látek. Ve výsledné databázi pak jen zbývalo přepsat názvy tabulek a obnovit mezi nimi relace.

Věřím, že podobný import lze naprogramovat přímo ve VBA. Protože jsem neměl dostatek času využil jsem raději jazyk PHP, ve kterém programuji a znám jeho možnosti. Vytvoření skriptu a samotný import mi tak trval pouze jeden den.

### 4.2 Import hodnoty CAS

Data byla importována parsováním HTML stránek, které obsahují informace o jednotlivých chemikáliích.

První verze parseru (soubor *extractCAS.php*) byla vytvořena pro parsování stránek stažených lokálně na pevný disk z adresy:

<http://www.verkon.cz/simple/katalog/chemie/cz/chemieM.html>.

Tato data jsem stáhl pomocí nástroje wget. Tím jsem získal sadu souborů, kde každý obsahoval informace o jedné chemikálii. Vlastní parser pak postupně čte soubory z adresáře, pomocí regulárních výrazů, najde hledané informace a uloží je do souboru ve formátu CSV (oddělovačem je středník). Tento soubor jsem pak importoval do aplikace MS Access a následně do MySQL. Samozřejmě jsem mohl získané informace ukládat přímo do databáze, ale ve fázi psaní parsovacího skriptu jsem ještě nevěděl, pro jaké informace budu schopen vytvořit regulární výraz, a proto jsem nebyl schopen vytvořit vhodnou tabulku v databázi.

Druhá verze parseru (soubor *extractCAS2.php*) vznikla přeprogramováním verze první tak, aby parsovala data z fakultních stránek: <http://www.ft.utb.cz/interni/nchl/default.htm>, které jsem si opět stáhnul lokálně na pevný disk. V této verzi jsem již získaná data ukládal přímo do databáze MySQL.

Posledním krokem bylo vložení získaných dat do tabulky *tblChemikálie*. Pro tento účel jsem opět napsal PHP skript *fillCAS.php*, který pro každou chemikálii v tabulce *tblChemikálie* najde odpovídající záznam v tabulkách *extrakt* a *extrakt2* na základě porovnání názvů chemikálií.

### 4.3 R-věty a S-věty

Standardní věty označující specifickou rizikovost (R-věty) a standardní pokyny pro bezpečné nakládání (S-věty) jsem získal ze stránky: <http://www.merci.cz/katalog.php?sekce=283>.

Protože věty jsou zde uvedeny v textovém formátu, kde číslo věty je od popisu věty odděleno různým počtem teček, musel jsem napsat PHP skript (soubor *rsparser.php*), který tyto tečky změní na oddělovač „|“. Skript tedy přečte zdrojový textový soubor, tečky nahradí znakem „|“ a uloží do souboru CSV (soubor *rsvety.csv*). Tento soubor jsem pak rozdělil na dva samostatné soubory, jeden pro R-věty a druhý pro S-věty. V těchto souborech jsem ještě odstranil první znak na řádku, který vždy obsahoval písmeno R nebo S podle toho, o který typ věty se jednalo. Toto jsem provedl pomocí regulárního výrazu v editoru ViM. Odstranění znaku bylo nutné, protože tento znak nebyl uveden datech poskytnutých zadavatelem (každý typ věty byl uveden ve zvláštním sloupci tabulky aplikace MS Excel) a nebylo by tedy možné jednoduše identifikovat R-, S-větu při importu dat (který popisují v kapitole 4.1).

#### 4.4 Klasifikační symboly

Tabulku s klasifikačními symboly jsem překopíroval z podobné aplikace, jako ta kterou jsem vytvořil, kterou mi poskytla vedoucí práce. K tabulce jsem pak vytvořil primární klíč *ID\_Symbol* a změnil názvy sloupců.

## ZÁVĚR

Cílem této práce bylo vytvoření databáze pro evidenci nebezpečných látek v laboratoři v prostředí aplikace Microsoft Access, import dat do databáze a odzkoušení všech jejích funkcí.

Hlavní část správy se odehrává ve formuláři *frmChemikálie*, jehož prostřednictvím uživatel edituje jednotlivé záznamy v tabulce *tblChemikálie*. Z tohoto formuláře je také možné otevřít formuláře pro přidání R- a S-vět a klasifikačních symbolů. Ostatní formuláře slouží k editaci ostatních tabulek databáze a jsou vytvořeny pomocí průvodců aplikace MS Access.

Navržená databázová struktura umožňuje snadno spravovat a udržovat konzistentní data a jednoduše aplikaci rozšiřovat přidáním dalších tabulek a vazeb mezi nimi. Ve vytvořené aplikaci je pak možné efektivně tyto data spravovat a tisknout štítky na lahve s chemikáliemi. Aplikace umožňuje správu ostatních informací, které se uvádějí u jednotlivých chemikálií (např. R a S věty, klasifikační symboly).

Údaje zobrazené na obrázcích jsou reálná data, která však nemusejí být aktuální.

Díky práci na tomto projektu jsem se velmi dobře seznámil s prostředím Microsoft Access. Bakalářská práce splnila všechny části zadání a je připravena k zajištění evidence nebezpečných látek v chemické laboratoři Fakulty Technologické UTB ve Zlíně.

## SEZNAM POUŽITÝCH ZDROJŮ

- 1) BARKER, F., S.: Microsoft Access 2002, programování databázových aplikací. 1. vyd., Praha: Computer Press, 2002. 631 s. ISBN: 80-7226-611-X.
- 2) FARANA, R.: Databázové systémy, MS Access 2.0 [online]. Poslední revize 1995. Dostupný z URL: <<http://www.fs.vsb.cz/books/dbacc20/Welcome.htm>>
- 3) FEDDEMA, H.: Mistrovství v Microsoft Access 2002. 1. vyd., Praha: Computer Press, 2002. 604 s. ISBN: 80-7226-725-6.
- 4) JAKUBEC, Z.: Zpracování metrologické databáze měřidel. Bakalářská práce: UTB ve Zlíně, Fakulta Technologická. 2004. 57 s.
- 5) KOCAN, M.: Databázové systémy neustrnuly, kráčejí směrem ke gridu [online]. Poslední revize 2003. Dostupný z URL:  
<<http://www.computerworld.cz/cw.nsf/ID/B667F2BA515CCF3DC1256E9400332CF7>>
- 6) KOŘENEK, J.: Systém pro správu a dotazování rozsáhlých slovníků ve formátu XML [online]. Poslední revize 2003. Dostupný z URL:  
<<http://nlp.fi.muni.cz/projekty/maxxl2/diplomka/maxxl/>>
- 7) POKORNÝ, J.: Databázová abeceda. SCIENCE, Brno, 1998. ISBN: 80-8603-02-0.
- 8) POKORNÝ, J., HALAŠKA, I.: Databázové systémy. Skriptum ČVUT, Praha, 1998.
- 9) SOUKUP, P.: Aplikace databázových technologií v digitální kartografii [online]. [cit. 2005-05-14]. Dostupný z URL: <<http://gama.fsv.cvut.cz/~soukup/dis/>>
- 10) STŘEDULINSKÝ, J.: Databázový systém pro sledování výpočetní techniky na IIT. Bakalářská práce: UTB ve Zlíně, Fakulta Technologická. 2005. 50 s.

**SEZNAM OBRÁZKŮ**

|                                                                                   |    |
|-----------------------------------------------------------------------------------|----|
| Obr. 1 Databázové systémy .....                                                   | 8  |
| Obr. 2 Ukázka diagramu typů entit a vztahů v E-R konceptuálním modelu .....       | 12 |
| Obr. 3 Diagram binárního vztahu s uvedením min. a max. počtu entit .....          | 13 |
| Obr. 4 Hierarchický model dat .....                                               | 15 |
| Obr. 5 Síťový model dat .....                                                     | 15 |
| Obr. 6 Relace mezi tabulkami v databázi .....                                     | 31 |
| Obr. 7 Formulář Chemikálie pro editaci záznamů tabulky <i>tblChemikálie</i> ..... | 33 |
| Obr. 8 Formulář frmPřidatRVěty pro přidání R-vět k aktuální chemikálii .....      | 34 |
| Obr. 9 Formulář pro výběr chemikálií k tisku štítků .....                         | 35 |

## **SEZNAM TABULEK**

|                                                                       |    |
|-----------------------------------------------------------------------|----|
| Tab. 1 Vybrané sloupce vstupních dat ze sešitu aplikace MS Excel..... | 29 |
|-----------------------------------------------------------------------|----|

## SEZNAM PŘÍLOH

P 1 : Příklad tiskové sestavy *rptChemikálie*